Universitat Politècnica de Catalunya

Departament de Llenguatges i Sistemes Informàtics

Programa de Doctorat de Software

Global Illumination Techniques for the Computation of High Quality Images in General Environments

PhD Dissertation

Frederic Pérez

Advisors: Prof. Xavier Pueyo and Dr. Ignacio Martín

Girona
February, 2003

Preface

Realistic image synthesis is the process of generating images that give a human observer the same visual impression that would be experienced in viewing the real scene. This process involves different areas, like physics (light, material's optical properties), human perception, mathematics (integral equations) and algorithms. In order to generate realistic images, global illumination models are required. These models account for the inter-reflection of light between the elements of the scene. Kajiya showed in 1986 that they are derived from the *rendering equation* [74]. Having its origins in the Radiative Transfer field [146], Kajiya reformulated the equation to model the (optical) physical phenomena of interest from the point of view of image synthesis. Thus, it does not consider for instance the phase of the light (diffraction is not studied, and the scattering is assumed to be *incoherent*, i.e. phase differences between scattered waves are not taken into account [23]), and fluorescence and phosphorescence are deemed unimportant.

The rendering equation in its primitive form considers scenes made up exclusively of surfaces. In this scenario light travels along straight lines ignoring absorption or scattering by the medium, interacting only at surfaces, and the governing equation is an integral equation. When this assumption of vacuum or clear air between objects is not valid, then light interacts not only at surface points but also at any point of the participating media, and the resulting governing equation is an integro-differential equation. Solving this equation efficiently for realistic rendering is the ultimate objective of this thesis (Chapters 4–6).

There are two main properties that make this problem difficult. The first is that a complete solution in the Radiative Transfer field involves temperatures, radiances, and other physical properties at *all points* within the medium. As a simplification, usually in the Computer Graphics field the temperatures are not considered as unknowns but as constant values. The second difficulty is that when dealing with gases, there is a strong variation of the properties with respect to the wavelength, so that may require a concise spectral analysis. For the visible part of the spectrum in which we are interested we do not do such a concise study, but the usual spectrum subdivision in a set of few intervals is used.

In the dissertation we use the term general environments to refer to environments that can potentially contain participating media and whose elements' optical properties are not restricted to isotropic models, i.e. diffuse surfaces and isotropic participating media. Two aspects are then relevant for us (Chapter 2):

- The interaction of light with surfaces. This is governed by the reflectance equation (Equation 2.2). Its kernel is the bidirectional reflectance distribution function (BRDF) that models how light is reflected.

- The processes that affect the illumination within the participating media: absorption, emission and scattering. The distribution of the scattering events is modeled by the phase function.

The resolution of the global illumination problem in scenes in vacuum has been studied extensively. Methods exist dealing with a variety of BRDFs, ranging from Lambertian to specular and intermediate glossy models. These methods are wide spread and known, whereas the treatment for participating media is not that known. This is the reason why we decided to study specifically the existing methods dealing with participating media (Chapter 3).

One of the first studies of radiation through participating media is that of the absorption and scattering in the atmosphere of the Earth. Lord Rayleigh treated the problem of the blue skies and sunset in 1871. In Computer Graphics participating media can be defined as any object which affects the light field in its interior region. Examples of participating media include haze, fog, smoke, steam, dust suspensions, clouds, the atmosphere and flames. On the front page a rendered image of a steaming coffee pot within a kitchen scene globally illuminated—generated with the methods presented in this dissertation—is shown.

Acknowledgments

I am very thankful to my advisors Xavier Pueyo and Ignacio Martín and to my tutor Isabel Navazo for their support and encouragement throughout the development of this work. I am also very grateful to François Sillion, whose insights and different points of view played a key role in my learning experience and had great influence on the dissertation.

I would like to thank my colleagues of the graphics section of the Universitat Politècnica de Catalunya (UPC) and of the Graphics Group of Girona (GGG) for their help and collaboration. I particularly remember my collaborations with Marc Vigo and Carles Mas at the UPC. Ignacio Martín wrote lots of great code at GGG, and I could take advantage of it. Mateu Sbert, Toni Sellarès and Pere-Pau Vázquez were sources of enthusiasm and discussion. Eva Cerezo read several chapters, during her postdoctoral time at the GGG, and provided useful feedback. Roel Martínez and Gonzalo Besuievsky added a plus for enjoying my time in the lab. I also want to express my gratitude to all the members of the *i*MAGIS group in Grenoble (France) for the terrific environment (both inside and outside the lab) they shared with me. I am especially pleased to have met Fred Drouet. Similarly, I had the pleasure of working for the "SIMULGEN" project, with very creative people including George Drettakis, Annette Scheel, Marc Stamminger and Neil Gatenby. I am also thankful to the unknown reviewers of the different submission I have been involved with, since their valuable suggestions contributed to improve this work. Additionally, my gratitude extends as well to the system managers I have met during these years, especially to Robert Valentí, for their quick help whenever I was in trouble. Lastly, Núria Castell and Mercè Juan of the Llenguatges i Sistemes Informàtics (LiSI) department of the UPC gave me more than a helping hand in the bureaucracy for the completion of this work.

This research has been supported by different grants and projects. Part of it has been developed at *i*MAGIS with a grant from the ERCIM Computer Graphics Network, an AIRE grant from the Generalitat de Catalunya, and within the ESPRIT Open LTR project #25772 "SIMULGEN". The rest of the thesis has been performed partly at LiSI in Barcelona and at GGG in Girona, funded with MCYT's projects TIC92-1031-C02-01, TIC95-0630-C05-05, TIC95-0614-C03-03, TIC98-0586-C03-02 and TIC2001-2392-C03-01, by 2001/SGR/00296 of the Generalitat de Catalunya, and by ESPRIT Open LTR project #35772 "SIMULGEN".

Finally, I want to thank my family and friends for their support during all these years, no matter where I was. I am especially grateful to Montse for her complicity and understanding, and to Mar for being my putative sister. Judit deserves my deepest gratitude for adding a lot of meaning to my life. I dedicate this thesis to her.

Frederic Pérez

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The objective of this thesis is the development of algorithms for the simulation of the light transport in general environments to render high quality still images. To this end, first we have analyzed the existing methods able to render participating media, with special emphasis on those that account for multiple scattering within the media. Next, we have devised a couple of two-pass methods for the computation of those images. For the first step we have proposed algorithms to cope with general environments. The second step uses the coarse solution of the first step to obtain the final rendered image. These proposed methods, together with the rest of the contributions of this thesis are briefly listed below. Next, the organization of the thesis is presented.

1.1 Contributions

In this thesis we have made the following contributions:

- A study of single and multiple scattering methods, characterizing them by identifying their base techniques, assumptions, limitations and range of utilization.

- Two first pass methods to solve the global illumination problem—the first based on finite elements (based on Hierarchical Radiosity with Clustering), and the second on Monte Carlo (based on Hierarchical Monte Carlo Radiosity). These methods are able to deal with scenes that are more complex than the methods they are based on.

- A specific design for the construction of PDFs for importance sampling (Link Probabilities) based on the results of a first pass execution. Their adaptive nature makes them appropriate for their use in final gathering algorithms, obtaining better samples than fixed schemes.

- Two methods based on conductance maps for progressive radiance computation. The algorithms are able to deal successfully with features like reflections, highly tessellated models and translucent objects.

1.2 Overview

The structure of the rest of this dissertation is outlined in the following.

Chapter 2: Global Illumination Fundamentals

In this chapter the principles of global illumination for general environments are reviewed, with the most important equations—the *rendering equation* and the *transport equation*—whose solution constitutes the *global illumination* problem.

In order to solve the global illumination problem, a certain number of multi-pass methods exist (e.g. [109, 177, 149, 144, 28, 166]). These methods combine the strengths of different existing strategies. Their objective is to be able to skip restrictions on the number of types of light paths that could be dealt with a single technique, or increase efficiency and/or accuracy. We have opted to follow this philosophy, and two two-pass methods have been developed for general environments.

Chapter 3: Participating Media Resolution Methods

This chapter includes the study of the methods that perform the single scattering approximation (defined in Chapter 2), and also the study of the ones that take into account multiple scattering. We have based the study of the latest on our survey of '97 [122], extended to include any method that did not exist at that moment.

Chapter 4: First Pass: Computation of a Coarse Solution

In our first pass a rough estimate of the global illumination is computed quickly. Knowing the benefits of hierarchical approaches, which started in the beginning of the nineties with the seminal work by Pat Hanrahan et al. [58], two concrete algorithms based on hierarchies have been extended to be more generic:

- *Hierarchical Radiosity with Clustering*. This algorithm is based on the work by François Sillion [147], where a unified algorithm was proposed for the simulation of light transfer between diffuse surfaces, isotropic participating media and object clusters, and by the work by Sillion et al. [148], a hierarchical algorithm capable of dealing with non-diffuse surfaces. From the reflection and the scattering equations we have identified the expressions needed to transport light between all kinds of objects (surfaces, media and clusters).

- *Hierarchical Monte Carlo Radiosity (HMCR)*. The HMCR algorithm by Bekaert et al. [14, 12] has been extended for surfaces with a combination of diffuse plus specular components, and for participating media.

Chapter 5: Second Pass: Link Probabilities

Using the coarse solution obtained by the first pass, our second pass computes a high quality solution from a given viewpoint. Radiances and source radiances are estimated using Monte Carlo processes in the context of path tracing acceleration and also for final gather. *Probability density functions* (PDFs) are created at ray intersection points. For such a task, we initially used constant basis functions for the directional domain. After realizing of their limitations we proposed the *Link Probabilities* (LPs), which are objects with adaptive PDFs in the links-space.

Chapter 6: Progressive Radiance Computation

In order to take advantage of the effort invested for the construction of the LPs presented in Chapter 5, we have devised two closely related progressive sampling strategies. In the second pass, instead of sampling each pixel individually, only a subset of samples is progressively estimated across the image plane. Our algorithms are inspired by the work of Michael D. McCool on anisotropic diffusion using conductance maps [97].

Chapter 7: Conclusions and Future Work

This chapter presents the conclusions of the thesis. Also possible lines of further research are suggested.

Appendix: Implementation

This thesis includes an appendix where some technological work arised from the needs of the thesis is presented.

Chapter 2

Global Illumination Fundamentals

In this chapter we introduce the principles of global illumination by means of its main equations. Excellent and comprehensive explanatory texts on the fundamentals of illumination can be found elsewhere [33, 150, 49, 173]. Notice that, for simplicity, we will omit the frequency dependency from the expressions that follow. We restrict ourselves to still images and static environments, i.e. we deal neither with walkthroughs nor animated scenes. Phenomena like fluorescence or phosphorescence are not within the scope of the dissertation. We start this chapter introducing the so-called rendering equation, for environments in vacuum. Next, participating media are considered and the corresponding transport equation is presented. Finally, the phase functions used in Realistic Rendering are reviewed.

2.1 The Rendering Equation

Energy equilibrium for a set of radiating surfaces is met when the reflected and the transmitted (or absorbed) energy are equal to the incident energy. This is expressed by the *global illumination equation* [150]:

$$\underbrace{L(x,\vec{\omega}_\mathrm{o})}_{\text{total radiance}} = \underbrace{L_\mathrm{e}(x,\vec{\omega}_\mathrm{o})}_{\text{emitted radiance}} + \underbrace{\int_\Omega \rho_\mathrm{bd}(x,\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\, L_\mathrm{i}(x,\vec{\omega}_\mathrm{i}) \cos\theta_\mathrm{i}\, d\sigma_{\vec{\omega}_\mathrm{i}}}_{\text{reflected radiance}}\,, \tag{2.1}$$

where $L(x,\vec{\omega}_\mathrm{o})$ is the radiance leaving point x in direction $\vec{\omega}_\mathrm{o}$, $L_\mathrm{e}(x,\vec{\omega}_\mathrm{o})$ is the self-emitted radiance, $L_\mathrm{i}(x,\vec{\omega}_\mathrm{i})$ is the incident radiance from direction $\vec{\omega}_\mathrm{i}$, $\rho_\mathrm{bd}(x,\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})$ is the bidirectional reflectance distribution function (BRDF) modeling the reflective properties at x, Ω is the set of directions of the hemisphere above x, θ is the angle between the normal of the surface at x and $\vec{\omega}_\mathrm{o}$, and $d\sigma_{\vec{\omega}_\mathrm{i}}$ is the differential solid angle corresponding to direction $\vec{\omega}_\mathrm{i}$.

The *reflectance equation* gives us the reflected radiance from the incoming illumination, and appears as the last term in Equation 2.1:

$$L_\mathrm{r}(x,\vec{\omega}_\mathrm{o}) = \int_\Omega \rho_\mathrm{bd}(x,\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\, L_\mathrm{i}(x,\vec{\omega}_\mathrm{i}) \cos\theta_\mathrm{i}\, d\sigma_{\vec{\omega}_\mathrm{i}}\,. \tag{2.2}$$

In absence of participating media, the visibility function $h(x,\vec{\omega})$ is introduced to couple incoming and outgoing radiances. Concretely, $y = h(x,\vec{\omega}_\mathrm{i})$ when y is the point visible from x in direction $\vec{\omega}_\mathrm{i}$. This allows Equation 2.1 to be rewritten as follows:

$$L(x,\vec{\omega}_\mathrm{o}) = L_\mathrm{e}(x,\vec{\omega}_\mathrm{o}) + \int_\Omega \rho_\mathrm{bd}(x,\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\, L(h(x,\vec{\omega}_\mathrm{i}),-\vec{\omega}_\mathrm{i}) \cos\theta_\mathrm{i}\, d\sigma_{\vec{\omega}_\mathrm{i}}\,. \tag{2.3}$$

Equations 2.1 or 2.3 are usually termed *rendering equations*. Other equivalent formulations employ the set of surfaces of the scene as the domain of the integration or using two-point transport quantities. For example, if $y = h(x, \vec{\omega})$, $v(x,y)$ is the visibility function between points x and y (0 if they are occluded, 1 if they are mutually visible), and $\cos\theta_y$ denotes the cosine between the normal at y and $-\vec{\omega}_\mathrm{i}$, then Equation 2.2 can be rewritten as follows:

$$L_\mathrm{r}(x, \vec{\omega}_\mathrm{o}) = \int_A \rho_\mathrm{bd}(x, \vec{\omega}_\mathrm{o}, \vec{\omega}_\mathrm{i}) \, \frac{\cos\theta_\mathrm{i} \, \cos\theta_y}{\|x-y\|^2} \, v(x,y) \, L(y, -\vec{\omega}_\mathrm{i}) \, dA_y \,, \tag{2.4}$$

where A denotes the set of surfaces of the scene.

Notice that, for the sake of simplicity, we have restricted the global illumination equation (Equation 2.1) to a reflecting (but not transmitting) surface. In order to be truly generic, a third term accounting for the transmitted radiance, similar to Equation 2.2, should be included.

2.2 Participating Media: The Transport Equation

In presence of participating media, further phenomena have to be taken into account, namely absorption, emission, and scattering of radiant energy within the media (see Figure 2.1) [146]. Absorption consists of a transformation of radiant energy into other energy forms. For a differential distance dx, the resulting reduction of radiance is given by $\kappa_\mathrm{a}(x)\,dx$, being $\kappa_\mathrm{a}(x)$ the *coefficient of absorption* of the medium at point x. Emission refers to the process of creation of radiant energy. Scattering means a change in the radiant propagation direction, and is generally split into out-scattering and in-scattering. Out-scattering reduces the radiance in the particular direction along dx by the factor $\kappa_\mathrm{s}(x)\,dx$, being $\kappa_\mathrm{s}(x)$ the *scattering coefficient*. Mathematically the reduction of radiance is expressed as $dL(x) = -\kappa_\mathrm{t}(x)\,L(x)\,dx$, where $\kappa_\mathrm{t} = \kappa_\mathrm{a} + \kappa_\mathrm{s}$ is the *extinction coefficient*. The solution of the previous differential equation is the *Bouguer's law* (also known as the *Beer's law*):

$$L(x) = L(x_0) \underbrace{e^{-\int_{x_0}^{x} \kappa_\mathrm{t}(u)\,du}}_{\tau(x_0,x)} = L(x_0)\,\tau(x_0,x)\,,$$

being the power term $\int_{x_0}^{x} \kappa_\mathrm{t}(u)\,du$ the so-called *optical thickness*, and $\tau(x_0,x)$ the *transmittance* from x_0 to x. Notice that the Bouguer's law simply models the reduction of radiance due to absorption and out-scattering. In order to derive the transport equation, though, more physical phenomena have to be accounted for.

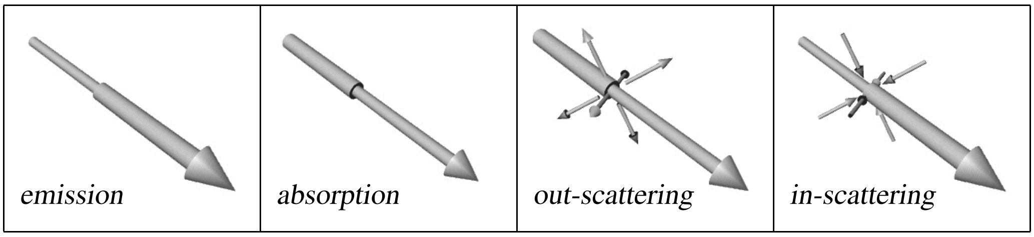


FIGURE 2.1 Interaction of light in a participating medium. Redrawn from [150, p. 175].

Radiance along the propagation direction is augmented because of in-scattering, i.e. because of light impinging on x that is scattered into the considered direction. The spatial distribution of the scattered light is modeled by the *phase function* $p(\vec{\omega}_\mathrm{o}, \vec{\omega}_\mathrm{i})$. Physically the phase function expresses the ratio of scattered radiance in direction $\vec{\omega}_\mathrm{o}$ to the incoming radiance from direction $\vec{\omega}_\mathrm{i}$ by the radiance that would be scattered if the scattering were isotropic (i.e. independent of the direction). Different phase functions have been proposed to model different media. These are reviewed in Section 2.3, and include approximations to the Mie and Rayleigh scattering theories, and empirical functions.

Light interaction with a participating medium is governed by the *transport equation*, describing the variation of radiance in dx around x in direction $\vec{\omega}_\mathrm{o}$ as follows—note that the $\vec{\omega}_\mathrm{o}$ parameter is omitted from the radiances in the following expressions to simplify them:

$$\begin{aligned}\frac{dL(x)}{dx} &= \kappa_\mathrm{t}(x)\,J(x) - \kappa_\mathrm{t}(x)\,L(x) \qquad (2.5)\\ &= \underbrace{\kappa_\mathrm{a}(x)\,L_\mathrm{e}(x)}_{\text{emission}} + \underbrace{\frac{\kappa_\mathrm{s}(x)}{4\pi}\int_\Omega L(x,\vec{\omega}_\mathrm{i})\,p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{i}}}_{\text{in-scattering}} - \underbrace{\kappa_\mathrm{a}(x)\,L(x)}_{\text{absorption}} - \underbrace{\kappa_\mathrm{s}(x)\,L(x)}_{\text{scattering}}\,,\end{aligned}$$

where $\kappa_\mathrm{t} = \kappa_\mathrm{a} + \kappa_\mathrm{s}$ is the *extinction coefficient*, Ω denotes here the set of directions on the sphere around point x, and $J(x)$ is the *source radiance*, which describes the local production of radiance, i.e. the radiance added to the point x due to self-emission and in-scattering. Concretely,

$$J(x) = \underbrace{(1-\Omega(x))\,L_\mathrm{e}(x)}_{J_\mathrm{e}(x)} + \frac{\Omega(x)}{4\pi}\int_\Omega L(x,\vec{\omega}_\mathrm{i})\,p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{i}}\,, \qquad (2.6)$$

where $\Omega = \frac{\kappa_\mathrm{s}}{\kappa_\mathrm{t}}$ is the so-called *scattering albedo*—sometimes also called *single scattering albedo*. The solution of Equation 2.5 is the *integral transport equation* (or *integrated form of the equation of transfer*) shown schematically in Figure 2.2 [146]:

$$L(x) = \underbrace{\tau(x_0,x)\,L(x_0)}_{L_\mathrm{ri}(x)} + \underbrace{\int_{x_0}^{x}\tau(u,x)\,\kappa_\mathrm{t}(u)\,J(u)\,du}_{L_\mathrm{m}(x)}\,, \qquad (2.7)$$

being $L_\mathrm{ri}(x)$ the *reduced incident radiance*, due to the radiance of a background surface (if any), and $L_\mathrm{m}(x)$ the *medium radiance*, due to the contribution of the source radiance within the medium [158]. The boundary conditions of the integral transport equation (Equation 2.7) are represented by the global illumination equation (Equation 2.1).

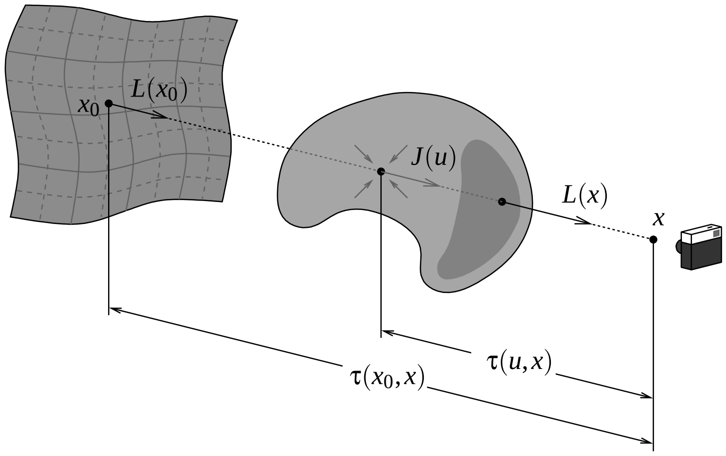


FIGURE 2.2 The integral transport equation: the radiance $L(x)$ at point x in a given direction is the sum of the reduced incident radiance $\tau(x_0,x)\,L(x_0)$ and the contribution of the source radiance within the medium.

The source radiance can be decomposed into three terms, accounting for self-emission, for the (first) scattering of reduced incident radiance, and for the scattering of the medium radiance. Mathematically this is expressed as follows:

$$J(x) = J_\mathrm{e}(x) + \underbrace{\frac{\Omega(x)}{4\pi}\int_\Omega L_\mathrm{ri}(x,\vec{\omega}_\mathrm{i})\,p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{i}}}_{J_\mathrm{ri}(x)} + \underbrace{\frac{\Omega(x)}{4\pi}\int_\Omega L_\mathrm{m}(x,\vec{\omega}_\mathrm{i})\,p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{i}}}_{J_\mathrm{m}(x)}\,. \tag{2.8}$$

By means of the change of variable $t = \int_0^u \kappa_\mathrm{t}(u)\,du$, Equation 2.7 can be rewritten more compactly [136, p. 295] [146, p. 688]:

$$L(t) = e^{-t}\left[L_0 + \int_0^t J(v)\,e^v\,dv\right]\,.$$

Special Case: No Scattering

A particular case is given when the participating media do not scatter (i.e. when $\kappa_\mathrm{s} = 0$). For example, in the first steps of an explosion there is a high emission of light and all kinds of scattering effects can be neglected [170]. Another example is the rendering of fire. Fire is a blackbody radiator (it absorbs but does not scatter) that creates low albedo smoke [105]. Under this assumption, $\kappa_\mathrm{t} = \kappa_\mathrm{a}$, $\Omega(x) = 0$ and $J(x) = L_\mathrm{e}$, and the integral transport equation (Equation 2.7) is considerably simplified:

$$L(x) = \tau_\mathrm{a}(x_0,x)\,L(x_0) + \int_{x_0}^{x} \tau_\mathrm{a}(u,x)\,\kappa_\mathrm{a}(u)\,L_\mathrm{e}(u)\,du\,, \tag{2.9}$$

with the transmittance term $\tau_\mathrm{a}(x_0,x)$ being $\exp(-\int_{x_0}^{x}\kappa_\mathrm{a}(u)\,du)$.

Under the non-scattering assumption, a simple depth of field effect can be achieved through the use of a homogeneous non-emitting medium—κ_a being constant and $L_\mathrm{e} = 0$. In this case Equation 2.9 reduces to

$$L(x) = e^{-\kappa_\mathrm{a}\|x_0-x\|}\,L(x_0)\,.$$

A more interesting effect can be obtained if the L_e term of Equation 2.9 is supposed to be the scattering of a constant ambient illumination in a homogeneous non-emitting medium. This would be a rough approximation of the single scattering case (see Equations 2.11 and 2.12):

$$L(x) = e^{-\kappa_\mathrm{a}\|x_0-x\|}\,L(x_0) + (1 - e^{-\kappa_\mathrm{a}\|x_0-x\|})\,L_\mathrm{e}\,. \tag{2.10}$$

Renderings of a scene based on the rings test scene by E. Haines [56], using Equation 2.10 are shown in Figure 2.3.

The Single Scattering Case

When the participating medium is optically thin (i.e. the transmittance through the entire medium is nearly one) or has low albedo, then the source radiance can be simplified to not take into account the *multiple scattering* within the medium, considering this term—$J_\mathrm{m}(x)$ in Equation 2.8—negligible. Representations for both the single and the multiple scattering cases are depicted in Figure 2.4.

Setting the general conditions under which the single scattering criterion is satisfied is difficult; it is not satisfied for example by clouds because of their high albedo [24, p. 9], being in 0.7–0.9 for cumulus and stratus [107, Sec. 3.1]; also Blinn states that multiple scattering cannot be neglected when $\Omega > 0.3$ [22, p. 28]. Under the single scattering assumption, at point x, the contribution of

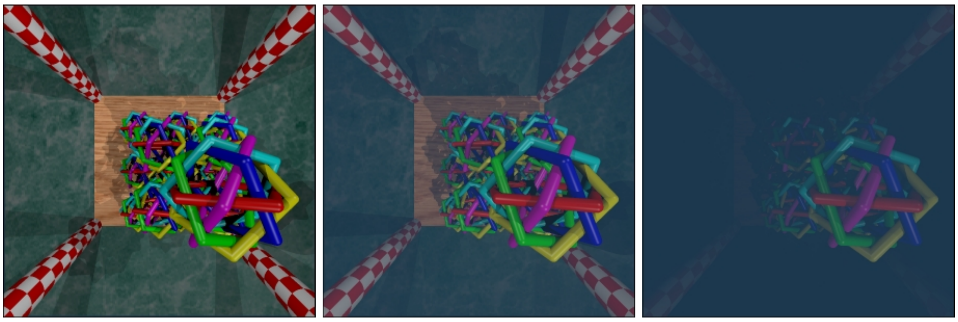


FIGURE 2.3 Renderings of a simple scene in clear air and within non-scattering fog with increasing extinction coefficient.

the scattering of the medium radiance $J_{\mathrm{m}}(x)$ to the source radiance is set to zero, considering that x is the first scattering point of the radiance coming from the background surfaces (*single scattering* case). The expressions for the source radiance (Equations 2.6 and 2.8) and the integral transport equation (Equation 2.7) in this case are simplified as follows:

$$J(x) \approx J_{\mathrm{ss}}(x) = J_{\mathrm{e}}(x) + \frac{\Omega(x)}{4\pi} \int_{\Omega} L_{\mathrm{ri}}(x, \vec{\omega}_{\mathrm{i}})\, p(\vec{\omega}_{\mathrm{o}}, \vec{\omega}_{\mathrm{i}})\, d\sigma_{\vec{\omega}_{\mathrm{i}}}\,, \tag{2.11}$$

$$L(x) = \tau(x_0, x)\, L(x_0) + \int_{x_0}^{x} \tau(u, x)\, \kappa_{\mathrm{t}}(u)\, J_{\mathrm{ss}}(u)\, du\,. \tag{2.12}$$

Single scattering represents a high simplification with respect to multiple scattering. Van de Hulst recognizes that even for the very simplest law of scattering for the individual particles (isotropic scattering—see Section 2.3.1) leads to complex mathematics in the multiple scattering problem [64, p. 383].

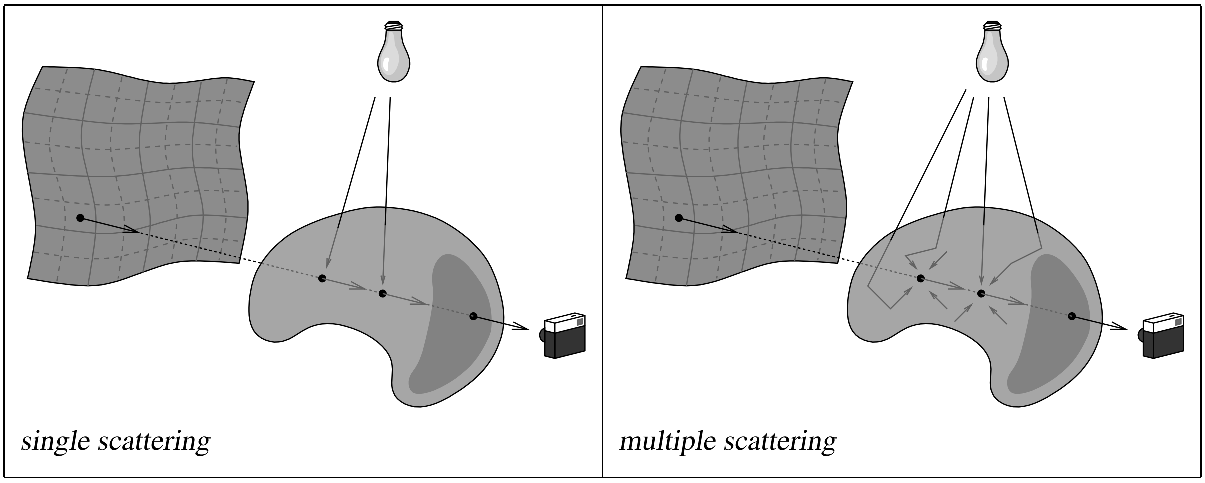


FIGURE 2.4 Schematic representations for the single and multiple scattering cases.

The goal of rendering algorithms is the resolution of the integral transport equation (Equation 2.7) and the global illumination equation (Equation 2.1), at least for the points and directions visible by the camera. As will be seen in Chapter 3, most of the methods that render scenes including participating media accounting for multiple scattering are view independent, and thus they use two stages: the *Illumination Pass*, in which the source radiance $J(x)$ (or other equivalent function) is computed, and the *Visualization Pass*, in which Equation 2.7 is solved for the points of the image plane, using the results of the Illumination Pass.

2.3 Phase Functions

The origins of the term *phase function* are found in astronomy, referring to lunar phases and having no relation with the phase of the light wave [67, p. 280]. A phase function describes the spherical distribution of the scattered light both in the Radiative Transfer and in the Computer Graphics fields. Physically it expresses the ratio of propagated energy in direction $\vec{\omega}_\mathrm{i}$ to the incoming energy from direction $\vec{\omega}_\mathrm{o}$, without taking into account the reflectivity—the albedo—of the participating medium [20, p. C204], or, in other words, it is the ratio of the scattered radiance into one direction by the radiance that would be scattered if the scattering were isotropic [146, p. 534]. It depends on the wavelength, and is a dimensionless and unbounded quantity.

Apart from our notation used for the phase function, $p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})$, a multitude of other different notations for it are found in the literature: $\varphi(\alpha)$, $P(\theta)$, $\varphi(V,V')$, $\Phi(\theta,\phi)$, etc. The parameters refer to input and/or output directions, expressed as vectors or angles. For instance, in the previous expressions $\vec{\omega}_\mathrm{o}$ denotes an outgoing direction, and $\vec{\omega}_\mathrm{i}$ an incoming direction; θ denotes a cone or polar angle, or even the angle formed by the incoming and the outgoing directions, and ϕ a circumferential or azimuth angle. In the case of a phase function that is symmetric around the incoming direction, i.e. when it depends only on the angle between the incoming and the outgoing directions, then it can be parameterized directly by this angle or by its cosine. This is the case in the above $\varphi(\alpha)$ and $P(\theta)$ notations, for example. Being the phase function the volumetric counterpart of the BRDF, some authors call this kind of phase function *isotropic*. However, the term *isotropic scattering* is more commonly found in the literature referring to scattering that is independent of the incoming/outgoing directions (i.e. for the constant phase function—Section 2.3.1).

Following its physical definition, the phase function is described mathematically as follows:

$$p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i}) = \frac{dL(\vec{\omega}_\mathrm{o})}{\frac{1}{4\pi}\int_\Omega dL(\vec{\omega})\,d\sigma_{\vec{\omega}}}\,, \quad [146, \text{Eq. 12-46, p. 534}]$$

being $dL(\vec{\omega})$ the radiance scattered from direction $\vec{\omega}_\mathrm{i}$ into direction $\vec{\omega}$.

A concept related to the phase function is the so-called *scattering function* $f(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})$, which specifies the fraction of radiance arriving from a certain direction $\vec{\omega}_\mathrm{i}$ which is scattered into direction $\vec{\omega}_\mathrm{o}$: [33, p. 326], [20, p. C203], [137, p. 524], [79, p. 217]

$$f(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i}) = \Omega\, p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,.$$

A physically plausible phase function satisfies the following two properties:

1. Following the Helmholtz reciprocity principle, p is symmetric:

$$\forall\vec{\omega}_\mathrm{i}\in\Omega,\ \forall\vec{\omega}_\mathrm{o}\in\Omega,\quad p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i}) = p(\vec{\omega}_\mathrm{i},\vec{\omega}_\mathrm{o})\,.$$

2. Because of energy conservation, the following normalization condition must be satisfied:

$$\forall\vec{\omega}_\mathrm{i}\in\Omega,\quad \frac{1}{4\pi}\int_\Omega p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{o}} = \frac{1}{4\pi}\int_0^{2\pi}\!\!\int_0^{\pi} p(\theta,\phi)\,\sin\theta\,d\theta\,d\phi = 1\,. \tag{2.13}$$

It should be noted that some authors do not include the $\frac{1}{4\pi}$ factor—for example in [24, p. 72], where the normalization condition used is $\int_\Omega p(\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{o}} = 1$.

In Computer Graphics, the phase functions commonly used are symmetric about the line of propagation of incident light. In this case Equation 2.13 can be simplified to be

$$\frac{1}{2}\int_0^{\pi} p(\theta)\,\sin\theta\,d\theta = 1\,, \tag{2.14}$$

and parameterizing the phase function with the cosine of the incoming and the outgoing directions, i.e. by $t = \cos\theta = \vec{\omega}_\mathrm{i} \cdot \vec{\omega}_\mathrm{o}$, the energy conservation condition can be further simplified to be

$$\int_{-1}^{1} p(t)\, dt = 2\,. \tag{2.15}$$

An *asymmetry parameter* (or *anisotropy factor* [181]) is defined to be the average cosine of the scattering angle of a phase function:

$$g = \langle p, \cos\theta \rangle = \int_{\Omega} p \cos\theta \, d\sigma_{\vec{\omega}}\,. \qquad [24, \text{p. } 72], [23]$$

The asymmetry parameter g lies between -1 (strict backward scattering) and $+1$ (strict forward scattering), being 0 in the case of isotropic scattering.

We summarize below the phase functions that have been proposed in Computer Graphics and some plots are given. For more information on phase functions (including plots) in the Radiative Transfer field, the reader is referred to [146, Section 12-9], "Scattering of energy by particles". As stated previously, phase functions in Computer Graphics are usually symmetric around the incident direction, so they can be parameterized by the angle θ between the incoming and outgoing direction, with $\theta = 0$ meaning strict forward scattering. This is the case in the phase functions presented below. For producing the drawings of the polar plots of the phase functions, care has to be taken on the range of the angular parameter—depending on the particular phase function, θ must be in $[-\pi, \pi)$ or between 0 and 2π. In Table 2.1 the different approximations for choosing the scattering theories for different situations are shown.

Condition	*Theory*
$r \ll \lambda$	Atmospheric absorption
$r < \lambda$	Rayleigh scattering
$r \approx \lambda$	Mie scattering
$r \gg \lambda$	Geometrical optics

TABLE 2.1 Criterion for selecting a phase function, comparing the characteristic radius r of the particles suspended in the medium with the characteristic light wavelength λ. Redrawn from [49, p. 759].

2.3.1 Isotropic Phase Function

The isotropic phase function is the simplest one since it is reduced to be a constant:

$$p(\theta) = 1\,.$$

It represents the counterpart of the diffuse BRDFs for participating media—the radiance is scattered equally in all directions. This is the reason why this phase function is used in the zonal method [136], which is an extension of the radiosity method.

2.3.2 Linear-Anisotropic Phase Function

The linear-anisotropic phase function is a simple function with a single parameter that allows a certain anisotropy, that is characterized by the parameter g [161] [146, p. 586] [18, p. 232] [27, p. 6] (see Figure 2.5):

$$p_g(\vec{\omega}_\mathrm{i}, \vec{\omega}_\mathrm{o}) = p_g(\vec{\omega}_\mathrm{i} \cdot \vec{\omega}_\mathrm{o}) = 1 + g\, \vec{\omega}_\mathrm{i} \cdot \vec{\omega}_\mathrm{o}, \quad -1 \leq g \leq 1\,.$$

This phase function is of particular interest in problems relating to planetary illumination [27].

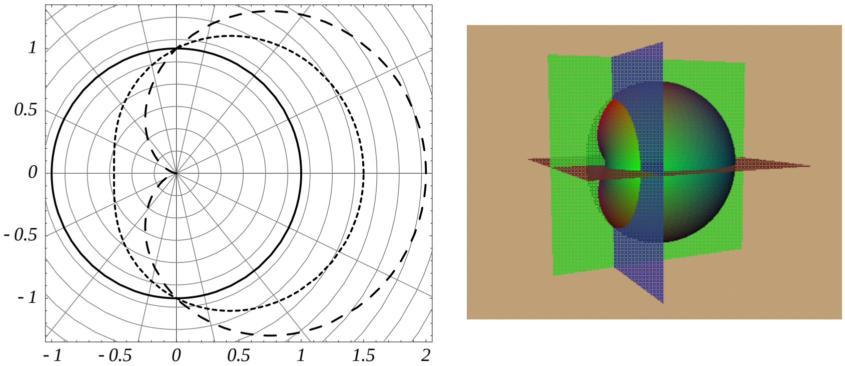


FIGURE 2.5 Left: Polar plots of the anisotropic phase function for g = 0, 0.5 and 1. Right: A 3D representation for p_1.

2.3.3 Rayleigh Scattering

Rayleigh scattering is defined as any scattering process produced by spherical particles whose radii are smaller than about one-tenth the wavelength of the scattered radiation [50]. Siegel and Howell give the approximate size limit for particles of radius r following this kind of scattering as $\frac{2\pi r}{\lambda_m} < 0.3$ (r being the radius of the spherical particles, and λ_m the wavelength inside the particle) [146, p. 578]. This is the case of the particles of the smoke of cigarettes—see [135, p. 14] for information about modeling these particles. It is also the case of the gas molecules of the atmosphere. In fact, if particles in the atmosphere did not scatter, the sky color would be black except in the direct direction to the sun (and to the moon). The Rayleigh scattering theory also explains the blue color of the sky for normal conditions and red color for sunset. This is due to the Rayleigh's law, which states that the scattering coefficient varies inversely with the fourth power of the wavelength—this is why Rayleigh scattering is also known as *selective* scattering [50].

The phase function for Rayleigh scattering is defined as follows [79, p. 216] [20, p. C204] [27, p. 6] (Figure 2.6):

$$p(\theta) = \frac{3}{4}\left(1 + \cos^2\theta\right) .$$

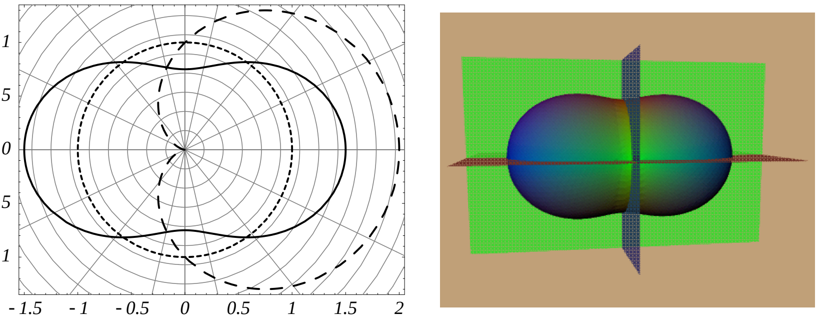


FIGURE 2.6 Left: Rayleigh phase function (solid line), and isotropic and anisotropic phase functions (dashed lines). Right: A 3D representation of Rayleigh phase function. Note the symmetry of Rayleigh scattering: forward scatter equals backward scatter.

2.3.4 Mie Scattering

Mie developed in 1908 a theory of the scattering of electromagnetic radiation by spherical particles [50]. This scattering depends heavily on the size and the electrical conductivity of the particles, and the phase function varies largely for small changes from the spherical shape. In practice approximations to the Mie phase function are used by mixing the properties of distributions of particles with different properties. This is the case of the Henyey-Greenstein functions (which will be reviewed below), that were first used to approximate the scattering of the interstellar dust medium. Other functions to approximate Mie scattering are given by Van De Hulst [63].

Although Mie's theory includes all possible diameter-to-wavelength ratios, it is used for scattering phenomena where the sizes of the particles are not too large, and are not so small to be treated with Rayleigh scattering, i.e. they are within the range $0.3 < \frac{2\pi r}{\lambda_\mathrm{m}} < 5$, where r is the radius of the spheres and λ_m the wavelength of the radiation in the material [146, p. 581]. Therefore the Mie theory can be applied to many meteorological optics phenomena like the scattering by particles responsible for the polluted sky, haze and clouds. It is also applied to scattering of radar energy by raindrops [50].

Approximations to Mie scattering can be of the form:

$$p(\theta) = k\left(1 + m\cos^n\frac{\theta}{2}\right), \qquad \text{[18, p. 230]}$$

$$p(\theta) = \frac{3}{5}\left[\left(1 - \frac{1}{2}\cos\theta\right)^2 + \left(\cos\theta - \frac{1}{2}\right)^2\right]. \qquad \text{[87, p. 13]}$$

Hazy Atmosphere Mie Approximation

Approximation of Mie scattering for *sparse* particle densities are modeled by the two following equivalent expressions (see Figure 2.7):

$$p(\theta) = \frac{1}{2}\left(1 + 9\cos^{16}\frac{\theta}{2}\right), \qquad \text{[108, p. 304], [168, p. C192]}$$

$$p(t) = \frac{1}{2} + \frac{9}{2}\left(\frac{1+t}{2}\right)^8, \; t = \cos\theta. \qquad \text{[20, p. C205]}$$

Murky Atmosphere Mie Approximation

In the case of *dense* particle densities the following two different approximations of the Mie scattering have been proposed (see Figure 2.7):

$$p(\theta) = \frac{33}{83}\left(1 + 50\cos^{64}\frac{\theta}{2}\right), \qquad \text{[108, p. 304], [168, p. C192]}$$

$$p(t) = \frac{1}{2} + \frac{33}{2}\left(\frac{1+t}{2}\right)^{32}, \; t = \cos\theta. \qquad \text{[20, p. C205]} \qquad (2.16)$$

Henyey-Greenstein Phase Function

The Henyey-Greenstein phase function (also known as one-term Henyey-Greenstein—OTHG—phase function) is another mathematically simple approximation to the Mie functions. It is expressed as the equation for an ellipse in polar coordinates, with the parameter g determining the

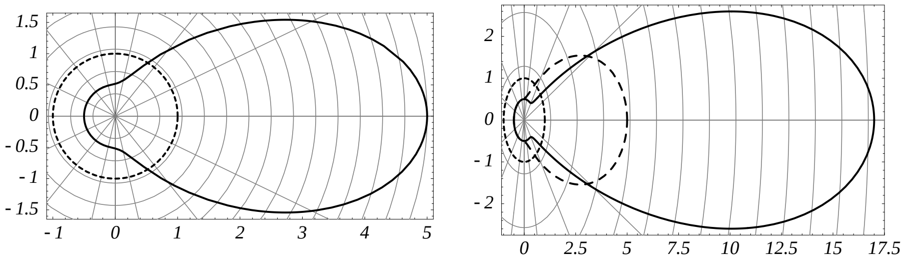


FIGURE 2.7 Left: Hazy approximation phase function (solid line) and isotropic phase function (dashed line). Right: Murky approximation phase function (solid line—using Equation 2.16), isotropic phase function and hazy atmosphere approximation (dashed lines). Notice the different aspect ratios of the plots.

eccentricity [62], [22, p. 25], [20, p. C205] (see Figure 2.8):

$$p_g(\theta) = \frac{1-g^2}{(1+g^2-2g\cos\theta)^{1.5}},\ g \in (-1,1)\,,$$

$$p_g(t) = \frac{1-g^2}{(1+g^2-2gt)^{1.5}},\ g \in (-1,1),\ t = \cos\theta\,.$$

This phase function has been used in the discrete ordinates and P-N methods (Chapter 3), matching closely the complete Mie scattering calculations [146, p. 587]. Max uses the value of $g = 0.55$ to approximate the exact Mie scattering from spherical water droplets [96, pp. 91 and 101]. Blinn used $g = 0.325$ for modeling the scattering of light by rough, sooty particles [22].

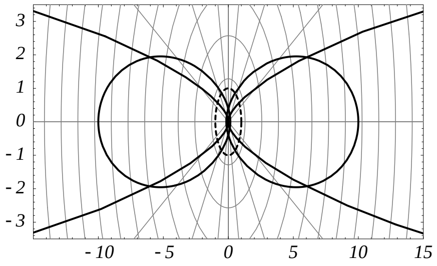


FIGURE 2.8 Henyey-Greenstein phase function for g being -0.9, -0.6, 0.6 and 0.9 (solid lines), and isotropic phase functions (dashed line).

In order to better approximate the shape of real phase functions, a two-term Henyey-Greenstein (TTHG) phase function was introduced [78]:

$$p_{r,g_1,g_2}(t) = r\,\frac{1-g_1^2}{(1-2g_1t+g_1^2)^{1.5}} + (1-r)\frac{1-g_2^2}{(1-2g_2t+g_2^2)^{1.5}}\,,$$
$$r \in [0,1],\ g_1, g_2 \in (-1,1),\ t = \cos\theta\,.$$

Cornette Phase Function

The Cornette phase function has physically more sense than the Henyey-Greenstein phase function, and is defined as follows [110, p. 177] (see Figure 2.9):

$$p_g(\theta) = \frac{3}{2}\,\frac{1-g^2}{2+g^2}\,\frac{1+\cos^2\theta}{(1+g^2-2g\cos\theta)^{1.5}}\,.$$

In this definition the asymmetry factor g is computed as a function of a variable u, which is determined by the atmospheric conditions, ranging between 0.7 and 0.85. The reader can refer to [110, p. 177] for the expression of u.

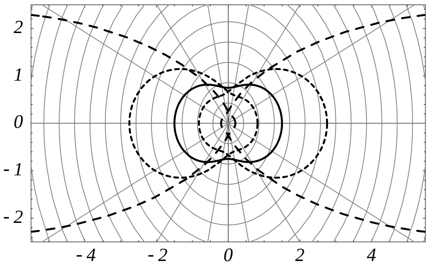


FIGURE 2.9 Cornette phase function for g being 0 (solid line), -0.2, 0.2, -0.6 and 0.6 (increasingly dash length lines).

Schlick Phase Function

Schlick defined the following phase function that is similar to the Henyey-Greenstein phase function, but is faster to compute, and very well suited to be used in Monte Carlo methods—by sampling it within an importance sampling scheme, because its corresponding inverse function can be evaluated quite inexpensively [20, p. C206] (see Figure 2.10):

$$p_g(t) = \frac{1-g^2}{(1-gt)^2},\ g \in (-1,1),\ t = \cos\theta\,.$$

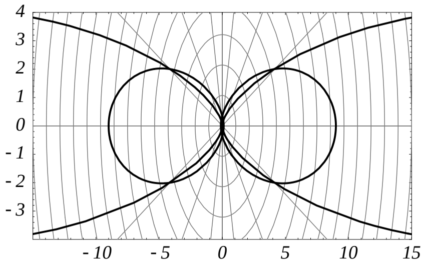


FIGURE 2.10 Schlick's phase function (solid lines) for $g = -0.95$, -0.8, 0, 0.8 and 0.95.

Following the same idea of the two-term Henyey-Greenstein phase function, using a two-term Schlick phase function (a normalized sum of two Schlick phase functions) it is possible to get good approximations to theoretical phase functions:

$$p_{r,g_1,g_2}(t) = r\,\frac{1-g_1^2}{(1-g_1 t)^2} + (1-r)\,\frac{1-g_2^2}{(1-g_2 t)^2}\,,$$
$$r \in [0,1],\ g_1, g_2 \in (-1,1),\ t = \cos\theta\,.$$

This normalized sum also satisfies the energy conservation law.

Blasi et al. approximate Rayleigh scattering by a two-term Schlick phase function with $r = 0.5$, $g_1 = -0.46$ and $g_2 = 0.46$ [20, p. C206] (see Figure 2.11). To approximate the Hazy Mie approximation phase function they set $r = 0.12$, $g_1 = -0.5$, and $g_2 = 0.7$, and to approximate the Murky Mie approximation phase function the fitted parameter values are $r = 0.19$, $g_1 = -0.65$, and $g_2 = 0.91$ [20, p. C206] (Figure 2.11).

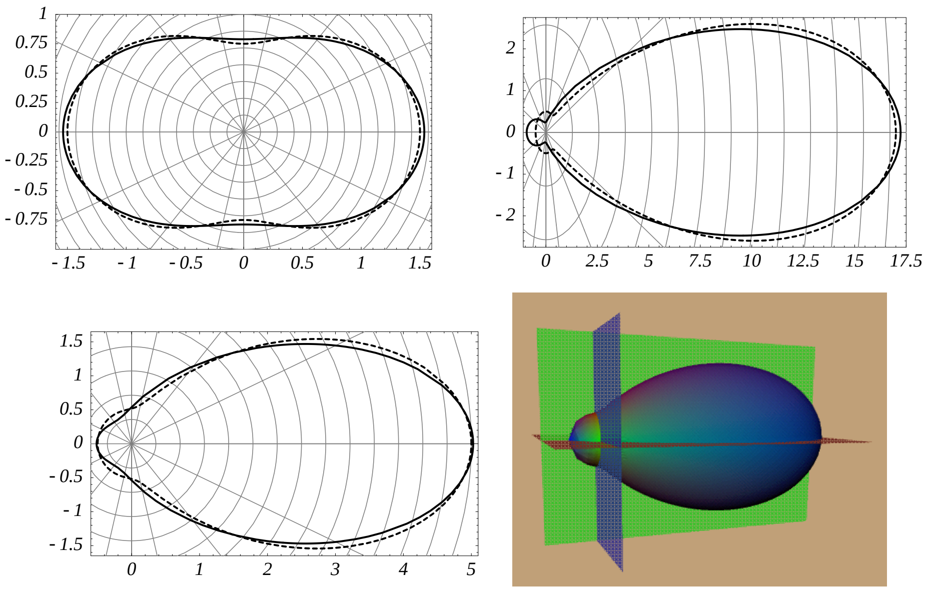


FIGURE 2.11 Top Left: Schlick approximation (solid line) of the Rayleigh phase function. Top Right: Schlick approximation (solid line) of Murky atmosphere approximation (dashed line). Bottom: Schlick approximation (solid line) of Hazy atmosphere approximation (dashed line), and its 3D representation.

2.3.5 Other Phase Functions

Large Diffuse Spheres

For large diffuse spheres, i.e. spheres with $\frac{\pi 2r}{\lambda_m} > 5$ (where r is the spherical particle radius, and λ_m the wavelength in the particle), the following phase function is given [135] [146, p. 573]:

$$p(\theta) = \frac{8}{3\pi} \left|\sin\theta - \theta\cos\theta\right|$$
$$p(\theta) = -\frac{8}{3\pi} \left|\sin\theta + (\pi - \theta)\cos\theta\right| . \qquad (2.17)$$

This phase function is plotted in Figure 2.12. Siegel and Howell explain that diffraction effects are produced for these kinds of spheres, but this is not considered for rendering [146, p. 568].

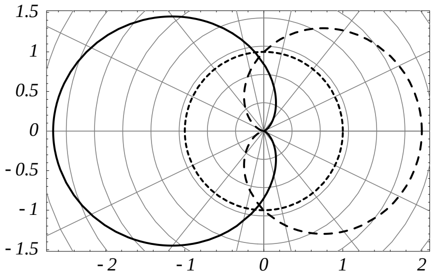


FIGURE 2.12 Phase function for diffuse spheres using Equation 2.17 (solid line), and isotropic and anisotropic phase functions (dashed lines).

Large Specular Spheres

A large specular sphere represents one of the simplest scattering geometries. Siegel and Howell derive the expression of the phase function for this case to obtain the following expression:

$$p(\theta) = \frac{\rho\left(\frac{\pi-\theta}{2}\right)}{\rho_{\mathrm{h}}}, \quad [146, \text{p. } 570]$$

where ρ_{h} is the hemispherical reflectivity and $\rho(\theta)$ is the directional specular reflectivity for incidence at angle θ.

Sakas Phase Function

Sakas proposes the use of the following—unnormalized—heuristic phase function [137, p. 524] (Figure 2.13):

$$\breve{p}_n(\theta) = 1 + |\cos^n\theta|\,,$$
$$\breve{p}_n(t) = 1 + |t^n|,\; t = \cos\theta\,,$$

where n determines the width of the scattered ray. The given expression must be corrected to satisfy the energy conservation law, that is, it must be normalized—see Equation 2.15:

$$p_n(t) = \frac{\breve{p}_n(t)}{\frac{1}{2}\int_{-1}^{1}\breve{p}_n(t)\,dt} = \frac{1+|t^n|}{\frac{1}{2}\int_{-1}^{1}(1+|t^n|)\,dt} = \frac{2\,(1+|t^n|)}{\frac{2(2+n)}{1+n}} = \frac{1+n}{2+n}(1+|t^n|)\;. \tag{2.18}$$

Notice that p_0 and p_2 are equivalent to the isotropic and to the Rayleigh phase functions respectively.

Gaussian Distribution

Another phase function proposed by Sakas is the following Gaussian distribution [137, p. 524] (Figure 2.13):

$$\breve{p}_\gamma(\theta) = \frac{\gamma}{\pi}e^{-\gamma\theta^2}\,,$$

where γ determines the width of the scattered ray. As in the Sakas's phase function, the previous expression should be normalized, using Equation 2.14:

$$p_\gamma(\theta) = \frac{\breve{p}_\gamma(\theta)}{\frac{1}{2}\int_0^\pi \breve{p}_\gamma(\theta)\sin\theta\,d\theta} = \frac{\frac{\gamma}{\pi}e^{-\gamma\theta^2}}{\frac{1}{2}\int_0^\pi \frac{\gamma}{\pi}e^{-\gamma\theta^2}\sin\theta\,d\theta} = \frac{2e^{-\gamma\theta^2}}{\int_0^\pi e^{-\gamma\theta^2}\sin\theta\,d\theta}\,.$$

Numerical values can be obtained for concrete values of γ for the integral in the denominator of the previous expression.

2.4 Participating Media Models

A model for a participating medium must allow the definition of medium emittance, phase function, extinction coefficient and scattering albedo as functions of position in the medium. It is also possible to use other combinations of media coefficients from which the extinction coefficient and the scattering albedo can be derived. This includes the utilization of mass coefficients instead of the

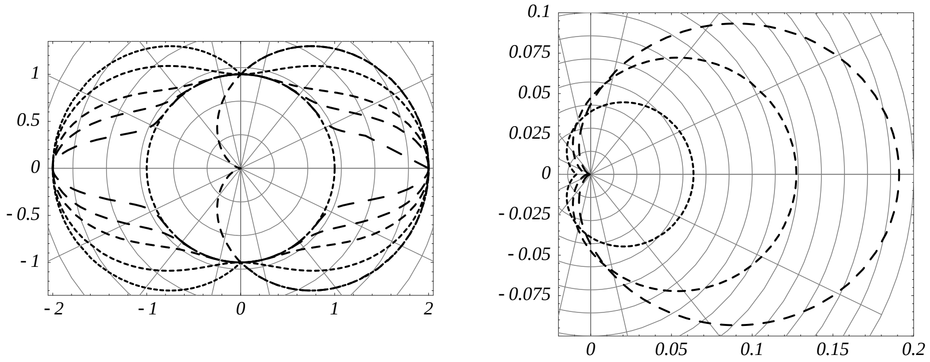


FIGURE 2.13 Left: Unnormalized Sakas's phase function for $n = 1$, 2, 5, 10 and 30 (dashed lines with progressively increasing dash length), and isotropic and anisotropic phase functions. Right: Unnormalized Gaussian Distribution phase function for $\gamma = 0.2$, 0.4 and 0.6 (increasingly dash length lines).

linear coefficients we introduced in Section 2.2. Mass coefficients are simply the result of dividing the corresponding linear coefficient by the material density. In the literature the mass absorption coefficient, for example, can be found to be represented by $\kappa_{a,m}$.

Differently to the case of surfaces, whose geometric and reflectance properties can be treated separately, the definition of the geometry and the optical properties for participating media are tightly related. If the extinction coefficient or the densities of the particles of the medium are given directly as a function of position in space, then the geometry of the medium is implied. Researches have used different representations for participating media. The simplest case is a homogeneous all pervading volume, or layers with constant properties—usually termed as *constant density medium* [22, 95, 79, 108, 186]. For inhomogeneous media heuristic functions have been used [47, 123, 42], as well as texturing functions and fractal algorithms [47, 42, 137, 138], particle systems [188], and blobs [157, 158].

Chapter 3

Participating Media Resolution Methods

In this chapter global illumination algorithms for environments including participating media are surveyed. Our objective is the characterization of those methods: Identification of their base techniques, assumptions, limitations and range of utilization. To this end, after reviewing the applications for rendering participating media, the algorithms for Realistic Rendering are grouped into three main classes: fake media methods, single scattering methods, and multiple scattering methods. Within each class, the algorithms are classified according to different categories and each method is briefly reviewed, followed with an overall discussion. We finish by discussing some applications as well as remaining areas for investigation.

Applications

The rendering of images containing participating media is important for a certain number of applications [134]. Simulations of interest can be made for the following areas:

- Safety analyses: Smoke filled rooms (visibility of exit signs); foggy environments (roadway lighting, relative contrast of target objects such as traffic signs in foggy driving).
- Military: Remote sensing (atmospheric effects attenuate and blur images of land surfaces acquired by distant sensors); underwater vision; battlefield smoke plumes.
- Industrial: Design of efficient headlamps for foggy driving.
- Commercial: Entertainment, virtual reality.
- Visual simulation systems for the training of drivers of cars or ships for which optical effects in participating media are of particular importance; also fire fighter training.

There are other applications where the techniques for dealing with participating media can be used in the Computer Graphics domain. These include the construction of approximate BRDFs of layered materials like paint coating and skin (see for example [57] for a model for subsurface scattering, and [72] for rendering wet materials), and the extension of functions typically related to participating media (like the coefficient of absorption, the albedo and phase function, described afterwards) for clusters (collections of surfaces and/or volumes—see for example Section A.1.1).

3.1 Fake Media Methods

There are some works that deal with the rendering of participating media but do not take into account the physical phenomena involved—although realistic (visually pleasant) results can even be obtained. This is completely acceptable for certain applications, such as the ones mentioned below. We term those methods that are not physically based but rather appearance based *fake media methods*. These include the work by Gardner [47], who has developed a model to generate synthetic clouds for economical visual simulation with enough realism for a wide variety of complex cloud formations. Gardner uses a 2D model with a *sky plane* far enough from the viewer, and a 3D model with ellipsoids and a mathematical function to generate textures. Perlin [123] focuses naturalistic visual complexity in a wider range, introducing a *Pixel Stream Editing language* for the creation of representations of clouds, fire, water, marble, etc., by means of noise and turbulence functions. Yaeger et al. [188] developed a method to display a simulation of the atmospheric flux of the atmosphere of Jupiter for the film 2010. This method mixes physical simulation of fluids dynamics (for the movement of flow field of the atmosphere) and visual simulation (a two-dimensional particle model to generate textures that are mapped onto a polygonalized sphere). Since the observer is restricted to be far away from the planet, its atmosphere is considered opaque, and thus there is no participating media treatment. In fact, Blinn [22] states that the cloudy surface of Jupiter follows the ideal Lambert law very closely.

Recently, in the context of real-time animation of fog, Biri et al. [19] have introduced a model of fog where a set of well-chosen functions for the extinction coefficient κ_t—functions that allow analytical integration—are used to achieve a realistic fog. Considering a constant source radiance within the fog (J_{fog}), Biri et al. extend Equation 2.10 for non-uniform fog: $L(x) = \tau(x_0,x)\,L(x_0) + (1-\tau(x_0,x))\,J_{\text{fog}}$. The transmittances $\tau(x_0,x)$ are efficiently computed because of the definition of κ_t, and the rendering schema uses graphics hardware to achieve animation in real-time.

3.2 Single Scattering

3.2.1 Scientific Visualization and Volume Rendering

Scientific Visualization is the representation of data graphically as a means of gaining understanding and insight into the data. One of the visualization techniques in Scientific Visualization is Volume Rendering, which is used to view 3D data directly. Voxels (volume elements) are used as representation of the volume to determine visual properties, such as opacity and color.

Methods that solve the rendering equation under the assumption of single scattering can be related to Volume Rendering, but there are important differences. The main affinity comes from the fact that the starting point of Volume Rendering is the so-called the *volume rendering integral*, which is the same that the rendering equation. The main distinction is that our concern is Realistic Rendering, whereas the objective of Volume Rendering is the generation of images that aid in the comprehension of a given volume model. Therefore, some assumptions can be done in Volume Rendering that are unacceptable in Realistic Rendering.

What is Rendered?

The unit volume that is rendered can be single valued, representing a certain property (sometimes referred to as a parameter) or multivalued, representing a set of properties. Property examples include temperature, pressure, oxygen concentration, humidity, etc. In Scientific Visualization the set of properties that the user is interested in are shown in the generated image. Filters can be applied to select ranges of values of interest (for example, to know where the temperatures are higher than 40 Celsius degrees). In Realistic Rendering the radiance field is the only property of interest.

The volume rendering integral expresses the radiance variation through a light path, and that is what volume rendering methods must solve. The rest is not applicable or has a slight connection.

Volume Rendering Methods

From the different steps in the volume rendering visualization pipeline, only the *volume viewing* stage has a true connection with Realistic Rendering. Volume rendering methods are commonly classified into two categories (Figure 3.1):

- Object order. In object order *splatting* methods, the final pixel color accumulation is done by image composition [125].
- Image order. Most of image order algorithms cast rays from the observer through each pixel to compute the pixel colors via composition. Sakas and others propose sending pyramids to reduce aliasing (see for example [140]). Usually samples are taken at regular intervals, computing their characteristics by trilinear interpolation of those from the vertices of the related voxels.
  Levoy [92] presents an expression back-to-front to composite colors and opacities of the samples along the line (for a certain wavelength):

$$C(x_i,y_j) = \sum_{k=0}^{n} \left[c(x_i,y_j,z_k)\, \alpha(x_i,y_j,z_k) \prod_{l=k+1}^{n} (1-\alpha(x_i,y_j,z_l)) \right] , \tag{3.1}$$

  where $c(x_i,y_j,z_k)$ and $\alpha(x_i,y_j,z_k)$ are the "intensity" and opacity of k-th sample for the (x_i,y_j) ray, with $\alpha(x_i,y_j,z_0) = 1$ and $c(x_i,y_j,z_n)$ being the background color. Upson and Keeler [171] propose a front-to-back traversal accumulating intensity and opacity using a trapezoid rule quadrature. Equation 3.1 can be demonstrated to be a discretized version of the integral transport equation (Equation 2.7) with c representing the source radiance J and α being one minus the transmittance between consecutive samples. Also c can be considered to be J_{ss} (Equation 2.11) when neglecting multiple scattering. For example, Sakas and Hartig [140] use an illumination model accounting for single scattering for interactive viewing of large scalar voxel fields. Figure 3.2 shows images obtained by ray casting scanned data.

3.2.2 Realistic Rendering

Since the beginning of the 80's, researches have proposed a vast quantity of single scattering methods for Realistic Rendering, focusing a set of different phenomena including participating media. We have categorized the most representative methods according to the technique used (deterministic or stochastic) and also to the type of media they deal with. Table 3.1 shows the resulting classification, with the different publications sorted by chronological order. Usually deterministic methods are applicable to very specific situations; for more general cases stochastic methods (based on some kind of random sampling) are used. In what follows we review each one of these single scattering methods, concluding with a general discussion.

Blinn [22] was the first researcher concerned with the visualization of participating media, in particular for the case of cloud layers (rings of Saturn and planet atmospheres) of spherical reflecting particles distributed uniformly (constant density). Blinn solves analytically the integral transport equation (Equation 2.12) for this case, for single scattering, and for a light source and viewer considered to be at infinity (Figure 3.3).

Kakiya and Von Herzen [75], in the first method introduced in their paper, extend the Blinn single scattering model for ray tracing, eliminating all viewing and lighting restrictions—thus,

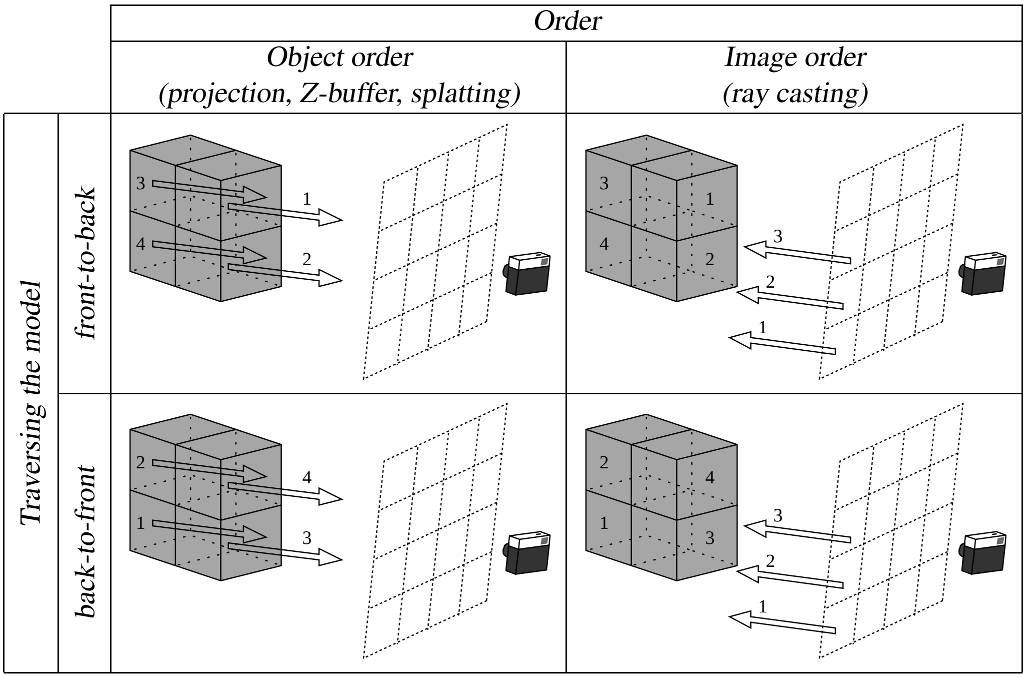


FIGURE 3.1 Volume viewing.

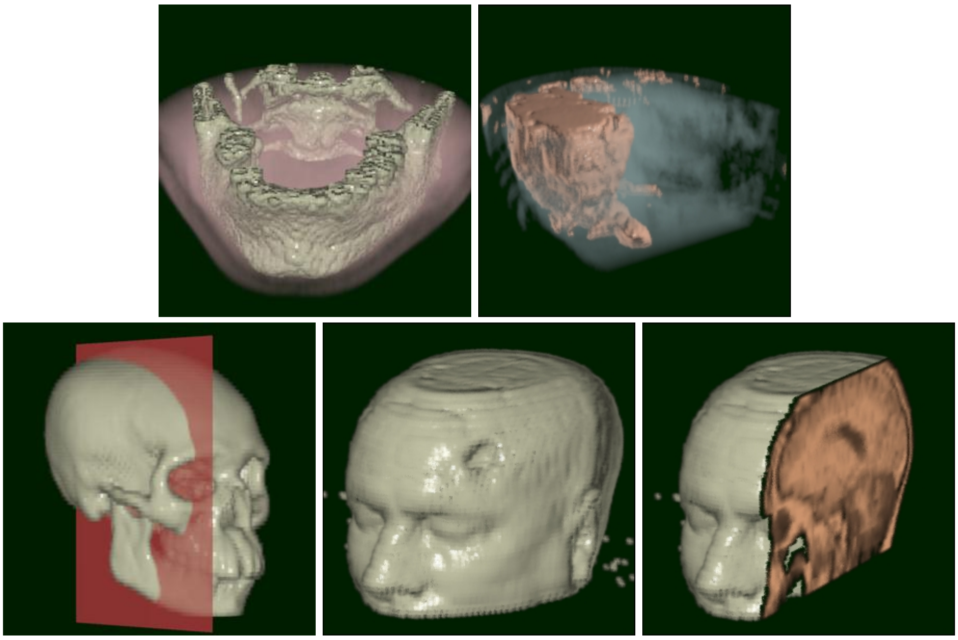


FIGURE 3.2 Ray tracing scanned data (jaw, brain including an isosurface, and head). The images on the top row accumulate intensities along samples using a front-to-back style.

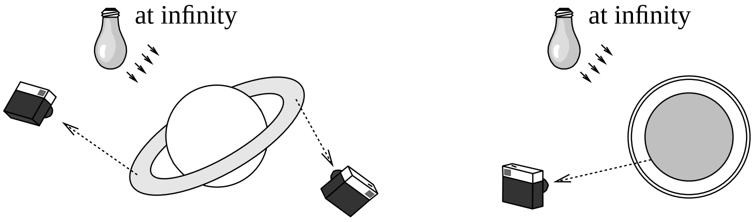


FIGURE 3.3 Configurations solved by Blinn [22].

				Type of Media						
						Atmospheric effects				
Reference	*Analytic*	*Deterministic*	*Stochastic*	*General*	*Smoke*	*Atmosphere*	*Fog*	*Clouds*	*Light shafts*	*Others*
Blinn [22]	✓	✓								✓
Kakiya and Von Herzen [75]		✓		✓				✓		
Max [95]		✓					✓	✓	✓	
Klassen [79]		✓				✓	✓			
Nishita et al. [108]		✓			✓		✓		✓	
Willis [186]	✓	✓					✓			
Rushmeier [133]		✓	✓	✓						
Inakage [65]		✓				✓				
Ebert and Parent [42]		✓		✓	✓			✓		
Sakas [137]		✓	✓	✓	✓			✓		
Kaneda et al. [77]		✓				✓				
Stam and Fiume [160]		✓			✓					
Tadamura et al. [168]		✓				✓				
Nishita et al. [110]		✓				✓				
Stam [157]			✓		✓					
Irwin [66]		✓				✓				
Lecocq et al. [90, 91]		✓					✓		✓	
Dobashi et al. [37]		✓				✓			✓	

TABLE 3.1 Single scattering methods.

the derivation of an analytic solution is infeasible. Representing the participating medium by a voxel model, the rendering procedure is separated into two steps: (1) computation of the radiance arriving at each voxel from all light sources (in other words, computation of the term L_{ri} per voxel in Equation 2.11), and (2) solving the eye radiance, using the intermediate results of the previous step, by an illumination model that is a discretized version of Equation 2.12. This illumination model was later used by Ebert and Parent [42] within a scan-line context.

Max [95] presents a scan-line based method that deals with shadow volumes to take into account the glow in haze. The radiance reaching the eye is computed by adding the contributions of the single scattering of light in the illumination volumes. Two scenarios are considered, shown in Figure 3.4: (1) a light source at infinity illuminating a fog layer with constant or layered density with any phase function; and (2) a point light source and medium of constant density with isotropic scattering.

Klassen [79] deals with two problems: sky color and fog. To render the sky and the sun, Klassen sets an observer on the Earth ground and considers spherical layers of haze-free and haze-filled air (for molecular and particle scattering respectively). Due to the relative distances of the atmosphere and the sun, parallel light rays are assumed. Single scattering is supposed to take place only in the haze-filled layer—however, multiple scattering is not negligible for the computation of sky color, as stated by Bohren [23]. Each pixel color is sampled at 33 wavelengths, taking samples each 10nm, to care for the selective scattering that would be wrong using a three-coordinate space. The fog is restricted to be lit by the sun, without shadows, using parallel light rays and approximating the geometry by a flat Earth. Since fog is not wavelength selective an RGB space can be

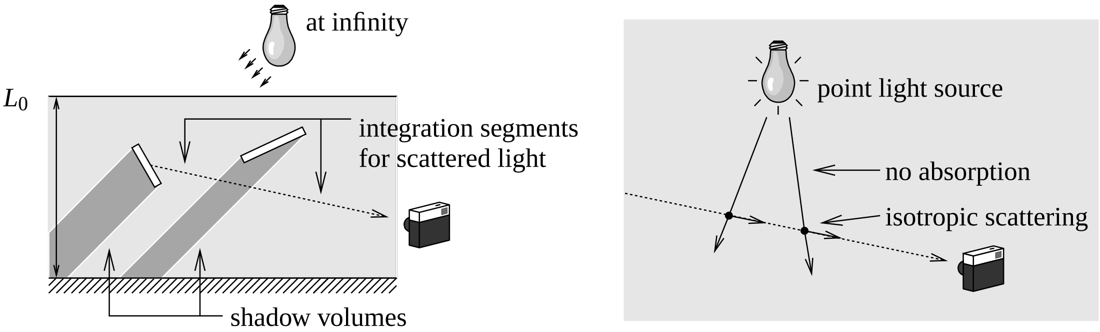


FIGURE 3.4 Two scenarios considered by Max [95].

used.

Nishita et al. [108] present a single scattering method that shares most of the ideas by Max [95]: use of shadow and illumination volumes (for scenes composed of polygons) and integration at lit segments. Shadow volumes only participate in the transmittance term. However, this new method is based in ray casting, while Max's method was based upon scan-line; also this is extended to deal with spotlights with angular distributions apart from parallel light sources (see Figure 3.5). Boundaries between layers of constant media are defined. The scattered light contribution is computed using sampling points along the segments that traverse illumination volumes (integration segments), to diminish aliasing problems and speed-up the whole process. Figure 3.6 shows images for a scene in vacuum and also considering single scattering using a simplified version of the algorithm by Nishita et al.—no shadow volumes were considered.

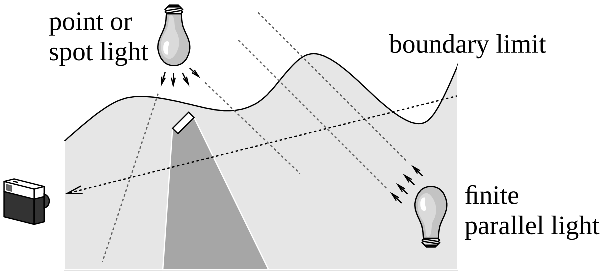


FIGURE 3.5 Schema of the case considered by Nishita et al. [108].

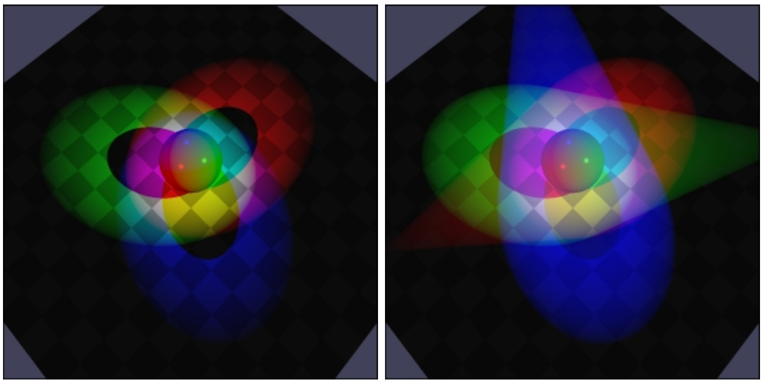


FIGURE 3.6 Ray casting a simple scene (left) without participating medium and (right) with a homogeneous medium using a single scattering approximation.

Willis [186] focuses the problem of flight simulators for scenes containing haze, mist and fog in daylight. His model is extremely simple since it does not consider light sources, but it is supposed that each point in the participating media emits isotropically a certain constant power, considering isotropic scattering.

Rushmeier [133] suggested using the zonal method (discussed in Section 3.3.1), in a first pass, to compute the source radiances. In order to deal with anisotropy, for the single scattering approximation, a Monte Carlo second pass is used as rendering step which accounts for one extra directional bounce toward the viewpoint.

Inakage [65] also considered the visualization of atmospheric effects, like blue skies, sunsets and rainbows. In his model, the atmosphere is constituted by particles of different sizes, that drive to the use of geometric optics and Rayleigh and Mie scattering theories for the computation of their scattering effects. The density and phase function of the scattering particles can vary from point to point.

Sakas [137] presents a method to render arbitrary distributions of volume densities by using projective polygonal rendering and solid texturing techniques. The general equation for the radiance reaching the eye for the single scattering case is derived, which can be solved with front-to-back Monte Carlo sampling. Also two simplifications—two special illumination geometries—are presented: (1) for very low density volumes, absorption of the radiance coming from the light sources to the scattering points is neglected; (2) volume of constant density. A Bresenham algorithm is used to traverse the participating medium that is modeled as a voxel field (better schemes are presented in [139]), and created using fractal techniques.

Kaneda et al. [77] extended Klassen's planar atmosphere model [79] applying it to concentric spherical layers. Air molecules and aerosols were accounted for—thus Rayleigh and Mie scattering were considered—with density distribution varying exponentially with altitude. This model was used for the generation of outdoor scenes including buildings, for which sunlight and skylight were calculated.

Stam and Fiume [160] deal with turbulent fields (i.e. wind) in gaseous phenomena. Their illumination model is similar to that of Nishita et al. [108], and use a front-to-back strategy (a blob renderer) in their rendering technique.

Tadamura et al. [168] present a method to render images of outdoor scenes, taking into account sky effects—of the whole atmosphere. Their work is divided into two parts: (1) computation of the distribution of the skylight intensity, taking into account the scattering and the absorption due to the particles of the atmosphere for different sun altitudes; and (2) a method to compute the illumination at an object's point, using a parallelepiped, and the results of the first part. This is performed by adding the contributions due to sky elements (which represent the light sources) at the point considered. The method presented in (1) is similar to Klassen's method [79]. The elements composing the participating media are air molecules and aerosols, whose density varies exponentially with altitude. The Rayleigh phase function is used to model the scattering due to air molecules, whereas approximations to the Mie theory, for hazy and murky atmospheres (as in [108]), are used for the scattering due to aerosols. Apart from ignoring the multiple scattering illumination, the method also neglects the effect of the ozone layer and the light reflected from the Earth.

Nishita et al. [110] present a method to visualize the Earth from outer space (not from the Earth's surface towards the sky, as for example in [79, 168]), accounting for atmospheric particles (air molecules and aerosols) and the water of the sea. Their method computes: (1) the spectra of the Earth viewed through the atmosphere—Earth illuminated by sunlight, which is affected by atmospheric scattering, (2) the spectra of the atmosphere taking into account absorption and scattering due a particles in the atmosphere, and (3) the spectra of the surface of the sea accounting for the light transmission through water molecules. Lookup tables are used to solve numeric integrations in (1) and (2); for (3) an analytic solution is given. The computation of the optical thickness is done by means of trapezoidal integration of sampled density—the atmosphere is non-uniformly subdivided into imaginary concentric spherical layers, established so that the density variation between two consecutive layers is below a certain threshold; using linear interpolation for the density of each sampling point. Fake clouds—that do not cast shadows onto the Earth—are added by mapping 2D fractal textures onto spherical layers. For the sea color of sea the light reflected on its surface and the contribution for single scattering of water molecules before being refracted towards the viewer are accounted for. The contribution of the color of the bottom of the sea is neglected, due to its depth. Recently, Dobashi et al. [37] have applied a similar method but aiming at obtaining appropriate textures. The final goal is to make the interactive rendering of atmospheric effects possible, by means of using graphics hardware. They also present a method that approximates multiple scattering by an ambient term to obtain images of typical shafts of light caused by sunlight.

Stam [157] introduces *stochastic rendering* of gaseous phenomena modeled as density fields, where the random element is transformed from the model to the rendering component. The statistics of the *intensity field* are related to the statistics of the phenomenon through the illumination equation. Instead of perturbing the model, the intensity is perturbed in a way that is consistent with both the model and the illumination equation. With a proper definition of an *average source radiance* $\bar{J}_{\mathrm{ss}}$, the integral transport equation for single scattering (Equation 2.12) is rewritten as $L(x) = \tau(x_0,x)\,L(x_0) + (1-\tau(x_0,x))\,\bar{J}_{\mathrm{ss}}(x_0,x)$. The transmittance $\tau(x_0,x)$ is computed using the properties of the density field, and for the average source radiance $\bar{J}_{\mathrm{ss}}(x_0,x)$ the algorithm of [160] is used.

Irwin [66] is another researcher concerned with the visualization of the Earth's atmosphere. He simulates the light of the sky and the Sun as seeing from the ground or from outer space, for an atmosphere consisting on pure air, with mass density falling exponentially. Only single scattering and Rayleigh scattering are considered—neglecting the light from the sky (not the sun) reflected from the Earth's surface and ozone absorption. With these restrictions, and modeling the atmosphere as a spherical shell, it is a matter of finding the appropriate line segments corresponding to the geometry to compute illumination paths. These are solved by numerical integration.

In the context of real-time rendering of participating media, Lecocq et al. [90] develop a method able to deal with mobile point light sources within a homogenous medium. Reformulating the transport equation of the model of Nishita et al. [108] in angular terms, Lecocq et al. identify a part of the equation suitable to be approximated by a polynomial. In order to deal with measured directional light sources, a numeric precomputation of the integrals of the developed expression is performed. For quick rendering, a 3D texture is computed (evaluating radiances sparsely at grid points) and added to the image after displaying the surfaces. The reported speed-up with respect to the method by Nishita et al. is 200 and 80 for isotropic and directional light sources respectively. The same authors have applied the method to a driving simulator that allows the testing of headlights (modeled as point light sources) in fog [91].

Discussion

Most of the methods that neglect multiple scattering are focused on atmospheric effects, i.e. in effects caused by the absorption or scattering of light in the air or in the particles present in the air. These methods make it possible to display various effects, such as the light beams caused by spotlights, shafts of light through clouds, foggy scenes, smoke, the sky viewed from the Earth or the Earth viewed from space. A few of them focus on other media such as water [66] or fire [157]. Some proposed methods do not obtain very accurate results, when the single scattering approximation does not hold. (The single scattering approximation is only valid in the case of low albedo or optically thin media—Section 2.2.) This is recognized by Jackèl and Walter [68], who account for a second order scattering to improve the accuracy of their proposal (see Section 3.3.1, p. 34).

A couple of these methods make so strong simplifications not only about the medium, but also about the sources and/or the viewer's position that an analytical approximation can be made [22, 186]. Nevertheless, in almost all applications these simplifications are unacceptable, so that more a complex treatment is needed. In any case, strong simplifications are made either regarding the medium optical properties or the light sources. Most of the methods consider only homogeneous media [22, 95, 108, 90, 91], or planar layers of constant density either [79, 95]. When rendering the Earth's atmosphere, spherical layers are common either of constant [77], exponential decreasing [77, 66] or linearly interpolated density [108]. Other researchers deal with voxels of constant properties [137] or even with properties varying from point to point [65].

Also the spectral dependency of scattering effects were not considered in the original methods (with the exception of the work by Klassen [79]), although in order to reach certain accuracy a certain number of wavelengths—more than three—have to be used. This has been progressively recognized, and more recent works do deal with a sufficient set of wavelengths when necessary.

Related to the light sources, it is common to work only with light sources placed at the infinity, such that the light angle is constant over the environment [22, 95, 79], but also with point light sources [95, 137, 108, 168] and with spotlights with angular distributions [108, 90, 91].

Even in the case of single scattering, the computational cost of calculating light scattering can be too expensive specially when trying to apply it to drive or space/flight simulators where interactive rates are needed. So, as in other areas, there is a general tendency to accelerate the rendering process by using graphics hardware. The intensities of the scattered light are computed in a pre-process and stored in look-up tables. These tables are used as 3D textures that are projected with bilinear interpolation to obtain the output images. This is for example the case of the work of Dobashi et al. [37] who have applied single scattering methods to different atmospheric effects to compute the appropriate textures, or the real-time animations of fog of Lecocq et al. [90, 91] and Biri et al. [19].

3.3 Multiple Scattering

Most of the methods that render scenes including participating media accounting for multiple scattering use two stages: the *Illumination Pass*, in which the source radiance $J(x)$ (or other equivalent function) is computed, and the *Visualization Pass*, in which Equation 2.7 is solved for the points of the image plane, using the results of the Illumination Pass. As we will discuss in Chapter 4 and Chapter 5, we have opted to follow this strategy.

Methods that deal with the more complex problem of multiple scattering (solving Equations 2.6 and 2.7) [122] are summarized and categorized as will be explained below in Tables 3.2 and 3.3. Entries in italic style denote methods that do not solve the global illumination problem, in the sense that in the scene there is only a single volume to illuminate.

We classify the existing methods into two main categories: deterministic and stochastic methods. Deterministic methods are further classified according to the space of directions, discerning between isotropic and anisotropic methods. All isotropic methods use constant basis functions for the computation of form factors. The very first of these methods is the zonal method [136], which is an extension to the classical radiosity method that accounts for isotropic emitting and scattering media. The zonal method has been improved by using hierarchies within the context of the progressive refinements method [17, 155] and also of the hierarchical radiosity (HR) [147].

Deterministic methods can deal with anisotropy by means of spherical harmonics (P-N methods), discrete ordinates, or some implicit representation. Kajiya and Von Herzen [75] expand the radiance in a truncated spherical harmonic basis and construct a system of partial differential equations. Bhate and Tokuta [18] extend the zonal method by using a spherical harmonic basis. Discrete ordinates refers to the discretization of the direction space into a set of bins [113, 86, 96]. Using a grid of voxels to model the participating media, the transport equation can be solved locally per voxel, by means of *local interactions* [113, 86]. At each step the exiting radiance of a voxel is updated, consequently changing the incoming radiance of its neighbors, which must in turn solve their exiting radiances. Alternatively, using *global interactions*, the energy exchange between all pairs of elements can be considered, as zonal method's extension, setting and solving a system of equations whose coefficients are form factors. Max [96] approximates the effects of the form factors avoiding their computation.

Finally some deterministic methods use an implicit representation of the directional distribution of radiance (encoded either in scattering patterns [107], considering only a specific set of directions [106, 59, 60], or by means of a diffusion equation [158, 159, 161]). In the method of Nishita et al. [107], the contributions to the radiance $L(x)$ (for the second and third orders of scattering) in the viewing direction in a point x interior of a participating medium come in the form of a set of extended form factors in a grid that forms a 3D-filter. These form factors must be multiplied by the energy at the related points and accumulated to obtain $L(x)$. Second order scattering is considered in the works by Nishita et al. [106] and by Harris and Lastra [59, 60] for a reduced number of incoming directions.

Stam [158, 159, 161] uses a *diffusion approximation* to solve the multiple scattering between blobs modeling the media. Restricting the medium source radiance of a blob to be of the simple form $J_{\mathrm{m}}(\vec{\omega}) = J^0 + \vec{J}^1 \cdot \vec{\omega}$, a diffusion equation can be written as a system of linear equations allowing the calculation of the source radiance for each blob.

Stochastic methods solve the transport equation by means of random sampling, using random paths along interaction points. We distinguish between the methods that set the interaction points by using a constant step distance [20, 21], from those that sample a function of κ_t, which include light tracing [114], bidirectional path-tracing [83], photon maps [71,84,44,7] and Metropolis Light Transport [116]. Another categorization is made according to the view dependency of the methods. We tag a method as *view dependent* if it is image based or if in the Visualization Pass some extra process is needed to get the value of $L(x_0)$ (e.g. by using a ray tracing).

Space of directions						
Isotropic		*Anisotropic*				
Constant basis functions		*Spherical harmonics*	*Discrete ordinates*			*Implicit representation*
Zonal method: [136]		*[75]* [18]	*Based on local solutions (local interactions)*		*Global interactions*	3D-filter/N directions: *[107, 106, 126, 59, 60]*
Hierarchy						
Progr. Ref.: [17, 155]	HR: [147]		Progr. Ref.: *[113]*	Sweeps: [86]	Sweeps: *[96]*	Diffusion: [158, 159, 161]

TABLE 3.2 Deterministic multiple scattering methods.

	Distance sampling	
	Constant	*Random*
view independent	Light tracing: *[20]*	Light tracing: [114]
view dependent	Light tracing: [21]	Bidirectional path-tracing: [83,39] Photon maps: [71,84,44,7] Metropolis Light Transport: [116]

TABLE 3.3 Stochastic multiple scattering methods.

3.3.1 Deterministic Methods

Constant Basis Functions

Zonal Method. The zonal method [136] is the extension to the classical radiosity method including isotropic participating media, which are modeled by voxels. The voxel radiosity is defined to include only the self-emitted plus the scattered energy (source function for a voxel). Form factors between volumes, and between volumes and surfaces are defined, and the form factors between surfaces are redefined to include a transmittance factor. They are computed by extending the hemicube technique. A system of s (for surfaces—patches) plus v (for volumes—voxels) related equations is constructed, and solved by the Gauss-Seidel iterative method. The direct application of the zonal method has a prohibitive cost: In a regular cube of n^3 voxels there are n^6 form factors; approximating them by the 1D integral along the centers of each pair of voxels in time $O(n)$ (i.e. number of intervening voxels) the computation of all form factors takes $O(n^7)$. Coherence between form factors has been exploited to compute them with lower cost [8].

Progressive Refinement Approach. These methods [17,155] establish a fixed hierarchy in a preprocessing step and thereafter use it in a shooting strategy. There are not further refinements of the hierarchy that would enable the computation of a cheaper coarse solution that could be iteratively improved by refining it, as proposed in [58].

Bhate's method [17] is a progressive refinement version of the zonal method, using hierarchies. These hierarchies are computed in two preprocessing steps, which consist in the subdivision of volumes and surfaces and the creation of links between volumes and volumes and also between surfaces and volumes, determining the level at which a pair of elements must interact. The proposed heuristics for the volume-volume refinement are: Total form factor (when a rough estimate of the form factor is below some specified threshold, then the related elements can interact at the current level), estimated visibility between the volumes, and the optical depth of the intervening medium. Basically, in these latest heuristics, when the transmittance between two elements is too high, there is no need for further refinements. The volume-surface refinement also includes a brightness factor heuristic (for light sources).

Some particular observations of the Bhate's method are:

- *Self-refinement*. Participating media are modeled by a set of *global* volumes. Each global volume is refined against each other, but not against itself (there is no self-refinement). Therefore links must be set between each pair of the smallest volume element considered within a given global volume, otherwise its self-illumination will not be correctly computed. A self-refinement strategy would be preferable since it would reduce the number of interactions.

- *Push-Pull*. There is no Push-Pull procedure to set correctly the values of the radiosity at all levels of the hierarchies. Such a procedure would be needed after each shooting iteration to obtain a correct solution.

In the Sobierajski's method [155] the volumetric data is represented by voxels that can model Lambertian surfaces, isotropic media, or a combination of both. Thus each voxel's BRDF is the sum of ideal diffuse reflection plus isotropic scattering. Depending on the specific coefficients for each component, a voxel can have a more translucent volumetric appearance or resemble more an opaque surface. Therefore, each voxel has a *diffuse* plus an *isotropic* radiosity. Form factors are defined to take into account the relationships between diffuse and isotropic components of the voxel's BRDF, and the surfaces.

The presented technique is an iterative shooting algorithm using hierarchies which—for the case of volumes—are built in a preprocessing step by combining eight neighboring voxels at a certain level to form one voxel of the parent level. A criterion is defined to decide if a parent voxel can be a good approximation of its descendents. The interaction between nodes depends on their levels, their averaged values and the amount of energy transferred between them. There are not explicit links between nodes, instead at each shooting iteration the best highest possible levels of interaction are found *on-the-fly*. After each shooting iteration a Push-Pull procedure assures the correct representation of the energies of all the nodes in the hierarchies.

Some specific aspects are:

- *Self-interaction*. A volume object cannot self-interact. The self-lighting should be considered but will be missed in volumes that are not convex solid objects.

- *Brightness-weighted interaction*. A hierarchy is computed independently for each "volumetric object", grouping the low level voxels recursively to form upper level voxels, and no links exist between different hierarchies. Those links are implicitly computed when finding the interaction levels at which two elements are allowed to interact. A test is done to check if the amount of radiosity shot from an element e_j to another element e_i is below a given threshold. This could be modified to check if the estimated energy shot from e_j *arriving* to e_i is under some threshold (using an estimated form factor). This technique would decrease dramatically the number of interactions when dealing with optically thick media.

- *Push-Pull*. The Push-Pull procedure presented limits the application of the method to media with constant albedo (pushing irradiance instead of radiosity would solve this drawback [147]). It should be noticed that maybe it would be more interesting to use a gathering technique instead of a shooting technique. Performing the Push-Pull procedure after each shoot might imply too high a cost.

Hierarchical Radiosity. Sillion [147] presents a hierarchical radiosity algorithm [58] adapted to include isotropic volumes. To represent energy exchanges within a volume, the self-link (link from the volume to itself) is introduced. This link is subdivided in a different way from links between different elements, since each child must include a self-link apart from the usual links between each pair of children. Furthermore, to avoid the quadratic cost of the initial link phase of the classical hierarchical method, that can be overwhelming in complex scenes, the transfer of energy between groups of objects (i.e. sets of surfaces and volumes) is allowed. These groups of objects compose abstract objects (clusters) that exchange energy as a whole. A hierarchy is created above the surface level, and then the initial linking phase is reduced to the creation of a single self-link from the top of the hierarchy to itself, representing the interactions taking place inside the global volume enclosing the scene. Once the initial link is refined by a recursive procedure, gathering and Push-Pull steps are performed until there is no significant change in the radiosities of any element. Care must be taken to perform correctly the Push-Pull procedure when dealing with inhomogeneous media and textured surfaces. Refinement of the links is done by bounding the radiosity transfer.

Spherical Harmonics

Kajiya and Von Herzen [75] present two methods. The first one deals with single scattering, and the second with multiple scattering within the participating media. The radiance is expressed in a truncated spherical harmonics basis, and a system of partial differential equations (PDEs) is constructed for the spherical harmonics coefficients. The system of PDEs is set and solved by relaxation. Only the constant phase function and the Rayleigh phase function are considered, and the expansion in spherical coordinates is truncated after the fourth coefficient (because “only the first few spherical harmonics are necessary for a convincing image”, but obviously the cost of the method depends largely on the number of coefficients). The effects between surfaces and volumes are not taken into account.

The method by Bhate and Tokuta [18] deals with the effects between surfaces and volumes missed in [75], being an extension to the zonal method, in which the assumption of the isotropy of the medium is eliminated (through a representation of the phase function and radiance by using spherical harmonics) and the surfaces remain ideal diffuse. The phase function is approximated by the first M terms of its spherical harmonics expansion (approximation to Mie scattering). Note that a large number of form factors will have to be computed, taking into account the spherical harmonics. These are calculated with the extended hemicube technique. Finally, a system of Mv equations for v volumes plus s related equations for s surfaces is set and solved using a Gauss-Seidel iterative technique. The direct application of this method is impractical because of its prohibitive cost: In a regular grid of $v = n^3$ voxels the cost to compute the form factors is $O(n^7 + M^2n^6)$.

Discrete Ordinates

Another possibility to account for directional functions is the use of discrete ordinates [146], i.e. a discretization of the full 4π solid angle into a set of bins. These represent particular directions, and it is supposed that, for sufficiently small volume elements, the properties are constant for each direction within each volume. The main problem of the discrete ordinates is the “ray effect” problem [89] since the energy is propagated through discrete directions instead of into the whole discrete solid angle.

Local Interactions.

Patmore [113] formulates the local solution of the transfer equation for the discrete directional model resulting of the subdivision of the volume (resulting in a cubic lattice) and the angular spaces (using in practice 6 or 26 directions). The participating medium considered does not emit light, since the objective is to render clouds. A global solution of the transfer equation is obtained through iteratively obtaining local solutions (related to points of the cubic lattice). As a consequence of a local solution the unshot energies of the related point are updated. A new local solution is computed for the lattice point adjacent and in the direction of the highest unshot energy of the previous one, thus effectively following importance-based paths, until the unshot energy is below some threshold or the path exits the volume. This method computes directly the radiances exiting the volume, so no integration of source radiances is needed in the visualization pass.

The method by Languénou et al. [86, 85] follows a progressive refinement approach. The usual shooting method for surfaces is extended to account for the transmittance through the media, and also source terms within the media are updated accordingly. The radiosities of the boundaries of the media are computed propagating the radiance (coming from the previously accumulated source terms) along all the discrete ordinates and using as many iterations as necessary to converge. Each iteration consists in a loop for each direction, in which a complete sweep of the voxel grid is performed to propagate the accumulated energy through adjacent voxels, starting from a convenient boundary voxel (related to the direction considered), where $O(Mv)$ is the cost per iteration (being M the number of direction bins). Finally, the radiance of the boundary faces of the medium is shot, using hemisphere interpolation. The whole process is repeated until convergence is met. The Visualization Pass computes the pixel radiances by using the source radiances of the voxels.

Cerezo and Serón [26] have extended the discrete ordinates method of [86] to include objects and sources inside the participating medium and to handle highly anisotropic phase functions. They also extend the original method to be able to consider volumetric inelastic processes, particularly fluorescence. This method has been applied to the rendering of underwater scenes where the sea is treated as a participating medium characterized by real experimental medium parametrizations.

Global Interactions. Max's method [96] is devoted to render clouds. The computation of the M^2v^2 form factors of the finite elements formulation is avoided by approximating their effects as the energy is propagated across the grid. For each bin, this propagation is made distributing the flux to the related neighbor voxels simultaneously for all voxels belonging to a layer, in time $O(v \log n)$, for v^2 interactions. The "ray effect" is reduced because the energy is propagated through the whole bin, not only through a single direction. The attenuation between two voxels is not accumulated along the straight line joining them, but along a set of possible propagation paths. The multiple scattering events produced within a single receiving element are accounted for. Since the time to scatter the received flux of a voxel to the direction bins is $O(Mv)$, the final cost per iteration is $O(Mv \log n + M^2v)$. Thus when the number of iterations required to converge is small compared to v, this method is better than computing the whole set of form factors (with a cost of $O(v^2n + v^2M)$) and solving the resulting system.

Implicit Representation

The directional distribution of radiance can be represented implicitly by a scattering pattern [107] or by a diffusion equation [158, 159, 161]. We also include in this category the methods that consider a set of specific directions [106, 59, 60], which can be considered as a restriction of the 3D-filter used in [107].

3D-Filter/Concrete Directions. Nishita et al. [107] propose a method to display clouds taking into account multiple scattering and skylight (light reaching the cloud due to the atmosphere's scattering plus the reflected light from the Earth's surface). Radiance from a cloud reaching the eye is computed from the sunlight multiple scattered plus the skylight single scattered by the particles of the cloud. Of the multiple scattering of the sunlight only the three first orders of scatterings are considered (to save computation time), computing separately the single scattering. For the contribution of the second and third order of scattering to the radiance in the viewing direction, the space including the cloud is subdivided into voxels, with the viewing direction as a principal axis. Instead of computing form factors between each pair of voxels, a smaller space with the mean density of the cloud is set, and the contribution ratios to the radiance of the center voxel (in the viewing direction) from the other voxels are computed, taking into account the sunlight direction. This is the *contribution-ratio pattern*, or 3D-filter. Since the scattering in clouds is mainly forward, most of the energy scattered at a point will lie within a relatively small solid angle. Using this fact it is possible to compute faster the extended form factors, concentrating the effort in those voxels which will effectively contribute to the center voxel, for paths having one or two scatterings (and using an stochastic method to select those voxels). The filter is applied to the voxels in the whole space storing for each voxel the light scattered due to the second and third order scattering in the viewing direction.

Nishita et al. [106] treat the rendering of the sky, viewed from the ground, taking into account multiple scattering. Both Rayleigh and Mie scattering are considered, for the scattering due to air molecules and to large particles such as aerosols and water droplets, respectively. Nishita et al. extend the work by Kaneda et al. [77], using their model of the atmosphere but computing the atmosphere's multiple scattering and the light reflected from the ground. Accumulated optical thickness as viewed from the sun is computed and stored in a preprocess to speed-up calculations—using interpolation—for skylight rendering for a certain viewpoint. Although Nishita et al. suggest a method to compute third and higher order scattering, since the source radiance due to higher order of scattering decreases drastically, only the second order scattering is effectively computed. This is estimated by considering only eight precise directions around any particular sampling point of the viewing ray. These correspond to the sunlight direction, the perpendicular direction to the sunlight, the horizontal direction to the Earth surface and the zenith direction. Some of these directions hit the Earth surface, and also the light scattered at those intersection points (due to direct sunlight and also to skylight) are taken into account.

Preetham et al. [126] consider the rendering of outdoor scenes including sunlight and skylight, and the effects of *aerial perspective* (de-saturation and color shift of distant objects). Two inexpensive analytic models are developed for these two objectives, whose results have been verified against standard literature from atmospheric science. The basic idea is to compute the sky spectral radiance function for a set of sun positions and turbidities (for which the method of Nishita et al. [106] was used), and then fit a parametric function.

The work by Harris and Lastra [59, 60] is concerned with the real-time rendering of high-quality static constant-shape clouds suitable for flight simulation and games. It is based on a two-pass method by Dobashi et al. [36], where clouds were modeled using particle systems, isotropic multiple scattering was approximated by a constant ambient term, and 2D-textures were used for rendering. The method by Harris and Lastra extends that method with an approximation to multiple forward scattering (only in the light direction) and anisotropic first-order scattering (in the eye direction). It allows the viewer to fly in and around clouds and to see other flying objects passing through or behind them. Since the clouds are static and of constant shape, a preprocess per cloud can be done to compute the multiple forward scattering illumination. This is approximated by only considering the most significative direction from the whole sphere of directions: the light direction—this represents a harder restriction of the space of directions than the one done by Nishita et al. [107]. The anisotropic first order scattering is calculated in run-time. The Rayleigh phase function was used in this work, although indeed it is far from being well-suited for clouds—more accurate phase functions should be used for more accurate results (see Section 2.3). Particles are rendered using splatting [185]. In order to achieve real-time rendering in scenes containing many complex clouds, dynamically generated impostors are used (see e.g. [141]).

Diffusion. Stam [158, 159, 161] solves the global illumination by progressive refinements by using shooting operations between patches, and between patches and blobs (which model the media). The shooting between blobs (that could be very expensive if the number of blobs v is large) is avoided by a set of v linear equations representing a diffusion equation. This is obtained by a *diffusion approximation* of the source radiance (due to the scattering of the medium radiance), i.e. it is characterized by only two functions: $J_{\mathrm{m}}(\omega) = J^0 + \vec{J}^1 \cdot \omega$. Solving the linear system allows the computation of the coefficients J^0 and $\vec{J}^1$ for each blob, and thus the multiple scattering between blobs. When v is not too large ($v < 1000$) the system can be solved with a direct LU-decomposition; for larger systems a relaxation scheme can be used, although the convergence is not guaranteed (but with a relatively small number of blobs good looking results are obtained). The proposed method uses far less memory and computation time than would be required by a grid method. Being a progressive method, when it deals with complex scenes composed of lots of surfaces, the cost of the progressive shooting of energy from the surface patches is quite expensive. A hierarchical approach would become necessary in such a case.

Others

Jackèl and Walter [68] focus highly realistic atmospheric rendering, allowing the modeling of real world atmospheres (composed by ozone, haze, dust, soot, sulphur acid, etc.). The combination of four sub-models, consisting of a number of concentric shells, determines the atmospheric model: clear air, aerosol, ozone and rain layer. As in the work by Nishita et al. [106], both Rayleigh and Mie scattering are accounted for. They approximate the second order scattering to obtain more accurate results, but unfortunately the authors do not explain how this approximation is performed.

3.3.2 Stochastic Methods

The global illumination stochastic methods basically trace random rays within the environment. The interaction points that limit the rays can be obtained by using a constant step distance [20, 21] or sampling a cumulative density function [114, 83].

Constant Distance Sampling

Blasi et al. [20, 21] describe methods to deal with participating media by using a simulation of the particle model of light (Monte Carlo light tracing). The first [20] deals with a single participating medium; the second [21] can render mixed scenes. Both take into account the multiple scattering within the media, using the Schlick phase function, specially defined in such a way that the importance sampling using it is quite inexpensive, while maintaining the possibility of approximating other phase functions. In [20] it is used an approximation to the Mie scattering as a combination of isotropic plus forward scattering components. In the scattering events, the scatter direction is given by optimal importance sampling of the scattering component, and the isotropic part is stored in the voxel. This isotropic part of the voxel is not considered for the illumination of the other voxels. Since the directional component is much more important than the isotropic component, it is expected that the resulting error will not be significant. A progressive refinement strategy could be used when this isotropic energy becomes too important. Bundles progress in steps of constant length δ. Therefore, at each interaction point there is a sampling process to decide if there is scattering in that point. Absorption is taken into account along the whole path of the bundle, decreasing its flux at each step by the transmittance due to absorption along distance δ.

In [21] a progressive technique is used to render mixed scenes. Surfaces are classified as "diffuse" or "specular" depending on a threshold. In the Illumination Pass, when a bundle hits a diffuse surface, its energy is stored there (and the bundle's path ends), whereas when it hits a specular one, it is reflected using importance sampling. Within the media the bundles progress as explained above, although only when a bundle exits the media (if it does) its energy is recorded (at the border voxel). Due to this storage scheme, the number of rays traveling through the volumes must be higher than it would be required with a storage per voxel, to get an accurate sampling of the energy leaving the volume.

Random Distance Sampling

Light Tracing.

The Monte Carlo light tracing by Pattanaik and Mudur [114] uses a sampling process to find the points of interaction (absorption or scattering) of the bundles within the volume, with the expression $1-\exp(-\int_0^S \kappa_t(u)\,du)$ as a cumulative distribution function, where S is the distance traveled. At those points, with the *Simple Absorption* method, another sampling process is performed to decide if the interaction is an absorption or a scattering event, based on Ω. On the other hand, with the *Absorption Suppression* method the bundles always scatter but they reduce their flux multiplying it by Ω. Different variance reduction techniques are proposed: Forced interaction of a bundle with each voxel, the yet mentioned Absorption Suppression method, and the Particle Divergence method (in which the outgoing bundle is split into many bundles at the scattering points). The storage scheme presented is suited for isotropic scattering, but can be changed to deal with anisotropic scattering.

Roysam et al. [132] focus the problem of visibility of lighted exit signs in buildings through fire-caused non-uniform smoke, proposing a parallelized Monte Carlo particle tracing method. A single step is performed in this method—just a particle tracing phase. Whenever a bundle hits the image plane (the receiver), the corresponding pixel is conveniently updated. Using a few number of bundles a coarse solution can be obtained. If the user requires a higher accuracy, the energy of the yet traced bundles is re-scaled and more bundles are shot. The scattering pattern corresponding to different smoke particles is measured experimentally and mixed into a tabulated phase function.

Bidirectional Path Tracing.

In [83] a bidirectional path tracing for non-emitting participating media is presented. Random walks are traced both from the light sources (light paths—light shooting) and from the eye point (eye paths—light gathering), being a combination of light tracing and eye tracing. Consequently this is an image-based method. After tracing a light path and an eye path, each intersection point of the respective paths are connected by shadow rays. Those shadow rays that are not occluded constitute a part of (complete) transport paths from the light sources to the eye, and an illumination contribution is computed for each transport path. These illumination contributions are combined to obtain an unbiased estimator for the radiance reaching the eye, taking into account the probability densities for generating the transport paths used. Concretely, the balance heuristic [174] is used to obtain the weights of the illumination contributions. Random walks (both light and eye rays) are traced computing interaction points within the media as in [114] (Simple Absorption case). For the scattering direction computation the Schlick phase function is used.

Dumont [39] is concerned with the design of solutions to improve road safety in foggy weather. A way to study the influence of fog effects is by means of simulations. The approach followed by Dumont is the use of a Monte Carlo algorithm for a homogeneous medium with constant phase function, extinction coefficient and albedo, being the theoretical equivalent values resulting from the combination of real data. The phase function is modeled by a goniometric distribution. In his work, he resorts to the use of an extension (to deal with participating media) of the light tracing combined with next event estimation for the direct computation of pixel radiances, based on [40]. Thus, at each reflection or scattering event, the direct contribution to the camera (or to the cameras, if more than one is used) is calculated—thus, this method represents in fact a subset of bidirectional path tracing. This direct contribution is computed taking into account the PDF of the event (to reflect/scatter in a particular direction) and the transmittance between the corresponding point and the camera.

Photon Maps. Jensen and Christensen [71] remove previous restrictions limiting the photon map method to surfaces [70] introducing a *volume photon map* containing photons in the participating media. This is a photon map different from the surface photon map because the way radiance is estimated from them is different—the authors also derive the radiance estimate expression for the volume photon map, using density estimation. The volume photon map is used only used to represent indirect illumination, that is, it only stores photons that have been reflected or transmitted by surfaces before interacting with the media, and photons that have been scattered at least once in the media. It is created in a particle tracing preprocess, like in the work by Pattanaik and Mudur [114]. To account for anisotropic scattering, the direction of incidence is stored at hit points to recover in the rendering step the source radiance toward the eye.

The same photon map method by Jensen and Christensen [71] was simultaneously suggested by Lange and Pietrek [84], although restricted to homogeneous isotropically scattering media—thus, in this case there is no need to store the incident direction in the photon map. Fedkiw et al. [44] use the method of [71] for visual simulation of smoke. In the second step, however, a forward instead of a backward ray marching algorithm is proposed, since it allows a more efficient culling of computations in smoke that is obscured by other smoke. Also a more efficient use of the photon map results by allowing to use less photons in the query as the ray gets deeper into the media.

Adabala and Manohar [7] use particle systems to model gaseous volumes, allowing the treatment of inhomogeneous anisotropically scattering media. By displacing the particles following fluid dynamics equations, the model can evolve in time, avoiding the use of grids. In order to manage the particles and their associated illumination field, a median kd-tree is used, dubbed by the authors *particle map*, storing information to be retrieved by associative searches. A particle tracing is used to distribute light within the media, in a way that is similar to the photon map approach. However, when a scattering event takes place, instead of saving an entry in a photon map, the related information is stored in the particle map. The incident energy is distributed among particles near the point of interaction. Since the storage of the incident direction replicated onto those nearby particles would be prohibitive, only the corresponding energy that would be scattered into the viewing direction is stored, sacrificing the viewpoint independency of photon maps. The integral transport equation and the source radiances are solved using the particle maps by using nearest neighbor queries.

Metropolis Light Transport. Pauly et al. [116] extend Metropolis Light Transport [175], which is based on the Metropolis sampling technique for handling difficult sampling problems in computational physics [98], to incorporate volumetric scattering. The resulting algorithm is unbiased, handles general geometric models and inhomogeneous media, accounts for multiple scattering, and uses little storage—at the expense of being a view dependent method. In order to render an image, in Metropolis Light Transport a sequence of light transport paths is generated by randomly mutating single current paths (e.g. by adding a new vertex to the path). In the Metropolis technique the *path space* (set of all finite-length paths with an associated *path space measure*) is explored locally, favoring mutations that make small changes to the current path, focusing on the light paths that contribute most to the rendered image. Each path deposits a certain amount of energy to the pixel it passes through, updating the image. Therefore, the paths are distributed proportionally to their contribution to the final image. An initialization step determining the total image brightness is performed by means of a bidirectional path tracing execution.

3.3.3 Discussion

Applications

As mentioned above, participating media functions (e.g. $\kappa_a(x)$, $\Omega(x)$, $p(\vec{\omega}_o, \vec{\omega}_i)$) can be extended for clusters. The computational cost of solving the global illumination problem in complex scenes can be dramatically reduced by the use of clusters, which represent both the transmission properties and the energy exchanges of their contained elements as a whole [148, 154]. Thus, a suitable *extended* phase function for clusters could be used, for example, in a directional clustered hierarchical radiosity algorithm (see [30]).

It is clear that applications for safety analyses will need the best possible solution at any cost, starting from a sufficiently precise input model (for these, the bidirectional path tracing could be used [83]), while for entertainment, for example, a visually pleasant image will be enough no matters if it is physically accurate (perhaps in this case isotropic media can be used, or the diffusion approximation [158, 159, 161]). Training systems must perform real-time rendering of images, so the computation of images must be very fast, possibly reducing computation time by using visibly acceptable approximations and not too complex (dynamic) models. This invalidates the use of Monte Carlo based methods since they are highly time consuming. Extensions of the existing methods should have to be studied for dealing with dynamic environments taking coherence into account, to achieve the necessary speed. Also the conditions of the particular problem to solve must be considered, like the complexity of the scene (in number of elements and their optical properties), types of image required, etc. For example, for a sequence of a foggy driving simulation, the viewpoint is at a fixed distance from the ground and should follow a restricted path (must be over the road), on the other hand the *importance* of the elements of the scene must vary according to their position (relative to the viewpoint) in each snapshot (frame).

Progressive Results

The multi-gridding technique can be used to compute a sequence of solutions stopping when a sufficiently accurate solution is obtained. The sequence starts with a very rapid computation of a first coarse solution that is improved in successive steps. This can be accomplished in hierarchical approaches like in [147]. If a relatively small quantity of time is given to compute a solution, then with the multi-gridding technique a coarse solution could be obtained. The Monte Carlo light tracing method of Pattanaik and Mudur [114], on the other hand, in a given short time will produce in the Illumination Pass a partial solution far from converged in the illumination elements, so in practice the image related to that partial results will not be of utility. This is due to the fact that each bundle follows its path in the scene until it dies. In the light tracing method by Blasi et al. [21], however, the reflection on diffuse surfaces is eliminated, so that whenever a bundle hits a diffuse surface its path ends, and that diffuse surface accumulates unshot energy. At each iteration a set of bundles representing the unshot energy of the element having the highest value is spread to the environment, thus being a progressive refinement algorithm. Note that this technique also introduces bias since the process of reflection at a given point is substituted by a shooting from a random point within the element. The progressive nature of [21] allows the computation of an iterative sequence of images. However, the illumination of the media will be far from converged unless a very high number of bundles have been used. In the bidirectional ray tracing, since it is a view dependent method where the illumination is solved directly per pixel, the quality of the image can be gradually improved, starting from a very crude approximate image and converging to the solution as the program progresses.

Partial results of the illumination of a volume treated by the methods which use discrete ordinates by sweeping of energy [86, 96] could be given between successive iterations. Methods that do progressive refinements using hierarchies could do a good job if an ambient term is used for display purposes (as a generalization of the ambient term of the classical radiosity) after the shoot of light of the most energetic elements. This can also be used by other progressive refinements methods like the diffusion approximation [158, 159, 161], and the method by Blasi et al. [21]..

Sampling Strategies

It should be noted that the sampling strategy of [21] drives to biased results, while that of [114] does not. This is because the bundles can only be scattered at distances which are multiple of δ, and thus the expected length before scattering will not be equal to the mean free path without absorption. The error is reduced as long as the value of δ is diminished. Unfortunately to assure results with a variance below some threshold the time required is approximately of the order of the inverse of δ. Moreover, we have checked that, for a same variance threshold and setting a value of δ relatively small to get a tolerable bias, the computation time using the sampling procedure of [114] is always lower than that of [21].

Isotropic Media

It seems clear that for applications in which the isotropic assumption can be used (i.e. non-realistic applications), the obvious choice is the hierarchical radiosity method [147] since it has the best performance in computation time; moreover it is more reliable than the progressive refinements methods, in which a fixed hierarchy is used. Also the diffusion approximation using blobs method [158, 159, 161] could be simplified for the isotropic scattering case, being then a good choice (less memory and computation time than grid based methods) when the number of blobs is relatively small. When that number of blobs is high, then a hierarchical approach becomes necessary.

Anisotropic Media

In the case of anisotropic media, all the existing methods commit errors; only the bidirectional path tracing, the photon maps approaches and Metropolis Light Transport are unbiased. It is then important to know what type of error can be accepted for each concrete application.

In the case of soft indirect illumination, the Metropolis method tends to use the same amount of computation time as brute force bidirectional path tracing. As Jensen and Christensen noticed [71], within participating media the illumination is mostly soft because of the continuous scattering taking place everywhere in the medium. Therefore the benefits of Metropolis with respect to bidirectional path tracing can be marginal, depending on the concrete lighting conditions.

Stam [158, 159, 161] utilizes a *diffusion approximation* (Section 3.3.1). This is only valid in the case of a high number of scattering events. This condition fails at the boundaries of the media, and thus the results are not so precise there; however, for certain applications (such as animations in which the objective is that things “look right”) they are accurate enough. Spherical Harmonics and Discrete Ordinates methods approximate directional functions by using a fixed set of bases. It could be interesting to use an adaptive number of bases in function of the accuracy required for the solution. Max [96] propagates the energy inside the medium in a way so that the “ray effect” (present in [113, 86]) is reduced, but computing an approximation of the true attenuation between two voxels; therefore although it produces visually better images it is not clear if that solution is less accurate than the direct Discrete Ordinates method (although for displaying clouds, for example, it is considered to be better). Obviously the bias of the Spherical Harmonics and Discrete Ordinates methods can be reduced by using a higher number of bases, but the computation time augments dramatically doing so. Thus there is a compromise between computation time and image quality.

Comparing the costs of the zonal Spherical Harmonics method [18] (form factor computation: $O(n^7 + M^2n^6)$) against Discrete Ordinates methods (iteration cost: $O(Mv)$ in [86], $O(Mv\log n + M^2v)$ in [96]—15 iterations are used approximately; form factor computation for the direct extension of the zonal method with discrete ordinates: $O(v^2n + v^2M)$), with $v \gg M$, we can sort the different methods starting from the cheaper as follows: Languénou et al.'s discrete ordinates [86], Max's discrete ordinates [96], direct extension of the zonal method with discrete ordinates, and the zonal spherical harmonics by Bhate and Tokuta [18]. However it should be noticed that the accuracy of the results follows the reverse order.

The photon map approach introduced by Jensen and Christensen [71] is a simple and efficient method able to deal with complex geometries and lighting conditions. However, it requires high amount of memory for difficult lighting situations, and suffers from various artifacts, such as blurred shadows and caustic borders.

Chapter 4

First Pass: Computation of a Coarse Solution

This chapter describes two methods suitable for a first pass. The basic requirements of a first pass algorithm are two: on one hand, it has to compute a view independent solution of the global illumination problem of scenes possibly including participating media, and on the other hand, this has to be performed in a relatively short computation time. From the set of strategies available in the literature, reviewed in Chapter 3, hierarchical methods are very attractive in order to keep memory and computation time as low as possible while obtaining a coarse solution. This is the reason why our two approaches are based on hierarchical strategies.

The Radiosity Method and Hierarchical Radiosity

In what follows we will briefly review the radiosity and hierarchical methods. Excellent books on radiosity exist to which the reader is referred for comprehensive coverage on this topic [33, 150].

The basic radiosity method was introduced by Goral et al. in the eighties [52]. The radiosity method assumes that the scene is made of diffuse surfaces only, so that the global illumination equation (Equation 2.1) can be greatly simplified. Substituting the general BRDF $\rho_{\mathrm{bd}}(x,\vec{\omega}_{\mathrm{o}},\vec{\omega}_{\mathrm{i}})$ by the concrete diffuse BRDF $\rho_{\mathrm{d}}(x)/\pi$—which is independent of directions—allows moving the reflectance term out of the integral of Equation 2.1. Defining radiosity as the unknown of the global illumination problem and meshing the scene assuming constant radiosity per patch, a set of linear equations result, with *form factors* coupling pairs of patches. A notable improvement on the resolution of this set of equations was the progressive refinement strategy, providing faster feedback [32].

A major breakthrough from the basic radiosity method was the introduction of the hierarchical radiosity method by Hanrahan et al. [58], who recognized the similarities between the radiosity problem and the N-body problem. *Links* are built on-the-fly while establishing relationships between nodes of quadtrees associated with the surfaces, trying to avoid connections whose related energy transfers are deemed insignificant. For commonly used scenes, this implies a reduction in the number of entries of the matrix associated with the initial problem and thus a reduction of the computational cost.

Object Hierarchies: Clustering

A further improvement to the radiosity algorithm is the incorporation of clustering, where disjoint objects (patches and/or clusters) are grouped into single entities that can interact with other entities [154, 30, 148, 147, 48]. Clustering represents a higher reduction on the computation cost. This is the reason why we have chosen this kind of algorithms as the basis of our work, extending them to be more generic and usable in our first stage of the two-pass algorithms.

As stated in the beginning of this chapter, two concrete algorithms based on hierarchies have been selected to be further improved to be usable in a broader scope: Hierarchical Radiosity with Clustering, and Hierarchical Monte Carlo Radiosity—the corresponding proposed algorithms are discussed in Section 4.2 and Section 4.3 respectively. After presenting previous work, we describe these two methods to be used as the first pass of two-pass algorithms, capable of computing solutions to the global illumination in general environments (diffuse or glossy surfaces, anisotropically scattering participating media). This is accomplished by two-pass methods that combine the strengths of hierarchical algorithms and Monte Carlo methods, as will be presented. Monte Carlo global illumination algorithms allow computation of unbiased solutions, and for general scenes, as long as they can be represented accurately mathematically. Unfortunately, naive Monte Carlo algorithms are slow. In order to accelerate the Monte Carlo computation, importance sampling can be used, by setting probability density functions (PDFs) that resemble the integrand of the integral equation to solve, as will be seen in Chapter 5. Thus, we propose to use hierarchical algorithms as a first pass methods that allow setting PDFs which will be used to accelerate a Monte Carlo-based second pass. For convenience (to ease the comparison between by the first and second pass—results obtained and resources used), tests performed by the proposed algorithms will be given in Chapter 5 in the context of the different algorithms presented there.

4.1 Previous Work

As reviewed in Chapter 3, deterministic methods that deal with scenes including anisotropically scattering media have the advantage of being fast, but they consume high amounts of memory and give biased results. On the other hand, stochastic methods consume less memory and are capable of obtaining unbiased results, but they take large amounts of computation time to converge. Our methods aim at obtaining the best of the two approaches. A first deterministic step is used to obtain a quick rough solution. A very detailed solution would be prohibitive due to memory requirements. Then this result is used by a second step, a stochastic one, to accelerate its convergence.

Apart from the literature reviewed in Chapter 3, other publications are related to our work, concretely focusing multiple pass methods. These works are briefly reviewed in this section.

Chen et al. introduced a multi-pass method combining finite element and Monte Carlo steps, with no restrictions on the surfaces' optical properties [28]. Their algorithm is based on a classification of light paths and the assignment of three techniques (progressive refinement radiosity, light and Monte Carlo path tracing) to those paths. The progressive refinement radiosity step provides an approximation of the overall illumination, and Monte Carlo path tracing refines the image for high frequencies (shadows, caustics) and low frequencies (color bleeding). The last pass, "low frequency refinement", uses the results of the radiosity pass for higher order reflections, by ending the paths of the Monte Carlo path tracing step on the second diffuse-like surface they interact with (and getting the radiosity of this surface from the results of the radiosity step). Chen et al.'s algorithm, by finishing diffuse paths at the second bounce from the eye using the solution of the first pass, increases the bias of the solution. Our method of Section 5.2, however, does not use the results of the first (finite element) step to finish the paths of the Monte Carlo step, but to drive the sampling of the reflected and scattered directions of the paths, thus introducing no bias. However, we can also use the finite element solution if bias is acceptable, making the method even faster (like in [70]—see Section 5.3).

Suykens and Willems presented weighted multi-pass methods that allow compositing overlapping transport between different methods [166]. This is different from the approach taken here, in which results from a first finite element step are used (but not combined) by a second Monte Carlo step.

4.2 Hierarchical Radiosity with Clustering

4.2.1 Overview

Our first concrete first pass algorithm is based on two finite elements algorithms: a unified algorithm for the simulation of light transfer between diffuse surfaces, *isotropic* participating media and object clusters [147], and a hierarchical algorithm capable of dealing with non-diffuse surfaces [148]. In our approach, the global illumination of scenes including inhomogeneous *anisotropically* scattering media can be solved efficiently by means of the clustering strategy [120, 121]. From the reflection equation (Equation 2.2) and the scattering equation (integral term in Equation 2.6) we have identified the expressions needed to transport *radiant intensity* between all kinds of objects (surfaces, media and clusters).

4.2.2 Radiance Clustering Algorithm

Our first first-pass method is a finite element method, concretely a *Radiance Clustering* algorithm that also takes into account inhomogeneous anisotropically scattering media. This first pass is based on [147], where a unified algorithm was proposed for the simulation of light transfer between diffuse surfaces, *isotropic* participating media and object clusters, and on [148], a hierarchical algorithm capable of dealing with non-diffuse surfaces. Surfaces and isotropic participating media are collected into clusters in a hierarchical subdivision of the 3D space. Each cluster, surface or participating medium element has associated a number of directional distributions that represent their radiant properties. Energy exchanges between all kinds of elements are treated in a uniform way. The extension proposed here takes into account the possible anisotropy of participating media, by considering the phase function in the scattering at the leaves (previous hierarchical methods used a constant phase function). To our knowledge, ours is the first finite element hierarchical algorithm that deals with inhomogeneous *anisotropically* scattering media. From the reflection and the scattering equations we have identified the expressions needed to transport radiant intensity between all kinds of objects (surfaces, media and clusters). Currently we deal with diffuse surfaces in this finite element step, but it is easily extensible to deal with glossy surfaces, since we have the machinery to deal with directional radiance distributions (see Appendix A).

Light Transport Equations

In the following we derive the expressions for the reflected and scattered radiances due to the different kinds of interactions: surface to surface, surface to volume, volume to surface and volume to volume. We denote radiant intensity by I, and its related extended form factor by F'; $\mathcal{S}$ and $\mathcal{R}$ denote the sender and receiver (they can be surfaces or clusters—see Figure 4.1); x and y are points in $\mathcal{S}$ and $\mathcal{R}$ respectively. As in [148], we redefine the notion of "form factor" from purely algorithmic considerations. The extended form factor F' associated to each link is simply the scalar quantity by which I (from the emitter) must be multiplied to obtain the irradiance. (Definitions for the different types of interactions are given in the expressions below, sharing nomenclature from Chapter 2.) Directions are denoted by vectors like $\vec{\omega}$; the differential solid angle related to $\vec{\omega}$ is represented by $d\sigma_{\vec{\omega}}$. For surfaces, the BRDF is denoted by ρ_{bd}. For participating media, the albedo is represented by Ω, the extinction coefficient by κ_{t}, and the phase function by p. Differential of area and volume at a point x is denoted by dA_x and dV_x respectively. $\tau(x,y)$ expresses the transmittance between points x (in sender $\mathcal{S}$) and y (in receiver $\mathcal{R}$).

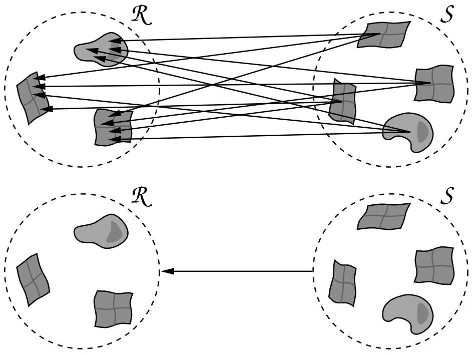


FIGURE 4.1 Accurate and approximate light transport between two clusters. Redrawn from [30, p. 97, Fig. 6.1], with the addition of participating media.

Surface→Surface Interaction.

We can identify I and F' in the reflectance equation for surfaces (Equation 2.4, substituting the visibility function v by the transmittance τ to also account for participating media):

$$L_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) = \int_{x\in\mathcal{S}} \rho_{\text{bd}}(y,\vec{\omega}_{xy},\vec{\omega})\,\underbrace{\underbrace{\tau(x,y)\frac{\cos\theta_y}{\|x-y\|^2}}_{F'}\underbrace{\cos\theta_x\,L(x,\vec{\omega}_{xy})\,dA_x}_{dI(x,\vec{\omega}_{xy})}}_{H(y,\vec{\omega}_{xy})=F'I=\cos\theta_y\,L_{\text{i}}(y,\vec{\omega}_{xy})\,\left[\frac{\text{W}}{\text{m}^2}\right]}\,.$$

Surface→Volume Interaction.

The expression for the scattered source radiance J at a point y in $\mathcal{R}$ due to radiance coming from surfaces follows (Equation 2.6, changing the domain of integration to integrate over surfaces):

$$J_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) = \int_{x\in\mathcal{S}} \frac{\Omega(y)}{4\pi}\,p(y,\vec{\omega},\vec{\omega}_{xy})\,\underbrace{\underbrace{\tau(x,y)\frac{1}{\|x-y\|^2}}_{F'}\underbrace{\cos\theta_x\,L(x,\vec{\omega}_{xy})\,dA_x}_{dI(x,\vec{\omega}_{xy})}}_{H(y,\vec{\omega}_{xy})=F'I\,\left[\frac{\text{W}}{\text{m}^2}\right]}\,.$$

Volume→Surface Interaction.

We start from the reflectance equation (Equation 2.2):

$$L_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) = \int_{\Omega} \rho_{\text{bd}}(y,\vec{\omega}_{\text{i}},\vec{\omega})\,L_{\text{i}}(y,\vec{\omega}_{\text{i}})\cos\theta_y\,d\sigma_{\vec{\omega}_{\text{i}}}\,. \tag{4.1}$$

The radiance $L_{\text{i}}(y,\vec{\omega}_{\text{i}})$ can be substituted by the integral transport equation restricted to volumes (Equation 2.7):

$$L(y) = \underbrace{\tau(y_0,y)\,L(y_0)}_{=0} + \int_{y_0}^{y} \tau(u,y)\,\kappa_{\text{t}}(u)\,J(u)\,du\,. \tag{4.2}$$

Now, in the reflectance equation, instead of integrating across directions we integrate across spatial positions and merge Equations 4.1 with 4.2:

$$\begin{aligned} L_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) &= \int_{\Omega} \rho_{\mathrm{bd}}(y,\vec{\omega}_{\mathrm{i}},\vec{\omega}) \int_{y_0}^{y} \tau(u,y)\,\kappa_{\mathrm{t}}(u)\,J(u,\vec{\omega}_{\mathrm{i}})\;du\,\cos\theta_y\;d\sigma_{\vec{\omega}_{\mathrm{i}}} \\ & \qquad \left\{ \begin{array}{l} dV^* = dS^*\,d\sigma_{\vec{\omega}}\,(S-S^*)^2 \quad \text{[146, p. 722]} \\ d\sigma_{\vec{\omega}} = \frac{dV^*}{dS^*\,(S-S^*)^2} \end{array} \right\} \\ &= \int_{x\in\mathcal{S}} \rho_{\mathrm{bd}}(y,\vec{\omega}_{xy},\vec{\omega})\,\tau(x,y)\,\kappa_{\mathrm{t}}(x)\,J(x,\vec{\omega}_{xy})\cos\theta_y\;\frac{dV_x}{\|x-y\|^2}\;. \end{aligned}$$

Rearranging terms:

$$L_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) = \int_{x\in\mathcal{S}} \rho_{\mathrm{bd}}(y,\vec{\omega}_{xy},\vec{\omega})\,\underbrace{\tau(x,y)\frac{\cos\theta_y}{\|x-y\|^2}}_{F'}\,\underbrace{\kappa_{\mathrm{t}}(x)\,J(x,\vec{\omega}_{xy})\,dV_x}_{dI(x,\vec{\omega}_{xy})}\;.$$

Volume→Volume Interaction. Again starting from the source radiance definition (Equation 2.6), changing the domain of integration to integrate over volumes, and operating as in the previous kind of interaction, we obtain:

$$J_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) = \int_{x\in\mathcal{S}} \frac{\Omega(y)}{4\pi}\,p(y,\vec{\omega},\vec{\omega}_{xy})\,\underbrace{\tau(x,y)\frac{1}{\|x-y\|^2}}_{F'}\,\underbrace{\kappa_{\mathrm{t}}(x)\,J(x,\vec{\omega}_{xy})\,dV_x}_{dI(x,\vec{\omega}_{xy})}\;.$$

The previous expressions are modified by using some approximations so that the light transport can be split in three parts, as in [30]:

1. Light of leaf surfaces and volumes are used to approximate light distributions of parent clusters (computation of $I(x,\vec{\omega})$ of an element by addition of that of its children).

2. Transport between clusters (multiplication by the F' term).

3. Light reaching clusters is immediately pushed down, as in [147, 148], so that at the leaves we compute the reflected/scattered light at surfaces/media. Alternatively the incoming light could be recorded at the cluster level, as in [30], and the push-pull stage would take into account the directionality of the incoming light of the parent cluster.

As stated in the beginning of this section, the current implementation of this step deals with diffuse surfaces, although we might well extend it for glossy surfaces by using our code for directional distributions.

Figure 4.2 shows the results of the application of the algorithm presented, for a box scene with additional occluding surfaces and containing an inhomogeneous medium for different phase functions. Different views have been set for the generation of these images based on the anisotropy of the phase functions used.

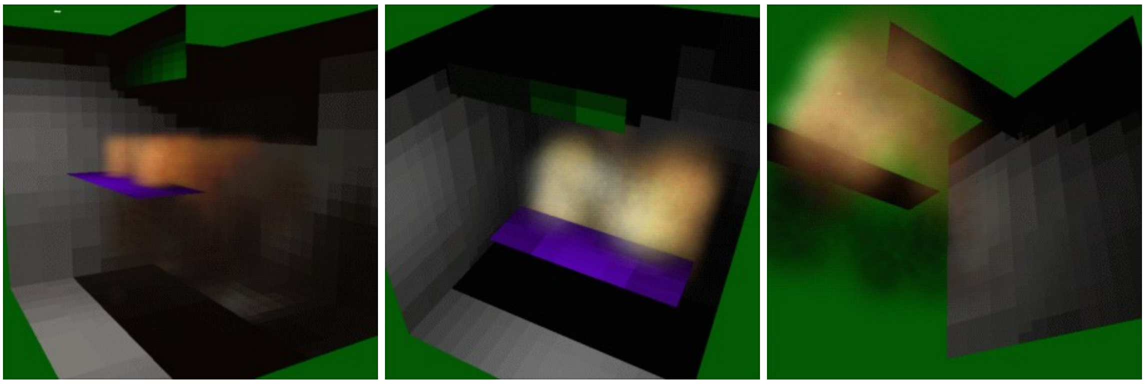


FIGURE 4.2 Scene including an inhomogeneous participating medium, for different phase functions. Left: Using the isotropic phase function. Middle: Using the Schlick phase function (see Section 2.3.4) with anisotropy parameter $g = -0.8$ (backward scattering). Right: Using the Schlick phase function with $g = 0.8$ (forward scattering).

4.3 Hierarchical Monte Carlo Radiosity

Since finite element methods like the presented in Section 4.2 are usually very complex and often have robustness problems, we have developed a second first-pass algorithm that computes a rough global illumination solution based on a particle tracing method: the Hierarchical Monte Carlo Radiosity (HMCR) method. The method is very simple and allows to quickly compute solutions that can be viewed interactively by means of 3D textures for the isotropic and diffuse components of the illumination. For this method we have developed a new second rendering step, described in Section 5.3, assisted by a final gathering procedure, that allows the computation of high quality images using the HMCR solution. Currently our implementation treats the interaction of light between isotropic participating media and diffuse or specular (or a combination of both) surfaces.

4.3.1 Overview

The Hierarchical Monte Carlo Radiosity algorithm is an extension of discrete Monte Carlo Radiosity that incorporates hierarchical refinement and clustering [14, 12]. We have generalized this algorithm to account for surfaces with a combination of diffuse plus specular components, and for participating media [117, 118]. The surfaces' diffuse component is handled as in HMCR, whilst the specular component is handled by following the path of particles through bounces until they eventually hit a diffuse surface (with no specular component), interact with a medium, or quit the scene. Particles interact randomly within participating media through a sampling process, according to the transmittance function that is based on the media's extinction coefficient. We use the *Absorption Suppression* strategy by Pattanaik and Mudur [114] at the interaction points. The meshes related to surfaces and volumes are adaptively refined, together with the implicit link structure. In our context, though, we *can* store the links explicitly to construct the Link Probabilities for the second pass.

4.3.2 Extending HMCR

HMCR

Bekaert et al. [12, 14] introduced HMCR, that is an extension of discrete Monte Carlo Radiosity (MCR) that incorporates hierarchical refinement and clustering. MCR algorithms generate lines (globally or locally), that connect mutually visible points x and y. Each of these lines represents a sample contributing to the form factor between the patch containing x and the patch containing y. This contribution to the form factor is immediately translated into an energy transfer between the patches. In HMCR, for each sample line, a decision is made to select the level of the element hierarchies at x and y is appropriate for energy transfer instead of simply taking the patches containing points x and y (Figure 4.3).

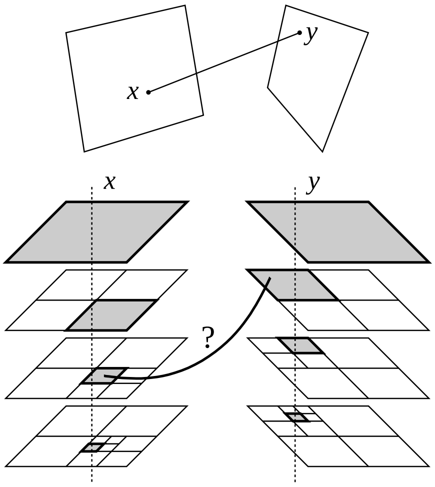


FIGURE 4.3 HMCR: for each sample line connecting two points x and y, the level of the element hierarchies at x and y is determined as being appropriate for computing the light transport from x to y. Redrawn from [14].

The essential difference between Hierarchical Radiosity (HR) [58] and HMCR is that at each refinement step, in HR *all* the sub-elements containing x and y are recursively refined, whereas in HMCR only the sub-elements containing x and y are considered for further refinement. The same refinement criteria and strategies as in HR can be used to check whether a candidate interaction between elements containing x and y is admissible, and, if not admissible, how it should be refined. The HMCR adaptively refined mesh and (implicit) link structure is very similar to those in HR, but are lazily constructed as needed during form factor sampling rather than beforehand as in HR.

As in [14], our concrete implementation of HMCR uses a breadth-first approach, so that the push-pull procedure only considers the new deposited radiosity and only the elements in the hierarchy that contain or are part of the receiver element need to be updated. (In the depth-first approach, the push-pull procedure should be called after *each* sample since the next sample will carry on the transported power, possibly starting from another element level. This implies that this approach would have an overhead.) Also we use a local line generation technique, and thus it contains a loop over the leaf elements of the element hierarchy since the most detailed representation of radiosity is available there.

HMCR for Participating Media and Specular Surfaces

Participating Media.

In the case of participating media the HMCR is modified so that the local lines can interact not only at surfaces but also in the volumes representing the media. As in the Monte Carlo light tracing by Pattanaik and Mudur [114], we use a sampling process to find the points of interaction (absorption or scattering) of the particles within the volume, with the expression $1 - \exp(-\int_0^S \kappa_t(u)du)$ as a cumulative distribution function, where κ_t is the extinction coefficient of the medium and S is the distance traveled. At those points, Pattanaik and Mudur define the *Simple Absorption* method, by which another sampling process is performed to decide if the interaction is an absorption or a scattering event, based on the albedo Ω at that point. Following this schema the local line would continue traversing the media until the first scattering event takes place. On the other hand, with the *Absorption Suppression* method the particles always scatter (there is no sampling process) but they reduce their flux multiplying it by Ω. We follow this schema, so the local line ends at the first interaction point, updating the unsent power of the receiving volume element at the level specified by the refiner. As in the case of the HMCR for surfaces, a refinement criterion is set to decide at which level of the hierarchies the transport of energy is appropriate (Figure 4.4).

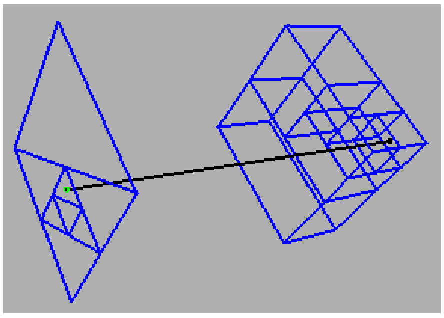


FIGURE 4.4 A line connecting two points for an interaction between a surface and a volume (participating medium). As in the case of surfaces, the algorithm determines which level of the two related hierarchies is appropriate to compute the light transport between the two points.

Currently we deal with isotropically scattering media, therefore when a leaf volume element is selected to shoot its unsent power, the local lines follow a uniform distribution in the space of directions. In the case of anisotropically scattering media, the phase function should be taken into account, together with the incoming direction (Illumination Samples would be effectively used under this circumstance [164]—see Section 7.3.1).

The push-pull procedure for volume elements in the HMCR method is like the push-pull of *Radiance Clustering*, where the incoming power is weighted in the elements of the hierarchy by their area factor $4\kappa_t V$ (with V being the volume of the element) [148].

Surfaces with Specular Component.

The integration of surfaces with specular component in their BRDF in the HMCR algorithm is very easy. When a particle hits one of these surfaces, apart from the refinement and transfer of energy taking place for the diffuse component, it is also bounced following the direction of the perfect reflection/refraction taking into account the normal at the hit point. The power of the particle is weighted in the bounce by the spectral "color" of the specular component. (In the case of a "white mirror", the spectral color would be (1, 1, 1) in RGB space, so the weight for the three channels is 1, and the particle would not loose energy. A "green mirror" would absorb the red and blue components, and only the green component of the particle's power would remain for the bounced particle.) The particle keeps bouncing as long as it hits surfaces with specular component. Notice that the refinement procedure has to be adapted to take into account the whole path, not simply the geometry between two points connected by a segment. Figure 4.5 shows an example of HMCR with surfaces with specular component. As another example, we have built a room scene (Figure 4.6), a kind of simplified replication of one of the scenes of E. Veach's dissertation [173], containing both direct (lamp on the right) and indirect illumination (lamp on the left, pointing upwards). The wall on the right has both (gray) diffuse and specular components. The direct illumination can be noticed on this wall as a gray contribution to the illumination imposed over the mirrored illumination. This will be more easily seen with the execution of the second pass (Figure 5.20). The first pass took 25.26 seconds to compute the solution (using a PC Linux SuSe with a PIII 550MHz processor), shown on the right of Figure 4.6.

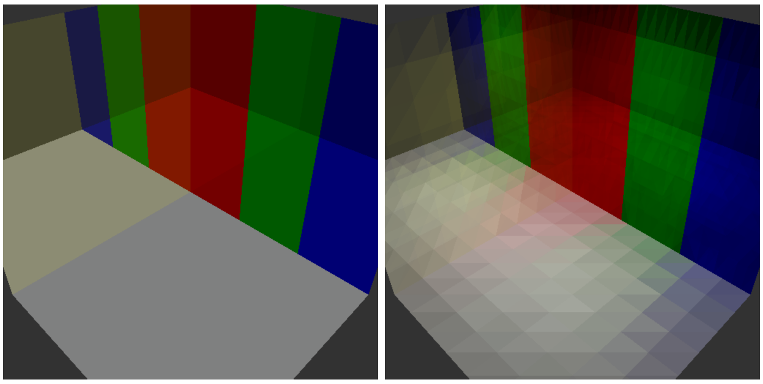


FIGURE 4.5 Left: Simple scene with a light source (not viewed because of culling), a diffuse floor and four walls with specular component (yellow—90% of specularity; red, green and blue—50% of specularity). Right: Result of HMCR (without Gouraud shading). Notice that there is color bleeding on the floor's edges due to the colored surfaces.

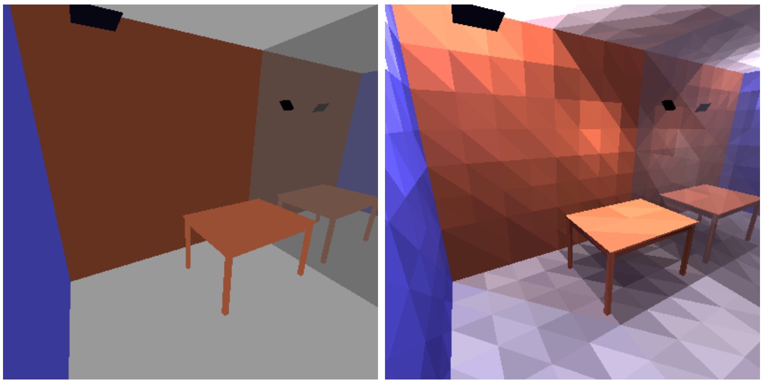


FIGURE 4.6 The room scene. Left: The input scene. Right: The result of the HMCR step (flat-shaded).

4.3.3 Interactive Viewing

To display the results of the first pass we use the OpenGL API [187], taking into account the diffuse/isotropic component of the illumination. After the HMCR step, to visualize isotropic participating media we create 3D textures from the source radiances stored at the leaves of the volume hierarchies. (See [35] for a comprehensive survey of 3D texturing.) A 3D texture is set per medium. The resolution of each texture is determined by the maximum level of subdivision reached. Since the volume can be subdivided more finely in some zones than in others, care must be taken to supply the 3D texture with the data of the related leaves replicated to reach the finest resolution. Since currently we deal with opaque surfaces, to render the results of the HMCR step we first display the set of 3D textures, followed by the surfaces. To simulate the light composition across a medium, the blending function must be set with the call glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA). We apply two possible minimization/magnification filters to the textures (Figure 4.7): GL_NEAREST (that shows the results HMCR computed straightforward) and GL_LINEAR (in which linear interpolation is used, being a 3D counterpart of Gouraud shading).

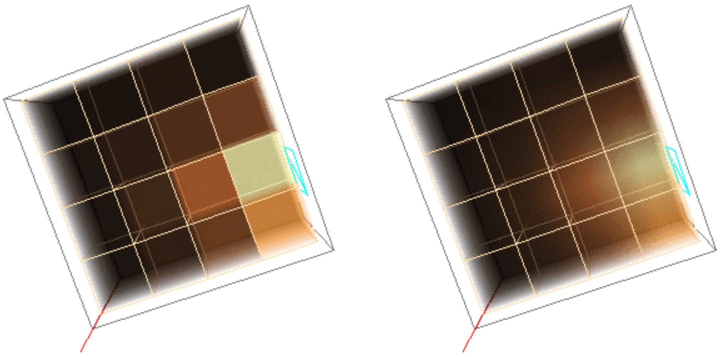


FIGURE 4.7 A light source illuminates a participating medium. On the left, the results are shown directly; interpolation is used on the right.

Cards supporting 3D textures by hardware allow interactive viewing of the scene; it is still possible to render by software the 3D textures, but then the interactivity is lost (the speed decreasing as the number of semi-transparent planes cutting the volumes is increased.) An example with two participating media is shown in Figure 4.8.

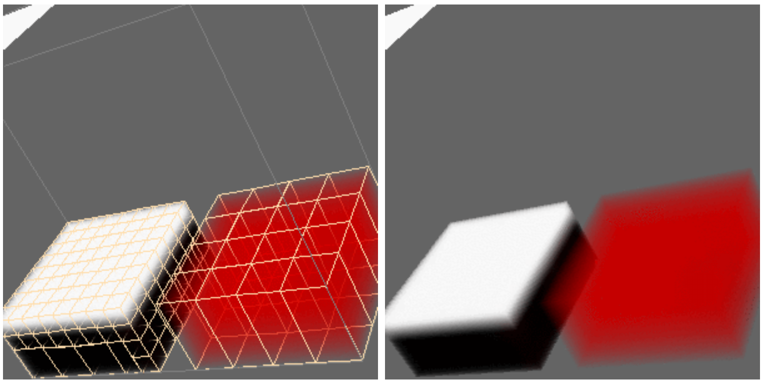


FIGURE 4.8 Two participating media visualized after the HMCR step. Left: The subdivision—mesh—of the volumes is shown. The medium on the left has a very high extinction coefficient, so most of the energy reaches only the border of its extent. Right: The volume subdivision is not shown. Notice the combination of energies in the area were both volumes visually overlap.

Chapter 5

Second Pass: Link Probabilities

The second pass is responsible of obtaining a high quality image from the coarse global illumination resulting from the first pass. In order to accomplish this, we have based our work on two view-dependent methods: a Monte Carlo path tracing algorithm using next event estimation [120, 121] and a ray tracer incorporating final gather [117, 118], both taking advantage of importance sampling. Importance sampling is usually performed based on the BRDF term for surfaces [80], and on the phase function for participating media [20], but not taking into account the incoming radiance. During the second pass, light impinging a certain point can be estimated using the results of the first pass. This permits the creation of discrete probability density functions (PDFs) at intersection points to perform importance sampling also accounting for the incoming radiance.

Our very first design of these PDFs (Section 5.2.1) used a constant basis representation using bins for the directional domain. However, their intrinsic nature—using a fixed number of immutable bins, as we will see—has a shortcoming: the coefficient of each single bin represents the *mixed* illumination arriving from the set of directions associated to the bin. This can drive to low efficiency cases. A better strategy is to construct the PDFs directly over the link space. This can be done for example using the set of links previously established in the first pass [121, 117]. Improved results are obtained by setting appropriate links for each particular intersection point, at the cost of a higher rendering time [118]. In any case, we propose the use of adaptive PDFs, in which the PDFs of the links are continuously improved (instead of remaining fixed), to better match the distribution of the irradiance at a certain point. As a consequence, the related estimations are improved by reducing their variance.

5.1 Previous Work

As will be seen afterwards, we will use importance sampling to accelerate convergence of Monte Carlo integration, by means of *probability density functions* (*PDF*s) that resemble the integrand. In the case of the reflectance equation, the integrand is the product of the BRDF (for an opaque surface), the incoming radiance and a cosine term. Most algorithms perform importance sampling based simply on the BRDF and the cosine, but there are methods that do take into account the incoming radiance. This can be done e.g. with a particle tracing first pass [69, 172], or directly by adaptively improving the PDFs [41, 82, 115]. In the case of the scattering equation, the kernel is constituted by the phase function and the incoming radiance. Monte Carlo methods perform importance sampling based on the phase function [114, 20, 83].

In the context of final gathering for radiosity, Ureña and Torres [172] construct adequate PDFs after gathering information during the computation of a low resolution radiosity solution. This information consists of a two-dimensional array whose entries couple pairs of patches, and accumulate the energy transferred between them. Thus, the method proposed is well suited for progressive radiosity and Monte Carlo particle tracing. A discrete PDF is established for each patch based on the built array. Next, to estimate the irradiance at a certain point, this PDF is used to select a patch to sample (sender). Finally, a second PDF based on the solid angle subtended by the patch is sampled. Notice that this scheme becomes prohibitive for scenes containing a high number of initial patches, due to the storage requirements of the array (quadratic on the number of patches).

Sturzlinger [165] proposes the use of the links established by a hierarchical radiosity first step to construct a PDF per patch similar to the one obtained by Ureña and Torres. After using this PDF to select a patch to sample, the visibility from that patch to the gathering point is computed to obtain the estimate of the irradiance due to the patch. If shaft culling was used during the hierarchical radiosity step, then links can be tagged with a boolean indicating if they connect unoccluded patches. Computational savings can be achieved by avoiding visibility recomputation for links with predetermined full visibility. Similar approaches are presented by Bekaert et al. [13], but using classical radiosity.

Szirmay-Kalos et al. [167] use a particle shooting step in a scene previously discretized to obtain an estimation of the radiance function, as Ureña and Torres do. Instead of storing a two-dimensional table, though, a set of links is constructed incrementally during the shooting step. These links simply connect mutually visible plain patches, without any kind of hierarchical structure. This fact renders the method impractical for scenes with a high quantity of patches. A second step is used for high quality rendering—a path tracing that uses importance sampling based on the link set. At each bounce the domain of links onto the domain of directions, using an approximation, to finally establish a PDF on the resulting domain of directions. The cost of this projection and PDF construction can lead to high computation times since it is performed at each interaction point of the random walk.

5.2 Monte Carlo Path Tracing

Our first second-pass is based on a view dependent Monte Carlo path tracing algorithm [74]. This method estimates the radiance for each pixel of the image by tracing random walks that start from the eye and traverse the pixel, bouncing in the scene. For this pass, our algorithm constructs so-called “informed” PDFs from the information contained in the results of the first step. These PDFs allow the Monte-Carlo step to generate a better set of random walks that reduce the noise of the final image. The key of the method is to reduce the variance of the estimators by using PDFs that are approximately proportional to

$$\rho_{\mathrm{bd}}(y,\vec{\omega}_{\mathrm{o}},\vec{\omega}_{\mathrm{i}})\,L_{\mathrm{i}}(y,\vec{\omega}_{\mathrm{i}})\cos\theta_{\mathrm{i}}$$

for surface points (kernel in Equation 2.2), and to

$$L(y,\vec{\omega}_{\mathrm{i}})\,p(\vec{\omega}_{\mathrm{o}},\vec{\omega}_{\mathrm{i}})$$

for points within the media (kernel of integral in Equation 2.6). Notice that, as in Section 4.2, we use the following notation: y denotes points on receivers (gathering points), and x will denote points in sender objects. The PDFs are computed using the links related to the hierarchical structure created in the first pass, to estimate L_{i} by means of a directional gathering procedure. Note that links to clusters must be taken into account too.

The key of our approach is to perform importance sampling taking into account the BRDF *and* the approximated irradiance computed in the first pass. This sampling is performed using PDFs that are tied to leaf objects of the hierarchical structure created in the first pass, and are computed using the links related to those objects to estimate L_{i} by means of a directional gathering procedure. Note that links to clusters must be taken into account too.

Our extended path tracing algorithm uses next event estimation and the usual importance sampling based on the BRDF term for surfaces without information about the incoming radiance [80], and on the phase function for participating media [20]. Our specific implementation sets PDFs taking into account the incoming light at Lambertian surfaces (based on $L_{\mathrm{i}}(y,\vec{\omega}_{\mathrm{i}})\cos\theta_{\mathrm{i}}$). The PDFs of volume elements containing participating media can be set taking into account the incoming radiance, the phase function, or their product.

5.2.1 Constant Basis Functions

As stated above, our first design of the PDFs uses a constant basis representation by means of bins covering the directional domain [41,120,167]. In the following we describe this design, and results of their use are also presented.

Constructing PDFs

Identification of "Informed" Polygons and Construction of its PDFs. For a certain view, we identify the "informed" polygons by constructing an auxiliary image ray casting the scene. The purpose of this ray casting is simply to create a set of identifiers of the polygons visible for that view. Thus, no local illumination model is used at all to create the resulting image, but only the identifier of the closest polygon (if any) for each viewing ray is stored per pixel. Our implementation uses a single ray per pixel, so that each pixel needs to store a single polygon identifier at most—if multi-sampling were used, then lists of identifiers would be necessary. We use an image resolution with the same aspect ratio as the resolution of the requested final accurate image (result of the complete two-pass algorithm), but with reduced size, to avoid spending too much time in this stage. Also a z-buffer could be used for this identification task, as long as it provides polygon identifiers. As a consequence of this procedure, we dispose a set of polygons that are *candidates* to be tagged as "informed" polygons. We do not consider "informed" polygons those that are emitters. Triangles that are "very far" from being equilateral are not considered either, since presumably their illumination can vary considerably within their areas (compared against well formed triangles), and thus their PDFs would not be meaningful. In our implementation, this quite-unequilateral property is measured by the predicate $\|l_i - \bar{l}\| > \varepsilon$, being l_i the length of edge i, $\bar{l}$ the mean of edge lengths, and ε an user defined threshold. Also those polygons whose projection is smaller than a certain percentage of the image size are not considered "informed". This is easily done by counting the number of hits per polygon.

Once the top level polygons have been identified, a recursive function is called to create CDFs at all leaves of those polygons, if they happen to have a link at any level of the hierarchy. If a polygon does not have any link, there is no need to create a CDF for it.

Identification of "Informed" Volume Elements and Construction of its PDFs. To identify the leaf clusters containing media for which we must set "informed" CDFs, we also employ a ray casting procedure for the selected view, similar to the one used for polygons. However, in this case, a list of identifiers of leaf clusters is directly updated when casting viewing rays through the scene, by considering the leaf clusters pierced by each ray. The construction of the CDF for each one of those leaf clusters must be performed by gathering energy from the links arriving to those cluster leaves (arriving directly or at an ancestor level).

Gathering Energy. PDFs are set for the selected leaf elements by considering the reflectance equation or the scattering equation and applying an importance sampling schema. For surfaces, we start by reminding the reflectance equation (Equation 2.2):

$$L(y,\vec{\omega}_\mathrm{o}) = \int_{\Omega} \rho_\mathrm{bd}(y,\vec{\omega}_\mathrm{o},\vec{\omega}_\mathrm{i}) L_\mathrm{i}(y,\vec{\omega}_\mathrm{i}) \cos\theta_y \, d\sigma_{\vec{\omega}_\mathrm{i}} \, . \tag{5.1}$$

In the case of Lambertian surfaces the bidirectional reflectance distribution function is a constant, therefore Equation 5.1 reduces to

$$L(y,\vec{\omega}_\mathrm{o}) = \rho_\mathrm{bd} \int_{\Omega} L_\mathrm{i}(y,\vec{\omega}_\mathrm{i}) \cos\theta_y \, d\sigma_{\vec{\omega}_\mathrm{i}} \, . \tag{5.2}$$

To solve Equation 5.2 with a Monte Carlo process, a sampling schema for the integrand $L_\mathrm{i}(y,\vec{\omega}_\mathrm{i}) \cos\theta_y \, d\sigma_{\vec{\omega}_\mathrm{i}}$ is needed to generate direction samples $\vec{\omega}_\mathrm{i}$. To this end, we propose the use of a discrete PDF. The directional domain is split into a certain number of meridians and parallels, thus obtaining a set of bins, with $\Omega = \bigcup_{i=0}^{\#\mathrm{bins}-1} \Omega_i$. Each bin will store a coefficient for the PDF (and another for the CDF).

To construct the PDF related to a receiver leaf element $\mathcal{R}$ (that can be a polygon or cluster), for each considered link (coming from a sender element $\mathcal{S}$) a sampling process is performed. Following Equation 5.2, we define the unnormalized PDF of a bin i to be

$$\breve{p}_i = \int_{\Omega_i} L_\mathrm{i}(y,\vec{\omega}_\mathrm{i}) \cos\theta_y \, d\sigma_{\vec{\omega}_\mathrm{i}} \, , \tag{5.3}$$

and thus the normalized PDF of the i-th bin is defined as $p_i = \breve{p}_i / \sum_{k=0}^{\#\mathrm{bins}-1} \breve{p}_k$. An approximation of Equation 5.3 can be obtained as follows:

$$\breve{p}_i = \int_{\Omega_i} L_\mathrm{i}(y,\vec{\omega}_\mathrm{i}) \cos\theta_y \, d\sigma_{\vec{\omega}_\mathrm{i}} \approx \overline{L}_{\mathrm{i},i} \cos\theta_i \int_{\Omega_i} d\sigma_{\vec{\omega}_\mathrm{i}} = \overline{L}_{\mathrm{i},i} \cos\theta_i \, \sigma_i \, , \tag{5.4}$$

where $\overline{L}_{\mathrm{i},i}$ is an approximation to the mean of the incoming radiance through bin i, θ_i is the angle between the normal of the surface and a representative direction of bin i (e.g. the center of the bin), and σ_i its solid angle. The computation of $\cos\theta_i$ and σ_i is straightforward, and $\overline{L}_{\mathrm{i},i}$ can be estimated using N samples as follows (see Figure 5.1):

$$\overline{L}_{\mathrm{i},i} = \frac{\int_{\Omega_i} L_\mathrm{i}(y,\vec{\omega}_\mathrm{i}) \, d\sigma_{\vec{\omega}_\mathrm{i}}}{\int_{\Omega_i} d\sigma_{\vec{\omega}_\mathrm{i}}} \approx \frac{\sum_{j=0}^{N} L_{\mathrm{i},i}^{j} \, \sigma_{i,j}}{\sigma_i} \, ,$$

being $L_{\mathrm{i},i}^{j}$ the j-th sample of $L_\mathrm{i}(y,\vec{\omega}_\mathrm{i})$, and σ_i its related solid angle.

Notice that substituting the final expression of $\overline{L}_{\mathrm{i},i}$ in Equation 5.4 we obtain a simple expression for the unnormalized PDF:

$$\breve{p}_i \approx \overline{L}_{\mathrm{i},i} \cos\theta_i \, \sigma_i = \frac{\sum_{j=0}^{N} L_{\mathrm{i},i}^{j} \, \sigma_{i,j}}{\sigma_i} \cos\theta_i \, \sigma_i = \cos\theta_i \sum_{j=0}^{N} L_{\mathrm{i},i}^{j} \, \sigma_{i,j} \, .$$

In order to account for the energy reaching a particular leaf $\mathcal{R}$, the hierarchy containing the ancestors of $\mathcal{R}$ has to be traversed, so that all links are considered. For each link, a set of $N = 32^2$ stratified sample points x_j at the source element $\mathcal{S}$ are taken. Estimates of the incoming radiance

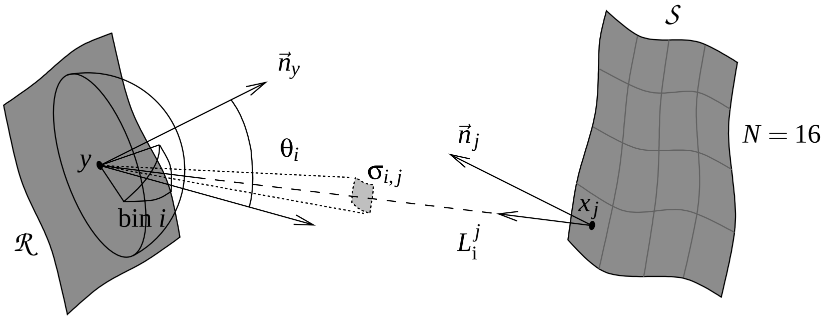


FIGURE 5.1 Example of sampling used in the construction of a PDF.

$L^j_{\mathrm{i},i}$ (from x_j to the center of $\mathcal{R}$) are computed by using the results of the first pass. Estimates of the solid angle related to each sample are computed as follows:

$$\sigma_{i,j} \approx \begin{cases} \frac{A_\mathcal{S}}{N}\left(\frac{y-x_j}{\|y-x_j\|}\cdot\vec{n}_j\right)\frac{1}{\|y-x_j\|^2} & \text{for } \mathcal{S} \text{ being a surface} \\ \frac{\sqrt[3]{V^2_\mathcal{S}}}{N}\,\frac{1}{\|y-x_j\|^2} & \text{for } \mathcal{S} \text{ being a volume,} \end{cases}$$

where $A_\mathcal{S}$ and $V_\mathcal{S}$ are the area and the volume of $\mathcal{S}$ respectively, and $\vec{n}_j$ denotes the normal vector at x_j.

In the case of volumes, the starting point is the scattering equation (integral term in Equation 2.6):

$$J_{\mathcal{S}\mathcal{R}}(y,\vec{\omega}) = \frac{\Omega(y)}{4\pi}\int_\Omega p(y,\vec{\omega},\vec{\omega}_{xy})\,L_\mathrm{i}(y,\vec{\omega}_\mathrm{i})\,d\sigma_{\vec{\omega}_\mathrm{i}}\,.$$

From this equation, three importance sampling strategies can be devised: based on the phase function [114, 20, 83], on the incoming radiance, or on their product. Following the same development as above, when using the incoming radiance one obtains $\breve{p}_i \approx \sum_{j=0}^{N} L^j_{\mathrm{i},i}\,\sigma_{i,j}$. The combination of phase function and incoming radiance can be performed straightforwardly, by multiplying the previous expression of $\breve{p}_i$ by the phase function, using the direction of the ray before reaching the interaction point, and a representative direction of each bin (i.e. the center), and normalizing the resulting distribution to obtain a probability density function. Unfortunately, this direct strategy for combining phase function and incoming radiance has a cost too big, We have performed experiments that show that the overall computational time using this scheme can be greater than e.g. using uniform sampling, for which samples are not that good, but many more samples can be obtained per second.

An offset is used to make sure that the PDF is positive in all the domain (half of the hemisphere for surfaces, the whole sphere for volumes), to avoid PDFs that potentially could drive to biased results. Notice that, because of inaccurate sampling, bins can have associated zero values where they should be positive. Figure 5.2 shows an example of this problem, represented in 2D, where a PDF is set for a certain polygon $\mathcal{R}$. From the finite element pass, it is known that $\mathcal{R}$ receives energy from polygons $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$ and $\mathcal{S}_4$, and from the geometry of this scene, $\mathcal{R}$ receives energy from all directions above its hemisphere. For simplicity, suppose that just a single sample is used for sending element. Thus, using constant basis functions to represent the directional distribution related to the PDF as said above, some bins are updated with positive values, but other bins have zero values at the end of this gathering process. Using the resulting “sampled PDF” would drive to biased results, since part of the domain would never be sampled. To overcome this problem, an offset is added, so that the whole domain would be used. In our implementation the offset is a fraction of the value of the sum of the unnormalized PDFs of the bins, i.e. $\alpha \sum_i^{\#\text{bins}} \breve{p}_i$. Note that if initially $\breve{p}_i > 0$ for all i, then the PDF is already positive in all the domain, and there is no need for the use of the offset.

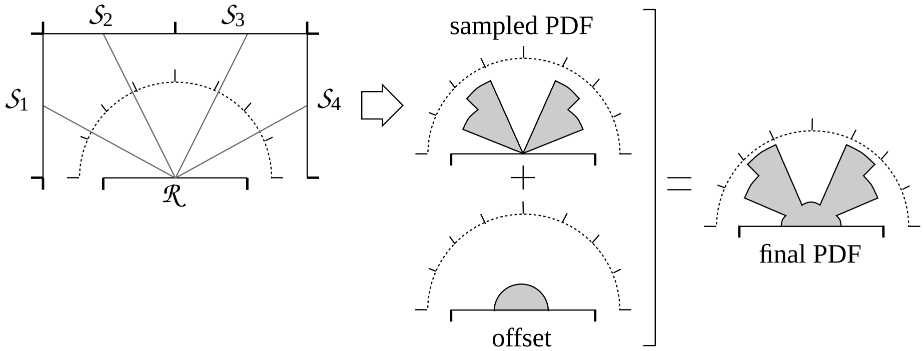


FIGURE 5.2 Inaccurate sampling problem, and offset workaround.

The creation of the CDF is straightforward once the discrete PDF is completed, by simple accumulation. Thus, for bin i its CDF would be $c_i = \sum_{k=0}^{i} p_k$. An example is shown in Figure 5.3.

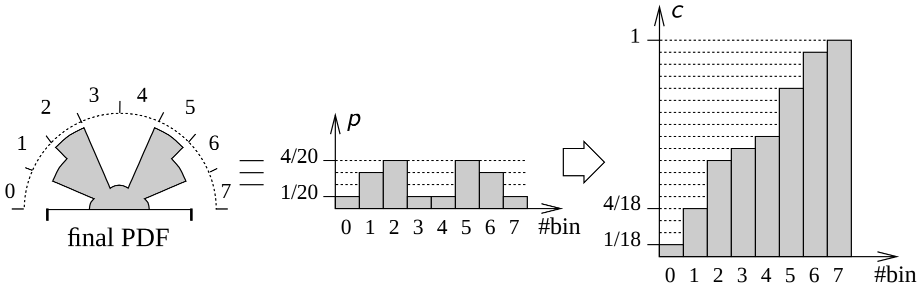


FIGURE 5.3 Example of a discrete PDF and its related CDF.

PDFs Put Into Use

Concrete methods to sample Phong-like BRDFs and unimportant Lambertian BRDFs simply use the typical importance sampling strategy, and for “informed” surfaces the constructed CDFs are used. The same scheme is employed for volume elements.

To generate a sample direction using the CDFs, all non-triangular bins are divided into two triangles. Therefore, the whole sphere of directions is tessellated into a set of spherical triangles. When requiring a sample direction, first the discrete PDF related to the triangular bins is sampled, by generating uniformly a random number between 0 and 1, and identifying the related bin through the CDF (i.e. the PDF is sampled by inversion of the CDF). Then a direction is obtained by uniformly sampling the spherical triangle [10]. To this end, we adapted the C-code provided by Arvo [9] to be integrated into our system. Probably a better sampling strategy for a reflected/scattered direction would be to interpolate the PDFs of the elements near the interaction point, as is done in [101, 53], at the expense of increasing the computation time.

Path Tracer

Our path tracer follows the work of [80], using importance sampling (see Section A.2.2). We implemented a stopping criterion based on the estimation of the average per-pixel variance [128], discussed in Appendix A.2.3.

Implementation and Results

The implementation of the two-pass method the classical Monte Carlo path tracing has been done using two systems. The finite element part of the software (Section 4.2) has been built on the BRIGHT system, a Radiance Clustering algorithm developed at *i*MAGIS. The Monte Carlo part is based in the SIR system, a framework for the development of global illumination algorithms at GGG [103]. The implementation of the second pass is currently limited to a single homogeneous participating medium, but it could be extended to deal with a set of media and inhomogeneous media.

Six tests have been performed obtaining images by using the classical path tracing and the new method. To obtain images of the same quality, the adaptive control of the number of required samples per pixel has been used, for several different values of the related error percent relative threshold. Each execution reported a set of statistics to compare the results of classical path tracing and our method. For each test we give a table summarizing those statistics. The meaning of the labels used in the tables is the following:

- *eprt*. Acronym for "error percent relative threshold".
- *uid*. Acronym for "use importance data", i.e., it indicates through a positive or negative sign depending on whether CDFs were used or not.
- *#paths*. Number of paths resulting from the execution. It is equal to the number of primary rays.
- *FE time*. Time spent in the finite elements stage (if any).
- *pc time*. Time spent in the creation of "informed" PDFs (after "PDF creation").
- *PT time*. Time spent in the path tracing stage.
- t^-/t^+. Ratio obtained by taking the tracing time spent by the execution that did not use "informed" CDFs and dividing it by the tracing time spent by the classical path tracing.

Figure 5.4 shows the statistics obtained for our set of tests, for a quite low image resolution (25×25 pixels), using "informed" PDFs or not. These tests are explained below.

Test A: Empty Cornell Box Scene. The very first test of the new method uses the well-known Cornell box scene. The error percent relative threshold was set to 200%, 100%, 50% and 25%. Statistics for these executions are given in Figure 5.4. Obviously, when reducing the value of the *eprt* threshold more samples are needed to meet the error criterion, and the quality of the resulting image is improved. As expected, for each used *eprt* value the number of paths is considerably higher for the classical Monte Carlo executions than for the two-pass method, due to the better sampling strategy used by the two-pass method. However, the speed-up achieved is not directly related to the number of paths times the mean path length, since the cost associated to the sampling process is higher in the two-pass method.

Test B: Adding Two Diffuse Boxes. The model used for the Test B was the Cornell box augmented with a pair of diffuse boxes. The statistics related to the *eprt* values of 200%, 100%, 50% and 25% are also shown in Figure 5.4. There has been an increase of scene complexity with respect to the Test A, and thus for the same *eprt* values this test reports higher number of paths needed to compute the images, and as a consequence the tracing times are higher too. Similar relative speed-ups are obtained.

Test C: Glossy Surfaces. Note that the Monte Carlo second pass can deal with all kinds of scenes. This stands both for radiometric and for geometric properties. It is possible to get rough

Test A

eprt	*uid*	*#paths*	*PT time*	t^-/t^+
200	–	61552	15.82s	
	+	21792	6.68s	2.37
100	–	240944	57.71s	
	+	95008	28.64s	2.02
50	–	968064	231.89s	
	+	458160	144.9s	1.60

Test B

eprt	*uid*	*#paths*	*PT time*	t^-/t^+
200	–	78880	22.04s	
	+	23216	7.58s	2.91
100	–	351616	99.39s	
	+	130448	42.18s	2.36
50	–	1474720	398.01s	
	+	692672	237.28s	1.68

Test C

eprt	*uid*	*#paths*	*PT time*	t^-/t^+
200	–	76336	21.43s	
	+	24064	8.19s	2.62
100	–	323424	87.16s	
	+	130384	44.1s	1.98
50	–	1454560	393.2s	
	+	781392	269.34s	1.46

Test D

eprt	*uid*	*#paths*	*PT time*	t^-/t^+
200	–	103168	30.87s	
	+	20064	9.69s	3.19
100	–	340512	99.12s	
	+	93760	46.5s	2.13
50	–	1386624	400.93s	
	+	480304	236.32s	1.70

Test E

eprt	*uid*	*#paths*	*PT time*	t^-/t^+
200	–	94672	28.58s	
	+	21808	8.82s	3.24
100	–	427024	132.49s	
	+	116016	47.15s	2.80
50	–	1819136	573.74s	
	+	658944	276.83s	2.07

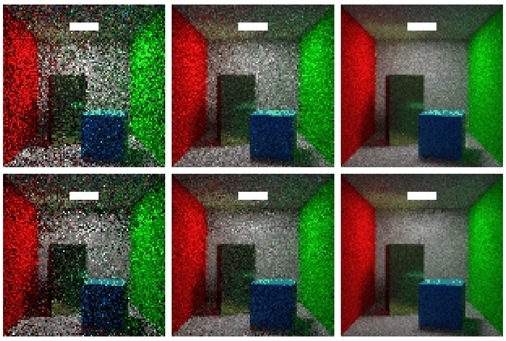
Test F

eprt	*uid*	*p*	*#paths*	*PT time*	t^-/t^+
200	–	p	74880	27.32s	
	+	p	24816	10.14s	2.69
	+	L_i	20560	8.6s	3.18
100	–	p	278160	94.83s	
	+	p	133824	55.66s	1.70
	+	L_i	100240	42.28s	2.24
50	–	p	1138944	364.31s	
	+	p	736816	289.77s	1.26
	+	L_i	578816	238.82s	1.53

FIGURE 5.4 Statistics for the set of tests.

results by using an *approximated* scene in the first pass (e.g. a mirror surface can be substituted by a highly glossy surface; a texture can be substituted by its mean value; bump maps can be set to zero) and then use the *exact* scene in the second pass. Also curved surfaces can be approximated by sets of planar surfaces in the first pass, and still use the true geometry in the second. Taking advantage of this fact, we have tested our method by using the Cornell box with a glossy box and a diffuse box. The first pass does not deal with the specular component of the glossy surfaces, but instead only the diffuse part is used to obtain the intermediate results. The second pass obviously takes into account the true model. Comparing the tables shown in Figure 5.4 for Test B and Test C, it is clear that this test takes more paths and time (for the two methods) than the previous test. Also it is derived from the tables that the speed-up achieved in this test is lower than the one obtained for the diffuse boxes. The reason is that the rough solution obtained with the first method is worse than it was for the previous test. This makes the sampling process for this scene worse than it was for the scene with diffuse boxes. Figure 5.5 shows the statistics and images of this test when generating images of 100×100 pixels.

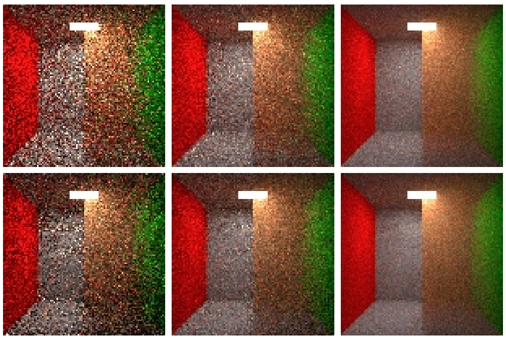
Test D: Isotropic Scattering Medium in Cornell Box. The first test of the two-pass method using a participating medium used the Cornell box with the addition of an isotropic participating medium in half of its space. New statistics and related 100×100 images are shown in Figure 5.6. Obviously the number of paths has been increased with respect to the first test with the empty Cornell box, since there are scattering events within the participating medium. Note that the



eprt	*uid*	*#paths*	*FE time*	*pc time*	*PT time*	t^-/t^+
200	$-$	1201936	.	.	332.62s	
	$+$	378896	0.30s	14.66s	127.46s	2.61
100	$-$	5100656	.	.	1377.89s	
	$+$	2165792	0.31s	14.66s	731.09s	1.88
50	$-$	23504512	.	.	6478.19s	
	$+$	12962384	0.30s	14.63s	4538.23s	1.43

FIGURE 5.5 Statistics and images of the Cornell box with a diffuse box and a specular box. Images: Top: Classical path tracing, Bottom: new method. From left to right: *eprt* = 200, 100 and 50.

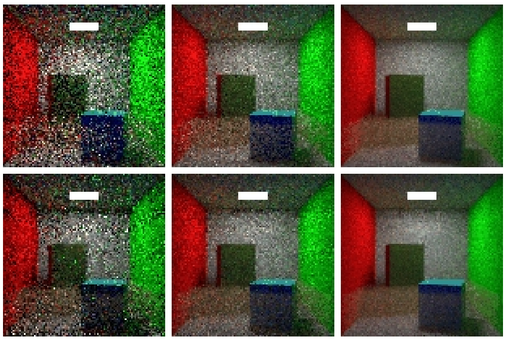
speed-up achieved is better than the achieved for the first test (empty Cornell box).



eprt	*uid*	*#paths*	*FE time*	*pc time*	*PT time*	t^-/t^+
200	$-$	1601504	.	.	478.87s	
	$+$	321280	45.11s	182.64s	162.54s	2.95
100	$-$	5570720	.	.	1637.15s	
	$+$	1491040	44.73s	182.65s	738.83s	2.22
50	$-$	22553536	.	.	6839.61s	
	$+$	8297216	44.86s	182.68s	4045.51s	1.69

FIGURE 5.6 Statistics and images of the Cornell box with half of its space filled with isotropic participating medium. Images: Top: Classical path tracing, Bottom: new method. From left to right: *eprt* = 200, 100 and 50.

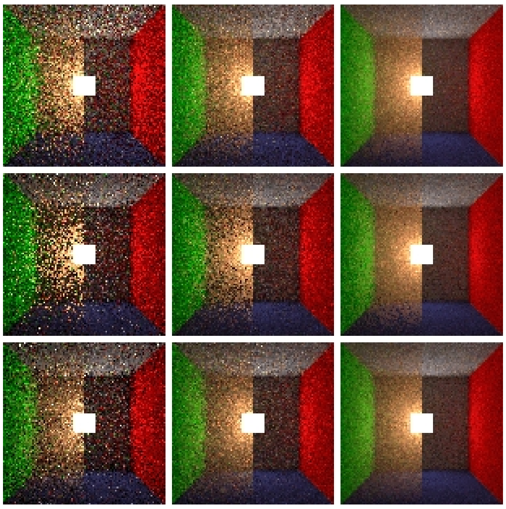
Test E: Isotropic Scattering Medium in Augmented Cornell Box. The mixing of participating media and surfaces has been tested with the Cornell box scene with the diffuse cubes, adding a slab of an isotropically scattering participating medium. New results for 100×100 images are given in Figure 5.7. The new method performs importance sampling based on the incoming illumination for both surfaces and medium, achieving a high reduction of the number of paths with respect to classical path tracing, and a high speed-up.



eprt	*uid*	*#paths*	*FE time*	*pc time*	*PT time*	t^-/t^+
200	$-$	1542000	.	.	463.52s	
	$+$	352112	8.90s	60.93s	142.71s	3.25
100	$-$	6747392	.	.	2049.51s	
	$+$	1891984	8.65s	60.83s	860.14s	2.38
50	$-$	29119872	.	.	9184.9s	
	$+$	10709424	8.79s	60.86s	4437.09s	2.07

FIGURE 5.7 Results for the Cornell box with boxes and a slab of participating medium. Images: Top: Classical path tracing, Bottom: new method. From left to right: *eprt* = 200, 100 and 50.

Test F: Anisotropic Scattering Medium. Finally, tests have been performed in a closed Cornell box with half of its space filled with an anisotropically scattering medium (see Figure 5.8 for results when generating 100×100 images). Concretely, the phase function used has been the Schlick phase function with parameter of anisotropy being 0.8 (forward scattering—see Section 2.3.4). For these tests the camera used pointed to the light source. The classical path tracing executions perform importance sampling based on the phase function at scattering points within the media. Two different sets of executions have been done with the new method, one using importance sampling at scattering points based on the phase function, and the second using importance sampling at those points based on the incoming illumination. This is shown in the table of Figure 5.8 in the p (the PDF) column being p (based on the phase function) or L_i (based on the incoming illumination). We derive from the table that the use of the phase function to guide scattering sample directions does not drive to a relevant gain in the execution time of the second pass of the new method (practically they spent the same time for the more converged case—the $eprt = 50$ case). However, the use of importance sampling based on the incoming illumination drives to a considerable speed-up, although not as high as in the previous tests.



eprt	*uid*	*p*	*#paths*	*FE time*	*pc time*	*PT time*	$\frac{t^-}{t^+}$
200	−	p	1209424	.	.	381.62s	
	+	p	393376	18.64s	33.49s	160.93s	2.37
	+	L_i	327600	18.6s	33.41s	140.55s	2.72
100	−	p	4314064	.	.	1371.97s	
	+	p	2069520	18.68s	33.43s	841.84s	1.63
	+	L_i	1613184	18.59s	33.57s	686.52s	2.0
50	−	p	18095312	.	.	5766.81s	
	+	p	12265904	18.62s	33.42s	5017.99s	1.15
	+	L_i	9640272	18.63s	33.58s	4092.49s	1.41

FIGURE 5.8 Statistics and images for the Cornell box with half of its space filled with anisotropic participating medium. Images: Top: Classical path tracing, Middle: new method with PDF based on phase function, Bottom: new method with PDF based on incoming illumination. From left to right: $eprt = 200$, 100 and 50.

Conclusions

In this section we have presented a first second-pass method for the computation of the global illumination in general scenes, including anisotropically scattering media. We conclude that for the scenes tested there is an acceleration of the ultimate path tracing stage and of the overall two-pass algorithm. The reduction in the number of paths is considerable with the new method with respect to classical path tracing, but the gain obtained in time is lower. This is due to the fact that the cost of processing a path in the proposed method is higher than the cost of processing a path in classical path tracing.

The speed-ups obtained in the tests for low image resolution (25×25) are kept for images of higher resolution (100×100). We could derive from this that for even higher resolutions the speed-up will remain similar.

5.2.2 Limitations of PDFs Using Constant Basis Functions

Tests performed with the approach presented in Section 5.2.1 led us to identify the restraints of the method. There are two main issues that make this method limited:

1. The intrinsic nature of the discrete directional representation of the informed PDFs. An exaggerated example is depicted in Figure 5.9: a simple scene with a small light source and a large leaf surface receiver, with its related PDF. When sampling this PDF, a certain spherical triangle will be uniformly sampled. Since the projection of the light source onto this spherical triangle is quite reduced, a high ratio of samples will not hit the light source, and thus the efficiency is reduced.

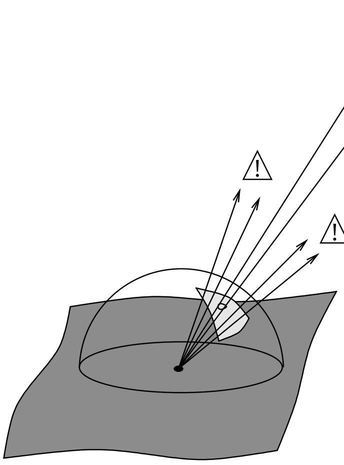


FIGURE 5.9 Small projection on the spherical triangle.

2. The extension of the use of the informed PDF. They extend across whole surface leaves. If a very rough subdivision of the scene resulted from the first step, distant points on the surfaces would probably receive (in a much more accurate solution) illumination quite differently, so that it would be more precise to have a set of informed PDFs instead. This problem is shown in Figure 5.10, where a large surface receives illumination from a small light source, and the first pass established a single link from the light source to the receiver. Great differences on the incoming angle can happen at the corner points of the receiver, but since a single informed PDF is constructed, its final effectiveness is greatly reduced.

To address these limitations we use the sampling scheme of Section 5.2.3, where the PDFs are constructed using the Illumination Samples (links arriving at leaves and ancestors, representing implicitly the space of directions), and per pixel basis PDFs are set for first bounces.

5.2.3 Link Probabilities

PDFs Based on Links

A technique for constructing and using informed PDFs to improve the convergence of the second pass is introduced here. The Link Probabilities refer to discrete PDFs constructed in the domain of the links. A set of links represents the sources of illumination.

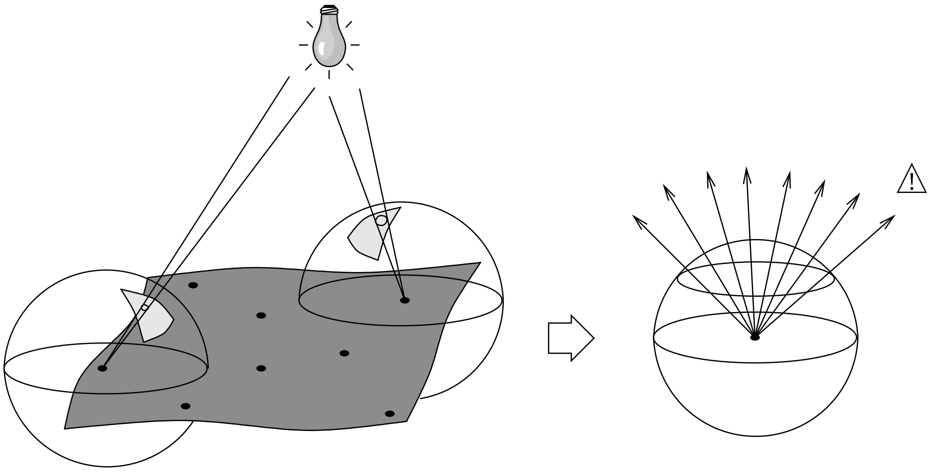


FIGURE 5.10 PDF for a leaf that is too large.

- Instead of using directional constant basis PDFs (Section 5.2.1), we use a new sampling scheme based directly on the Illumination Samples [163], i.e. on the links arriving at leaves and ancestors [165, 172]. The PDF is represented by an array of pairs containing the links (that implicitly represent the space of directions) and the discrete PDF values. These PDF values result from the estimation of the irradiance through the links after normalization.

- Constructing the informed PDFs is performed in a per leaf basis (for secondary bounces of the random walk), and also in a per pixel basis (for the first bounce). This means to recompute the values of the discrete PDFs of the leaf element *at the interaction point* when the radiance for a new pixel has to be estimated.

- The computation of PDFs is performed not only for visible leaf elements, but for all of them, in order to be used for secondary bounces (to improve convergence). Only PDFs corresponding to first interactions (i.e. the first hit point in the path tracing step) are recomputed in a per pixel basis, for better accuracy. However, PDFs for following interactions are constructed in a per leaf basis.

As stated above, informed PDFs are represented by an array of links with their associated probabilities (Figure 5.11). This set of links represents all the sources of illumination for a given leaf of the surface or volume hierarchy. The construction of informed PDFs has three steps:

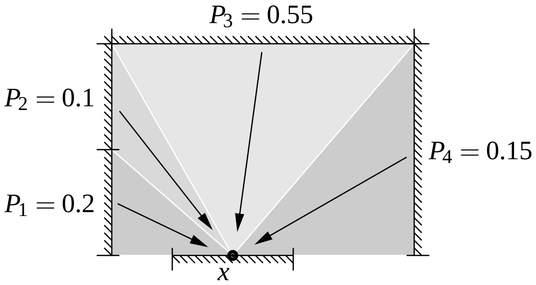


FIGURE 5.11 PDF based on links for a given interaction point x. In this example four links contribute to the illumination at x. Probabilities are normalized, so their sum is one.

1. Identification of the leaf element of the hierarchy corresponding to the hit point of the random walk.

2. Collecting all the links that contribute that leaf element by traversing the full hierarchy (from the leaf to the root cluster) computed in the first pass. This step is performed only once per leaf, since successive hits on the same leaf reuse the same information.

3. Computation of a probability for each link. Irradiances coming from each link's sender are estimated, accumulated and normalized to obtain a valid PDF. The PDF value associated to each link represents the probability of the random walk to choose a direction within the solid angle subtended by the element acting as shooter in the given link. As stated above, this step is performed in two different ways depending whether:

   - the hit point is the first bounce of the random walk. This case corresponds to interaction points on visible elements. The PDF values are recomputed once per pixel to take into account the variation of the irradiance across the extent of the leaf. That is, the point hit by the viewing ray is used to re-evaluate the approximated irradiances coming from each link to set their related probabilities. The PDF values are not recomputed for each first bounce of the random walks used for the same pixel because the variation of irradiance across hit points on the same leaf for the same pixel is hardly noticeable.
   - the hit point is a secondary bounce of the random walk. The PDF values are computed once per leaf. (Therefore, for efficiency reasons, the link collecting and the per leaf PDF computation are performed in a preprocess for all leaves, using for approximate irradiance computation for instance the centers of the sender and the receiver.) This PDF is used for all the hit points of the same leaf.

In order to obtain a sample (reflected/scattered) direction, first the discrete PDF is sampled to choose a link. Next, a direction must be determined towards the link's sender $\mathcal{S}$. If $\mathcal{S}$ happens to be a surface, then this can be done by uniformly sampling its area. Alternatively, the solid angle subtended by $\mathcal{S}$ can be sampled to get a source point [10, 180]. However, if the link's sender is a participating medium, the use of solid angle measure to sample it is rather complicated; instead we sample the related volume according to the volume measure. Care must be taken when sampling directions that do not hit $\mathcal{S}$ (due to occluders). This is depicted in Figure 5.12 through a simple example. Suppose we are sampling a reflected direction at point y in receiver $\mathcal{R}$. Also suppose that l_2 is chosen to get a direction sample, and a point x is sampled uniformly in $\mathcal{S}_2$ to get a direction $\vec{\omega}_{yx}$. If this direction was used, the estimated contribution must be zero because the radiance coming from x does not contribute to the irradiance at y. This situation is identified in the program because the random walk going from y to x hits another point (x') on its way.

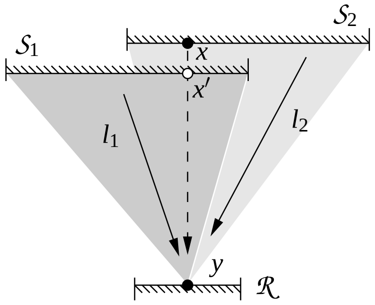


FIGURE 5.12 The contribution of the radiance at x to the irradiance at y is zero since x is occluded by $\mathcal{S}_1$.

Adaptive PDFs

An improvement of the use of the array of pointers to links with their related constant PDFs consists of the use of adaptive PDFs, in which the PDFs of the links are continuously improved as the execution progresses (instead of remaining fixed), to better match the distribution of the irradiance at a certain point. An example of the limitation of the use of constant PDFs is shown in Figure 5.13, where a leaf $\mathcal{R}$ receives light from links l_1 and l_2. Notice that point x should consider positive probabilities for the two links, but ideally for point y the PDF of l_1 should be zero to avoid wasting samples trying to hit $\mathcal{S}_1$ from y, because $\mathcal{S}_1$ does not contribute (directly) to the illumination of y. Otherwise the estimated value of the irradiance due to $\mathcal{S}_1$ must be set to zero, increasing the variance of the total irradiance at y. Thus, to reduce the variance the PDF must be adaptively updated in order to match the correct irradiance distribution. Notice that, in order to obtain unbiased results, the informed PDFs should consider only those links that really contribute to the illumination at the point where the PDF is applied. Senders related to links arriving at a leaf receiver $\mathcal{R}$ can be partially occluded for points on $\mathcal{R}$ (like e.g. point y in Figure 5.13). Therefore, to guarantee unbiasedness, for each point in the walk step a process should be performed to discard those links whose sender is completely occluded. Currently, our implementation does not perform this step, which can be very costly depending on the scene.

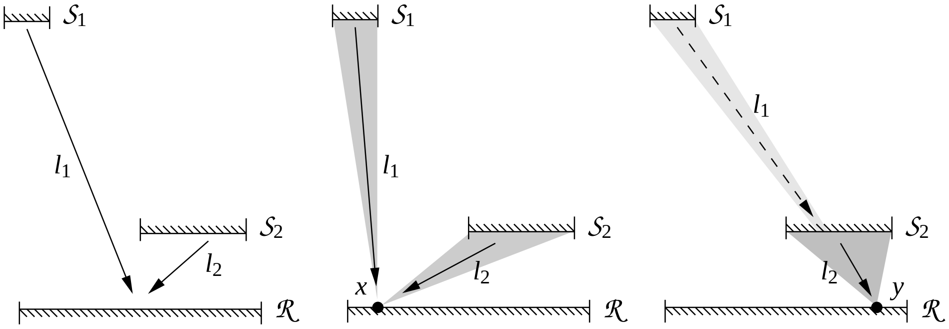


FIGURE 5.13 Left: The finite elements first pass established two links arriving at leaf $\mathcal{R}$. Middle: At point x the PDF must consider both links. Right: The PDF at point y should ideally set to zero the probability of link l_1 since sender $\mathcal{S}_1$ is fully blocked.

The PDFs are updated by taking into account the number of failures (choosing a direction that does not hit the link's sender) with respect to the number of attempts to get a sample direction once a link is sampled. Due to the extra cost of the maintenance of the adaptive PDFs, these are only used in the PDFs per pixel basis (first bounce). Informed PDFs for secondary bounces are not updated.

When a certain number of failures is reached when sampling the adaptive PDF per pixel, the basic expression for updating the probability of link i is

$$P_i = \begin{cases} \frac{\#\text{tries}_i - \#\text{failures}_i}{\#\text{tries}_i} \times \widehat{H}_i & \text{if } \#\text{tries}_i > 0 \\ \varepsilon & \text{otherwise}, \end{cases} \tag{5.5}$$

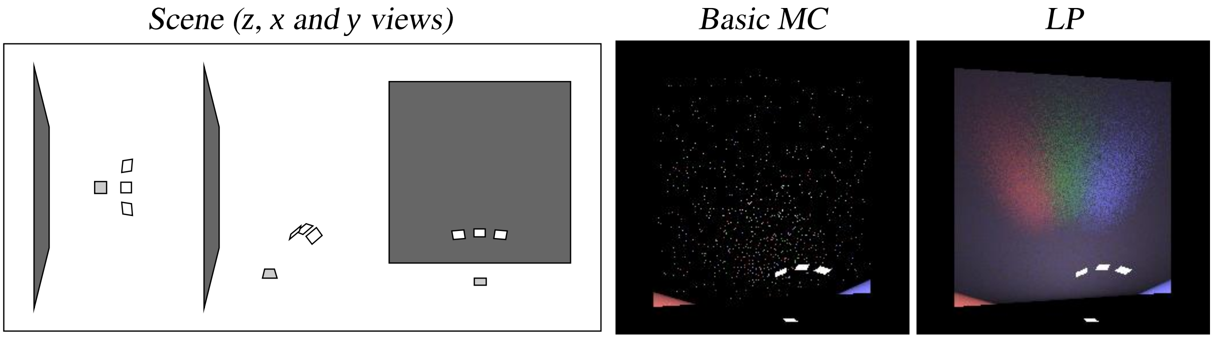
where $\widehat{H}_i$ is the estimated irradiance for link i from the first pass, and ε is a small positive value to avoid P_i from becoming zero (since other directions can contribute to the illumination). Equation 5.5 is evaluated for all the links stored in the PDF, and then the results are normalized.

Results

We have implemented a first version of the algorithm to demonstrate the benefits of the method, using the first pass of Section 4.2, which deals only with diffuse surfaces, the second pass not considering links arriving from volumes containing participating media.

We have tested the method with three different scenes, to show the performance of the PDFs constructed using Link Probabilities.

For the first test, a very simple (exaggerated) test scene has been used to show the differences between the standard path tracing algorithm and the Link Probabilities method. Results are shown in Figure 5.14. A small glossy reflector is lit by three light sources, illuminating a large diffuse screen. Sixteen samples per pixel were used in each execution, using next event estimation (direct illumination is estimated at each bounce of the random walk). The image on the left was obtained using Monte Carlo path tracing with importance sampling based on the BRDF, whereas the one on the right was generated by the approach based on Link Probabilities (LP; PDFs based on Illumination Samples). The number of samples for each image was set so that both took the same time (68 seconds) to be computed (33 samples for the basic Monte Carlo and 16 for the LP method). It is clear from Figure 5.14 that LP gives much better results than the basic Monte Carlo method.


Scene (z, x and y views)

Basic MC

LP

FIGURE 5.14 Three light sources illuminating a small glossy reflector, which in turn illuminates a large diffuse receiver.

To show that the Link Probabilities approach works well for scenes with difficult lighting conditions were classical methods fail, in the second test we compute the global illumination in a room illuminated indirectly by a lamp. Light leaving the light source needs at least three bounces to reach the eye. Two images computed with standard path tracing and with the new approach are shown in Figure 5.15. A standard path tracing algorithm is unable to get a meaningful image even with a high number of samples, while the LP approach can render an image with a reasonable quality.

Finally, tests have been also performed with an “office” scene containing some tables and chairs (585 initial polygons), with indirect illumination. Figure 5.16 shows the results computed with the pure Monte Carlo path tracing and the LP approach, using next event estimation. Timings for the LP approach include the time spent in the finite element step (26 seconds), the time to construct the PDFs per leaf basis (1 second), and the time of the Monte Carlo path tracing with PDFs per pixel construction. Unfortunately, there are some spike pixels on the images using the LP approach, since sometimes a poor importance sampling drives to overestimation (see [189]). This can be due to the use of PDFs for secondary bounces in a per leaf basis, instead of using more precise PDFs.

The plot in Figure 5.17 (left) shows the running times of the Basic Monte Carlo and the LP executions for the office scene, for a set of increasing number of samples. Notice that there is an overhead of the LP method with respect to the basic Monte Carlo method, and that the running

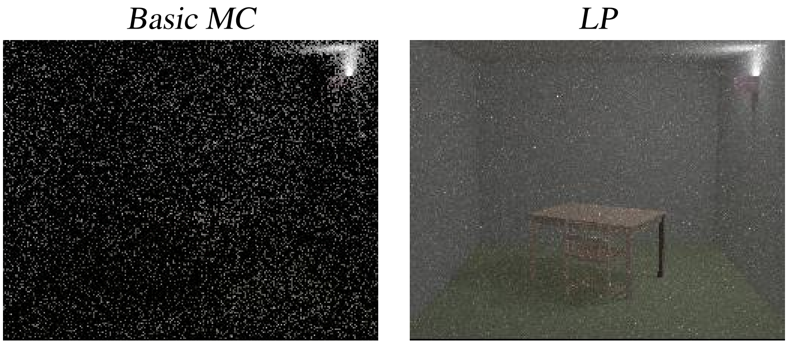


FIGURE 5.15 Room with indirect illumination. Both images computed with the same number of samples per pixel (250). Image with standard path tracing took 6001 seconds, and image with LP approach took 11034 seconds.

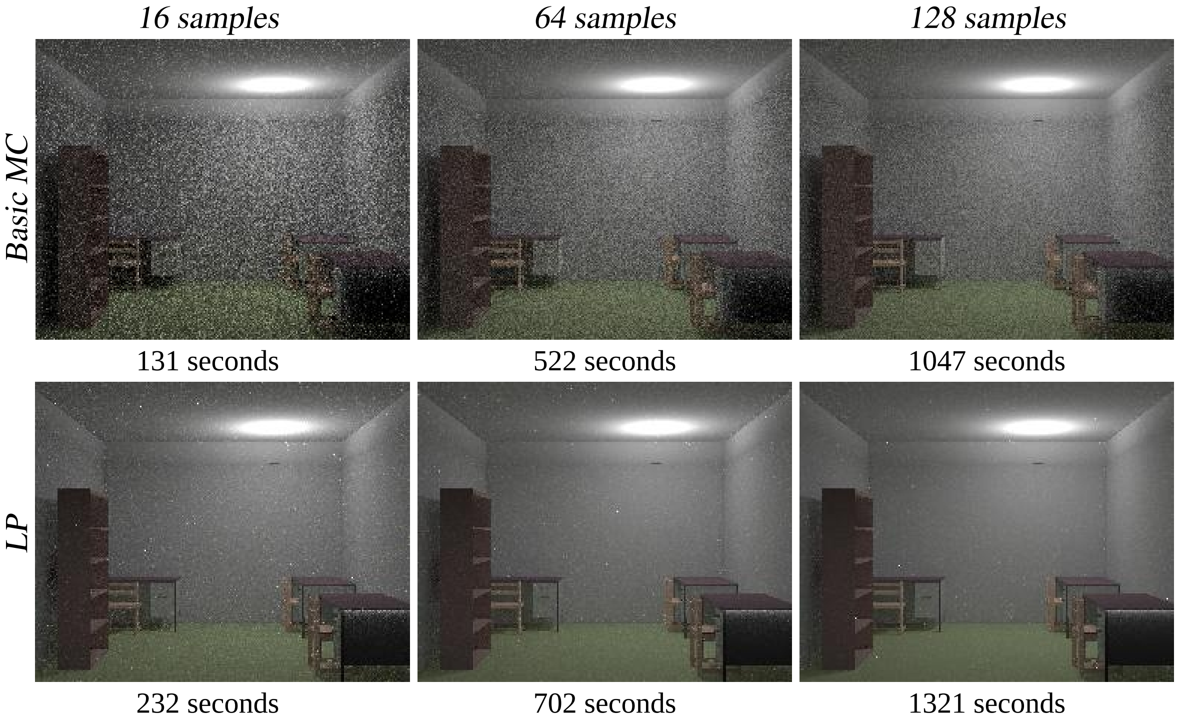


FIGURE 5.16 Office room with indirect illumination.

time of both methods is linear. The plot on Figure 5.17 (right) shows the ratio between the running times of Figure 5.17 (left). The relative overhead is diminished as the number of samples per pixel grows, mainly because of the amortization of the time spend in the finite element stage, and also because of the PDF construction and update for the first bounce is performed only once per pixel.

Conclusions

The method presented in this section provides a great enhancement of the path tracing method by providing importance sampling based on the estimated irradiances. This sampling allows to follow light paths accurately and provides good results for very different lighting conditions. A very interesting property of the new method is that the quality of the images does not strongly depend on the lighting conditions. It means that the method can follow the right light paths even with scenes like the one used in Figure 5.15. For similar visual appearance of the images, the LP approach takes less time than the basic Monte Carlo path tracing, even though there is an overhead

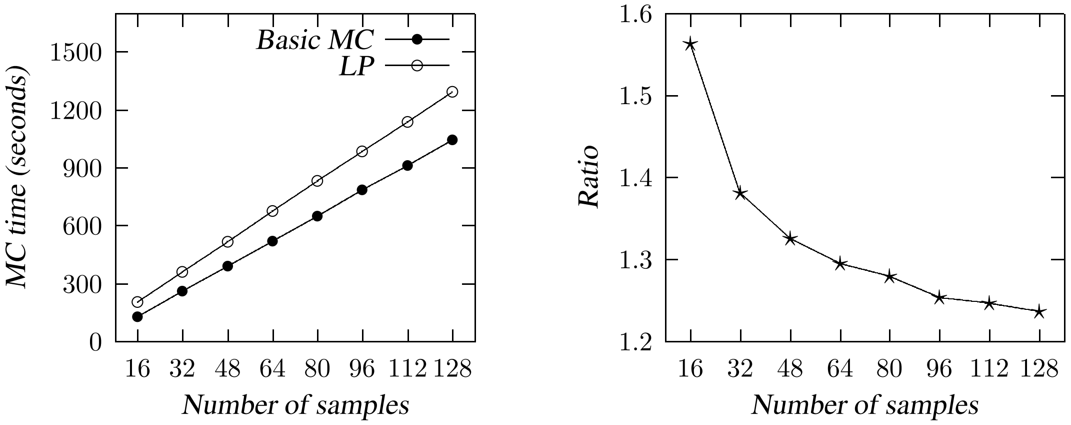


FIGURE 5.17 Left: Execution time of the basic Monte Carlo and LP for the office scene for different number of samples. Right: Ratio of execution times of plot in top.

in the LP approach due to the informed PDF construction and update.

5.3 Ray Tracing Assisted by Local Gather

This section presents our second second-pass algorithm: a modified ray tracing that uses the results of the first step to estimate indirect illumination. The principal components of the ray tracer with respect to the illumination model are introduced in what follows, starting with the discussion of the case of scenes in vacuum (i.e. composed only by surfaces). In those scenes the radiances reaching the eye are those from the visible surfaces, governed by the global illumination equation. We suppose that the BRDF ρ_{bd} can be expressed as a sum of a perfect diffuse part plus a glossy part. Then, the integral part of the global illumination equation (Equation 2.1),

$$L(x,\vec{\omega}_{\mathrm{o}}) = L_{\mathrm{e}}(x,\vec{\omega}_{\mathrm{o}}) + \int_{\Omega} \rho_{\mathrm{bd}}(x,\vec{\omega}_{\mathrm{o}},\vec{\omega}_{\mathrm{i}})\, L_{\mathrm{i}}(x,\vec{\omega}_{\mathrm{i}}) \cos\theta_{\mathrm{i}}\, d\sigma_{\vec{\omega}_{\mathrm{i}}} ,$$

can be split into three components: a perfectly specular part (L_{spec}), direct illumination (L_{dir}) and indirect illumination (L_{ind}). Dropping the parameters, we have:

$$L = L_{\mathrm{e}} + L_{\mathrm{dir}} + L_{\mathrm{spec}} + L_{\mathrm{ind}} .$$

The computation of these terms is discussed below.

Self-Emission. The self-emission L_{e} is directly available from the scene description, so it causes no problems.

Perfectly Specular Term. The L_{spec} component is computed by following the viewing path. A reflected ray is traced to retrieve the incoming energy at the interaction point. This energy is weighted accordingly by the specular fraction of ρ_{bd}.

Direct Illumination Term. The L_{dir} component can be computed for example by sampling the light sources directly. Alternatively, it can also be computed in conjunction with L_{ind} as explained below (our implementation offers both possibilities).

Indirect Illumination Term. The computation of the L_{ind} component is more laborious than the previous terms. L_{ind} is estimated by using the solution of the first step. In our implementation, we have used the HMCR algorithm of Section 4.3 as a first pass algorithm to provide the rough solution for the ray tracer. A straightforward approach is to use the results of the HMCR pass directly, or perhaps after Gouraud shading, as Bekaert et al. do [14]. However, this leads to disturbing artifacts like over estimating lighting in shadow areas and light leaks. A better—though more costly—solution is to use the HMCR results to estimate L_{ind} (or $L_{\text{dir}} + L_{\text{ind}}$) with a gathering procedure, by means of Link Probabilities to accelerate its convergence (Section 5.2.3), using the sets of links stored in the first pass. This is a gathering process similar to other final gathering schemes. Our final gather procedure differs from others by the fact that we set a different PDF for each viewed point (not for patches) and they are adaptively modified to better match the incoming distribution of energy (to alleviate visibility problems—see Section 5.2.3). This gathering could be extended to be a Monte Carlo path tracing (Section 5.2), but if some bias can be admitted, accurate images can be obtained. In order to do this gathering, a Link Probabilities instance is constructed from the *pushed* links arriving at the leaf elements related to the interaction points. These Link Probabilities are then sampled to estimate the irradiance at the interaction points to finally obtain a better estimate of L_{ind}.

To be able to use Link Probabilities in the second pass, links can be explicitly stored at receiver elements in the HMCR step. This is done in the refinement procedure, concretely when the levels of interaction between two points are established. Upon completion of the HMCR pass, the links—that are stored at different levels of the hierarchies—are pushed down to the leaves. In this way the links are available for quick access in the construction of Link Probabilities at the target points. However, this is not a good strategy. Note that the stochastic nature of the HMCR step gives a chance to miss links, mostly when dealing with very small patches (of tiny objects, for example). This leads to underestimation of irradiance, and also discontinuities in the illumination function across boundaries of patches. This is fixed by delaying the link construction to the rendering phase—the HMCR step not storing links at all. Now, in the rendering pass, for a gathering point x, the links are reconstructed by refining the scene against x, perhaps subdividing patches in this process. We use the same refinement schema as the refinement of HMCR, taking into account that now we do not consider element-to-element refinement but point-to-element refinement. This refinement thus establishes the set of links that serves as a basis for the Link Probabilities objects. Finally, the Link Probabilities are sampled to estimate the irradiance at the interaction points to obtain a better estimate of L_{ind} at last.

An example of the result of the refinement procedure is depicted in Figure 5.18, for the room scene of Figure 4.6. Yellowish lines represent links with higher probability in the related Link Probabilities object (see Section 5.2.3). As expected, the links with highest probabilities for the receiving point on the wall are those related to the visible light source. It should be noticed that delaying the link construction to the second pass saves computation time of the HMCR step because it does not have to collect links. In our room scene example, this represents the 4.4% of the time of the first pass. Unfortunately, the refinement needed per gathering point increments the time of the second step. (This can be alleviated by discarding the elements with very low illumination, with the consequence of more biased results.) Notice that the refinement is mandatory to fix the problems of under estimated illumination and discontinuities in the illumination between patches. This is shown in Figure 5.19, where we have rendered images for a zoom on a corner of the table of the room scene, collecting links in the HMCR step and delaying link construction to the second step, using the refinement per gathering point. We can see, in false-color, the zones of the images that differ the most. The mentioned discontinuities can be easily seen at the lines joining certain patches, due to the fact that missed different links. Also it can be seen that using the refinement per gathering point, the penumbra area has been improved substantially.

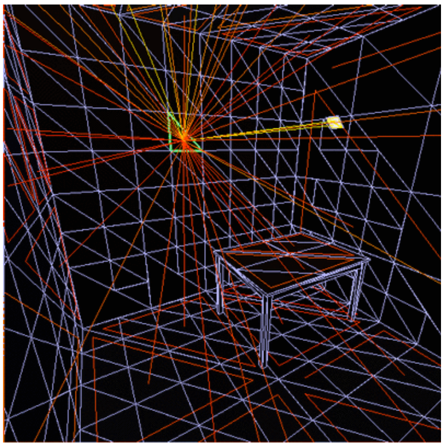


FIGURE 5.18 Mesh obtained by the execution of the HMCR step for the room scene, plus links established for an example point after refinement.

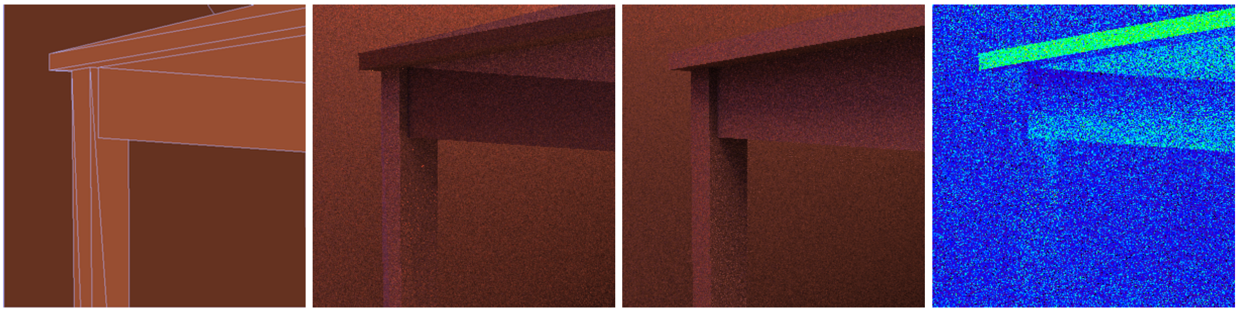


FIGURE 5.19 Left: Zoom of the table corner, showing the mesh. Middle Left: Result of the rendering step using links collected in the HMCR step. Middle Right: Result of the rendering step using links established by the refinement procedure. Right: Differences in false-color.

For scenes including participating media, care must be taken because of the transmittances through the media and the source radiances inside the media. The rendering step must then solve the *integral transport equation* (Equation 2.7 and Figure 2.2):

$$L(x) = \underbrace{\tau(x_0,x)\,L(x_0)}_{L_{\mathrm{ri}}(x)} + \underbrace{\int_{x_0}^{x} \tau(u,x)\,\kappa_{\mathrm{t}}(u)\,J(u)\,du}_{L_{\mathrm{m}}(x)} ,$$

The radiance at the background surface, $L(x_0)$, is estimated as explained above. To estimate $L_{\mathrm{m}}(x)$, a set of *interaction points* is chosen along the line of sight (jittered samples along the pierced participating media). As in the case of scenes in vacuum, the estimates of the source radiances resulting from the first pass could be used directly to estimate L_{m}. However, if a more accurate image is required, a gathering strategy is used at those interaction points in the media, similar to the gathering for points in surfaces.

Results

We have tested our method with very different scenes to examine its behavior. We started by testing the effects of progressively using a higher number of samples, followed by two scenes including participating media with diffuse surfaces, and finally we have used scenes in vacuum with surfaces having different material properties.

In Figure 5.20 we show a set of images obtained by progressively doubling the number of samples in the second pass for the room scene. Obviously, it can be seen that the noise is reduced as long as more samples are used. The timings of this set of images are shown in a logarithmic plot in Figure 5.21, with the image resolution fixed to 400 × 400 pixels. It can be observed that the rendering time is not doubled as the number of samples is doubled (this hypothetically doubling in the rendering time is shown in dashed lines). This is due to the refinement procedure made for the gathering points, and to the construction of the related Link Probabilities object. When using more and more samples, this cost is amortized, so the timing curve approaches the doubling of time.

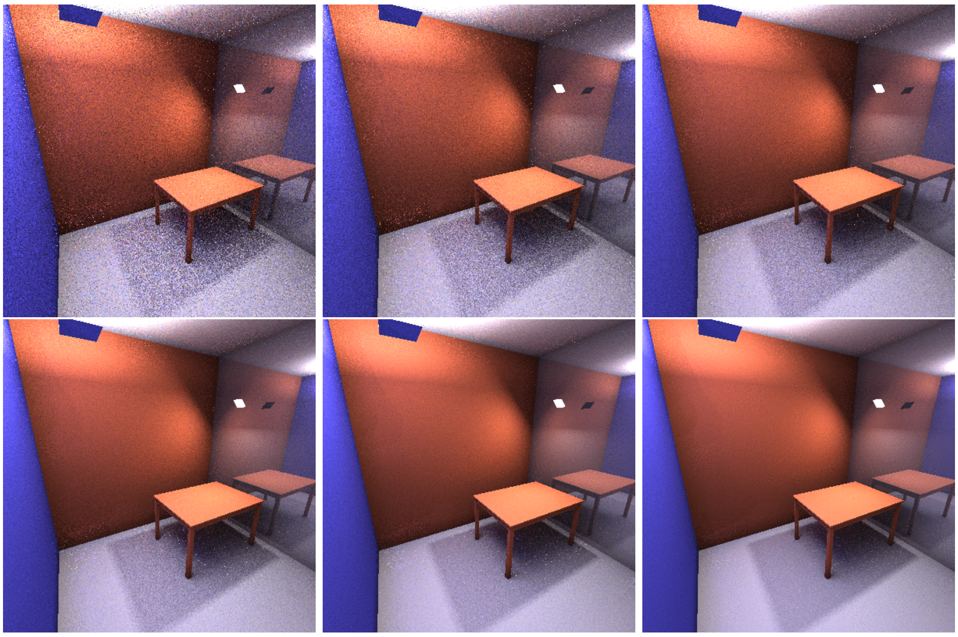


FIGURE 5.20 (From left to right and from top to bottom) Results of the second step using 4, 8, 16, 32, 64 and 128 samples per pixel for the room scene.

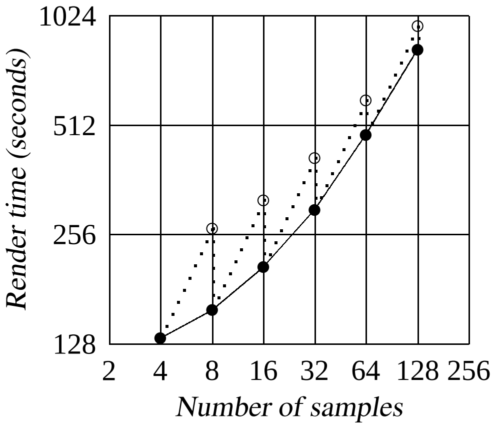


FIGURE 5.21 Rendering time per number of samples.

Our first scene incorporating a participating medium is a very simple scene with a cube lying on a floor. This setting is directly illuminated by three colored (red, green and blue) light sources focusing the cube, disposed in a way that different color combinations result. Images of a resolution of 200×200 have been computed in vacuum and in presence of a participating medium (Figure 5.22). For the scene in vacuum the HMCR step took 25 seconds, and the rendering step—using 24 gathering samples at surface hits—took 42 seconds. With the addition of the participating medium, the HMCR step raised up to 35 seconds (obviously because of the interaction of light inside the medium and between the surfaces and the medium), whilst the rendering step—with 6 samples at surface interaction points and 9 samples at volume interaction points—needed 1046 seconds. The rendering time is very high because of the high number of radiance estimates that are computed at the volume, and still some noise is very perceptible.

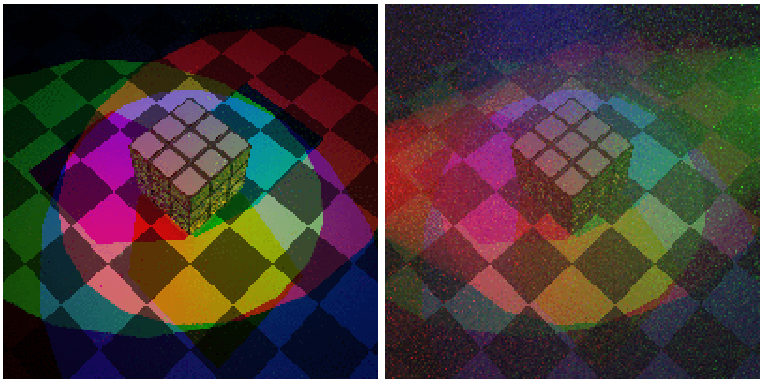


FIGURE 5.22 An illuminated cube in vacuum and within a participating medium.

The next scene allows us to observe more precisely the light interactions between surfaces and media. This scene contains a box of participating media, a pair of cubes and a wall, all floating in space (Figure 5.23). The HMCR step was executed in 7 seconds, obtaining the meshes showed on the left of Figure 5.23. Notice also the links arriving at a triangle on the right cube: illumination comes directly from the (unseen) light source and from the participating medium. The rendering step computed an image—with a size of 200×200 pixels—in 242 seconds, using 8 samples to gather illumination at surface points and 8 at volume points (right image of Figure 5.23). This image shows clearly the effect of the transmittance through the medium on the wall, plus the shadows both in the medium and on the wall caused by the cubes. Also the color bleeding from the medium on face of the cube on the left is observed.

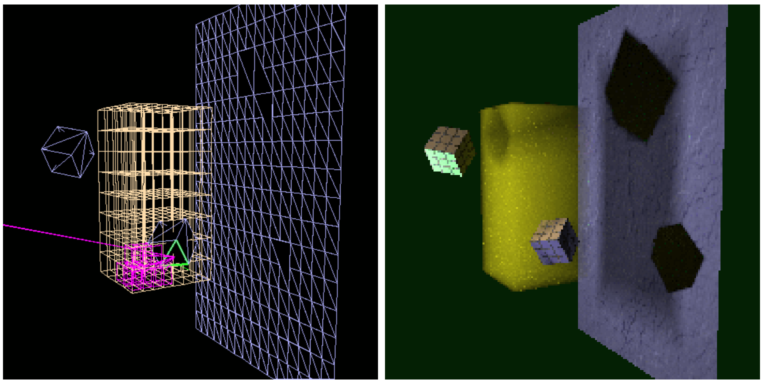


FIGURE 5.23 A box of participating medium. Left: Mesh and example links after the HMCR step. Right: Result of the rendering step.

For the next test we used another office scene, that contains specular surfaces (attenuated mirrors on a wall, and at the door), and diffuse plus specular components of the BRDF for the top of a table. Notice that most of the illumination comes to the ceiling, since there is only a single lamp pointing upwards. Running the HMCR algorithm takes 19.25 seconds for a rough estimate of the global illumination. (Previously, though, when collecting links in the HMCR pass, we were forced to shoot more lines to get also an acceptable set of links, so it took 67 seconds.) The resulting radiances of the patches are also shown on the left of Figure 5.24. Notice again that this solution (or after applying Gouraud shading) does not show illumination features like shadows close to the bottom of the table's legs, etc. However, after executing the rendering step gathering energy at interaction points, those features are captured, as shown on the middle of Figure 5.24 (built in 551.6 seconds). (When collecting links in the first pass, though, the second step only took 461 seconds, but some areas—mostly small patches on chairs—had underestimated illumination, which is fixed now.) For this image, of a resolution of 400×400 pixels, 16 samples have been used for estimating $L_{\mathrm{dir}} + L_{\mathrm{ind}}$. Some noise is still perceptible, for instance in Figure 5.24 (right), at the corner made by the two walls and at the region of the wall close to the desk. This noise was more acute when collecting links in the first pass. Again, this undesirable effect can be reduced by using more samples at gathering points, especially at critical points, instead of using a fixed number of samples. The noisy points can be detected for instance by controlling the variance of the estimation. Also, considering the visibility in the refinement procedure for the gathering process would help substantially.

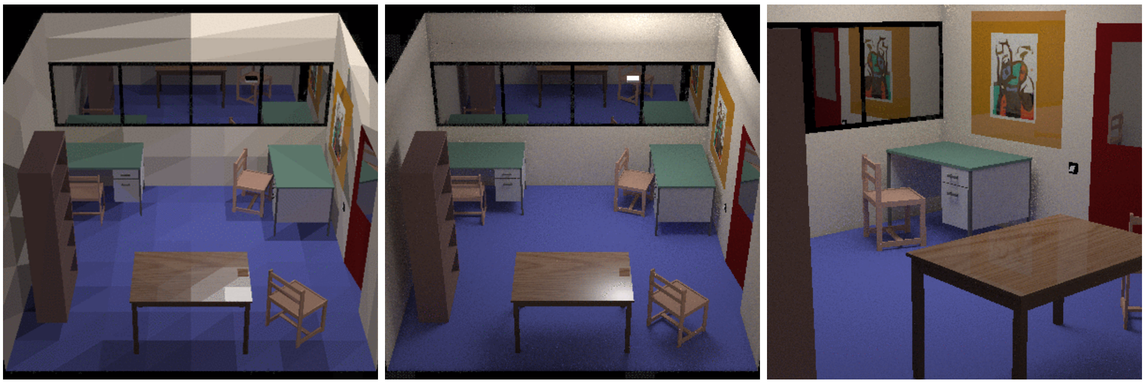


FIGURE 5.24 Office room with indirect illumination. Left: The result of the HMCR step. Middle: Solution after the second pass. Right: Solution after the second pass for a second view.

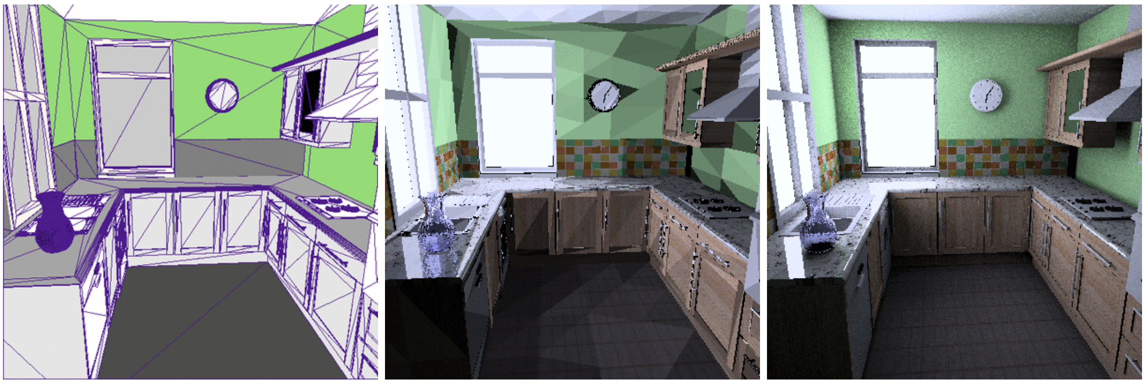


FIGURE 5.25 The global illumination is roughly computed for a kitchen model (left—courtesy of LightWork Design) with an extension of the Hierarchical Monte Carlo Radiosity algorithm (middle—shown flat-shaded). Finally, with the use of our local gathering procedure, we obtain a high quality image (right).

Finally, we have tested our algorithm with a more complex model, a kitchen scene (shown on the left of Figure 5.25). This scene has two large windows that act as diffuse light sources. The extended HMCR step took 12 seconds to compute a rough solution of the global illumination (middle of Figure 5.25). Notice that currently our implementation of the HMCR pass does not deal with clusters for light exchange, and thus there are patches that have not received any hit (so they have no energy), for example in the vase. However, this solution is still usable by the rendering step, that spent 1651 seconds to compute an image of 400 × 400 pixels, with 64 samples per pixel. Two more views, that took 1794 and 1891 seconds respectively, are shown in Figure 5.26.

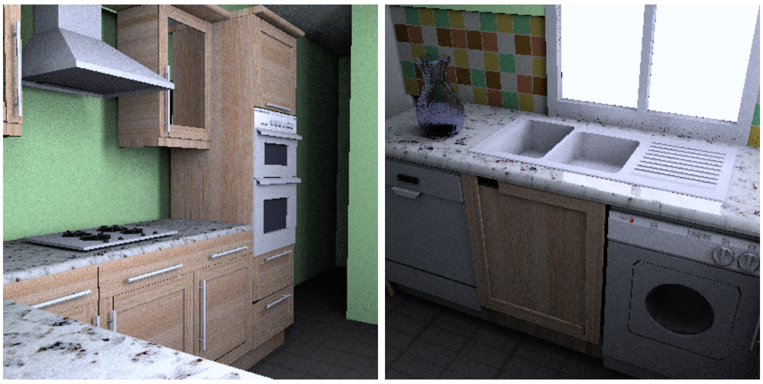


FIGURE 5.26 Two more views of the kitchen, after the second step.

Conclusions

A second final rendering step has been developed that constructs a high quality image from the HMCR based solution, using importance sampling, by means of Link Probabilities. This method provides good results for very different lighting conditions.

Chapter 6

Progressive Radiance Computation

The exhaustive rendering—computing the radiance for every single pixel—of a large image of a scene accounting for global illumination can still be extremely time consuming. The concrete magnitude of the rendering time depends on the input scene model and the complexity of the illumination paths that the rendering algorithm used is able to deal with. In our case, the construction of the Link Probabilities has a significant cost during the second pass. High rendering times are the reason why, in the design process for creating a high quality image, low resolution images can be generated iteratively after every time the designer changes the model—instead of generating high resolution images which would slow down dramatically the design process. Another possibility for a designer is the use of a *progressive computation*—also called progressive refinement or progressive sampling—scheme. Progressive computation is a means to obtain usable images in short time that are being refined gradually. The designer can then stop the rendering process at any time he or she understands the model needs further changes, or wants to vary the viewing parameters. The basic idea of progressive computation is to use a subset of samples across the image plane, instead of sampling each pixel individually, and keep growing this subset by the addition of well-chosen new samples. Although motivated to reduce the cost of the LPs construction by creating less LPs, the presented algorithms can be plugged to any other kind of global illumination method—it is effectively independent of the way radiances are computed. Different methods for progressive sampling can be found in the literature, and it is used in packages like RADIANCE [4,88], RenderPark [15] and Rayshade [5].

Actually, the goals of the progressive computation of an image are two-fold. On one hand, it is an aid to the designer as explained above. On the other hand, it allows to obtain a visually satisfactory image in far less computation time that would be required if the exhaustive rendering took place. Depending on the concrete application, possibly this visually satisfactory image (computed for example with only the 20% of the total number of pixels of the image) can be usable as a final image, with no need to wait for the computation of the radiances of the 100% of the pixels.

In this chapter we introduce the use of conductance maps to drive the setting of the sample points on the image plane to compute subsequent radiance values [119]. Conductance maps are functions that estimate the coherence of an image. We use conductance maps to aid selecting appropriate new sample points by identifying possible radiance discontinuities, in cooperation with a related triangulation whose vertices are the shaded pixels—the pixels with already computed radiance. Our two algorithms differ basically on how this triangulation is created. In the first algorithm, the *Straight Method*, it is composed of two triangles covering the rendering area. In the second algorithm, dubbed the *Regions Method*, the triangulation is the result of a segmentation process. The Straight Method is easier to implement than the Regions Method and it benefits from the use of conductance maps as will be shown. The Regions Method, although needs a small extra time for preprocessing, achieves even better results by establishing edges at boundaries. Both algorithms are efficient and easy to implement.

6.1 Previous Work

Progressive refinement for image generation has been used since the eighties. Bergman et al. [16] used a succession of stages, first displaying only vertices of the polygons, followed by displaying edges, then a sequence of improving shaded facets, and performing antialiasing at the end. Mitchell [100] decoupled the sampling process (where to locate the next sample) from the generation of the image samples themselves. Painter and Sloan [112] adaptively subdivided the image plane in their antialiased ray tracing, being the first who used recursive image subdivision for progressive refinement.

In the following subsections we quickly review the different strategies developed for progressive refinement. They can be grouped in those that use recursive image subdivision, those that use some kind of triangulation, and others.

6.1.1 Recursive Image Subdivision

Some progressive refinement approaches are based on recursive image subdivision. A first example is the work by Painter and Sloan [112] mentioned above, that uses a kd-tree. Maillot et al. [94] use a sequential probability ratio test to drive their sampling scheme, employing a quadtree. Guo [54] introduced the Directional Coherence Map (DCM), using oriented finite elements for interpolation. A DCM encodes the directional coherence in image space by means of a partition of the image with a quadtree, uses little memory, and achieves good results for discontinuities involving non-polygonal geometry or caused by secondary lighting. Lately Scheel et al. [142] have enhanced DCMs to handle thin objects and textures in an efficient way.

6.1.2 Triangulation Based

Pighin et al. [124] constructed a constrained Delaunay triangulation of the image plane that is updated with new radiance samples, for polyhedral scenes illuminated by point light sources. Notkin and Gotsman [111] and Reisman et al. [130, 131] also use a Delaunay triangulation together with its corresponding Voronoi diagram to look for the next sample in the progressive algorithm, based on a "farthest point strategy" [43]. In their parallel algorithm there is a preprocess in which they construct a *complexity map* of the image, that is used to distribute tasks among processors—it is not used to guide the adaptive sampling scheme. Also Simmons and Séquin [152] use Delaunay reconstruction of the image plane in their *tapestry* (a dynamic 3D triangle mesh) work, for interactive viewing. The main shortcoming of their approach is the lack of sharp edges at discontinuities. Unfortunately, ways to overcome this problem seem to be too costly and thereafter inadequate in an interactive system. Our proposed algorithms, as the one by Pighin et al. [124], are based on a preprocess which is acceptable for still images, that detects those sharp edges.

6.1.3 Others

A light-field like and an Image-Based approach are the *Holodeck Ray Cache* [183] and the *RenderCache* [178] respectively. Both are strategies devised to deal with camera or object movements (by reusing old ray samples), that is not our case—we deal with static images. One of the techniques to reconstruct images from the Holodeck Ray Cache uses Voronoi regions from the samples, constrained by the values of a depth buffer to respect silhouette edges; another technique uses a triangle mesh representation [183]. Conservative *interpolants* to approximate radiance [169, 11] have also been used. More information can be found in the review of the state of the art in interactive ray tracing by Wand and Slusallek [176].

Finally, in the context of the creation of impostors for real-time visualization of urban scenery, Sillion et al. extract a contour from a depth image of the distant scenery, to which a constrained Delaunay triangulation is applied [151]. This contour extraction is similar to the segmentation of the conductance maps that we perform in the Regions Method (Section 6.5), although for a completely different purpose.

Our two algorithms basically differ from the previous work by the fact that hard boundaries are detected as quickly as possible, during a preprocess. All methods do find these boundaries later, during the progressive computation, with the exception of the work by Pighin et al. [124], although in their work polygonal models and point light sources are required—this enforcement is not necessary in our algorithms, which set no restrictions on the input scene models. As stated above, similarities are also found in the Holodeck Ray Cache [183]. However, in our algorithms we take into account orientations and materials for establishing the conductance maps that will help in choosing sample points, whilst in the Holodeck Ray Cache only the depth is used to respect silhouettes. As a consequence of using more information, our algorithms guarantee a higher quality of the obtained results.

6.2 Overview

Using the information of conductance maps which we pre-compute for a certain view, our aim is to develop simple procedures to sample progressively the image plane with as low overhead both in memory and computation time as possible. Conductance maps are presented in detail in Section 6.3.

Our algorithms are based on a constrained Delaunay triangulation of the image plane, for which we use the Open Source GTS library [2]. As in the work by Pighin et al. [124], we select the next location in the sequence of sampling along an edge of the triangulation (concretely, from the edge that has the highest error measure). Another possibility would be to consider the whole area of each triangle, but since our objective is to construct a simple algorithm, we rely on the edges.

The basics of the two methods presented in this chapter are as follows. During a preprocess, a set of false-color images is computed for the selected view by a modified ray tracing procedure. These false-color images convey information on orientation, depth and materials for the points projected onto each pixel of the image. Conductance maps for the horizontal and vertical axes are then computed from the false-color images. Our two proposed techniques basically differ on how an initial progressive Delaunay triangulation is set and managed thereafter:

- *Straight Method.* The progressive triangulation is set by shading and connecting the four vertices of the image by means of two triangles. The method iterates by choosing the edge with highest error measure to set the next sample location, which will be shaded and added to the triangulation. The conductance maps computed during the preprocess play a key role in the setting of the error measure for edges.
- *Regions Method.* From the conductance maps a segmentation of the image plane is performed, which is used to establish an initial triangulation. The procedure iterates in a way similar to the Straight Method.

Differently from some other previous methods, we do not deal with interactive ray tracers (although we potentially could) but with a global illumination solver. Concretely we use the extension of Hierarchical Monte Carlo Radiosity as a preprocess (Section 4.3), and with a final gathering step (Section 5.2). In this case, the cost of a sample eye radiance is much higher than that of a simple interactive ray tracer because our rendering system takes into account global illumination effects—more realistic images are obtained. Also our proposal is able to deal not only with the usual perspective projection, but with any kind of projection: panoramic, cylindrical, omnimax, spherical, etc.

6.3 Conductance Maps

We have modeled a very simple room scene for explanation purposes in this and following sections. The model of this scene is shown in Figure 6.1, being a simple room scene with a table in it, containing both direct (lamp on the right) and indirect illumination (unseen lamp on the left, pointing upwards).

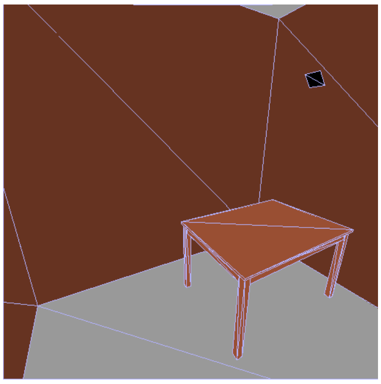


FIGURE 6.1 A very simple room scene.

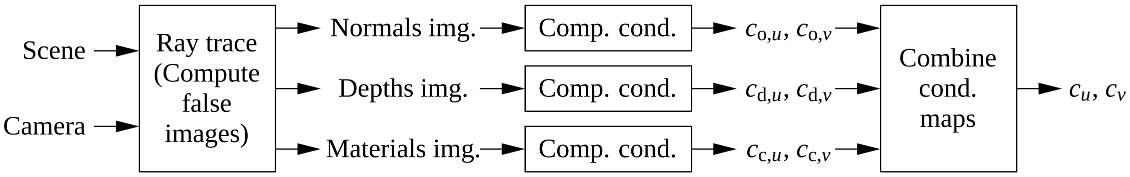


FIGURE 6.2 Common preprocess in Straight Method and Regions Method.

There is a common preprocess in the Straight Method and the Regions Method to obtain the conductance maps c_u and c_v—for the horizontal and the vertical directions—from a certain view. This preprocess is depicted in Figure 6.2. Conductance maps (or conductance functions) are maps that estimate the coherence of an image. A conductance function is established in the domain of the pixels of the image and ranges between zero and one. They are used in nonlinear anisotropic diffusion processes, basically measuring how well certain data is spread when filtered. We refer to McCool for a nice detailed explanation of conductance functions and related aspects in the context of anisotropic diffusion for Monte Carlo noise reduction [97]. In the preprocess, three false-color images are created, all three resulting from a single execution of a ray tracing for the selected view:

- Normals image: It stores the value of the normal at the first intersected point for the primary direction related to each pixel.
- Depths image: It records the distance from the viewpoint to the first intersected point, for each pixel.

- Materials image: Each pixel stores the result of ray tracing the scene with neither light nor shadow rays, and considering the BRDF as the source of the color at the intersection points.

From each of these false-color images a pair of conductance maps (for the horizontal and vertical directions) is obtained: the orientation conductance maps $c_{\mathrm{o},u}$ and $c_{\mathrm{o},v}$ from the normals image, the depth conductance maps $c_{\mathrm{d},u}$ and $c_{\mathrm{d},v}$ from the depths image, and the color conductance maps $c_{\mathrm{c},u}$ and $c_{\mathrm{c},v}$ from the materials image. In Figure 6.3 we show the set of conductance maps for our example scene. Finally, these conductance maps are combined (by multiplying them) to obtain the final conductance maps c_u and c_v. Figure 6.4 shows our concrete conductance maps for the example scene.

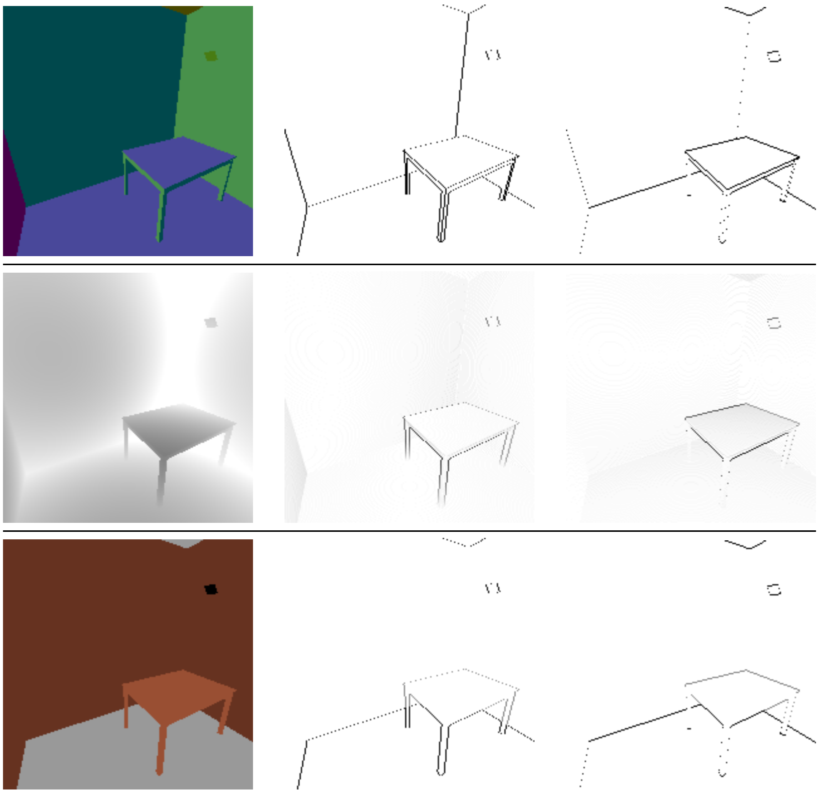


FIGURE 6.3 False-color images and related conductance maps. Top: Normals image and orientation conductance maps for u and v directions. Middle: Depths image and depth conductance maps. Bottom: Materials image and color conductance maps.

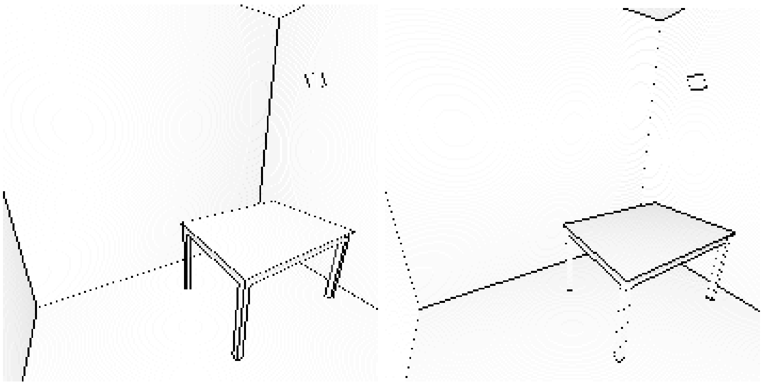


FIGURE 6.4 Final conductance maps for the u and v directions (c_u and c_v).

6.4 Straight Method: Four Initial Vertices

The Straight Method starts by considering the four corners (vertices) of the image that define an initial triangulation built with only two triangles. As commented previously, to select the next pixel to be shaded and included in the triangulation, we establish an error measure (or priority measure) for each edge of the triangulation. A priority queue for the triangulation with edges sorted by the error measure is used to select the edge where the next sample will lie. One of the components in the definition of the error measure of edge e is the *conductance* of e, or e_{cond}. The value of e_{cond} is set by traversing the pixels pierced by e, by means of a Bresenham procedure, and multiplying the precomputed conductances c_u and c_v at the related inter-pixel locations. Figure 6.5 depicts two example scenarios for the computation of edge conductances. In the first case the edge traverses a set of pixels never passing through a pixel corner. We set e_{cond} as the product of the "pierced conductances", from neighbor pixels, that happen to be horizontal or vertical conductances only. In the second case, though, the edge passes through a pixel corner. Since we do not dispose of diagonal conductances, these are estimated by taking the minimum of the product of the vertical/horizontal conductances of the two possible stepwise paths. In the example case of Figure 6.5, that would be the minimum of the paths E+S (going to the East and then to the South) and S+E.

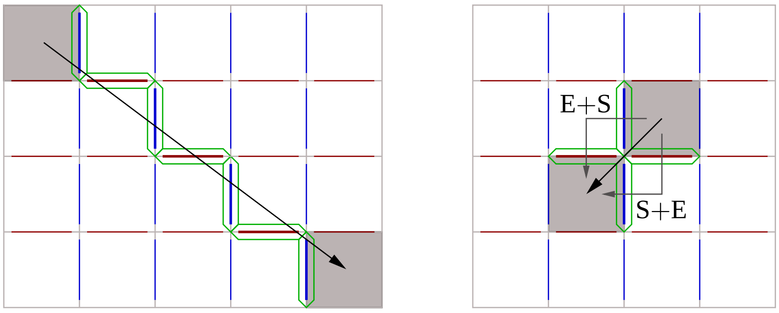


FIGURE 6.5 Two example cases for the computation of edge conductances. Blue interpixel segments represent c_u, whereas red interpixel segments represent c_v.

The error measure for an edge e is computed simply by taking into account the length of e (concretely the square of its length, $e_{\text{lengthSqr}}$), the difference in color of the radiance samples of the vertices of e ($e_{\text{colorDiff}}$—for which we use the distance of the radiances converted to the L_{ab} color space), and the conductance of e. This is done to steer the refinement process to split larger regions before smaller ones (as in the work by Painter [112]) but also taking care of color differences and possible radiance discontinuities along the edges. In our concrete implementation we use the following heuristic expression: $e_{\text{measure}} = (1. - e_{\text{cond}}) \times e_{\text{lengthSqr}} \times (e_{\text{colorDiff}} + 1)$. Notice that probably better and more complex strategies can be devised, though, like taking into account contrast [100] and other perceptual issues (masking effects, etc.—see Section 7.3.2).

The priority queue sorted by the edge error measured is maintained during the execution of the progressive refinement procedure. In order to select the edge with maximum error, the top of this queue is retrieved. Then, the location where that edge will be split still has to be computed. A possibility would be choosing the edge midpoint pixel, as Pighin et al. do [124]. However, we take into account the values of the conductance maps and establish the next sampling point to be the mean of the edge midpoint pixel and the pixel sharing the minimum conductance. Experimentally this turned to perform better than simply selecting the pixel next to the minimum conductance. The triangulation is updated with the new vertex, and normally new edges will appear. Error measures are computed for these newly created edges, updating the edge priority queue accordingly.

There are two possibilities to reconstruct an image from a set of samples: Reconstructing from samples via filtering, and interpolating between samples. Since we display the progressive image using the Gimp Toolkit (GTK) with OpenGL [187], for simplicity, we have opted for the interpolation scheme using the capabilities of OpenGL.

In order to be able to distinguish the pixels and the conductance maps, we have executed the progressive radiance algorithm for a very small image, with a resolution of 60×60 pixels, for the simple room scene. In Figure 6.6 we see the evolution of the computed image and the triangulation for the 2%, 4%, 8%, 16% and 32% of the total of pixels computed. As can be seen, the progression converges to a solution covering all the pixels. It can be noticed that the triangulation is gradually adapted to try to follow the edges of the room and of the table, because of the use of the conductance maps for the error measure.

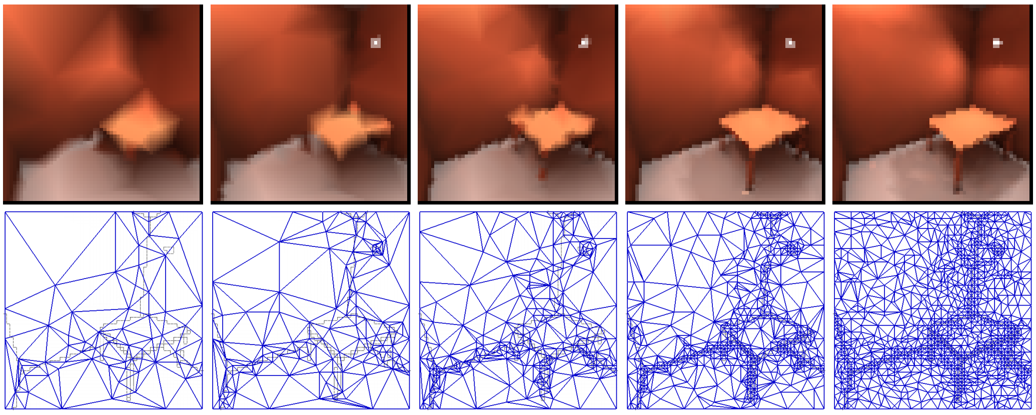


FIGURE 6.6 Progressive series for the simple room scene (Straight Method).

We have also tested the Straight Method with a more complex scene: a kitchen environment (courtesy and copyright of Goods, S.L.) made up of surfaces of different materials. In Figure 6.7 the model is shown, together with the combined conductance maps for the horizontal and the vertical directions. Notice that the conductance maps have detected discontinuities within the area of the oven, which is modeled by a texture. Also in Figure 6.7 an exhaustive rendering—the radiance has been computed for all the pixels—for the chosen view is shown. This rendered image of a size of 400×400 pixels was obtained by using 128 gathering rays at each hit point, after the hierarchical Monte Carlo step. For this image, each pixel was sampled (100% of sampling coverage), and took 2731.21 seconds to be completed.

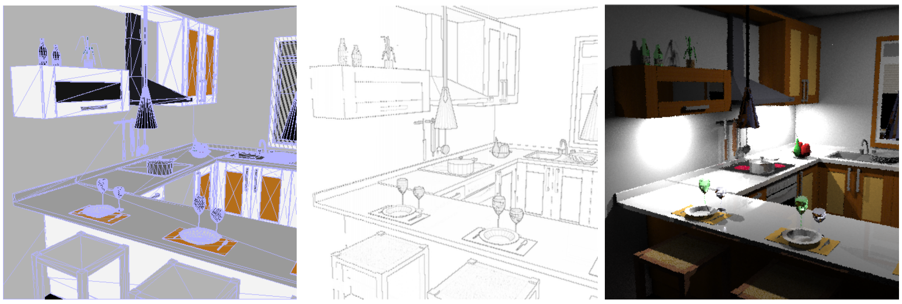


FIGURE 6.7 A kitchen model (left—courtesy and copyright of Goods, S.L.), combined *uv* conductances (middle), and complete rendering (right).

Executing our algorithm for this scene using the view of Figure 6.7, we obtain images that are progressively of better quality, as expected. For example, images of the same resolution (400×400 pixels) for the first 5% and 10% of the pixels sampled are shown in Figure 6.8 (on top), accompanied by their respective triangulations (at the bottom). The computation of these images took 308.98 and 345.53 second for the 5% and 10% of the pixels sampled respectively. Notice that since we do not use any kind of perceptual measure, the darkest portion of the images of this view (bottom left part of the rendered images) is quite densely subdivided (as can be seen in the images of the meshes produced), although their effect is not important.

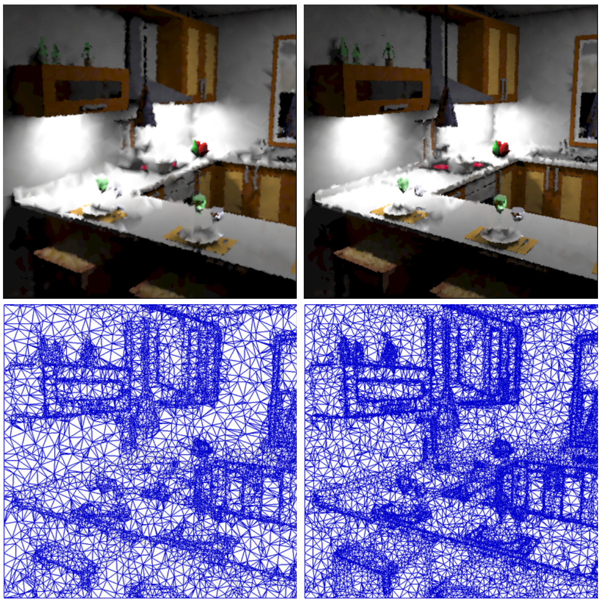


FIGURE 6.8 Snapshots of the evolutionary images at 5% and at 10% of the total of image pixels computed using the Straight Method, and the obtained triangulations.

6.5 Regions Method: Segmented Conductance Map

The use of different regions can enhance the visual impact of the images at early stages. The idea is to use the conductance maps obtained in the common preprocess depicted in Figure 6.2 to segment the image in a set of regions (connected sets of pixels). The limits established by the conductance maps are related with potential radiance discontinuities. Thus, the illumination related to the different regions on the image plane is estimated independently. A set of sparse samples can interpolate the radiance of a region if the radiance function varies smoothly in its interior—similarly to the *interpolants* of Bala et al. [11]. Regions where the radiance function varies more rapidly should be sampled more densely.

It is customary to categorize the segmentation process into three types: segmentation based on *global* knowledge concerning pixel intensities, *region-based* segmentation (“growing” a region while neighboring pixels are similar) and *edge-based* segmentation (using edge detection processes) [184]. In our case, we have two conductance map images (for the *u* and *v* directions), with values ranging from zero to one. Thus, we perform a kind of *gray scale thresholding* segmentation (a region-based segmentation) where conductances below a certain small threshold denote edges. As schematically shown in Figure 6.9, we implemented “contour encoding” (using intermediate *chain codes* to represent boundaries) [51, 184]. For simplicity and robustness, we segregate the possible appendices—outer lines with no related area—of the chains. After converting each chain to a polygonal representation, the resulting contours are used as constraints to create an initial single triangulation. Also the vertices of those polygons are inserted into a priority queue (different from the priority queue for edges, which is also used by the Regions Method, with the same edge error measure as in the Straight Method). The priority queue for vertices sorts its entries with a measure of the area associated to each vertex, since in principle the vertices with related greater areas will have a more impact in the rendering of the image when shaded (and after interpolating). Notice that later, in the progressive radiance computation, vertices will be picked to be shaded in the order established in this priority queue. Figure 6.10 shows the segmentation for a low resolution image (result of the "Compute contours" black box of Figure 6.9), and the related first initial triangulation for our room example scene.

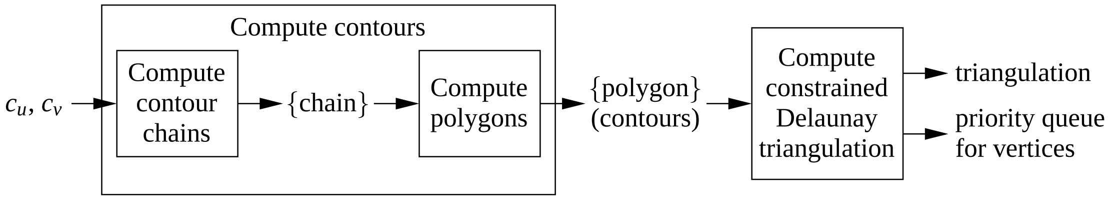


FIGURE 6.9 Further preprocess for the Regions Method.

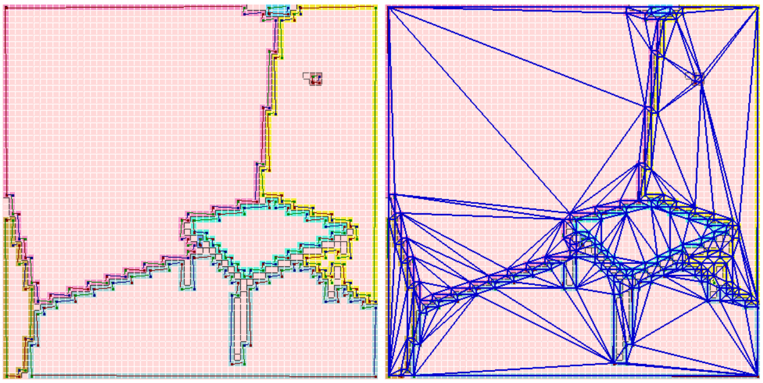


FIGURE 6.10 Contours of the regions computed for the room scene (left) and related triangulation (right).

The vertices resulting from the preprocess can be shaded in a row, and then generate the error measures for the edges of the triangulation to create the priority queue for edges. Alternatively, after some of the initial vertices have been shaded, probably shading points within edges can have more impact in the rendered image. We have opted to implement this second case. In order to do so, notice that only when the two vertices related to an edge have been shaded that edge (with its error measure) can be inserted into the priority queue for edges. After a percentage of the initial vertices have been shaded, while there are initial vertices to be shaded, our concrete implementation consecutively shades a vertex of the initial triangulation or a point within an edge with shaded endpoints. To show the interpolated image to the user, for those of the initial vertices that have not been shaded a fake color is computed combining the colors of neighbor vertices and the conductance of the related edges.

As for the Straight Method case, we have executed the progressive radiance algorithm for the simple room scene with low resolution, now for the Regions Method. This is shown in Figure 6.11 with both the partial computed images and their related triangulations for the 2%, 4%, 8%, 16% and 32% of the total of pixels computed. As expected, the edges "appear" earlier in this series than in the Straight Method (compare with Figure 6.6), due to the establishment of vertices around the different regions.

We have also applied the Regions Method to the kitchen model shown in Figure 6.7, with the same percentages of pixels computed as in the Straight Method. The resulting images, together

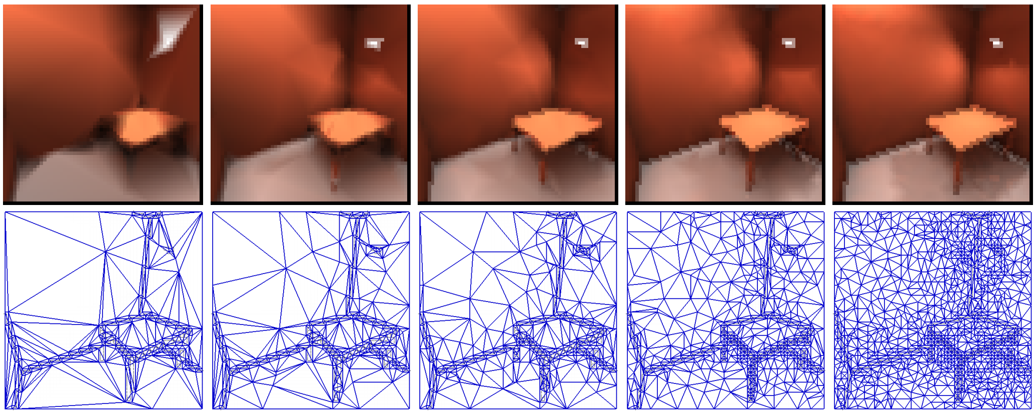


FIGURE 6.11 Progressive series for the simple room scene (Regions Method).

with their corresponding triangulations, are shown in Figure 6.12. Comparing these results with Figure 6.8 it is clear again that the Regions Method outperforms the Straight Method with respect to the images obtained (however, the Straight Method is far easier to implement than the Regions Method).

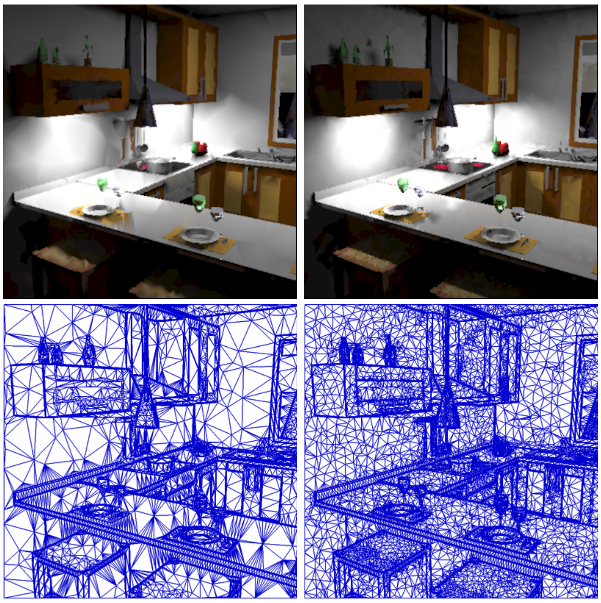


FIGURE 6.12 Snapshots of the evolutionary images at 5% and at 10% of the total of image pixels computed using the Regions Method, and the obtained triangulations.

Table 6.1 gives the timings for the kitchen scene, for both the Straight and the Regions Method. All timings in this chapter are on a PC Linux SuSe with a P-IV 1.80GHz processor. Notice that the further preprocess of the Regions Method took less than a pair of seconds, and that the management of the triangulation and related priority queues (denoted in Table 6.1 as "△ care") is not very high in any of the two methods, compared with the time spent in shading the selected pixels. The shading time for the same percentage of pixels computed can vary significantly for each method. This is due to the fact that they shade different pixels and the cost of shading a pixel across the image is not constant.

A set of three more executions completes this section. These executions focus on the ability of the algorithm to deal with different features like reflections (Figure 6.13), highly tessellated models (Figure 6.14) and translucent objects (Figure 6.15). These features will be shown to be

	Common	*Further*	*Shading*	*△ care*	*Total time*	*Shading*	*△ care*	*Total time*
	preprocess	*preprocess*	*time at 5%*	*at 5%*	*at 5%*	*time at 10%*	*at 10%*	*at 10%*
Straight Method	6.1s	–	147.47s	2.93s	150.40s	303.11s	5.87s	308.98s
Regions Method	6.1s	1.5s	175.59s	0.96s	176.55s	343.14s	2.39s	345.53s

TABLE 6.1 Timings for the comparison of Straight Method and Regions Method for the kitchen scene.

clearly captured by our algorithm in the initial stages of the progressive refinements—earlier than other methods thanks to the use of the conductance maps. Perceptually acceptable answers are obtained with about 10% or 20% of the pixels computed, depending on the scene and the view.

The same kitchen used in Figure 6.12 has been rendered from another viewpoint, roughly pointing towards a glass bottle, and showing part of a window. The results are shown in Figure 6.13. Notice that the portion of the reflection on the window and on the marble is detected by the conductance map, and is improved like the rest of the image. After 10% of the pixels computed, the related triangulation and rendered image are shown on the right of Figure 6.13.

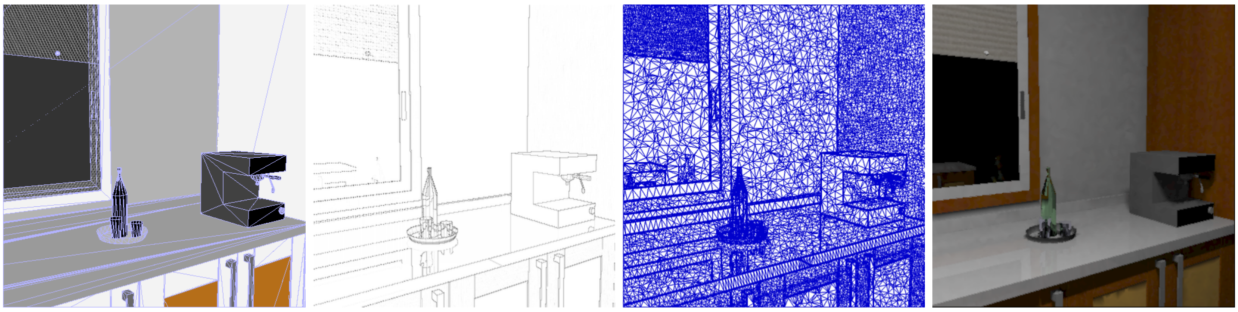


FIGURE 6.13 Another view of the kitchen model.

A rather complex model with curvilinear elements is the Jaguar car, shown in Figure 6.14. The conductance map for the chosen view catches the major discontinuity lines regardless of the fine tessellation of the model. Again 10% of the pixels have been computed for this view. Notice in the triangulation achieved at that point the detection of the discontinuity of the hard shadow due to the illumination of the scene by a small light source.

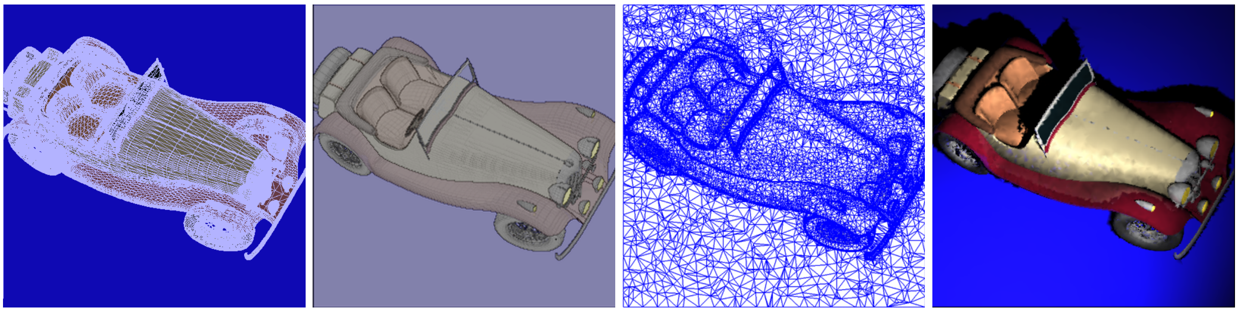


FIGURE 6.14 Rendering of a Jaguar (copyrighted by Helmut Schaub; downloaded from PlanIT 3D).

Finally, our last rendering is a close-up of a vase in a kitchen scene (courtesy and copyright of LightWork Design Ltd.) showing transparencies, shown in Figure 6.15. In this case, 20% of the pixels of the image have been computed. Due to the transparencies, the shading times are higher than in previous examples—rays have to be transmitted through the glass. It can be seen that the most important texture discontinuities of the marble have been detected in the conductance map.

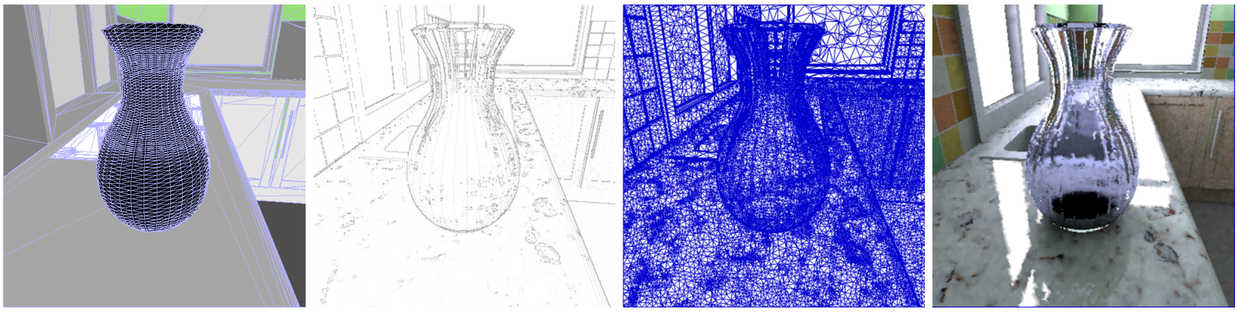


FIGURE 6.15 Rendering of a vase in a kitchen (courtesy and copyright of LightWork Design Ltd.).

Table 6.2 gives the most representative timings for the set of executions performed. The time spent in the common preprocess—for the generation of the conductance maps—is increased for each case due to the increased complexity of the related scenes (in the number of triangles of the Jaguar, or due to the optical properties of the vase in the kitchen). The management of the triangulation and other data structures again is not very high compared with the rendering times.

	Common preprocess	*Further preprocess*	*Shading time*	*△ care*	*Total time*
Kitchen detail	05.88s	1.30s	329.65s	3.23s	332.88s
Jaguar	14.40s	1.25s	412.13s	2.44s	414.57s
Vase in a kitchen	23.02s	1.28s	502.60s	5.92s	508.52s

TABLE 6.2 Timings of the renderings for the different scenes using the Regions Method.

Conclusions

In this chapter two algorithms to progressively compute radiance samples accounting for global illumination effects were presented, using a partition of the image plane and taking advantage of conductance maps that are computed in a preprocess. These conductance maps are computed from three false images that take care of orientations, distances and the BRDFs of the surfaces. This permits the progressive computation of images that are perceptually more accurate than those obtained by previous methods in earlier stages of the executions. The algorithms are easy to be implemented, and can be improved by means of a better specification of the edge measure.

Chapter 7

Conclusions and Future Work

We present in this chapter the conclusions and the main contributions of this thesis, and also some possible directions for future research.

7.1 Conclusions

In this thesis we have focused the development of algorithms for the simulation of the light transport in general environments to render high quality still images. After reviewing briefly the main equations modeling the light transport in general environments (Chapter 2), we have analyzed the existing methods able to render participating media, both for the single and the multiple scattering cases (Chapter 3). The rest of the thesis introduced new rendering algorithms, concretely a couple of two-pass methods (Chapters 4 and 5) and also a pair of algorithms aimed at computing progressively the final image (Chapter 6). For the first step we have proposed algorithms to cope with general environments, which produce low resolution solutions. The second pass uses the coarse solution of the first step to obtain a final high quality image, by means of a so-called radiance reconstruction, using in its core a final gathering scheme. Finally, we focused progressive computation—also called progressive refinement or progressive sampling—because exhaustive rendering of all the pixels of an image has a high cost when solving the global illumination problem. The idea of progressive computation is attractive since it is a means to obtain usable images in short time that are refined gradually. Two methods have been presented following this philosophy.

We summarize below the algorithms proposed in this dissertation:

- In Chapter 4 we first pointed out the relative benefits and drawbacks of hierarchical and stochastic algorithms. This draw the decision of establishing two pass algorithms for solving efficiently the global illumination problem, and devoted the rest of the chapter to the introduction of two hierarchical algorithms to be used as a first pass of a two pass algorithm. More specifically, two concrete algorithms based on hierarchies have been extended to be more generic:
  - *Hierarchical Radiosity with Clustering (HRC)*. This algorithm is based on two finite elements algorithms: a unified algorithm for the simulation of light transfer between diffuse surfaces, *isotropic* participating media and object clusters [147], and a hierarchical algorithm capable of dealing with non-diffuse surfaces [148]. In our approach, the global illumination of scenes including inhomogeneous *anisotropically* scattering media can be solved efficiently by means of the clustering strategy. We have identified the expressions needed to transport *radiant intensity* between all kinds of objects (surfaces, media and clusters) starting from the reflection and the scattering equations. In this case the resulting light flow in the

scene is represented by a hierarchy of links representing light transfer between the different types of objects.

- *Hierarchical Monte Carlo Radiosity (HMCR)* The HMCR algorithm by Bekaert et al. [14, 12] has been extended for surfaces with a combination of diffuse plus specular components, and for participating media.

These two first pass algorithms create in very short time an approximate representation of the light flow in the scene.

- Chapter 5 focuses second pass algorithms, which are responsible of obtaining the high quality final renderings, using the coarse global illumination resulting from the first pass. The algorithms employed are a Monte Carlo path tracing using next event estimation, and a ray tracer incorporating final gather. The kernel of the two proposed algorithms is a gathering procedure based on the construction of *probability density functions* (PDFs) at intersection points for importance sampling. These PDFs are based on links and are progressively adapted based on the visibility sampled during irradiance estimation. The use of this strategy drives to significant improvements with respect to naive techniques. Although this second pass has a computational cost much higher than the first pass, this is not necessarily an overwhelming problem, as discussed below.

- In Chapter 6 the literature related to progressive computation was reviewed before introducing the last two algorithms, which follow a progressive rendering approach. They are based on conductance maps, for the horizontal and vertical axes, obtained through a set of false-color images computed for the selected view by a modified ray tracing procedure. These false-color images convey information on orientation, depth and materials for the points projected onto each pixel of the image. The conductance maps are then used to establish and guide a progressive Delaunay triangulation of image space, for producing a sparse sampling of the image. Our two proposed techniques basically differ on how the initial triangulation is set and managed thereafter:

  - *Straight Method.* The progressive triangulation is set by shading and connecting the four vertices of the image by means of two triangles. The method iterates by choosing the edge with highest error measure to set the next sample location, which will be shaded and added to the triangulation. The conductance maps computed during the preprocess play a key role in the setting of the error measure for edges.
  - *Regions Method.* From the conductance maps a segmentation of the image plane is performed, which is used to establish an initial triangulation. The procedure iterates in a way similar to the Straight Method.

  As a result of this approach high quality images are generated with essentially one-tenth the computational cost of the complete solution.

7.2 Summary of Original Contributions

In this thesis we have made the following contributions:

- A study of single and multiple scattering methods, characterizing them by identifying their base techniques, assumptions, limitations and range of utilization.

- Two first pass methods to solve the global illumination problem using finite elements (based on Hierarchical Radiosity with Clustering) and Monte Carlo (based on Hierarchical Monte Carlo Radiosity). These methods are able to deal with scenes that are more complex than the methods they are based upon.

- A specific design for the construction of PDFs for importance sampling (Link Probabilities) based on the results of a first pass execution. Their adaptive nature makes them appropriate for their use in final gathering algorithms, obtaining better samples than fixed schemes.

- Two methods based on conductance maps for progressive radiance computation. The algorithms are able to deal successfully with features like reflections, highly tessellated models and translucent objects.

7.3 Future Research

7.3.1 Improved Two Pass Algorithm

We envisage dealing with the issues discussed in what follows, in order to achieve a more efficient two pass algorithm capable of dealing with a large range of possible input scenes.

Noise Reduction.

The noise due to the presence of participating media (e.g. Figure 5.22) can be reduced by devising a better way to sample radiance coming from the volume elements (using the solid angle measure instead of using the volume measure).

Better LP construction.

Currently the refinement procedure in the rendering step does not take into account visibility, relying on the adaptiveness of the Link Probabilities to handle occlusion. We should investigate the inclusion of the visibility in the refinement, in such a way that does not increase prohibitively the rendering time. Also we should consider the use of shaft culling for objects O with high illumination to quickly discard receivers that do not see O. For glossy (not perfect specular) BRDFs, the refinement should take into account the BRDF for the outgoing direction. For the estimation of the irradiance, the use of Quasi-Monte Carlo random sequences would improve its convergence.

Caustics and General Materials.

Caustics could be easily rendered by identifying the paths that cause them. In the HMCR pass, when bouncing off of glossy objects, the illumination caused by the related particles should be stored separately, for instance using a photon map (this would allow caustics on participating media) [71]. In the rendering step, when estimating L_{ind}, the caustic component could be added by retrieving it, with density estimation if using a photon map.

So far our implementation deals with surfaces with a combination of purely diffuse plus purely specular components, and for isotropic scattering media. In order to handle glossy surfaces and anisotropically scattering media, the directionality of the radiant properties has to be represented. The HMCR step could use Illumination Samples [164], storing them only for glossy/anisotropic objects. When one of these objects was chosen to shoot its unshot energy, the related Illumination

Samples would be combined with the BRDF or phase function to draw reflected/scattered directions. For the visualization of the results of the HMCR step the final set of Illumination Samples can be used to reconstruct eye radiances. Another possibility would be to construct directional distributions based on the Illumination Samples for faster rendering from different viewpoints.

Natural Lighting. We are actively working on an extension of the HMCR pass that integrates sunlight and skylight. Apart from allowing the rendering of scenes directly lit by natural sources, we consider illumination through light pipes.

7.3.2 Progressive Radiance Computation

The proposed algorithms for progressive radiance computation can be improved by devising better specifications of the edge measure. The kernel procedure can also be extended by incorporating more information to obtain error measures of edges. As stated in Chapter 6, for example it could be enhanced by including perceptual issues—e.g. [55] (see below). Further conductance maps could be considered, apart from the orientation, depth and color conductance maps. For example, in the case of scenes with point light sources, a conductance map based on shadow maps (or even perspective shadow maps [162]) could be included. Aliasing can be easily addressed for example by adaptive supersampling on the pixels identified by the conductance map. Another way to do antialiasing would be repeating the process a certain number of times (with different directions per pixel), and performing an average of the results. More complex possibilities can be studied.

Additionally, the following topics could be investigated: generation of stereo images, addition of participating media, etc. Also the next sample location could be located within the triangles, not just on the edges, by means of a new measure that would be established for the interior of the triangles.

Accounting for Masking Effects

As stated previously, the algorithms presented above can be improved by taking into account perceptual issues. In particular, one possibility is the use of a *threshold elevation factor* to establish the required quality of the sample radiance of each pixel. This can be done by using a different number of primary estimators for pixel radiances instead of using a fixed number of those estimators—as we currently do in the methods explained above.

The basic idea is to use the current view to obtain a *complexity image* from which a threshold map would be derived by means of an algorithm based on a Discrete Cosine Transform as in the work by Walter et al. [179], but treating the complexity image as if it was a single input texture. This schema is depicted in Figure 7.1. As a complexity image the materials false-color image could be used. Other possibilities include the use of an image incorporating the results of the HMCR step or an estimation of the overall illumination computed as the sum of direct illumination and an estimate of the indirect illumination by means of an ambient term, which would also permit the use of a tone mapping operator prior the execution of the threshold map extractor.

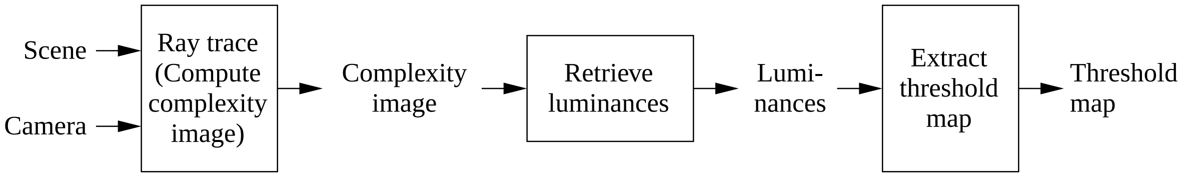


FIGURE 7.1 Computation of the threshold map for the selected view.

This schema is simple, fast, and allows the identification of the ability of masking by complex fine geometry viewed from the selected viewpoint—imagine for example the visualization of scenes with plants. It also detects the natural masking produced by the textures present in the scene (see the work by Walter et al. [179]). We can also take into account reflection effects (mirrors, glasses) in this schema.

The resulting masking map can be used in a way similar to the work by Ramasubramanian et al. [129]: in their path tracing algorithm the number of rays is reduced depending on the value of the masking map at each pixel. In our specific case, in the second pass the precision required for pixel samples could be modified according to the visual mask factor—currently our algorithm uses a fixed number of primary estimations (samples) for the estimation of the pixel radiance.

7.4 Publications

As stated throughout the dissertation, during the development of this thesis a certain number of papers have been produced:

- Global Illumination Techniques for the Simulation of Participating Media (in *Rendering Techniques '97*) [122]

- Acceleration of Monte Carlo Path Tracing in the Presence of Anisotropic Scattering Media (technical report) [120]

- Acceleration of Monte Carlo Path Tracing in General Environments (in *Proceedings of Pacific Graphics 2000*) [121]

- Hierarchical Monte Carlo Radiosity for General Environments (technical report) [117]

- High Quality Final Gathering for Hierarchical Monte Carlo Radiosity for General Environments (in *Advances in Modelling, Animation and Rendering*) [118]

- Progressive Radiance Computation Based on Conductance Maps (submitted) [119]

The author has also contributed, by means of partial work developed for the thesis, to the following publications:

- The SIR Rendering Architecture (in *Computers & Graphics*) [103]

- Efficient Glossy Global Illumination with Interactive Viewing (in *Graphics Interface '99* and extended in *Computer Graphics Forum*) [163, 164]

Appendix A
Implementation

During the development of the thesis a series of tools or software modules have been devised. These include a library for directional distributions, a set of programs dealing with stochastic processes, a program to perform image comparisons, and the extension of the Materials and Geometry Format (MGF) description language [182] to account for participating media [73]. Some of these software modules have contributed to different publications: the directional distributions library was used within the context of glossy global illumination with interactive viewing [163, 164], and stochastic algorithms and the participating media definitions for the extended MGF were considered in the SIR rendering architecture developed by our group [103].

A.1 Library for Directional Distributions

Within the context of the Esprit project SIMULGEN [6] we have developed and implemented an interface in C++ for directional distributions (i.e. non-negative functions). These have been used for example in the extension of Hierarchical Radiosity with Clustering (Section 4.2), and within the context of glossy global illumination [163, 164].

We distinguish between uni- and bi-directional distributions. By unidirectional distributions we mean those distributions that depend on a single direction parameter $\vec{\omega}$, whilst bidirectional distributions depend on a pair of direction parameters $(\vec{\omega}_o, \vec{\omega}_i)$. Unidirectional distributions are able to represent radiances, power, radiant intensities, etc.; bidirectional distributions can model reflection and scattering functions.

The interface takes into account the spectral nature of some radiometric quantities, using a certain number of wavelength samples. Thus, the definitions of classes for the uni- and bi-directional distributions are related to spectra for efficiency reasons. For example, the radiance of a surface could be represented by (1) *three scalar* unidirectional distributions, or better, by (2) a *single spectral* unidirectional distribution. When computing the radiance reflected by that surface, if using (1) we would need to update each one of the three scalar unidirectional distributions in turn, without any possibility of sharing computations; however (2) allows for it. *Gray* unidirectional distributions (i.e. wavelength independent) are also part of the interface, since they turn out to be useful to model some quantities, e.g. when dealing with gray media.

Also the set of classes that compose the interface are meant to be *independent* of any concrete class, i.e. they are able to be used for different spectral representations. This allows BRIGHT (radiosity system in *i*MAGIS), Vision [153] or SIR [103] to share the same code for the interface, and also to be able to change the spectral type (e.g. to change from an RGB type to a spectrum type with 42 samples) without requiring any change of the interface's implementation. This is achieved with the use of class templates. Also the interface is not tied to a particular class modeling the 3D-vectors used for directions, and a type template parameter is used instead. To allow the interface to implement its functionality, a minimum set of requirements on the type template parameters should be met by the formal parameters.

A.1.1 Unidirectional Distributions

Unidirectional distribution functions ($\{f_\lambda(\vec{\omega})\}$, for the 4π steradians) can be used to model

- the radiant intensity of non-lambertian surfaces, of any anisotropic medium, and/or of clusters
- incoming light for clusters, when these clusters are approximated by a single point, in the case that this incoming light is stored
- extinction coefficient of a cluster
- directional intracluster visibility

As stated above, for efficiency reasons, we distinguish between *gray* and spectral unidirectional distributions, with gray denoting independence of wavelength. Notice that when dealing with clusters that do not contain participating media both the extinction coefficient and the directional intracluster visibility depend only on geometry. Also in the case of dealing with gray media there is no need of spectral unidirectional distributions for those quantities. However, for other quantities, or when using non-gray media, it is necessary to use spectral unidirectional distributions.

To be truly generic, our design is based on class templates, so that the implementation is able to work in BRIGHT, Vision and SIR without any change. The design is shown in Figure A.1, using UML notation [25]. Notice that the DirDistrBasis class is an abstract base class, shared both by the DirDistr and DirDistrGray class templates (for spectral and gray unidirectional distributions respectively), from which concrete class templates for dealing with different representations are derived (DirDistrCtBasis for constant basis functions, DirDistrHaarTriBasis for Haar triangular basis—developed by GDV/University of Erlangen, etc.). The formal parameters SpectralType (which will be called TS henceforth, for short), 3DVectorType (or TV in the following) and nbSamples are classes to represent a spectrum, a vector with three components, an unsigned int denoting the dimension of the set of distributions required respectively.

To make the schema easily extensible and avoiding as much recompilation as possible, a map of factories has been used so that the DirDistrBasis does not depend on its derived classes. This is related to the factory pattern [46]. It has been implemented by following a schema given in [31], extended to deal with class templates.

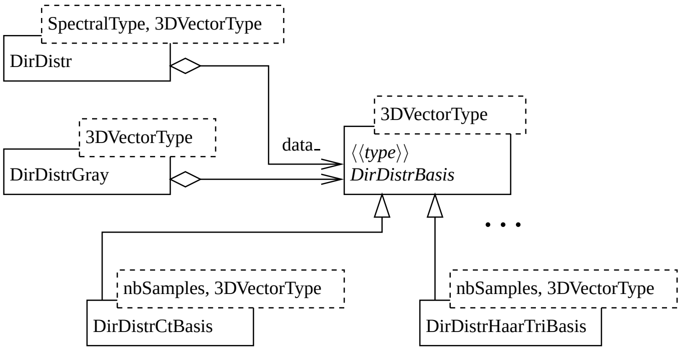


FIGURE A.1 Class diagram for unidirectional distributions.

Thus, the interface for spectral unidirectional distributions is implemented by “template⟨class TS, class TV⟩ class DirDistr”, and the interface for gray ones is “template⟨class TV⟩ class DirDistrGray”.

For example, we can compute the gray directional distribution representing the extinction of a cluster containing three surfaces forming a “delta object” (Figure A.2) using constant basis functions or with Haar triangular basis for different accuracy levels (Figure A.3).

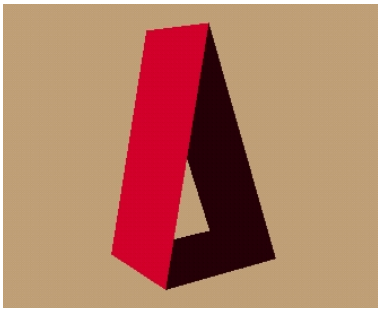


FIGURE A.2 Three oriented surfaces forming a cluster.

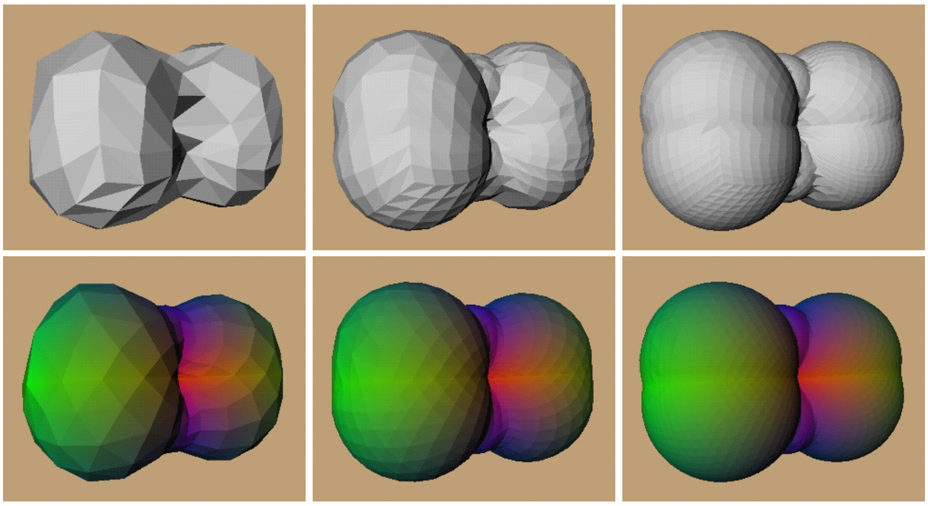


FIGURE A.3 Extinction of the example cluster for increasing accuracy levels, using DirDistrCtBasis objects (top), and DirDistrHaarTriBasis objects (bottom).

Spectral Unidirectional Distributions

Our template⟨class TS, class TV⟩ class DirDistr can be seen as a generalization of the angular distribution function of the ray tracing framework by Shirley et al. [145].

Apart from specifying a set of methods and operators to resemble built-in types for client convenience, there are some important methods related to rendering that are discussed below. The interface allows the addition of objects of the *same* template class (i.e. an instantiation of the class template). This means that it is not allowed the sum of objects belonging to different classes like the ones resulting of instantiating different spectral classes (ditto for the 3D-vector type). In practice this means that the user cannot mix in the program distributions related to, for example, an RGB-related class with other distributions related to spectra with 42 samples. It should be noticed that also (linear) color spaces like the CIE XYZ could be used in substitution of TS.

```
▶ template<class TS, class TV>
  void
  DirDistr<TS,TV>::AddScaledImpulsionalResponse(const ReflectDistr<TS,TV>* rd,
                                               const TS& irr, const TV& inDir, const TV& normal,
                                               double scale);
```

ReflectDistr being a bidirectional distribution (see Section A.1.2) adds to the object the result of evaluating the scaled reflectance equation for a given incoming direction:

$$\text{scale} \times \underbrace{\underbrace{\rho_{\mathrm{bd}}^{\lambda}}_{\mathsf{rd}_\lambda}(\underbrace{\vec{n}}_{\text{normal}}, \vec{\omega}, \underbrace{\vec{\omega}_{\mathrm{i}}}_{\mathsf{inDir}}) \underbrace{L_{\mathrm{i}}^{\lambda}(\vec{\omega}_{\mathrm{i}}) \cos\theta_{\mathrm{i}}\, d\sigma_{\vec{\omega}_{\mathrm{i}}}}_{\mathsf{irr}}}_{L^{\lambda}(\vec{\omega})\text{ (reflected radiance)}} \qquad \forall\lambda, \vec{\omega}\,. \tag{A.1}$$

```
▶ template<class TS, class TV>
  void
  DirDistr<TS,TV>::AddScaledImpulsionalResponseTimesCos(const ReflectDistr<TS,TV>* rd,
                                                          const TS& irr, const TV& inDir, const TV& normal,
                                                          double scale);
```

combines Equation A.1 with a cosine component,

$$\text{scale} \times (\vec{\omega} \cdot \underbrace{\vec{n}}_{\text{normal}}) \times \underbrace{\rho_{\mathrm{bd}}^{\lambda}}_{\mathsf{rd}_\lambda}(\vec{n}, \vec{\omega}, \underbrace{\vec{\omega}_{\mathrm{i}}}_{\mathsf{inDir}}) \underbrace{L_{\mathrm{i}}^{\lambda}(\vec{\omega}_{\mathrm{i}}) \cos\theta_{\mathrm{i}}\, d\sigma_{\vec{\omega}_{\mathrm{i}}}}_{\mathsf{irr}} \qquad \forall\lambda, \vec{\omega}\,, \tag{A.2}$$

allowing the computation of the radiant intensity (with scale being the area of the surface related to the unidirectional distribution).

```
▶ template<class TS, class TV>
  void
  DirDistr<TS,TV>::AddScaledImpulsionalResponse(const ScattDistr<TS,TV>* sd,
                                                 const TS& irr, const TV& inDir,
                                                 const TS& scale);
```

adds the result of evaluating the scattering equation for a given incoming direction (ScattDistr being a bidirectional distribution—see Section A.1.2)

$$\text{scale}_\lambda \times \underbrace{\underbrace{\frac{\Omega_\lambda}{4\pi} p_{\mathrm{bd}}^{\lambda}(\vec{\omega}, \underbrace{\vec{\omega}_{\mathrm{i}}}_{\mathsf{inDir}})}_{\mathsf{sd}_\lambda} \underbrace{L_{\mathrm{i}}^{\lambda}(\vec{\omega}_{\mathrm{i}})\, d\sigma_{\vec{\omega}_{\mathrm{i}}}}_{\mathsf{irr}}}_{J^{\lambda}(\vec{\omega})\text{ (scattered radiance)}} \qquad \forall\lambda, \vec{\omega} \tag{A.3}$$

(here, if scale is equal to $4\kappa_{\mathrm{t}}^{\lambda} V$ we update the radiant intensity of a medium with volume V).

Some methods have been defined for debugging and to visualize results. For instance,

```
▶ template<class TS, class TV>
  void
  DirDistr<TS,TV>::WriteIvFile(const char* fileNameIv,
                               const TV* center=NULL,
                               const SbColor* dColor=NULL) const;
```

writes an Open Inventor file with name fileNameIv containing a visual representation of the DirDistr object, centered at center and using dColor as a diffuse color, whereas

```
▶ template<class TS, class TV>
  SoSeparator*
  DirDistr<TS,TV>::ObtainScaledSoSeparator(double scale=1.,
                                           const TV* center=NULL,
                                           const SbColor* dColor=NULL) const;
```

simply returns a pointer to an SoSeparator object containing that representation. This allows for example a client to save a single file containing all the unidirectional distributions (e.g. radiances) of a set of elements (e.g. surfaces) of a scene.

Note that another possible design would be to make DirDistr an ordinary class (not a class template) by dealing directly with arrays of samples, but the use of the template together with the associated concept leads to a cleaner code (obviously from the point of view of the user of the class). Also the signatures could deal with triplets (double x, double y, double z) instead of directly with objects of type TV, but using the latter there is no need for the user to do any conversion, and also it allows using formal arguments being const TV& vector which in principle should be more efficient than passing three doubles by value.

Gray Unidirectional Distributions

The direct design of the gray unidirectional distribution would be a partial specialization of the DirDistr class template. Unfortunately at the time of the development of this code, some compilers (like the MIPSpro 7.1 C++ compiler) do not support it (others do support it, like the publicly available egcs-1.0 [1] and the MIPSpro 7.2 C++ compiler). In order to be able to use the MIPSpro 7.1 C++ compiler we have specified a separate DirDistrGray template class that still makes use of the DirDistrBasis abstract base class, as shown in Figure A.1.

A.1.2 Bidirectional Distributions

Bidirectional distributions ($\{f_\lambda(\vec{\omega}_o, \vec{\omega}_i)\}$) are also implemented through hierarchies of class templates (Figure A.4), and are used to model

- BRDFs (for surfaces): with ReflectDistr and derived classes.
- Phase functions (for participating media): with ScattDistr (phase function times albedo divided by 4π) and derived classes.

These classes are used for example when computing (or updating) the radiant intensity or radiance of an element. Currently we have adapted the code by Heckbert [61] for the Phong model, but doing that for other models should be straightforward, for example to use physically plausible BRDFs [143, 81]. For scattering, implementations for the Schlick and Henyey-Greenstein phase functions exist.

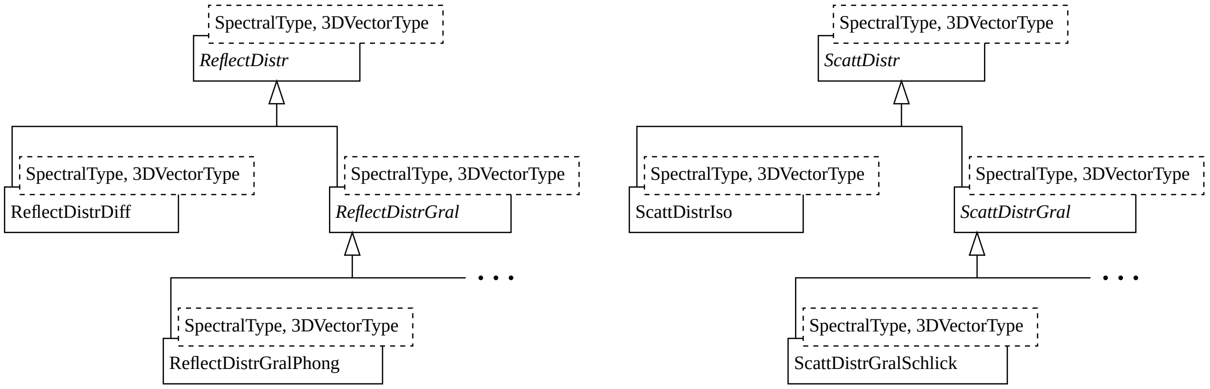

SpectralType, 3DVectorType
ReflectDistr
SpectralType, 3DVectorType
ReflectDistrDiff
SpectralType, 3DVectorType
ReflectDistrGral
...
SpectralType, 3DVectorType
ReflectDistrGralPhong
SpectralType, 3DVectorType
ScattDistr
SpectralType, 3DVectorType
ScattDistrIso
SpectralType, 3DVectorType
ScattDistrGral
...
SpectralType, 3DVectorType
ScattDistrGralSchlick

FIGURE A.4 Hierarchy of classes for bidirectional distributions.

The most important (virtual) methods of ReflectDistr and ScattDistr are

```
▶ template<class TS, class TV>
  TS
  ReflectDistr<TS,TV>::Evaluate(const TV& normal, TV& outDir, TV& inDir) const =0;
```

and

```
▶ template<class TS, class TV>
  TS
  ReflectDistr<TS,TV>::Evaluate(TV& outDir, TV& inDir) const =0;
```

which are defined in derived concrete classes. An example of the result of the use of these distributions is depicted in Figure A.5.

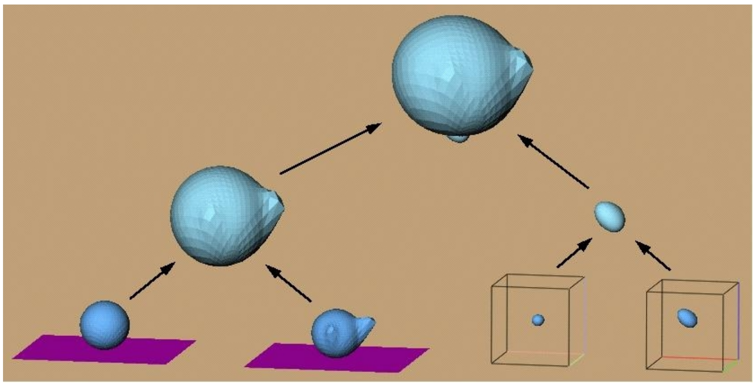


FIGURE A.5 Example of the utilization of uni- and bidirectional distributions: hierarchically pushing radiant intensities (DirDistr objects) of surfaces (using ReflectDistrs objects to model the reflection) and participating media (with ScattDistr objects modeling the scattering).

A.2 Stochastic Simulation of Scattering Processes

A.2.1 Stochastic Scattering: 'scatter' and 'mcs3d'

In order to consider different phase functions (isotropic, Schlick, and combination of two Schlick phase functions [20]), a program scatter was written that simulates the directions after scattering events, including a 3D representation of the results. This program allowed the verification of the kernel code (the scattering itself) prior its inclusion into another software. The 4π sr of the sphere of directions are subdivided in a set of small solid angles, using a number of parallels and meridians set by the user (Figure A.6). Three parts can be distinguished according to that subdivision. Firstly, we have the set of spherical triangles (say triangular solid angles) related to the north pole (that is the direction where the cone angle—or zenith—θ is zero). Secondly, there is a set of spherical quadrilaterals between the first parallel (which is a degenerated one, as the latest parallel, because they are located at the poles) and the previous to the last parallel. Finally, there is another set of spherical triangles related to the south pole ($\theta = \pi$).

Once the subdivision is set, a certain number or particles of energy are sent using any constant incoming direction and scattered at the center of the world (i.e. the center of the sphere) with an initial flux of 1 Watt per particle, within some solid angle element, and updating the outgoing flux through that solid angle. At the end the accumulated flux of all the solid angles is divided by the total number of the simulation particles used. To represent and show the results an Open Inventor file is written. The resulting flux per solid angle is represented by a planar triangle or quadrilateral at a distance from the center directly proportional to that flux and inversely proportional to the solid angle. Examples of results are shown in Figures A.7 and A.8.

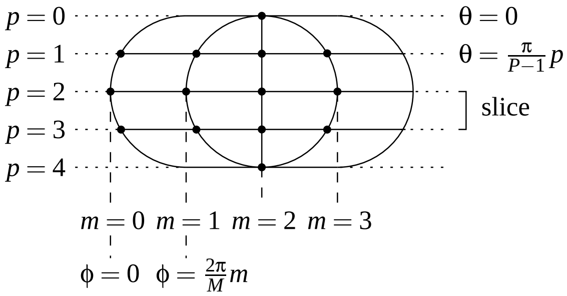


FIGURE A.6 Sphere solid angle subdivision for $M=4$ meridians and $P=5$ parallels (unrolled sphere), and related cone or polar angle θ and circumferential or azimuth angle ϕ.

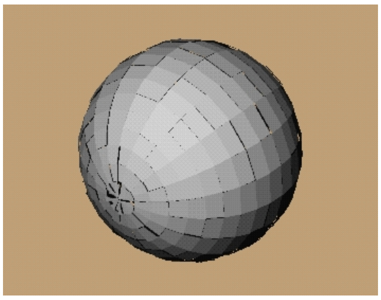


FIGURE A.7 Sampling the isotropic phase function.

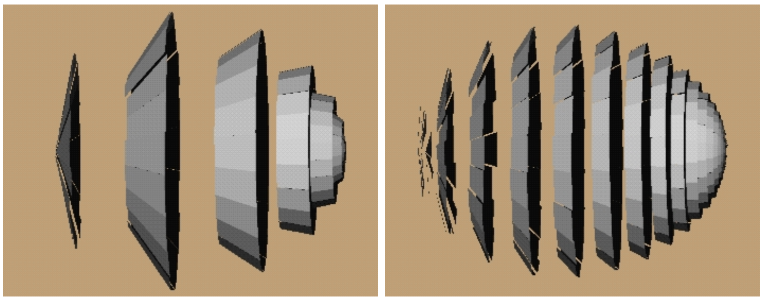


FIGURE A.8 The same anisotropic phase function (Schlick phase function with $k=0.5$) using 9 (left) and 30 (right) parallels. In both cases 20 meridians were used.

Computing Solid Angles

To obtain the value of the solid angles we obtain the solid angle of a slice (angle ϕ between 0 and 2π, cone angle θ between the values related to two consecutive parallels, let us say θ between α and β) and then dividing this value by the number of meridians used. Using the definition of the differential solid angle ($d\sigma_{\vec{\omega}} = \sin\theta\, d\theta\, d\phi$) it is straightforward to obtain a closed expression for such a value:

$$\sigma_{\vec{\omega},\text{slice }\alpha\to\beta} = \int_{\theta\in[\alpha,\beta]}\int_{\phi\in[0,2\pi]} \sin\theta\, d\theta\, d\phi = \int_{\phi\in[0,2\pi]} d\phi \int_{\theta\in[\alpha,\beta]} \sin\theta\, d\theta = 2\pi\int_{\theta\in[\alpha,\beta]} \sin\theta\, d\theta = 2\pi(\cos\alpha - \cos\beta)\,.$$

For example, the solid angle of the spherical triangles around the poles is

$$\frac{\sigma_{\vec{\omega},\text{slice }0\to\frac{\pi}{P-1}}}{M}\,,$$

and the one of the quadrilaterals between parallels p (parallel $p=0$ is the north pole) and $p+1$ is

$$\frac{\sigma_{\vec{\omega},\text{slice }p\frac{\pi}{P-1}\to(p+1)\frac{\pi}{P-1}}}{M}\,.$$

Implemented Phase Functions

- Isotropic: $p(\theta) = 1$.
  As indicated in [156, Eq.s 40–41, p. 72] [99, p. 703] [114, p. 138], the formulas to compute the angles ϕ and θ are:

$$\begin{aligned}\phi &= 2\pi\xi_\phi \\ \theta &= \operatorname{acos}(1-2\xi_\theta)\,.\end{aligned}$$

- Schlick phase function: $p_k(t) = \frac{1-k^2}{(1-kt)^2}$, $k \in (-1,1)$, $t = \cos\theta$.
  From the above expression the way to compute the angle θ is given by

$$\begin{aligned}t &= \frac{k+2u-1}{k(2u-1)+1}\,, \\ \theta &= \operatorname{acos}(t)\,.\end{aligned}$$

- Combination of Schlick phase functions: $p_{r,k,l}(t) = r\,p_k(t) + (1-r)p_l(t)$, $r \in [0,1]$, $k,l \in (-1,1)$, $t = \cos\theta$.
  As the expression for computing t directly has been found to be extremely complex if doing direct importance sampling of the function $p_{r,k,l}(t)$, we simply use the next algorithm:

$$\begin{aligned}&\text{if } (\xi_r < r)\ \theta = \operatorname{acos}\left(\tfrac{2\xi_\theta+k-1}{2k\xi_\theta-k+1}\right) \\ &\text{else } \theta = \operatorname{acos}\left(\tfrac{2\xi_\theta+l-1}{2l\xi_\theta-l+1}\right)\,.\end{aligned}$$

Comparison of Sampling Strategies

As stated in Chapter 3, stochastic multiple scattering methods can classified by their *distance* sampling strategy, that can be constant (CDS) [20, 21] or random based on κ_t (RDS$_t$) [114, 83] (represented in Figure A.9). Also random distance sampling based on κ_s is possible (RDS$_s$—Figure A.10). In order to compare these sampling strategies, including RDS$_s$, we set a simple testbed scene (Figure A.11) and implement their kernels, in a program called mcs3d (from Monte Carlo simulation for three-dimensional media).

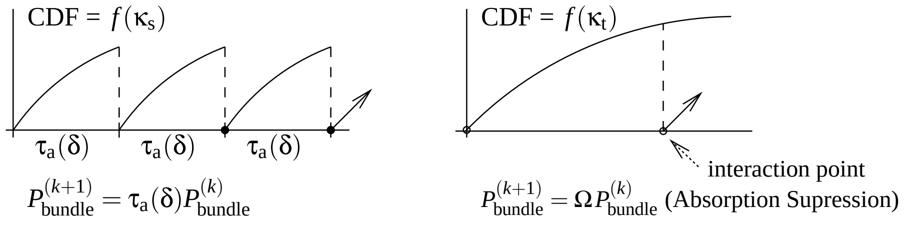

CDF = $f(\kappa_s)$
$\tau_a(\delta)$
$\tau_a(\delta)$
$\tau_a(\delta)$
$P^{(k+1)}_{\text{bundle}} = \tau_a(\delta)P^{(k)}_{\text{bundle}}$
CDF = $f(\kappa_t)$
interaction point
$P^{(k+1)}_{\text{bundle}} = \Omega P^{(k)}_{\text{bundle}}$ (Absorption Supression)

FIGURE A.9 Distance sampling strategies: constant (CDS–left) and randomly, based on κ_t (RDS$_t$–right).

Unbiasedness of the Random Distance Sampling Methods.

Here we demonstrate that the random distance sampling methods are unbiased, that is, the expected values of their results match the physical ones. The derivation of the expression of the flux scattered along the differential length dS after traveling a distance S follows. A bundle with initial flux W after traveling a distance

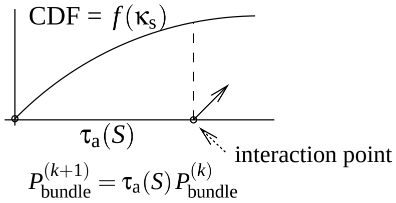


FIGURE A.10 Another random distance sampling strategy, based on κ_{s} (RDS$_{\mathrm{s}}$).

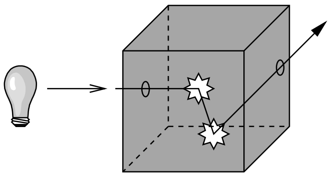


FIGURE A.11 Participating media testbed scene. Bundles of unit power are sent with constant direction towards a cube of media, they are possibly scattered, and the scattered flux exiting the cube is recorded at the (possibly subdivided) limiting borders.

S has a flux of $We^{-\int_0^S \kappa_{\mathrm{t}}(v)dv}$. This travels then an infinitesimal distance dS, and it is reduced by absorption and scattering by the fraction $e^{-\int_S^{S+dS}\kappa_{\mathrm{t}}(v)dv}$, so a fraction $1-e^{-\int_S^{S+dS}\kappa_{\mathrm{t}}(v)dv}$ of the energy at S is absorbed and scattered. Concretely, the portion $\frac{\kappa_{\mathrm{a}}(S)}{\kappa_{\mathrm{t}}(S)}(1-e^{-\int_S^{S+dS}\kappa_{\mathrm{t}}(v)dv})$ is absorbed and the portion $\frac{\kappa_{\mathrm{s}}(S)}{\kappa_{\mathrm{t}}(S)}(1-e^{-\int_S^{S+dS}\kappa_{\mathrm{t}}(v)dv}) = \Omega(S)\,(1-e^{-\int_S^{S+dS}\kappa_{\mathrm{t}}(v)dv})$ is scattered. Thus, we can write

$$P_{\mathrm{sca}}(S,S+dS) = We^{-\int_0^S \kappa_{\mathrm{t}}(v)dv}\Omega(S)\,(1-e^{-\int_S^{S+dS}\kappa_{\mathrm{t}}(v)dv}) = W\Omega(S)(e^{-\int_0^S \kappa_{\mathrm{t}}(v)dv} - e^{-\int_0^{S+dS}\kappa_{\mathrm{t}}(v)dv})\,.$$

For simplicity we will suppose a homogeneous medium for calculating the scattered flux for the RDS$_{\mathrm{t}}$ and RDS$_{\mathrm{s}}$ strategies. For the RDS$_{\mathrm{t}}$ strategy we have

$$\begin{aligned} P^{\mathrm{RDS_t}}_{\mathrm{sca}}(S,S+dS) &= \int_S^{S+dS} W\Omega\, p_{\kappa_{\mathrm{t}}}(u)\,du = W\Omega\int_S^{S+dS}\kappa_{\mathrm{t}}e^{-\kappa_{\mathrm{t}}u}du = W\Omega\kappa_{\mathrm{t}}\left(-\frac{e^{-\kappa_{\mathrm{t}}u}}{\kappa_{\mathrm{t}}}\right)\Bigg|_S^{S+dS} \\ &= W\Omega(e^{-\kappa_{\mathrm{t}}S} - e^{-\kappa_{\mathrm{t}}(S+dS)})\,, \end{aligned}$$

and we obtain the same results for the RDS$_{\mathrm{s}}$ strategy:

$$\begin{aligned} P^{\mathrm{RDS_s}}_{\mathrm{sca}}(S,S+dS) &= \int_S^{S+dS} We^{-\kappa_{\mathrm{a}}u}p_{\kappa_{\mathrm{s}}}(u)\,du = W\int_S^{S+dS} e^{-\kappa_{\mathrm{a}}u}\kappa_{\mathrm{s}}e^{-\kappa_{\mathrm{s}}u}du = W\kappa_{\mathrm{s}}\int_S^{S+dS} e^{-\kappa_{\mathrm{t}}u}du \\ &= W\kappa_{\mathrm{s}}\left(-\frac{e^{-\kappa_{\mathrm{t}}u}}{\kappa_{\mathrm{t}}}\right)\Bigg|_S^{S+dS} = W\kappa_{\mathrm{s}}\left[-\frac{e^{-\kappa_{\mathrm{t}}S}}{\kappa_{\mathrm{t}}} + \frac{e^{-\kappa_{\mathrm{t}}(S+dS)}}{\kappa_{\mathrm{t}}}\right] \\ &= W\Omega(e^{-\kappa_{\mathrm{t}}S} - e^{-\kappa_{\mathrm{t}}(S+dS)})\,. \end{aligned}$$

Therefore they are unbiased estimators. Note that the CDS strategy is biased since it only allows the scattering of the bundles at certain points. Considering the random events (or outcomes of random events) “scattering in latest step” (E_1) and “no scattering in the latest step” (E_0) we have

E_i	$p_i = \mathrm{P}\{E_i\}$
E_1	$1-e^{-\int_0^S \kappa_{\mathrm{s}}(u)du} = p$
E_0	$e^{-\int_0^S \kappa_{\mathrm{s}}(u)du} = 1-p = q$

When $\kappa_s(u) = \kappa_s$ (homogeneous medium) and we consider steps of length δ then we have $p = 1 - e^{-\kappa_s \delta}$ and $q = e^{-\kappa_s \delta}$. This is then a binomial distribution (if we relate a random variable with value 1 for E_1 and value 0 for E_0) [76, p. 13], but note that we carry out the experiment "is the photon scattered" repeatedly and independently, the results can be success or failure, and as long as we obtain failures we repeat the experiment, this is a geometrical distribution, being x the number of experiments we have done until the success appears [76, pp. 14–15].

$$\mathrm{P}\{x = n\} = q^{n-1} p, \qquad n = 1, 2, \ldots$$

The average number of experiments is

$$\langle x \rangle = \sum_{n=1}^{\infty} n q^{n-1} p = \frac{p}{(1-q)^2} = \frac{1}{p} = \frac{1}{1 - e^{-\kappa_s \delta}}\,,$$

therefore the average length $\langle l_{[21]} \rangle$ before an scattering event is

$$\langle l_{[21]} \rangle = \langle x \rangle\, \delta = \frac{\delta}{1 - e^{-\kappa_s \delta}}\,.$$

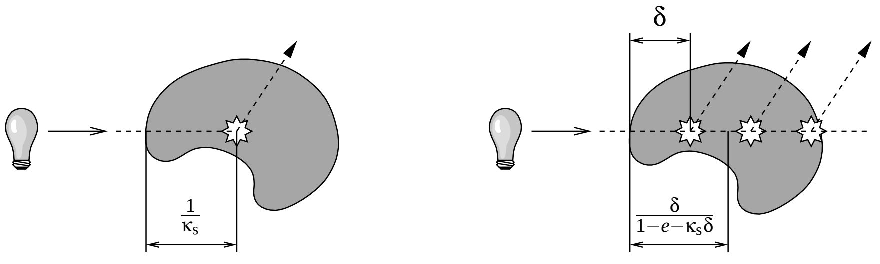
(This expression has been checked experimentally with a very simple program.) This must be compared with the expectance of the distance before an scattering event occurs when sampling directly the CDF $1 - e^{-\int_0^S \kappa_s(u)du}$. This should be equal to the mean free path without absorption:

$$\langle l_{\mathrm{RDS_s}} \rangle = \frac{1}{\kappa_s}\,.$$

The error committed by the CDS strategy is then (Figure A.12)

$$|\langle l_{\mathrm{CDS}} \rangle - \langle l_{\mathrm{RDS_s}} \rangle| = \left| \frac{\delta}{1 - e^{-\kappa_s \delta}} - \frac{1}{\kappa_s} \right|\,.$$

The only way to make $\langle l_{\mathrm{CDS}} \rangle - \langle l_{\mathrm{RDS_s}} \rangle = 0$ is by setting δ equal to zero, which has no sense.



Mean free path for scattering

FIGURE A.12 Error of the constant distance approach.

The multiple scattering process, being just a concatenation of the simple scattering process, will also give unbiased results for methods $\mathrm{RDS_t}$ and $\mathrm{RDS_s}$.

In order to see the behavior of the error by the CDS strategy for different δ values, the very simple scene of Figure A.11 was used, initially filled with a homogeneous medium, with constant coefficients of absorption and scattering. Energy gets into the medium only from a single point and a single direction. As expected, as the value of δ was reduced for different simulations, the estimate of the scattered energy (of all the cube) was closer to the true value. Also, the lower the δ

the higher the simulation time (because the paths are longer), that is roughly speaking of the order of the inverse of δ. However, RDS_t was cheaper in computation time. After performing a set of executions changing the properties of the medium, subdividing it, using different phase functions, our conclusion is that is more efficient and accurate to use the random distance sampling approach, and that RDS_t outperforms RDS_s in most of the cases (except when dealing with media with high κ_s).

A.2.2 Monte Carlo Path Tracing with Multiple Scattering

In the SIMULGEN context, our group has implemented a Monte Carlo path tracing algorithm (mcpt) over the SIR system [103], following the work of [80] and containing several enhancements to make it usable for moderately complex scenes with participating media. With minor extensions (i.e. through class derivation to redefine virtual methods conveniently) this path tracer constitutes the second pass of our two-pass algorithms.

In [80] the classical path tracing algorithm is presented as a mathematical tool for solving the rendering equation. However, the basic method is not practical to be used in even simple scenes. The author proposes several improvements that allow to compute much more accurate solutions with the same computational effort. The path tracer herein presented has some of them:

- Importance sampling based on the BRDF term for surfaces with neither information about the incoming radiance [80], nor on the phase function for participating media [20].

- Next event estimation. With this technique direct illumination is computed with much less number of samples than with classical path tracing. For each path, a point on a light source is sampled, and the direct illumination at each intersection point of the path is computed.

There are other features that have been incorporated to the path tracer that allow better functionality and produce more accurate images:

- Adaptive determination of the number of radiance samples per pixel (Section A.2.3). A threshold is given to the program and each pixel is progressively sampled until the estimated error is lower than the threshold. This is very useful to get uniform images because more work will be done in those pixels whose radiance is more difficult to compute.

- Quasi-Monte Carlo pixel sampling. Samples for the directions of the primary rays per pixel are generated using quasi-Monte Carlo number sequences [127]. Quasi-Monte Carlo Sobol sequences generate samples in an N dimensional space that are guaranteed to uniformly sample the unitary N-hypercube.

- Forced interaction. When rendering participating media standard path tracing needs to compute a huge amount of samples due to the high variance of the pixels that “see” the media. This is due to the fact that paths sent through the same pixel can interact with the media along the line that starts at the pixel and ends in the first surface. This set of points usually has a high range compared with that of those pixels that do not “see” the media. This is especially problematic when the media has a low extinction coefficient, i.e. the probability of interaction is very low. In this case most of the paths do not interact with the media making the variance high because a lot of samples are needed to get a good approximation of the pixel radiance. Forced interaction [114] is a variance reduction technique that forces interactions with the medium instead of computing a probability of interaction, so that all paths sample the media and there is a better use of the pixel samples (Figure A.13). For each primary ray with forced interaction with the media two contributions must be computed: the radiance coming from behind the interaction point, and the radiance in-scattered at that point.

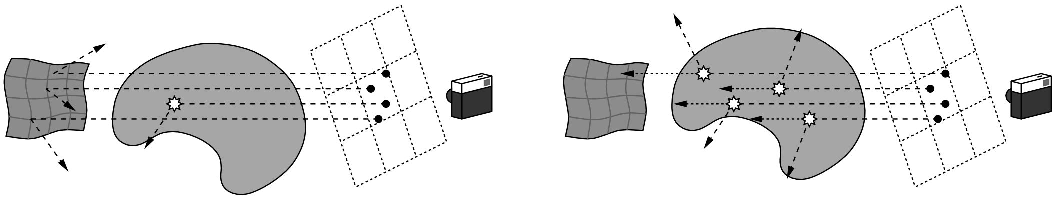


FIGURE A.13 On the left the normal Monte Carlo sampling is depicted, with only a few paths interacting with the media. On the right is represented the forced interaction technique that forces interaction with the media for primary rays.

An image computed by the path tracer is shown in Figure A.14.

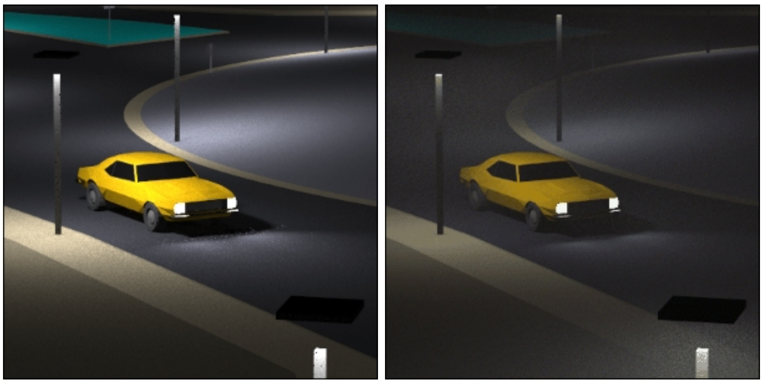


FIGURE A.14 Images generated by mcpt for a scene in vacuum and with homogeneous medium.

A.2.3 Stopping Criteria for Stochastic Methods

Stochastic methods compute estimates of random variables, having a certain variance that depends on the number of samples and on the samples themselves. Thus, a certain accuracy can be achieved by using (adaptively) as many samples as needed to arrive to an estimated error below a certain threshold. This is done by our Monte Carlo path tracer (both mcpt and the second pass presented in Section 5.2) and by mcs3d.

Estimating the Estimate's Error

The basics of this section was studied by Purgathofer in [128]. To control the error, as a program progresses, an estimation of the variances of the random variables is computed [93, pp. 62–65]. This implies an overhead, but the number of estimations (for instance the number of primary rays in mcpt) does not have to be adjusted. Let $\hat{\mu}_{\mathrm{t}}$ be the estimate of the mean μ_{t} of the random variable. Our estimate is computed simply as a mean with the following expression,

$$\hat{\mu}_{\mathrm{t}} = \langle \hat{\mu}(\cdot) \rangle_{n_{\mathrm{t}}} = \frac{1}{n_{\mathrm{t}}} \sum_{i=1}^{n_{\mathrm{t}}} \hat{\mu}_i \,, \tag{A.4}$$

being n_{t} the total number of primary estimators used in the simulation until the current checkpoint. Its variance can be empirically estimated by the square of the standard deviation

$$\hat{s} = \sqrt{\frac{1}{n_{\mathrm{t}}-1}\sum_{i=1}^{n_{\mathrm{t}}}(\hat{\mu}_i - \hat{\mu}_{\mathrm{t}})^2}\,,$$

and it is an unbiased estimator, i.e. $\mathrm{E}[\hat{s}] = \sigma$.

The *standard error of the mean* $s_{\hat{\mu}_{\mathrm{t}}}$ is an unbiased estimate of $\sigma/\sqrt{n_{\mathrm{t}}}$:

$$s_{\hat{\mu}_{\mathrm{t}}} = \frac{\hat{s}}{\sqrt{n_{\mathrm{t}}}} = \sqrt{\frac{1}{n_{\mathrm{t}}\,(n_{\mathrm{t}}-1)}\sum_{i=1}^{n_{\mathrm{t}}}(\hat{\mu}_i - \hat{\mu}_{\mathrm{t}})^2}\,.$$

Therefore the *square of the standard error of the mean* is

$$\hat{s}^2_{\hat{\mu}_{\mathrm{t}}} = \frac{1}{n_{\mathrm{t}}\,(n_{\mathrm{t}}-1)}\sum_{i=1}^{n_{\mathrm{t}}}(\hat{\mu}_i - \hat{\mu}_{\mathrm{t}})^2\,, \tag{A.5}$$

but that expression cannot be applied generally as we should store all the $\hat{\mu}_i$'s. However, a convenient formulation is achieved after some transformations:

$$\begin{aligned}\sum_{i=1}^{n_{\mathrm{t}}}(\hat{\mu}_i - \hat{\mu}_{\mathrm{t}})^2 &= \sum_{i=1}^{n_{\mathrm{t}}}\left(\hat{\mu}_i^2 - 2\,\hat{\mu}_i\,\hat{\mu}_{\mathrm{t}} + \hat{\mu}_{\mathrm{t}}^2\right) = \sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - 2\,\hat{\mu}_{\mathrm{t}}\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i + n_{\mathrm{t}}\frac{1}{n_{\mathrm{t}}^2}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^2\\ &= \sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - 2\frac{1}{n_{\mathrm{t}}}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^2 + \frac{1}{n_{\mathrm{t}}}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^2 = \sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - \frac{1}{n_{\mathrm{t}}}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^2\\ &= \mathsf{accEstSq} - \frac{1}{n_{\mathrm{t}}}\mathsf{accEst}^2\,.\end{aligned}$$

Thus we obtain the final expression

$$\hat{s}^2_{\hat{\mu}_{\mathrm{t}}} = \frac{1}{n_{\mathrm{t}}\,(n_{\mathrm{t}}-1)}\left[\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - \frac{1}{n_{\mathrm{t}}}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^2\right]\,. \tag{A.6}$$

We can use directly Equation A.6 just defining a threshold and whenever the estimate $\hat{s}^2_{\hat{\mu}_{\mathrm{t}}}$ is below that threshold, the generation of samples ends.

Another possibility is to define a "relative error" $\hat{\varepsilon}_{\mathrm{t}}$ as follows:

$$\begin{aligned}\hat{\varepsilon}_{\mathrm{t}} = \frac{\hat{s}^2_{\hat{\mu}_{\mathrm{t}}}}{\hat{\mu}_{\mathrm{t}}^2} = n_{\mathrm{t}}^2\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^{-2}\frac{1}{n_{\mathrm{t}}(n_{\mathrm{t}}-1)}\left[\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - \frac{1}{n_{\mathrm{t}}}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^2\right] = \frac{n_{\mathrm{t}}}{n_{\mathrm{t}}-1}\left[\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^{-2}\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - \frac{1}{n_{\mathrm{t}}}\right] =\\ = \frac{1}{n_{\mathrm{t}}-1}\left[n_{\mathrm{t}}\left(\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i\right)^{-2}\sum_{i=1}^{n_{\mathrm{t}}}\hat{\mu}_i^2 - 1\right] = \frac{1}{n_{\mathrm{t}}-1}\left[n_{\mathrm{t}}\,\mathsf{accEst}^{-2}\,\mathsf{accEstSq} - 1\right]\,.\end{aligned} \tag{A.7}$$

If we use this relative error, we can stop the simulation when $\hat{\varepsilon}_{\mathrm{t}} < \varepsilon$, being ε a threshold explained below. It should be noticed that we can instead use $\sqrt{\hat{\varepsilon}_{\mathrm{t}}} = \frac{\hat{s}_{\hat{\mu}_{\mathrm{t}}}}{\hat{\mu}_{\mathrm{t}}}$ (standard error of the mean divided by the sample mean) and stop the simulation when $\sqrt{\hat{\varepsilon}_{\mathrm{t}}} < \sqrt{\varepsilon}$.

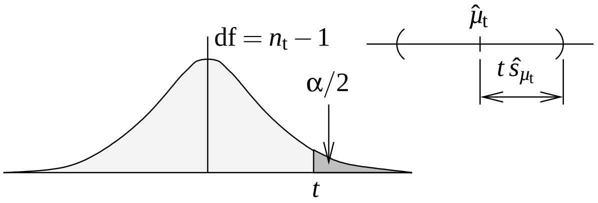


FIGURE A.15 Student's *t* distribution and confidence interval.

When estimating the mean with population standard deviation unknown, the formula for the (1-α)% confidence interval (see Figure A.15) is given by

$$\hat{\mu}_t \pm t\left(\text{df}, \frac{\alpha}{2}\right) \frac{\hat{s}}{\sqrt{n}} = \hat{\mu}_t \pm t\left(n_t - 1, \frac{\alpha}{2}\right) \hat{s}_{\hat{\mu}_t} \,,$$

being t the Student distribution.

It would be interesting to stop the simulation when

$$t\left(n_t - 1, \frac{\alpha}{2}\right) \hat{s}_{\hat{\mu}_t} < \varepsilon \qquad \text{or} \qquad \frac{t\left(n_t - 1, \frac{\alpha}{2}\right) \hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \varepsilon$$

for an "absolute" or a "relative" error measure. But when the value of n_t is large then $t\left(n_t - 1, \frac{\alpha}{2}\right)$ can be practically considered a constant (for a given α). Then we can write

$$K\hat{s}_{\hat{\mu}_t} < \varepsilon \Leftrightarrow \hat{s}_{\hat{\mu}_t} < \varepsilon' \qquad \text{or} \qquad \frac{K\hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \varepsilon \Leftrightarrow \frac{\hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \varepsilon' \,.$$

We continue developing further the last expression, using $K' = 2 > K$ (corresponding to $\alpha = 0.05$):

$$\frac{t\left(n_t - 1, \frac{\alpha}{2}\right) \hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \frac{K\hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \frac{2\hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \varepsilon_{\text{relative}} \quad \Rightarrow \quad \frac{2\hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} \times 100 < \varepsilon_{\text{relative}} \times 100 = \varepsilon^{\%}_{\text{relative}}$$

$$\frac{\hat{s}_{\hat{\mu}_t}}{\hat{\mu}_t} < \frac{\varepsilon^{\%}_{\text{relative}}}{200}$$

$$\hat{\varepsilon}_t = \frac{\hat{s}^2_{\hat{\mu}_t}}{\hat{\mu}^2_t} < \varepsilon \equiv \frac{\left(\varepsilon^{\%}_{\text{relative}}\right)^2}{4 \cdot 10^4} \,. \tag{A.8}$$

Using Equation A.8 with a certain value of $\varepsilon^{\%}_{\text{relative}}$ (in percentage) set by the user it is expected that

$$\frac{|\hat{\mu}_t - \mu|}{\mu} \times 100 < \varepsilon^{\%}_{\text{relative}}$$

in the 95% of the executions.

We have checked experimentally Equation A.8 with a set of 50 executions using a light tracer. We only got 2 executions with error greater than 1%, therefore 48 of them had an error below the 1%, that means the 96% of executions. This is what we expected since we imposed a (1-α)% confidence interval with α=0.05 (that is, a 95% confidence interval).

The Problematic Case

If we use Equation A.6 notice that when $\hat{\mu}_i = 0\ \forall i$ then $\hat{s}^2_{\hat{\mu}_t} = 0$, so it will always meet the requirement of being below any given threshold. Care must be taken in the initial steps of the process, when n_t is small enough so that we can have all $\hat{\mu}_i = 0$ while μ can be potentially positive.

Also notice that if we use the relative error $\hat{\varepsilon}_{\mathrm{t}}$ (Equation A.7) when $\hat{\mu}_i = 0\ \forall i$ then $\hat{\mu}_{\mathrm{t}} = 0$ (Equation A.4) and $\hat{s}^2_{\hat{\mu}_{\mathrm{t}}} = 0$ (Equation A.5), therefore $\hat{\varepsilon}_{\mathrm{t}} = \hat{s}^2_{\hat{\mu}_{\mathrm{t}}}/\hat{\mu}^2_{\mathrm{t}}$ is undetermined. This special case can occur in practice (e.g. when estimating the direct illumination of a point in shadow, or in the initial stages of the process, when n_{t} is small enough to have all $\hat{\mu}_i = 0$).

A simple way to deal with this case is to define a threshold maxNoSamplesEq0 so that when $\sum_{i=1}^{n_{\mathrm{t}}} \hat{\mu}_i$ is zero and n_{t} is greater than the threshold, we fix our estimate to be zero, and do not continue generating samples. Otherwise we can compute Equations A.6 and/or A.7 without problem.

The Type of the Estimates

In the image based algorithms, for a given pixel in general the estimates can be of a composed type related to irradiance or radiance values for a set of wavelengths, e.g. Spectrum⟨float⟩. If avoiding the direct solution (i.e. computing monochromatic images—one per wavelength—, and at the end merging them), then the comparison between the error and the threshold (e.g. Equation A.8) must be specified for the set of wavelengths. In C++ terms we could probably speak of overloading of the operator "$<$". This can be done defining a threshold value per wavelength, and the global comparison evaluating true when all the local comparisons are true. Another possibility is to do a single comparison, just combining the estimate values of the different wavelengths (e.g. with the same weight, i.e. the mean, or with different weights, like the Y channel of the YIQ color model). Indeed, for samples on pixels, the control should be on the mapped display values, not on the radiances themselves [29, p. 28].

A.3 Image Comparisons: `diffrgbe'

In order to compare images (for example, to compare not converged images against a converged image representing a solution), we have written some image comparison algorithms within the context of the SIR architecture. SIR writes output image files in RADIANCE's RGBE format [88]. This allows performing comparisons both in direct radiance units (or any other physical units), and in "display units" after the use of a tone mapping operator.

Although we plan to implement methods that take into account the human perception [34, 45, 102, 104], for now diffrgbe contemplates quite simple methods like (R)MSE and Drettakis's error measure (percentage of pixels with an absolute difference greater than a certain tolerance) [38].

The program diffrgbe has been initially developed in Linux using Qt [3] for the graphical user interface and then ported to UNIX without any problem.

A.4 Participating Media in the SIR Framework

The SIR framework was developed from scratch at the GGG to aid in the programming of algorithms to solve the global illumination problem [103]. Thus, in the SIR kernel, abstract classes for geometric objects (sirGeomObj), optical properties (sirOpticalProp) and radiance representations (sirRadObj) were specified to serve as interfaces. Derived classes take into account the surface or volume nature of the related concept. Specific classes for participating media are thus considered in sirGOVol (from "Geometric Object–Volume"; for instance it could be limited by a rectangular prism), sirOPVol (from "Optical properties–volume"; it contains information on the albedo, extinction, the phase function, inspectors to sample it, etc.), sirROVol (from "Radiance Object–Volume") and their descendants. Phase functions are considered in classes derived from the abstract base class sirBSDF (from “Bidirectional Scattering Distribution Function”): BSDFIsotropic, BSDFRayleigh and BSDFSchlick.

Both the Monte Carlo path tracing mcpt (Section A.2.2) and the second pass presented in Section 5.2 were developed upon the SIR framework.

A.5 Participating Media in the MGFE Format

The Materials and Geometry Format Extended (MGFE) is a description language for 3D environments expressly suited to visible light simulation and rendering [73]. Developed by our group, it is an extension of Greg Ward’s MGF language [182] embedding MGF and adding new functionalities: cameras, *participating media* and animation. (The author’s contribution is limited to the definition of the part of the language related to the participating media.) The GGG has written a parser following the same philosophy of the MGF parser (currently used by SIR applications) and a Maya plug-in to export scenes to MGFE.

The idea is to describe scenes with one or more MGFE files. Therefore MGFE must support ways to define geometry (surfaces and volumes), materials, textures, participating media, animation, etc. Obviously it must handle spectral and anisotropic characteristics of absorbing-emitting-scattering media. The quantities $\kappa_t(x)$, $\Omega(x)$, $J_e(x,\vec{\omega})$, $p(x,\vec{\omega}_o,\vec{\omega}_i)$ should be given, or computable from the given data.

As in MGF, MGFE is based on *entities* and *contexts*. An entity is any non-blank line, starting with a command keyword possibly followed by arguments. A *context* describes the current state of the interpreter, and affects or is affected by certain entities as they are read in. MGFE adds to the MGF contexts the new contexts phf and mpm to define the phase function and the “material” of the medium. MGFE entities and contexts are shown in Tables A.1 and A.2.

Entities that exist in MGF but that have (slightly) different meaning in MGFE are:

- cyl and cone: They are open-ended in MGF, but in MGFE they are closed, to contain the medium.

Entities that did not exist in MGF are:

- phf: For the phase function context
- type, k, l and r: These entities are related to the phase function context, and they set its type and parameters for the concrete type:
  - Isotropic (needs no parameters)
  - Schlick k
  - Schlick2 r k l
  - Henyey-Greenstein k
- mpm: Material context for a participating medium.
- albedo: The rd entity counterpart for participating media.
- em: The ed entity counterpart for participating media (‘em’ from ‘emission’).
- kt: To define the extinction coefficient κ_t.
- ktgridfile: Its parameter is a binary file containing weights to multiply the value of κ_t (defined with the kt entity) so that the medium will be inhomogeneous. The format of this file could be:
  - A header starting with a magic number to identify this type of file, followed by three ints expressing the dimensions of the data array in the x, y and z axis (sizeX, sizeY and sizeZ).

Keyword	Arguments	Interpretation
phf	[id [= [template]]]	get/set phase function context
type	Isotropic|Schlick|Schlick2|Henyey-Greenstein	set the type for current phase function
k	value	set the k parameter value for the current phase function
l	value	set the l parameter value for the current phase function
r	value	set the r parameter value for the current phase function
mpm	[id [= [template]]]	get/set material context for participating medium
albedo	omega	set albedo for current material
em	epsilon	set emittance for current material
kt	kappa_t	set extinction for current material
ktgridfile	filename	set the grid file to modulate kappa_t for current material
albedogridfile	filename	set the grid file to modulate Omega for current material

TABLE A.1 MGFE participating media entities and their arguments. Arguments in brackets are optional. Arguments in curly braces mean one of the given choices must appear. Ellipsis mean that any number of arguments may be given.

- The body of the file, i.e. the data. This can be e.g. stored as unsigned ints (other possibilities could be implemented, just adding after the magic number some identifier related to the type of data stored thereafter), in z-slices, so that the code to read this data could be:

```
unsigned char* buffer = new unsigned char [sizeX*sizeY*sizeZ];
unsigned char* t=buffer;
for (int k=0; k < sizeZ; ++k) {
  read(fd,(char*)t,sizeX*sizeY*sizeof(unsigned char));
  t += sizeX*sizeY;
}
```

- albedogridfile: Just like the ktgridfile entity, but to weight the albedo.
- rectprism: To define a rectangular prism containing a participating medium.
- fullspace: Express that the participating medium is everywhere.

Context	Cntl. Entity	Default Value	Field Entities	Affects
Phase function	phf	isotropic	type, k, l, r	sph, cyl, cone, prism, torus, rectprism, fullspace
Material for participating media	mpm	black	albedo, em, kt, ktgridfile, albedogridfile	sph, cyl, cone, prism, torus, rectprism, fullspace

TABLE A.2 MGFE participating media contexts and their related entities and default values.

A simple MGFE participating media example follows:

```
# Establish a new phf context called "cloudyPhF"
phf cloudyPhF =
  type Schlick2
  k 0.3
  l -0.4
  r 0.5
# We’re done, the current phf context is now "cloudyPhF"
# Create new named material for participating media context
mpm matPsychoCloud =
  # Set a certain albedo
  c
    # Spectrum measured in 10 nm increments from 400 to 700 nm
    cspec 400 700 35.29 44.87 47.25 47.03 46.87 47.00 47.09 \
          47.15 46.80 46.17 46.26 48.74 51.08 51.31 51.10 \
          51.11 50.52 50.36 51.72 53.61 53.95 52.08 49.49 \
          48.30 48.75 49.99 51.35 52.75 54.44 56.34 58.00
  albedo 0.97
  # Set neutral color
  c
  # Set extinction coefficient
  kt .04
  # Modulate with a file of our own
  ktgridfile myGridFileApart

# The cloud corner vertices:
v corner.xyz =
        p 0 0 0
v corner.XYZ =
        p 48 24 48

# The cloud:
# Push object name
o cloud
  # Get previously defined matPsychoCloud material
  m matPsychoCloud
  # Get previously defined cloudyPhF phase function
  phf cloudyPhF
  # Polygonal face using defined vertices
  rectprism corner.xyz corner.XYZ
# Pop object name
o
```

Bibliography

[1] Egcs project. World Wide Web: http://www.cygnus.com/egcs/. 97

[2] GTS (GNU Triangulated Surface Library). World Wide Web: http://gts.sourceforge.net. 77

[3] Qt. World Wide Web: http://www.troll.no/products/qt.html. 107

[4] RADIANCE. World Wide Web: http://floyd.lbl.gov/radiance/. 75

[5] Rayshade. World Wide Web: http://graphics.stanford.edu/˜cek/rayshade/rayshade.html. 75

[6] SIMULGEN. Realistic Simulation of Light for General Environments. ESPRIT project #35772. World Wide Web: http://iiia.udg.es/Simulgen/. 93

[7] Neeharika Adabala and Swami Manohar. Modeling and Rendering of Gaseous Phenomena Using Particle Maps. *Journal of Visualization and Computer Animation*, 11(5):279–294, 2000. 28, 29, 36

[8] Didier Arquès and Sylvain Michelin. Proximity Radiosity: Exploiting Coherence to Accelerate Form Factor Computations. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 143–152, New York, NY, 1996. Springer-Verlag/Wien. 29

[9] James Arvo. Code of "Stratified Sampling of Spherical Triangles". Available from http://www.cs.caltech.edu/˜arvo/software.html, 1995. 56

[10] James Arvo. Stratified Sampling of Spherical Triangles. In *Computer Graphics Proceedings, Annual Conference Series, 1995 (ACM SIGGRAPH '95 Proceedings)*, pages 437–438, 1995. 56, 63

[11] Kavita Bala, Julie Dorsey, and Seth Teller. Radiance Interpolants for Accelerated Bounded-Error Ray Tracing. *ACM Transactions on Graphics*, 18(3):213–256, July 1999. 77, 82

[12] Philippe Bekaert. *Hierarchical and Stochastic Algorithms for Radiosity*. PhD thesis, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, 1999. 2, 46, 46, 88

[13] Philippe Bekaert, Philip Dutré, and Yves D. Willems. Final Radiosity Gather Step Using a Monte Carlo Technique with Optimal Importance Sampling. Technical Report CW275, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, November 1998. 52

[14] Philippe Bekaert, László Neumann, Attila Neumann, Mateu Sbert, and Yves D. Willems. Hierarchical Monte Carlo Radiosity. In G. Drettakis and N. Max, editors, *Rendering Techniques '98 (Proceedings of Eurographics Rendering Workshop '98)*, pages 259–268, New York, NY, 1998. Springer Wien. 2, 46, 46, 47, 47, 68, 88

[15] Philippe Bekaert (web editor). RenderPark. World Wide Web: http://www.cs.kuleuven.ac.be/cwis/research/graphics/RENDERPARK/. 75

[16] Larry Bergman, Henry Fuchs, Eric Grant, and Susan Spach. Image Rendering by Adaptive Refinement. *Computer Graphics (SIGGRAPH '86 Proceedings)*, 20(4):29–37, August 1986. 76

[17] Neeta Bhate. Application of Rapid Hierarchical Radiosity to Participating Media. In *Proceedings of ATARV-93: Advanced Techniques in Animation, Rendering, and Visualization*, pages 43–53, Ankara, Turkey, July 1993. Bilkent University. 27, 28, 29, 29

[18] Neeta Bhate and A. Tokuta. Photorealistic Volume Rendering of Media with Directional Scattering. In *Third Eurographics Workshop on Rendering*, pages 227–245, Bristol, UK, May 1992. 11, 13, 28, 28, 31, 39, 39

[19] Venceslas Biri, Sylvain Michelin, and Didier Arquès. Real-Time Animation of Realistic Fog. In *Rendering Techniques 2002 (Proceedings of the Thirteenth Eurographics Workshop on Rendering)*. ACM Press, June 2002. Poster paper. 20, 27

[20] Philippe Blasi, Bertrand Le Saëc, and Christophe Schlick. A Rendering Algorithm for Discrete Volume Density Objects. *Computer Graphics Forum (Eurographics '93)*, 12(3):C201–C210, September 1993. 10, 10, 12, 13, 13, 14, 15, 15, 15, 28, 29, 34, 34, 34, 34, 51, 51, 53, 55, 98, 100, 103

[21] Philippe Blasi, Bertrand Le Saëc, and Christophe Schlick. An Importance Driven Monte-Carlo Solution to the Global Illumination Problem. In *Fifth Eurographics Workshop on Rendering*, pages 173–183, Darmstadt, Germany, June 1994. 28, 29, 34, 34, 34, 34, 37, 37, 38, 38, 38, 100, 102, 102

[22] James F. Blinn. Ligth Reflection Functions for Simulation of Clouds and Dusty Surfaces. *Computer Graphics (ACM SIGGRAPH '82 Proceedings)*, 16(3):21–29, 1982. 8, 14, 14, 18, 20, 21, 22, 23, 27, 27, 27

[23] Craig F. Bohren. Multiple Scattering of Light and Some of its Observable Consequences. *American Journal of Physics*, 55(6):524–533, 1987. iii, 11, 23

[24] Craig F. Bohren and Donald R. Huffman. *Absorption and Scattering of Light by Small Particles*. John Wiley & Sons, 1993. 8, 10, 11

[25] Grady Booch, Ivar Jacobson, and Jim Rumbaugh. Unified Modeling Language (UML). World Wide Web: http://www.rational.com/uml/ (UML Resource Center). 94

[26] Eva Cerezo and Francisco J. Serón. Rendering Natural Water: Merging Computer Graphics with Physics and Biology. In J. Vince and R. Earnshaw, editors, *Advances in Modelling, Animation and Rendering (Proc. of Computer Graphics International 2002)*. Springer, 2002. 32

[27] Subrahmanyan Chandrasekhar. *Radiative Heat Transfer*. Dover Publications, New York, 1960. 11, 11, 12

[28] Shenchang Eric Chen, Holly E. Rushmeier, Gavin Miller, and Douglass Turner. A Progressive Multi-Pass Method for Global Illumination. In *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, volume 25, pages 164–174, July 1991. 2, 42

[29] Kenneth Chiu and Peter Shirley. Rendering, Complexity, and Perception. In *Fifth Eurographics Workshop on Rendering*, pages 19–33, Darmstadt, Germany, June 1994. 107

[30] Per Henrik Christensen. *Hierarchical Techniques for Glossy Global Illumination*. PhD thesis, Seattle, Washington, 1995. 37, 42, 44, 45, 45

[31] Allan D. Clarke. "C++ Tip-of-the-Day", on polymorphic constructors. World Wide Web: http://cpptips.hyperformix.com/cpptips.html (poly_ctor file within the archive), 2000. 94

[32] Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg. A Progressive Refinement Approach to Fast Radiosity Image Generation. In *Computer Graphics (ACM SIGGRAPH '88 Proceedings)*, volume 22, pages 75–84, August 1988. 41

[33] Michael F. Cohen and John R. Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press Professional, Boston, MA, 1993. 5, 10, 41

[34] Scott Daly. Visible Differences Predictor: An Algorithm for the Assessment of Image Fidelity. In A. Watson, editor, *Digital Images and Human Vision*, pages 179–206. MIT Press, 1993. 107

[35] Jean-Michel Dischler and Djamchid Ghazanfarpour. A Survey on 3D Texturing. *Computers & Graphics*, 25(1):135–151, 2001. 49

[36] Yoshinori Dobashi, Kazufumi Kaneda, Hideo Yamashita, Tsuyoshi Okita, and Tomoyuki Nishita. A Simple, Efficient Method for Realistic Animation of Clouds. *Proceedings of SIGGRAPH 2000*, pages 19–28, July 2000. 33

[37] Yoshinori Dobashi, Tsuyoshi Yamamoto, and Tomoyuki Nishita. Interactive Rendering of Atmospheric Scattering Effects Using Graphics Hardware. In S. N. Spencer, editor, *Proceedings of the 17th Eurographics/SIGGRAPH workshop on graphics hardware (EGGH-02)*, pages 99–108, New York, 2002. ACM Press. 23, 26, 27

[38] George Drettakis and Eugene Fiume. Concrete Computation of Global Illumination Using Structured Sampling. In *Third Eurographics Workshop on Rendering*, pages 189–202, Bristol, UK, May 1992. 107

[39] Eric Dumont. Semi-Monte Carlo Light Tracing Applied to the Study of Road Visibility in Fog. In *Proceedings of the Third International Conference on Monte Carlo and Quasi Monte Carlo Methods in Scientific Computing*, Lecture Notes in Computational Science and Engineering, Berlin, Germany, 1998. Springer Verlag. 29, 35

[40] Philip Dutré, Eric Lafortune, and Yves D. Willems. Monte Carlo Light Tracing with Direct Pixel Contributions. In *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93)*, pages 128–137, Alvor, Portugal, December 1993. 35

[41] Philip Dutré and Yves D. Willems. Potential-Driven Monte Carlo Particle Tracing for Diffuse Environments with Adaptive Probability Density Functions. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 306–315, New York, NY, 1995. Springer-Verlag. 51, 53

[42] David S. Ebert and Richard E. Parent. Rendering and Animation of Gaseous Phenomena by Combining Fast Volume and Scanline A-buffer Techniques. In *Computer Graphics (ACM SIGGRAPH '90 Proceedings)*, volume 24, pages 357–366, August 1990. 18, 18, 23, 23

[43] Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Zeevi. The Farthest-Point Strategy for Progressive Image Sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315, September 1997. 76

[44] Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. Visual Simulation of Smoke. In *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2001)*, pages 15–22, August 2001. 28, 29, 36

[45] Ajeetkumar Gaddipatti, Raghu Machiraju, and Roni Yagel. Steering Image Generation with Wavelet Based Perceptual Metric. *Computer Graphics Forum (Eurographics '97 Proceedings)*, 16(3):241–252, 1997. 107

[46] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison Wesley Professional Computing Series, 1994. 94

[47] Geoffrey Y. Gardner. Visual Simulation of Clouds. *Computer Graphics (ACM SIGGRAPH '85 Proceedings)*, 19(3):297–303, July 1985. 18, 18, 20

[48] Simon Gibson and Roger J. Hubbold. Efficient Hierarchical Refinement and Clustering for Radiosity in Complex Environments. *Computer Graphics Forum*, 15(5):297–310, December 1996. 42

[49] Andrew S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann, San Francisco, CA, 1995. 5, 11

[50] Daniel R. Glover, Jr (web editor). *Dictionary of Technical Terms for Aerospace Use*. Cleveland, Ohio. Available from http://sulu.lerc.nasa.gov/dictionary/content.html. 12, 12, 13, 13

[51] Rafael C. Gonzalez and Paul Wintz. *Digital Image Processing*. Addison-Wesley, Reading, MA, 1977. 82

[52] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modelling the Interaction of Light Between Diffuse Surfaces. In *Computer Graphics (ACM SIGGRAPH '84 Proceedings)*, volume 18, pages 212–222, July 1984. 41

[53] Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. The Irradiance Volume. *IEEE Computer Graphics and Applications*, 18(2):32–43, March/April 1998. 56

[54] Baining Guo. Progressive Radiance Evaluation Using Directional Coherence Maps. In *Computer Graphics (ACM SIGGRAPH '98 Proceedings)*, pages 255–266, 1998. 76

[55] Jörg Haber, Karol Myszkowski, Hitoshi Yamauchi, and Hans-Peter Seidel. Perceptually Guided Corrective Splatting. In *Computer Graphics Forum (Proceedings of Eurographics 2001)*, volume 20, pages 142–152, September 2001. 90

[56] Eric Haines. A Proposal for Standard Graphics Environments. *IEEE Computer Graphics and Applications*, 7(11):3–5, November 1987. 8

[57] Pat Hanrahan and Wolfgang Krueger. Reflection from Layered Surfaces Due to Subsurface Scattering. In *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pages 165–174, 1993. 19

[58] Pat Hanrahan, David Salzman, and Larry Aupperle. A Rapid Hierarchical Radiosity Algorithm. In *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, volume 25, pages 197–206, July 1991. 2, 29, 30, 41, 47

[59] Mark J. Harris and Anselmo Lastra. Real-Time Cloud Rendering. In *Computer Graphics Forum (Proceedings of Eurographics 2001)*, volume 20, pages 76–84, September 2001. 28, 28, 28, 32, 33

[60] Mark J. Harris and Anselmo Lastra. Real-Time Cloud Rendering for Games. In *Proceedings of Game Developers Conference 2002*, March 2002. 28, 28, 28, 32, 33

[61] Paul S. Heckbert. ‘brdfview’: a reflection model viewer written by students in the course 15-860. Source code available from http://www.cs.cmu.edu/afs/cs/user/ph/www/src/illum/. 97

[62] Louis George Henyey and Jesse L. Greenstein. Diffuse Radiation in the Galaxy. *Astrophysical Journal*, 88:70–73, 1940. 14

[63] Hendrik C. van de Hulst. *Multiple Light Scattering, Tables, Formulas, and Applications*. Academic Press, 1980. 13

[64] Hendrik C. van de Hulst. *Light Scattering by Small Particles*. Dover Publications, New York, NY, 1981. 9

[65] Masa Inakage. An Illumination Model for Atmospheric Environments. In R. A. Earnshaw and B. Wyvill, editors, *New Advances in Computer Graphics*, pages 533–548. Springer-Verlag, New York, NY, 1989. 23, 25, 27

[66] John Irwin. Full-Spectral Rendering of the Earth’s Atmosphere Using a Physical Model of Rayleigh Scattering. In *Proceedings of the 1996 Eurographics UK Conference*, pages 103–115, 1996. 23, 26, 26, 27

[67] Akira Ishimaru. *Electromagnetic Wave Propagation, Radiation, and Scattering*. Prentice-Hall, 1991. 10

[68] Dietmar Jackèl and Bruce Walter. Modeling and Rendering of the Atmosphere Using Mie-Scattering. *Computer Graphics Forum*, 16(4):201–210, 1997. 27, 34

[69] Henrik Wann Jensen. Importance Driven Path Tracing Using the Photon Map. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques ’95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 326–335, New York, NY, 1995. Springer-Verlag. 51

[70] Henrik Wann Jensen. Global Illumination Using Photon Maps. In *Rendering Techniques ’96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30, New York, NY, 1996. Springer-Verlag/Wien. 36, 42

[71] Henrik Wann Jensen and Per H. Christensen. Efficient Simulation of Light Transport in Scenes with Participating Media Using Photon Maps. In *Computer Graphics (ACM SIGGRAPH ’98 Proceedings)*, pages 311–320, 1998. 28, 29, 36, 36, 36, 38, 39, 89

[72] Henrik Wann Jensen, Justin Legakis, and Julie Dorsey. Rendering of Wet Materials. In D. Lischinski and G. W. Larson, editors, *Rendering Techniques ’99 (Proceedings of the Tenth Eurographics Workshop on Rendering)*, pages 273–282, New York, NY, 1999. Springer Wien. 19

[73] Roberto Jiménez, Ignacio Martín, and Frederic Pérez. Materials and Geometry Format – Extended. World Wide Web: http://ima.udg.es/iiia/GGG/doc/mgfehtml/mgfe.shtml. 93, 108

[74] James T. Kajiya. The Rendering Equation. In *Computer Graphics (ACM SIGGRAPH ’86 Proceedings)*, volume 20, pages 143–150, August 1986. iii, 52

[75] James T. Kajiya and Brian P. Von Herzen. Ray Tracing Volume Densities. In *Computer Graphics (ACM SIGGRAPH ’84 Proceedings)*, volume 18, pages 165–174, July 1984. 21, 23, 28, 28, 31, 31

[76] Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo Methods. Volume I: Basics*. Wiley-Interscience, 1986. 102, 102

[77] Kazufumi Kaneda, Takashi Okamoto, Eihachiro Nakamae, and Tomoyuki Nishita. Photorealistic Image Synthesis for Outdoor Scenery Under Various Atmospheric Conditions. *Visual Computer*, 7(5):247–258, 1991. 23, 25, 27, 27, 33

[78] George W. Kattawar. A Three-Parameter Analytic Phase Function for Multiple Scattering Calculations. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 15:839–849, 1975. 14

[79] R. Victor Klassen. Modeling the Effect of Atmosphere on Light. *ACM Transactions on Graphics*, 6(3):215–237, 1987. 10, 12, 18, 23, 23, 25, 25, 25, 27, 27, 27

[80] Eric P. Lafortune. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*. PhD thesis, Leuven, Belgium, February 1996. 51, 53, 56, 103, 103, 103

[81] Eric P. Lafortune and Yves D. Willems. Using the Modified Phong BRDF for Physically Based Rendering. Technical Report CW197, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, November 1994. 97

[82] Eric P. Lafortune and Yves D. Willems. A 5D Tree to Reduce the Variance of Monte Carlo Ray Tracing. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 11–20, New York, NY, 1995. Springer-Verlag. 51

[83] Eric P. Lafortune and Yves D. Willems. Rendering Participating Media with Bidirectional Path Tracing. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 91–100, New York, NY, 1996. Springer-Verlag/Wien. 28, 29, 34, 35, 37, 51, 55, 100

[84] Thorsten Lange and Georg Pietrek. Rendering Participating Media Using the Photon Map. Technical Report 678/1998, Fachbereich Informatik, Universität Dortmund, 1998. 28, 29, 36

[85] Eric Languénou. *Radiosité Hiérarchique et Transfer Radiatif Dans Les Milieux Semi-Transparents*. PhD thesis, Université de Rennes, October 1994. Available (in French) from http://www.sciences.univ-nantes.fr/info/perso/permanents/languenou/renderWork.html. 31

[86] Eric Languénou, Kadi Bouatouch, and Michael Chelle. Global Illumination in Presence of Participating Media with General Properties. In *Fifth Eurographics Workshop on Rendering*, pages 69–85, Darmstadt, Germany, June 1994. 28, 28, 28, 31, 32, 38, 39, 39, 39

[87] Eric Languénou, Michael Chelle, and Kadi Bouatouch. Simulation d'Eclairage dans un Environnement Contenant des Milieux Semi-Transparents. Research Report 800, INRIA, Rennes, France, Mars 1994. 13

[88] Greg Ward Larson and Rob Shakespeare. *Rendering with Radiance: The Art and Science of Lighting Visualization*. Morgan Kaufmann, San Francisco, CA, 1998. 75, 107

[89] Kaye D. Lathrop. Ray Effects in Discrete Ordinates Equations. *Nuclear Science and Engineering*, 32:357–369, 1968. 31

[90] Pascal Lecocq, Sylvain Michelin, Didier Arquès, and Andras Kemeny. Simulation d'Éclairage en Présence de Milieux Participatifs : Vers une Solution Temps-Réel. *AFIG '00 (Actes des 13ièmes journées de l'AFIG)*, December 2000. 23, 26, 27, 27, 27

[91] Pascal Lecocq, Sylvain Michelin, Andras Kemeny, and Didier Arquès. Real Time Lighting Simulation in Presence of Fog: Applications for Driving Simulation. In *Proceedings of the Driving Simulation Conference 2002*, Paris, France, September 2002. 23, 26, 27, 27, 27

[92] Marc Levoy. Display of Surfaces From Volume Data. *IEEE Computer Graphics and Applications*, 8(3):29–37, May 1988. 21

[93] Ivan Lux and László Koblinger. *Monte Carlo Particle Transport Methods: Neutron and Photon Calculations*. CRC Press, 1991. 104

[94] Jean-Luc Maillot, Laurent Carraro, and Bernard Péroche. Progressive Ray-Tracing. In D. Paddon, A. Chalmers, and F. Sillion, editors, *Rendering Techniques '92*, Eurographics, pages 9–20. Consolidation Express Bristol, 1992. 76

[95] Nelson L. Max. Atmospheric Illumination and Shadows. In *Computer Graphics (ACM SIGGRAPH '86 Proceedings)*, volume 20, pages 117–124, August 1986. 18, 23, 23, 24, 24, 27, 27, 27, 27

[96] Nelson L. Max. Efficient Light Propagation for Multiple Anisotropic Volume Scattering. In *Fifth Eurographics Workshop on Rendering*, pages 87–104, Darmstadt, Germany, June 1994. 14, 28, 28, 28, 32, 38, 38, 39, 39

[97] Michael D. McCool. Anisotropic Diffusion for Monte Carlo Noise Reduction. *ACM Transactions on Graphics*, 18(2):171–194, April 1999. 2, 78

[98] Nicholas Constantine Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087–1092, 1953. 36

[99] W. J. Minkowycz, E. M. Sparrow, G. E. Schneider, and R. H. Pletcher, editors. *Handbook of Numerical Heat Transfer*. John Wiley & Sons, New York, NY, 1988. 100

[100] Don P. Mitchell. Generating Antialiased Images at Low Sampling Densities. *Computer Graphics (ACM SIGGRAPH '87 Proceedings)*, 21(4):65–72, 1987. 76, 80

[101] Sudhir P. Mudur and Sumanta N. Pattanaik. Multidimensional Illumination Functions for Visualization of Complex 3D Environments. *The Journal of Visualization and Computer Animation*, 1(2):49–58, 1990. 56

[102] Karol Myszkowski. The Visible Differences Predictor: Applications to Global Illumination Problems. In G. Drettakis and N. Max, editors, *Rendering Techniques '98 (Proceedings of Eurographics Rendering Workshop '98)*, pages 233–236, New York, NY, 1998. Springer Wien. 107

[103] Ignacio Martín, Frederic Pérez, and Xavier Pueyo. The SIR Rendering Architecture. *Computers & Graphics*, 22(5):601–609, 1998. 57, 91, 93, 93, 103, 107

[104] László Neumann, Kresimir Matkovic, and Werner Purgathofer. Perception Based Color Image Difference. *Computer Graphics Forum (Proc. Eurographics '98*, 17(3):233–241, September 1998. 107

[105] Duc Quang Nguyen, Ronald P. Fedkiw, and Henrik Wann Jensen. Physically Based Modeling and Animation of Fire. *ACM Transactions on Graphics*, 21(3):721–728, 2002. 8

[106] Tomoyuki Nishita, Yoshinori Dobashi, Kazufumi Kaneda, and Hideo Yamashita. Display Method of the Sky Color Taking Into Account Multiple Scattering. In *Proceedings of the Fourth Pacific Conference on Computer Graphics and Applications (Pacific Graphics '96)*, pages 66–79, 1996. 28, 28, 28, 32, 33, 33, 34

[107] Tomoyuki Nishita, Yoshinori Dobashi, and Eihachiro Nakamae. Display of Clouds Taking Into Account Multiple Anisotropic Scattering and Sky Light. In *Computer Graphics Proceedings, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings)*, pages 379–386, 1996. 8, 28, 28, 28, 32, 32, 32, 33

[108] Tomoyuki Nishita, Yashuhiro Miyawaki, and Eihachiro Nakamae. A Shading Model for Atmospheric Scattering Considering Luminous Intensity Distribution of Light Sources. In *Computer Graphics (ACM SIGGRAPH '87 Proceedings)*, volume 21, pages 303–310, July 1987. 13, 13, 18, 23, 24, 24, 25, 25, 26, 27, 27, 27, 27

[109] Tomoyuki Nishita and Eihachiro Nakamae. Continuous Tone Representation of Three-Dimensional Objects Taking Account of Shadows and Interreflection. In *Computer Graphics (ACM SIGGRAPH '85 Proceedings)*, volume 19, pages 23–30, July 1985. 2

[110] Tomoyuki Nishita, Takao Sirai, Katsumi Tadamura, and Eihachiro Nakamae. Display of the Earth Taking Into Account Atmospheric Scattering. In *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pages 175–182, 1993. 14, 15, 23, 25

[111] Irena Notkin and Craig Gotsman. Parallel Progressive Ray-Tracing. *Computer Graphics Forum*, 16(1):43–55, March 1997. 76

[112] James Painter and Kenneth Sloan. Antialiased Ray Tracing by Adaptive Progressive Refinement. *Computer Graphics*, 23(3):281–288, July 1989. 76, 76, 80

[113] Chris Patmore. Simulated Multiple Scattering for Cloud Rendering. In S. P. Mudur and S. N. Pattanaik, editors, *Graphics, Design and Visualization (IFIP Transactions B-9)*, pages 59–70, Amsterdam, The Netherlands, 1993. North-Holland. 28, 28, 28, 31, 39

[114] Sumanta N. Pattanaik and Sudhir P. Mudur. Computation of Global Illumination in a Participating Medium by Monte Carlo Simulation. *The Journal of Visualization and Computer Animation*, 4(3):133–152, July - September 1993. 28, 29, 34, 35, 35, 36, 37, 38, 38, 46, 47, 51, 55, 100, 100, 103

[115] Sumanta N. Pattanaik and Sudhir P. Mudur. Efficient Potential Equation Solutions for Global Illumination Computation. *Computers & Graphics*, 17(4):387–396, 1993. 51

[116] Mark Pauly, Thomas Kollig, and Alexander Keller. Metropolis Light Transport for Participating Media. In B. Péroche and H. Rushmeier, editors, *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)*, pages 11–22, New York, NY, 2000. Springer Wien. 28, 29, 36

[117] Frederic Pérez, Ignacio Martín, and Xavier Pueyo. Hierarchical Monte Carlo Radiosity for General Environments. Technical Report IIiA 02-01-RR, Institut d'Informàtica i Aplicacions, Universitat de Girona, Girona, Spain, November 2001. Available from http://ima.udg.es/˜frederic/publications.html. 46, 51, 51, 91

[118] Frederic Pérez, Ignacio Martín, and Xavier Pueyo. High Quality Final Gathering for Hierarchical Monte Carlo Radiosity for General Environments. In J. Vince and R. Earnshaw, editors, *Advances in Modelling, Animation and Rendering (Proc. of Computer Graphics International 2002)*, pages 425–437. Springer, 2002. 46, 51, 51, 91

[119] Frederic Pérez, Ignacio Martín, and Xavier Pueyo. Progressive Radiance Computation Based on Conductance Maps, August 2002. Submitted to *The Visual Computer* journal. 75, 91

[120] Frederic Pérez, Ignacio Martín, François Sillion, and Xavier Pueyo. Acceleration of Monte Carlo Path Tracing in the Presence of Anisotropic Scattering Media. Technical Report IIiA 98-25-RR, Institut d'Informàtica i Aplicacions, Universitat de Girona, Girona, Spain, November 1998. Available from http://ima.udg.es/~frederic/publications.html. 43, 51, 53, 91

[121] Frederic Pérez, Ignacio Martín, François X. Sillion, and Xavier Pueyo. Acceleration of Monte Carlo Path Tracing in General Environments. In *Proceedings of Pacific Graphics 2000*, Hong Kong, PRC, October 2000. 43, 51, 51, 91

[122] Frederic Pérez, Xavier Pueyo, and François X. Sillion. Global Illumination Techniques for the Simulation of Participating Media. In J. Dorsey and Ph. Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 309–320, New York, NY, 1997. Springer Wien. 2, 27, 91

[123] Ken Perlin. An Image Synthesizer. *Computer Graphics (ACM SIGGRAPH '85 Proceedings)*, 19(3):287–296, August 1985. 18, 20

[124] Frédéric P. Pighin, Dani Lischinski, and David Salesin. Progressive Previewing of Ray-Traced Images Using Image-Plane Discontinuity Meshing. In J. Dorsey and Ph. Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 115–125, New York, NY, 1997. Springer Wien. 76, 76, 77, 77, 80

[125] Thomas Porter and Tom Duff. Compositing Digital Images. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 253–259, July 1984. 21

[126] Arcot J. Preetham, Peter Shirley, and Brian E. Smits. A Practical Analytic Model for Daylight. In *Computer Graphics Proceedings, Annual Conference Series (Proc. SIGGRAPH '99)*, pages 91–100, August 1999. 28, 33

[127] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing, Second Edition*. Cambridge University Press, Cambridge, 1992. 103

[128] Werner Purgathofer. A Statistical Method for Adaptive Stochastic Sampling. In A. A. G. Requicha, editor, *Proceedings of Eurographics '86*, pages 145–152, 1986. 56, 104

[129] Mahesh Ramasubramanian, Sumanta N. Pattanaik, and Donald P. Greenberg. A Perceptually Based Physical Error Metric for Realistic Image Synthesis. In *Computer Graphics Proceedings, Annual Conference Series (Proc. SIGGRAPH '99)*, pages 73–82, August 1999. 91

[130] Amit Reisman, Craig Gotsman, and Assaf Schuster. Parallel Progressive Rendering of Animation Sequences at Interactive Rates on Distributed-Memory Machines. In J. Painter, G. Stoll, and K.-L. Ma, editors, *IEEE Parallel Rendering Symposium*, pages 39–48. IEEE, November 1997. 76

[131] Amit Reisman, Craig Gotsman, and Assaf Schuster. Interactive-Rate Animation Generation by Parallel Progressive Ray-Tracing on Distributed-Memory Machines. *Journal of Parallel and Distributed Computing*, 60(9):1074–1102, September 2000. 76

[132] Badrinath Roysam, Andrew Cohen, Philip Getto, and Peter Boyce. A Numerical Approach to the Computation of Light Propagation Through Turbid Media: Application to the Evaluation of Lighted Exit Signs. In *IEEE Conference on Industry Applications*, pages 661–669, Detroit, MI, 1993. 35

[133] Holly E. Rushmeier. *Realistic Image Synthesis for Scenes with Radiatively Participating Media*. PhD thesis, Ithaca, NY, 1988. 23, 25

[134] Holly E. Rushmeier. Rendering Participating Media: Problems and Solutions from Application Areas. In *Fifth Eurographics Workshop on Rendering*, pages 35–56, Darmstadt, Germany, June 1994. 19

[135] Holly E. Rushmeier. Input for Participating Media. In *ACM SIGGRAPH '95 Course Notes - Realistic Input for Realistic Images*, 1995. 12, 16

[136] Holly E. Rushmeier and Kenneth E. Torrance. The Zonal Method for Calculating Light Intensities in the Presence of a Participating Medium. In *Computer Graphics (ACM SIGGRAPH '87 Proceedings)*, volume 21, pages 293–302, July 1987. 8, 11, 27, 28, 29

[137] Georgios Sakas. Fast Rendering of Arbitrary Distributed Volume Densities. In C. E. Vandoni and D. A. Duce, editors, *Eurographics '90*, pages 519–530, Amsterdam, Netherlands, 1990. North-Holland. 10, 17, 17, 18, 23, 25, 27, 27

[138] Georgios Sakas. Modeling and Animating Turbulent Gaseous Phenomena Using Spectral Synthesis. *The Visual Computer*, 9(4):200–212, January 1993. 18

[139] Georgios Sakas and Matthias Gerth. Sampling and Anti-Aliasing of Discrete 3-D Volume Density Textures. In *Eurographics '91*, pages 87–102, 527, Amsterdam, North-Holland, September 1991. Elsevier Science Publishers. 25

[140] Georgios Sakas and Jochen Hartig. Interactive Visualization of Large Scalar Voxel Fields. In A. Kaufman and G. M. Nielson, editors, *Proceedings Visualization '92*, Boston, Massachusets, October 1992. 21, 21

[141] Gernot Schaufler. Dynamically Generated Impostors. *GI Workshop on Modeling, Virtual Worlds, Distributed Graphics*, pages 129–139, 1995. 33

[142] Annette Scheel, Marc Stamminger, Jörg Putz, and Hans-Peter Seidel. Enhancements to Directional Coherence Maps. In *Ninth International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media (WSCG 2001)*, Plzen, Czech Republic, February 2001. University of West Bohemia. Available from http://wscg.zcu.cz/wscg2001. 76

[143] Christophe Schlick. A Survey of Shading and Reflectance Models. *Computer Graphics Forum*, 13(2):121–131, June 1994. 97

[144] Peter Shirley. A Ray Tracing Method for Illumination Calculation in Diffuse-Specular Scenes. In *Proceedings of Graphics Interface '90*, pages 205–212, San Francisco, CA, May 1990. Morgan Kaufmann. 2

[145] Peter Shirley, Kelvin Sung, and William Brown. A Ray Tracing Framework for Global Illumination Systems. In *Proceedings of Graphics Interface '91*, pages 117–128, San Francisco, CA, June 1991. Morgan Kaufmann. 95

[146] Robert Siegel and John R. Howell. *Thermal Radiation Heat Transfer, 3rd Edition*. Hemisphere Publishing Corporation, New York, NY, 1992. iii, 6, 7, 8, 10, 10, 11, 11, 12, 13, 14, 16, 16, 17, 31, 45

[147] François Sillion. A Unified Hierarchical Algorithm for Global Illumination with Scattering Volumes and Object Clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):240–254, September 1995. 2, 27, 28, 30, 30, 37, 38, 42, 43, 43, 45, 87

[148] François Sillion, George Drettakis, and Cyril Soler. A Clustering Algorithm for Radiance Calculation in General Environments. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 196–205, New York, NY, 1995. Springer-Verlag. 2, 37, 42, 43, 43, 43, 45, 48, 87

[149] François Sillion and Claude Puech. A General Two-Pass Method Integrating Specular and Diffuse Reflection. In *Computer Graphics (ACM SIGGRAPH '89 Proceedings)*, volume 23, pages 335–344, July 1989. 2

[150] François Sillion and Claude Puech. *Radiosity and Global Illumination*. Morgan Kaufmann, San Francisco, CA, 1994. 5, 5, 6, 41

[151] François Sillion, George Drettakis, and Benoit Bodelet. Efficient Impostor Manipulation for Real-Time Visualization of Urban Scenery. *Computer Graphics Forum (Eurographics '97 Proceedings)*, 16(3):207–218, 1997. 77

[152] Maryann Simmons and Carlo H. Séquin. Tapestry: A Dynamic Mesh-based Display Representation for Interactive Rendering. In B. Péroche and H. Rushmeier, editors, *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)*, pages 329–340, New York, NY, 2000. Springer Wien. 76

[153] Philipp Slusallek. *Vision - An Architecture for Physically Based Image Synthesis*. PhD thesis, Erlangen, Germany, June 1995. 93

[154] Brian Smits, James Arvo, and Donald Greenberg. A Clustering Algorithm for Radiosity in Complex Environments. In *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings)*, pages 435–442, 1994. 37, 42

[155] Lisa Marie Sobierajski. *Global Illumination Models for Volume Rendering*. PhD thesis, Stony Brook, NY, August 1994. 27, 28, 29, 30

[156] Ilya M. Sobol. *Método de Montecarlo, Second Edition*. Mir, Moscow, 1983. 100

[157] Jos Stam. Stochastic Rendering of Density Fields. In *Proceedings of Graphics Interface '94*, pages 51–58, San Francisco, CA, May 1994. Morgan Kaufmann. 18, 23, 26, 26

[158] Jos Stam. *Multi-Scale Stochastic Modelling of Complex Natural Phenomena*. PhD thesis, University of Toronto, Dept. of Computer Science, 1995. 7, 18, 28, 28, 28, 32, 33, 37, 38, 38, 38

[159] Jos Stam. Multiple Scattering as a Diffusion Process. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 41–50, New York, NY, 1995. Springer-Verlag. 28, 28, 28, 32, 33, 37, 38, 38, 38

[160] Jos Stam and Eugene Fiume. Turbulent Wind Fields for Gaseous Phenomena. In *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pages 369–376, 1993. 23, 25, 26

[161] Jos Stam and Eugene Fiume. Depicting Fire and Other Gaseous Phenomena Using Diffusion Processes. In *Computer Graphics Proceedings, Annual Conference Series, 1995 (ACM SIGGRAPH '95 Proceedings)*, pages 129–136, 1995. 11, 28, 28, 28, 32, 33, 37, 38, 38, 38

[162] Marc Stamminger and George Drettakis. Perspective Shadow Maps. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002 Annual Conference)*, 21(3):557–562, 2002. 90

[163] Marc Stamminger, Annette Scheel, Xavier Granier, Frederic Pérez, George Drettakis, and François Sillion. Efficient Glossy Global Illumination with Interactive Viewing. In *Graphics Interface '99*, San Francisco, CA, June 1999. Morgan Kaufmann. 62, 91, 93, 93

[164] Marc Stamminger, Annette Scheel, Xavier Granier, Frederic Pérez, George Drettakis, and François Sillion. Efficient Glossy Global Illumination with Interactive Viewing. *Computer Graphics Forum*, 19(1):13–25, 2000. 48, 89, 91, 93, 93

[165] Wolfgang Sturzlinger. Optimized Local Pass Using Importance Sampling. In *WSCG 96 (Fourth International Conference in Central Europe on Computer Graphics and Visualization)*, volume 2, pages 342–348, Plzen, Czech Republic, 1996. University of West Bohemia. 52, 62

[166] Frank Suykens and Yves D. Willems. Weighted Multipass Methods for Global Illumination. In *Computer Graphics Forum (Proc. Eurographics '99)*, volume 18, pages C–209–C–220, September 1999. 2, 43

[167] László Szirmay-Kalos, Balázs Csébfalvi, and Werner Purgathofer. Importance Driven Quasi-Random Walk Solution of the Rendering Equation. In V. Skala, editor, *WSCG '98 (Sixth European Conference in Central Europe on Computer Graphics and Visualization)*, pages 379–385, Plzen, Czech Republic, 1998. University of West Bohemia. 52, 53

[168] Katsumi Tadamura, Eihachiro Nakamae, Kazufumi Kaneda, Masshi Baba, Hideo Yamashita, and Tomoyuki Nishita. Modeling of Skylight and Rendering of Outdoor Scenes. In *Computer Graphics Forum (Eurographics '93)*, volume 12, pages C189–C200, Barcelona, Spain, September 1993. 13, 13, 23, 25, 25, 27

[169] Seth Teller, Kavita Bala, and Julie Dorsey. Conservative Radiance Interpolants for Ray Tracing. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 257–268, New York, NY, 1996. Springer-Verlag/Wien. 77

[170] Fabrice Uhl and Jacques Blanc-Talon. Rendering Explosions. In *SCS/SMC 97*, April 1997. 8

[171] Craig Upson and Michael Keeler. V-BUFFER: Visible Volume Rendering. In *Computer Graphics (ACM SIGGRAPH '88 Proceedings)*, volume 22, pages 59–64, August 1988. 21

[172] Carlos Ureña and Juan C. Torres. Improved Irradiance Computation by Importance Sampling. In J. Dorsey and Ph. Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 275–284, New York, NY, 1997. Springer Wien. 51, 51, 62

[173] Eric Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, December 1997. Available from http://graphics.stanford.edu/papers/veach_thesis. 5, 48

[174] Eric Veach and Leonidas J. Guibas. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Computer Graphics Proceedings, Annual Conference Series, 1995 (ACM SIGGRAPH '95 Proceedings)*, pages 419–428, 1995. 35

[175] Eric Veach and Leonidas J. Guibas. Metropolis Light Transport. In *Computer Graphics (ACM SIGGRAPH '97 Proceedings)*, volume 31, pages 65–76, 1997. 36

[176] Ingo Wald and Philipp Slusallek. State of the Art in Interactive Ray Tracing. In *State of the Art Reports, EUROGRAPHICS 2001*, pages 21–42. EUROGRAPHICS, Manchester, United Kingdom, 2001. 77

[177] John R. Wallace, Michael F. Cohen, and Donald P. Greenberg. A Two-Pass Solution to the Rendering Equation: A Synthesis of Ray Tracing and Radiosity Methods. In *Computer Graphics (ACM SIGGRAPH '87 Proceedings)*, volume 21, pages 311–320, July 1987. 2

[178] Bruce Walter, George Drettakis, and Steven Parker. Interactive Rendering using the Render Cache. In D. Lischinski and G. W. Larson, editors, *Rendering Techniques '99*, Eurographics, pages 19–30. Springer-Verlag Wien New York, 1999. 76

[179] Bruce Walter, Sumanta N. Pattanaik, and Donald P. Greenberg. Using Perceptual Texture Masking for Efficient Image Synthesis. *Computer Graphics Forum (Proceedings of Eurographics 2002)*, 21(3), September 2002. 90, 91

[180] Changyaw Wang. Physically Correct Direct Lighting for Distribution Ray Tracing. In D. Kirk, editor, *Graphics Gems III*, pages 307–313. Academic Press Professional, Boston, MA, 1992. 63

[181] Lihong Wang and Steven L. Jacques. Monte Carlo Modeling of Light Transport in Multi-layered Tissues in Standard C (Manual of the program 'mcml'), 1992. 11

[182] Gregory J. Ward. Materials and Geometry Format. World Wide Web: http://radsite.lbl.gov/mgf/HOME.html. 93, 108

[183] Gregory J. Ward and Maryann Simmons. The Holodeck Ray Cache: An Interactive Rendering System for Global Illumination in Nondiffuse Environments. *ACM Transactions on Graphics*, 18(4):361–398, 1999. 76, 77, 77

[184] Alan Watt and Fabio Policarpo. *The Computer Image*. Addison-Wesley, 1998. 82, 82

[185] Lee Westover. Footprint Evaluation for Volume Rendering. *Computer Graphics*, 24(4):367–376, August 1990. 33

[186] Philip J. Willis. Visual Simulation of Atmospheric Haze. *Computer Graphics Forum*, 6:35–42, 1987. 18, 23, 24, 27

[187] Mason Woo, Jackie Neider, Tom Davis, and OpenGL Architecture Review Board. *OpenGL Programming Guide: the Official Guide to Learning OpenGL, version 1.2, Third Edition*. Addison-Wesley, Reading, MA, 1999. 49, 81

[188] Larry Yaeger, Craig Upson, and Robert Myers. Combining Physical and Visual Simulation - Creation of the Planet Jupiter for the Film 2010. *Computer Graphics (ACM SIGGRAPH '86 Proceedings)*, 20(4):85–93, August 1986. 18, 20

[189] Kurt Zimmerman. *Density Prediction for Importance Sampling in Realistic Image Synthesis*. PhD thesis, Indiana University, May 1998. 65