UNIVERSITY OF CALIFORNIA, SAN DIEGO

EFFICIENT MONTE CARLO METHODS FOR LIGHT TRANSPORT IN SCATTERING MEDIA

A dissertation submitted in partial satisfaction of the

requirements for the degree

Doctor of Philosophy

in

Computer Science

by

Wojciech Jarosz

Committee in charge:

Henrik Wann Jensen, Co-Chair
Matthias Zwicker, Co-Chair
Samuel R. Buss
Per H. Christensen
David J. Kriegman
Falko Kuester

2008

The dissertation of Wojciech Jarosz is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____
Co-Chair

_____
Co-Chair

University of California, San Diego

2008

To Dorota and Krzysztof

TABLE OF CONTENTS

LIST OF TABLES

ACKNOWLEDGEMENTS

I owe many people an immense debt of gratitude for the help and support they have provided me over the course of my life and academic career.

I would like to thank my friends and family for supporting me along the way. First and foremost I would like to thank my parents Dorota and Krzysztof, from whom I inherited the interest in a topic that straddles the artistic and analytic sides of my brain, and without whom I would not be here today. This dissertation is dedicated to them.

It has been a great privilege to work with both of my advisors, Henrik Wann Jensen and Matthias Zwicker. Their complementary research approaches and advising styles have been immensely influential in the development of my work. The combined guidance I received from them has been invaluable in the culmination of this dissertation. I also owe a debt of gratitude to the other members of my dissertation committee, Sam Buss, Per Christensen, David Kriegman, and Falko Kuester, for agreeing to take their valuable time to read and evaluate this dissertation. I would furthermore like to thank my undergraduate research advisors John C. Hart and Michael Garland for introducing me to computer graphics research and giving me hands-on experience at an early stage in my academic career. I would also like to thank my high school programming and calculus teacher James Hindelang.

This dissertation would not have been possible if not for the stimulating environment created by the other members and visitors of the UCSD graphics lab, in particular: Neil Alldrin, Will Chang, Piotr Dollar, Diego Gutierrez, Toshiya Hachisuka, Neel Joshi, Arash Keshmirian, Alex Kozlowski, Wan-Yen Lo, Krystle de Mesa, Iman Mostafavi, Vincent Rabaud, Steve Rotenberg, and Iman Sadeghi. I thank each of them for their fruitful suggestions, countless conversations, and for agreeing to read my papers. I'd also like to thank everyone with whom I had the privilege of collaborating: Craig Donner, Tomas Akenine-Möller, Petrik Clarberg, Kevin Dale, Frédo Durand, Greg Humphreys, Oleg Kozhushnyan, Wojciech Matusik, Sylvain Paris, and Richard Weistroffer.

I would like to thank Industrial Light & Magic for providing me with three exciting and stimulating summer internships and for making the movies that sparked my interest in computer graphics in the first place. I would particularly like to thank Florian Kainz with whom I worked

for three summers and whose knowledge and experience made them so rewarding. I would also like to thank my friend of countless years, Eric Froemling, who first inspired me to take up 3D modeling and animation, which led to where I am now.

Finally, I would particularly like to thank Kelly, my fiance, for sharing this chapter of my life with me and embarking with me on the next one.

Portions of this dissertation are based on papers which I have co-authored with others. My contributions to each of these papers is listed below.

- Chapter 5 is a reproduction of the material published in the article:

  Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. "Radiance Caching for Participating Media." In *ACM Transactions on Graphics*, 27(1):1–11, 2008.

  I was the primary investigator and author of this paper.

- Chapter 6 is based on the work that appears in the paper:

  Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "Irradiance Gradients in the Presence of Participating Media." In *Computer Graphics Forum (Proceedings of EGSR 2008)*, 27(4), 2008.

  I was the primary investigator and author of this paper.

- Chapter 8 is based on material published in the following paper and extended technical report:

  Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "The Beam Radiance Estimate for Volumetric Photon Mapping." In *Computer Graphics Forum (Proceedings of Eurographics 2008)*, 27(2):557–566, 2008.

  Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "The Beam Radiance Estimate for Volumetric Photon Mapping." Technical Report CS2008-0914, University of California, San Diego, 2008.

  I was the primary investigator and author of both papers.

VITA

2003	Bachelor of Science, *highest honors*, University of Illinois, Urbana–Champaign
2003–2008	Teaching Assistant, Department of Computer Science and Engineering, University of California, San Diego
2006	Master of Science, University of California, San Diego
2008	Doctor of Philosophy, University of California, San Diego

PUBLICATIONS

Toshiya Hachisuka, Wojciech Jarosz, Richard Peter Weistroffer, Kevin Dale, Greg Humphreys, Matthias Zwicker, and Henrik Wann Jensen. "Multidimensional Adaptive Sampling and Reconstruction for Ray Tracing." In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008)*, 27(3), 2008.

Sylvain Paris, Will Chang, Oleg I. Kozhushnyan, Wojciech Jarosz, Wojciech Matusik, Matthias Zwicker, and Frédo Durand. "Hair Photobooth: Geometric and Photometric Acquisition of Real Hairstyles." In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008)*, 27(3), 2008.

Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "Irradiance Gradients in the Presence of Participating Media." In *Computer Graphics Forum (Proceedings of EGSR 2008)*, 27(4):1087–1096, 2008.

Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "The Beam Radiance Estimate for Volumetric Photon Mapping." In *Computer Graphics Forum (Proceedings of Eurographics 2008)*, 27(2):557–566, 2008.

Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. "Radiance Caching for Participating Media." In *ACM Transactions on Graphics*, 27(1):1–11, 2008.

Petrik Clarberg, Wojciech Jarosz, Tomas Akenine-Möller, and Henrik Wann Jensen. "Wavelet Importance Sampling: Efficiently Evaluating Products of Complex Functions." In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, 24(3):1166–1175, 2005.

John C. Hart, Ed Bachta, Wojciech Jarosz, Terry Fleury. "Using Particles to Sample and Control More Complex Implicit Surfaces." In *Proceedings of Shape Modeling International 2002*, May 2002, 129–136.

ABSTRACT OF THE DISSERTATION

EFFICIENT MONTE CARLO METHODS FOR LIGHT TRANSPORT IN SCATTERING MEDIA

by

Wojciech Jarosz

Doctor of Philosophy in Computer Science

University of California San Diego, 2008

Henrik Wann Jensen, Co-Chair

Matthias Zwicker, Co-Chair

In this dissertation we focus on developing accurate and efficient Monte Carlo methods for synthesizing images containing general participating media. Participating media such as clouds, smoke, and fog are ubiquitous in the world and are responsible for many important visual phenomena which are of interest to computer graphics as well as related fields. When present, the medium *participates* in lighting interactions by scattering or absorbing photons as they travel through the scene. Though these effects add atmosphere and considerable depth to rendered images they are computationally very expensive to simulate. Most practical solutions make simplifying assumptions about the medium in order to maintain efficiency. Unfortunately, accurate and efficient simulation of light transport in general scattering media is a challenging undertaking. In this dissertation, we address this problem by introducing two complementary techniques.

We first turn to the irradiance caching method for surface illumination. Irradiance caching gains efficiency by computing an accurate representation of lighting only at a sparse set of locations and reusing these values through interpolation whenever possible. We derive the mathematical concepts that form the foundation of this approach and analyze its strengths and weaknesses. Drawing inspiration from this algorithm, we then introduce a novel volumetric

radiance caching method for efficiently simulating global illumination within participating media. In developing the technique we also introduce efficient methods for evaluating the gradient of the lighting within participating media. Our gradient analysis has immediate applicability for improved interpolation quality in both surface and media-based caching methods.

We also develop a novel photon mapping technique for participating media. We present a theoretical reformulation of volumetric photon mapping, which provides significant new insights. This reformulation makes it easier to qualify the error introduced by the radiance estimate but, more importantly, also allows us to develop more efficient rendering techniques. Conventional photon mapping accelerate the computation of lighting at any *point* in the scene by performing density estimation. In contrast, our reformulation accelerates the computation of accumulated lighting along the length of entire *rays*. This algorithmic improvement provides for significantly reduced render times and even the potential for real-time visualization of light transport in participating media.

1

Introduction

"Make it so."

—Captain Jean-Luc Picard, 2305–

EVERYDAY we are exposed to an immense amount of visual information. The natural world is filled with beautiful and complex phenomena which have sparked the curiosity of humankind for millennia. Throughout history we have struggled to find new ways to understand, manipulate, and, in fact, create the world around us for purposes ranging from communication, to medicine, to art and entertainment.

The visual arts and sciences have been following parallel paths in our quest for understanding the nature of the world around us. These seemingly distinct disciplines have been inextricably linked in the development of this understanding. Artists have readily utilized existing scientific and mathematical knowledge to create more convincing works of art. For instance, the development of contrapposto by classical Greek sculptors such as Polykleitos in the 4th century BCE relied heavily on a system of ideal mathematical proportions initially investigated by Pythagoras and Euclid [Tanner, 2006]. The understanding of the behavior of light perhaps bridges these two disciplines most since it is light which determines the visual appearance of the natural world. Often discoveries were made independently in both fields, like when Renoir and Monet rediscovered indirect illumination by observing that shadows are not brown or black, but the color of the surrounding environment. At other times the physical sciences have only been able to explain phenomena years after artists had developed an intuitive understanding. Leonardo Da Vinci, for instance, introduced *aerial perspective* in his *Treatise on Painting* [1651] to describe

the effects of light scattering in the atmosphere centuries before a scientific understanding of volumetric scattering was established. In fact, these disciplines are so intertwined that historically, practitioners of one discipline were often heavily ingrained in or influenced by the other. This quality is perhaps embodied most by Da Vinci, but other notable examples include Archimedes, Su Song, Michelangelo, Piero della Francesca, Albrecht Dürer, Vermeer, and M.C. Escher. Recently we have seen a convergence between many aspects of these disciplines with the introduction of photography, and now the computer.

Today, the understanding fueled by these two disciplines, combined with the advent of the computer, has provided us with unprecedented potential for simulating our world. The introduction of computers has spawned a new discipline at the crossroads of these two fields: computer graphics. The goal of computer graphics is to develop algorithms that allow a user to digitally synthesize and manipulate visual content. As in art, the visual content we are typically interested in synthesizing relates to the natural world and therefore depends on the physical sciences.

Physically based rendering is concerned with generating, or rendering, images by modeling the way the physical world behaves. A related goal is photorealistic rendering, which strives to synthesize images that are indistinguishable from photographs. Both of these disciplines are primarily concerned with simulating the physical behavior of light as it interacts in virtual environments in order to generate realistic images. Applications of realistic image synthesis include design, architecture, and education. In particular, we have recently witnessed an explosion in the widespread use of computer graphics and realistic image synthesis in journalism, games, television, movies, as well as virtually all forms of media. This is made evident by the entertainment industry's extensive use of computer generated imagery and special effects.

As the field has matured, computer generated images in many of the aforementioned areas are becoming increasingly complex, and the desire for unparalleled realism is always increasing. Theoretical understanding combined with increased computational power has allowed us to simulate virtually every observable or imaginable phenomenon. Unfortunately, many of the most interesting phenomena are also some of the most computationally intensive, fueling the need for more efficient, general purpose algorithms. One example of particular interest to the computer

graphics community is simulating participating media.

Participating, or scattering, media influences virtually everything we see around us: from the color of the sky at sunset, to the appearance of milk, to the reduced visibility we encounter in fog or rain. Participating media gives rise to pyrotechnic effects such as fire, smoke, clouds, and explosions, which are of particular interest to the special effects industry. Furthermore, paper, marble, and virtually all organic materials exhibit subsurface scattering, which is a special case of the volumetric scattering inherent to participating media. See Figure 1.1 for a few examples of the striking and visually complex effects caused by participating media. Unfortunately, rendering images with participating media is an especially difficult problem. The behavior of light transport becomes significantly more complex when participating media is considered, since photons may interact at any point within the volume instead of just at surface boundaries. For this reason, many rendering algorithms assume that all objects are in a vacuum in order to simplify the lighting simulation needed. At times this can be a reasonable approximation, but there are many situations when the effects from participating media have a significant visual impact (see Figure 1.1). These are not pathological cases by any means, and are in fact of considerable interest to many disciplines outside of computer graphics, including heat transfer and neutron transport.

Due to increased computational power and the ubiquity of participating media, there has been a steady increase in techniques capable of rendering volumetric effects. Due to the complexity of the problem, many of the most successful techniques have made simplifying assumptions about the medium being simulated. For example, homogeneous media with a high scattering albedo, such as milk, can be modeled accurately using a diffusion approximation. Diffusion was originally developed to explain the transport of neutrons in nuclear reactors [Glasstone and Sesonske, 1955], but also leads to very efficient rendering algorithms [Stam, 1995; Jensen et al., 2001b]. On the other extreme, the assumption of non-scattering but absorbing media makes the light transport problem significantly more tractable. Tenuous media with a low scattering albedo can also be effectively approximated by assuming only a single scattering event between the light source and the eye [Blinn, 1982; Nishita et al., 1987]. In isotropic media, which scatters light equally in all directions, the lighting simulation can be computed in a preprocess and rapidly visualized from any arbitrary viewpoint [Rushmeier and Torrance, 1987]. Many efficient

Figure 1.1: Volumetric scattering due to participating media is responsible for the appearance of a number of striking visual effects including aerial or atmospheric perspective (top), shafts of light (left), clouds and crepuscular rays (right), as well as interstellar dust and nebula (bottom). The bottom image is courtesy of T.A.Rector (NOAO/AURA/NSF) and the Hubble Heritage Team (STScI/AURA/NASA).

algorithms have been developed which work exceptionally well for specific classes of scattering materials. Unfortunately, not all participating media effects fall within these narrow confines, in which case we must resort to more general, but typically less efficient, approaches.

The goal of this dissertation is to develop efficient, general-purpose methods for solv-

ing light transport in scenes containing participating media. To meet our goal of generality, we concentrate on Monte Carlo methods. These methods can easily handle arbitrary medium properties and make few underlying assumptions about the scenes being simulated. Unfortunately, unbiased Monte Carlo techniques suffer from long computation times and significant noise. To satisfy our goal of efficiency we focus on caching-based Monte Carlo methods. These methods exploit the often smooth nature of illumination by computing and caching lighting at a small set of locations. These cache values are then reused to estimate lighting at all locations needed to render the image. Aside from this property, the methods developed in this dissertation make no assumptions about the properties of the medium being rendered.

1.1 Summary of Original Contributions

The work in this dissertation builds upon the irradiance caching and photon mapping methods. We outline our major contributions below.

Volumetric Radiance Caching. Our first contribution is a novel radiance caching method for efficiently rendering participating media using Monte Carlo ray tracing. The *volumetric radiance caching* method handles all types of light scattering, including anisotropic scattering, and it works in both homogeneous and heterogeneous media. Our method gains efficiency by sparsely sampling and interpolating radiance within the medium. The method provides several orders of magnitude speedup compared to path tracing and produces higher quality results than volumetric photon mapping. Furthermore, it is view-driven and well suited in large scenes where methods such as photon mapping become costly.

Accurate Illumination Gradients in Participating Media. Another key contribution in this dissertation is a technique for computing gradients of radiance evaluated in the presence of participating media. These gradients describe how the incident radiance field changes with respect to translation both on surfaces and within the medium. Our gradient derivations take the full path of the scattered light into account including the changing properties of the medium in the case of heterogeneous media. We compute gradients for single scattering from lights and surfaces and

for multiple scattering. For surface irradiance, we additionally take into account the effects of visibility on the gradient. We apply our gradient derivations to provide higher quality interpolation within both the irradiance caching and volumetric radiance caching methods.

The Beam Radiance Estimate. We also introduce a new photon mapping method for more efficiently simulating the scattering of light within participating media. We present a theoretical reformulation of volumetric photon mapping, which allows us to develop a novel photon gathering technique for participating media. This approach explicitly estimates the accumulated radiance along the length of an entire ray instead of sampling the radiance at individual points. This optimization provides a significant savings making it substantially faster than regular volumetric photon mapping.

1.2 Organization of the Dissertation

The dissertation is divided into nine chapters. In the remaining chapters, we first focus on background material for the first two contributions above. In Chapter 2 we introduce the fundamentals of light transport in the absence of participating media. We describe irradiance caching and derived methods, which form the inspiration for our volumetric radiance caching approach, in Chapter 3. We also provide derivations which are missing from the literature in previous work. In Chapter 4 we extend our theoretical understanding by describing how light interacts with participating media. Based on this foundation, in Chapter 5 we develop the volumetric radiance caching method in detail and derive expressions for computing the gradient of in-scattered radiance within participating media. In Chapter 6 we extend our derivations for illumination gradients by considering the behavior of surface irradiance in the presence of participating media and occlusions. We then move on to the photon mapping method, which we describe in detail in Chapter 7. After highlighting the shortcomings of the volumetric photon mapping method, we present a theoretical reformulation in Chapter 8, which allows us to develop the beam radiance estimate. Chapter 9 summarizes our contributions and discusses possible avenues of future work.

In the body of this dissertation, we use a number of mathematical tools while providing

only minimal overview and no derivations of associated properties. We therefore provide three appendices which cover, in more detail, these mathematical concepts. Appendix A provides a detailed introduction to Monte Carlo integration upon which all our methods are based. We cover spherical harmonics in Appendix B and review their benefitial properties and associated operations. Lastly, we review the theory behind density estimation methods such as photon mapping in Appendix C.

2

Fundamentals of Light Transport

"He who loves practice without theory is like the sailor who boards ship without a rudder and compass and never knows where he may cast."

—Leonardo Da Vinci, 1452–1519

THE goal of rendering algorithms is to synthesize images of virtual scenes. Global illumination algorithms solve this problem by mimicking the physical behavior of light as it is emitted from light sources, scattered by elements in the scene, and ultimately detected by a virtual sensor or "camera." In this chapter we first explore the necessary background and develop the terminology needed to understand the physical properties of light. We then explain the expressions that govern the local and global behavior of light as it interacts with surfaces in a vacuum. We conclude the chapter with a short overview of techniques that have been developed in the computer graphics literature to simulate these effects, and summarize our notation in Table 2.2.

2.1 Assumptions About the Nature of Light

Our current understanding of the behavior of light relies on a progression of increasingly complete yet complicated models of light. These are: ray optics, wave optics, electromagnetic optics, and quantum optics (see Figure 2.1) [Saleh and Teich, 2007]. Computer graphics typically relies on the simplest of these models, ray optics (also called geometric optics). This model makes several simplifying assumptions about the behavior of light that limit the types of phenomena that can be simulated. In essence, in this model light can only be emitted, reflected, and transmitted. Additionally, light is assumed to travel in straight lines and at infinite speed. This means that

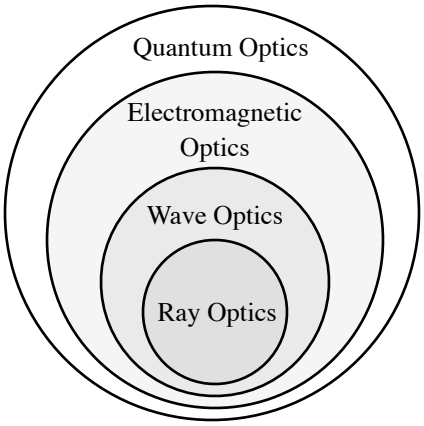


Figure 2.1: The theory of light is described by a series of increasingly complete optical models, where each successive model is able to account for more optical phenomena. In computer graphics and this dissertation, we will restrict ourselves to the simplest model, ray optics.

effects explained by the higher-level models cannot (easily) be incorporated into our simulations. In ray optics, effects such as diffraction and interference (wave optics), polarization and dispersion (electromagnetic optics), and fluorescence and phosphorescence (quantum optics) are completely ignored. In spite of these limitations, we are still able to correctly simulate a wide range of physical phenomena.

2.2 Radiometry

Radiometry is the study of the physical measurement of electromagnetic radiation, including visible light. It defines a common terminology for the physical quantities and units that are used by global illumination algorithms. In this section we review the basic radiometric quantities and illustrate them in Figure 2.2. All of these radiometric quantities in reality depend on the wavelength of light; however, we omit this in our notation.

2.2.1 Radiometric Quantities

Flux. The most fundamental radiometric quantity is radiant power, or flux. Flux, denoted Φ, expresses the amount of energy flowing across a surface over time. This is the same quantity that is used to describe the power of light bulbs and is expressed in terms of watts [$W = J \cdot s^{-1}$].

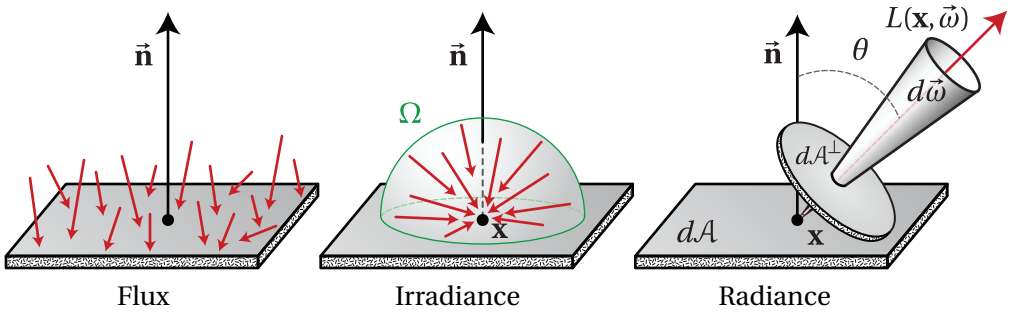


Figure 2.2: Flux measures the amount of light that hits a surface over a finite area from all directions, irradiance integrates the light arriving at a single point over the whole hemisphere, and radiance expresses the amount of light arriving at a single point from a differential solid angle.

Irradiance. A related quantity, irradiance, expresses the amount of incident power hitting a surface per unit surface area. Hence, it has units of $[W \cdot m^{-2}]$ and can be expressed in terms of flux as:

$$E(\mathbf{x}) = \frac{d\Phi(\mathbf{x})}{d\mathcal{A}(\mathbf{x})}. \tag{2.1}$$

Irradiance is always measured at some surface position $\mathbf{x}$ with a surface normal $\vec{\mathbf{n}}$. The term irradiance implies a measure of the flux *arriving* at a surface location $\mathbf{x}$. The term *radiant exitance* (M) or *radiosity* (B) is used instead if the flux is leaving a surface.

Radiance. Perhaps the most important quantity in global illumination is radiance. Intuitively, radiance expresses how much light arrives from a differential direction $d\vec{\omega}$ onto a hypothetical differential area *perpendicular* to that direction $d\mathcal{A}^{\perp}$. Radiance has units of $[W \cdot sr \cdot m^{-2}]$. This five-dimensional function of position and direction can be expressed as

$$L(\mathbf{x},\vec{\omega}) = \frac{d^2\Phi(\mathbf{x},\vec{\omega})}{d\vec{\omega}\, d\mathcal{A}^{\perp}(\mathbf{x})}. \tag{2.2}$$

In practice we are typically interested in measuring radiance at an actual surface instead of a hypothetical surface perpendicular to the flow of light. In this case the relationship $d\mathcal{A}^{\perp} =$

$(\vec{\mathbf{n}} \cdot \vec{\omega})\, d\mathcal{A}$ can be used to obtain:

$$L(\mathbf{x}, \vec{\omega}) = \frac{d^2\Phi(\mathbf{x}, \vec{\omega})}{(\vec{\mathbf{n}} \cdot \vec{\omega})\, d\vec{\omega}\, d\mathcal{A}(\mathbf{x})}. \tag{2.3}$$

The foreshortening cosine term $(\vec{\mathbf{n}} \cdot \vec{\omega})$, which returns the cosine of the angle between $\vec{\omega}$ and the surface normal $\vec{\mathbf{n}}$, takes into account the spreading of light at glancing angles.

It is also possible to interpret radiance as a density over projected solid angle $d\vec{\omega}^{\perp} = (\vec{\mathbf{n}} \cdot \vec{\omega})\, d\vec{\omega}$, instead of projected surface area $d\mathcal{A}^{\perp}$:

$$L(\mathbf{x}, \vec{\omega}) = \frac{d^2\Phi(\mathbf{x}, \vec{\omega})}{d\vec{\omega}^{\perp}\, d\mathcal{A}(\mathbf{x})}. \tag{2.4}$$

This can often be a more convenient way of describing radiance since it uses the natural differential surface area $d\mathcal{A}$ of the surface at $\mathbf{x}$.

Radiance is most directly related to the perceived brightness of colors to the human eye and is the quantity that needs to be computed for each pixel in a rendered image.

We summarize the radiometric quantities, their symbols, and units in Table 2.1.

2.2.2 Radiometric Relationships

Radiance is so instrumental in rendering that it is useful to express the other radiometric quantities in terms of radiance. Equation 2.2 expresses radiance in terms of flux, but it is also possible to invert this relationship by integrating both sides over the hemisphere of directions Ω

Table 2.1: Definitions of the fundamental radiometric quantities with their associated symbols and units.

Symbol	Units	Description
Φ	W	Radiance flux, or power
E	$W \cdot m^{-2}$	Irradiance
M	$W \cdot m^{-2}$	Radiant exitance (outgoing)
B	$W \cdot m^{-2}$	Radiosity (outgoing)
L	$W \cdot m^{-2} \cdot sr^{-1}$	Radiance

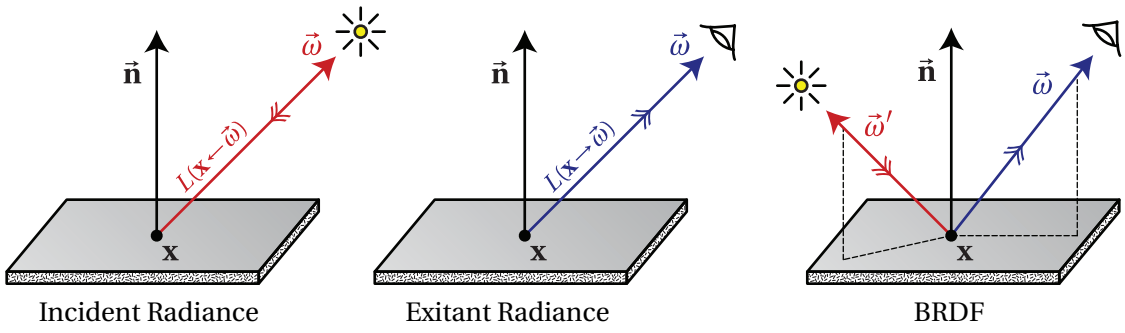


Figure 2.3: We distinguish between two types of radiance at a point $\mathbf{x}$. $L(\mathbf{x} \leftarrow \vec{\omega})$ represents incident radiance at $\mathbf{x}$ from direction $\vec{\omega}$ (left), whereas $L(\mathbf{x} \rightarrow \vec{\omega})$ represents exitant radiance at $\mathbf{x}$ in direction $\vec{\omega}$ (middle). The bidirectional reflectance distribution function (right) describes the relationship between these two quantities and is a function of both an incoming and an outgoing direction vector. In our notation, direction vectors always point out of $\mathbf{x}$ regardless of the actual direction of light flow.

and area $\mathcal{A}$ to arrive at:

$$\Phi = \int_{\mathcal{A}} \int_{\Omega} L(\mathbf{x} \rightarrow \vec{\omega}) \, (\vec{\mathbf{n}} \cdot \vec{\omega}) \, d\vec{\omega} \, d\mathcal{A}(\mathbf{x}). \tag{2.5}$$

Combining this with Equation 2.1 allows us to also express irradiance and radiosity in terms of radiance:

$$E(\mathbf{x}) = \int_{\Omega} L(\mathbf{x} \leftarrow \vec{\omega}) \, (\vec{\mathbf{n}} \cdot \vec{\omega}) \, d\vec{\omega}, \tag{2.6}$$

$$M(\mathbf{x}) = B(\mathbf{x}) = \int_{\Omega} L(\mathbf{x} \rightarrow \vec{\omega}) \, (\vec{\mathbf{n}} \cdot \vec{\omega}) \, d\vec{\omega}. \tag{2.7}$$

2.2.3 Incident and Exitant Radiance Functions

In this dissertation we use the convention that $L(\mathbf{x} \rightarrow \vec{\omega})$ denotes *exitant* radiance *leaving* $\mathbf{x}$ in direction $\vec{\omega}$, and $L(\mathbf{x} \leftarrow \vec{\omega})$ represents *incident* radiance *arriving* at $\mathbf{x}$ from direction $\vec{\omega}$. In both situations the direction vector $\vec{\omega}$ points *out* of the surface. This is illustrated in Figure 2.3. In addition to representing radiance flowing in different directions, this distinction is more fundamental since these two quantities measure different sets of photon events. Incident radiance measures photons just before they arrive at a surface, whereas exitant radiance measures the

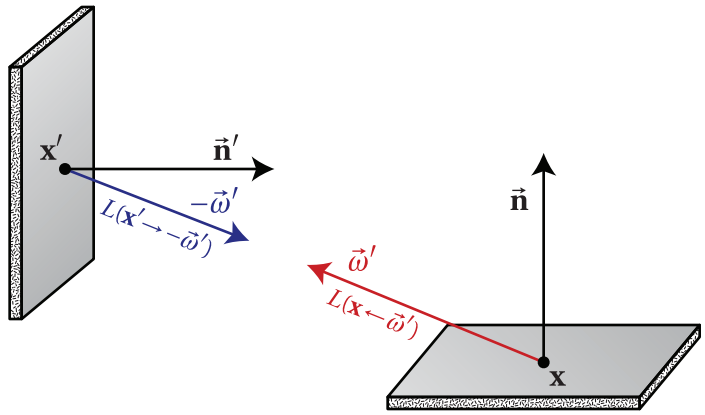


Figure 2.4: In a vacuum, the incident radiance at $\mathbf{x}$ from direction $\vec{\omega}$ is equal to the exitant radiance from the nearest visible surface in that direction.

photons departing. Hence, in general

$$L(\mathbf{x} \rightarrow \vec{\omega}) \neq L(\mathbf{x} \leftarrow \vec{\omega}). \tag{2.8}$$

Within a vacuum, photons propagate unobstructed. This means that all the radiance incident at a point from direction $\vec{\omega}$ will continue on as exitant radiance in direction $-\vec{\omega}$

$$L(\mathbf{x} \leftarrow \vec{\omega}) = L(\mathbf{x} \rightarrow -\vec{\omega}). \tag{2.9}$$

Hence, radiance remains constant until it interacts with a surface. This observation allows us to form a simple relation between the exitant and incident radiance functions between two distinct points. To accomplish this we introduce the *ray casting function* $\mathbf{r}(\mathbf{x}, \vec{\omega}) = \mathbf{x}'$, which returns $\mathbf{x}'$, the point on the closest surface from $\mathbf{x}$ in the direction $\vec{\omega}$. In the case that the space between surfaces is a vacuum, radiance remains constant along straight lines. This property means that the incident radiance at a point $\mathbf{x}$ from direction $\vec{\omega}$ is equal to the outgoing radiance from the closest visible point in that direction. This is illustrated in Figure 2.4 and can be expressed as:

$$L(\mathbf{x} \leftarrow \vec{\omega}) = L(\mathbf{x}' \rightarrow -\vec{\omega}). \tag{2.10}$$

It is important to remember that Equations 2.9 and 2.10 are only valid within a vacuum.

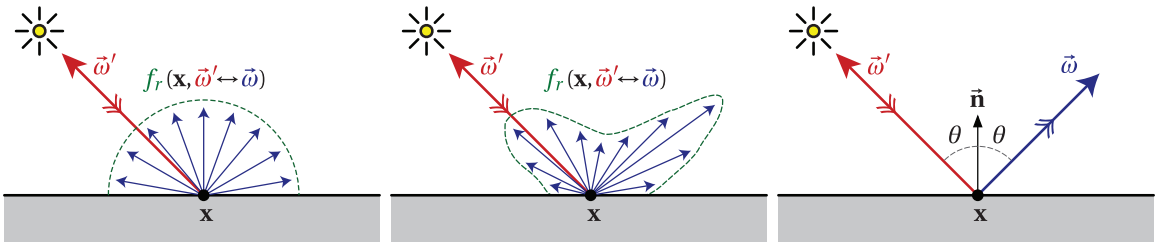


Figure 2.5: The ideal diffuse, or Lambertian, BRDF (left) reflects light equally in all directions. On the other extreme, perfectly specular materials (right) reflect all light in the mirror direction. General BRDFs (center) define a 2D reflectance distribution function for each incoming direction $\vec{\omega}'$.

We will extend this model in Chapter 4 to include effects from participating media.

2.3 Interaction of Light with Surfaces

In the remainder of this chapter we will explore the interaction of light with surfaces.

2.3.1 The BRDF

When light encounters objects in the scene, it interacts with the surfaces by being reflected, refracted, or absorbed. If we make the simplifying assumption that light striking a surface location will reflect at the same location, then the interaction between the light and the surface can be described using a six-dimensional function called the *bidirectional reflectance distribution function*, or BRDF. At a high level, the BRDF describes how "bright" a surface will look from a particular direction $\vec{\omega}$ when being illuminated by a light from another direction $\vec{\omega}'$. More precisely, the BRDF expresses the relationship between irradiance and reflected radiance at a point $\mathbf{x}$:

$$f_r(\mathbf{x}, \vec{\omega}' \rightarrow \vec{\omega}) \equiv \frac{dL(\mathbf{x} \rightarrow \vec{\omega})}{dE(\mathbf{x} \leftarrow \vec{\omega}')} = \frac{dL(\mathbf{x} \rightarrow \vec{\omega})}{L(\mathbf{x} \leftarrow \vec{\omega}')\,(\vec{\mathbf{n}} \cdot \vec{\omega}')\,d\vec{\omega}'}. \tag{2.11}$$

The last step results from substituting Equation 2.6 into the denominator. Three example BRDFs are illustrated in Figure 2.5.

Properties of the BRDF

1. **Domain.** For a particular surface point $\mathbf{x}$, the BRDF is a four-dimensional function: two dimensions to specify the incoming direction, and two dimensions to specify the outgoing

direction. Furthermore, if the BRDF is allowed to vary spatially over an object's surface, this leads to an additional two dimensions.

2. **Range.** The BRDF can take on any positive value.

3. **Reciprocity.** The value of the BRDF remains unchanged if the incident and outgoing directions are swapped. Mathematically, Helmholtz's law of reciprocity states that:

$$f_r(\mathbf{x}, \vec{\omega}', \vec{\omega}) = f_r(\mathbf{x}, \vec{\omega}, \vec{\omega}'). \tag{2.12}$$

Because of this property, we use the following notation for the BRDF to indicate that the directions can be interchanged:

$$f_r(\mathbf{x}, \vec{\omega}' \leftrightarrow \vec{\omega}). \tag{2.13}$$

From a practical point of view, reciprocity means that surface reflection is invariant to the direction of light flow, i.e, the reflected radiance remains unchanged if the light and camera positions are swapped. This property is essential for many global illumination algorithms and allows light to be traced either in the forward or backward direction.

4. **Relationship between incident and reflected radiance.** The information in the BRDF can be used to derive the relationship between incident and reflected radiance. By multiplying both sides of Equation 2.11 by the denominator and integrating over all directions we can derive an expression for computing the reflected radiance at a point **x**:

$$L(\mathbf{x} \rightarrow \vec{\omega}) = \int_{\Omega} f_r(\mathbf{x}, \vec{\omega}' \leftrightarrow \vec{\omega})\, L(\mathbf{x} \leftarrow \vec{\omega}')\, (\vec{\mathbf{n}} \cdot \vec{\omega}')\, d\vec{\omega}'. \tag{2.14}$$

In essence, the reflected radiance off of a surface can be computed by integrating all the incident radiance arriving over the hemisphere of directions. This expression describes the local behavior of light as it interacts with surfaces and is know as the *local illumination* model.

5. **Energy conservation.** Due to energy conservation, a surface cannot reflect more light than

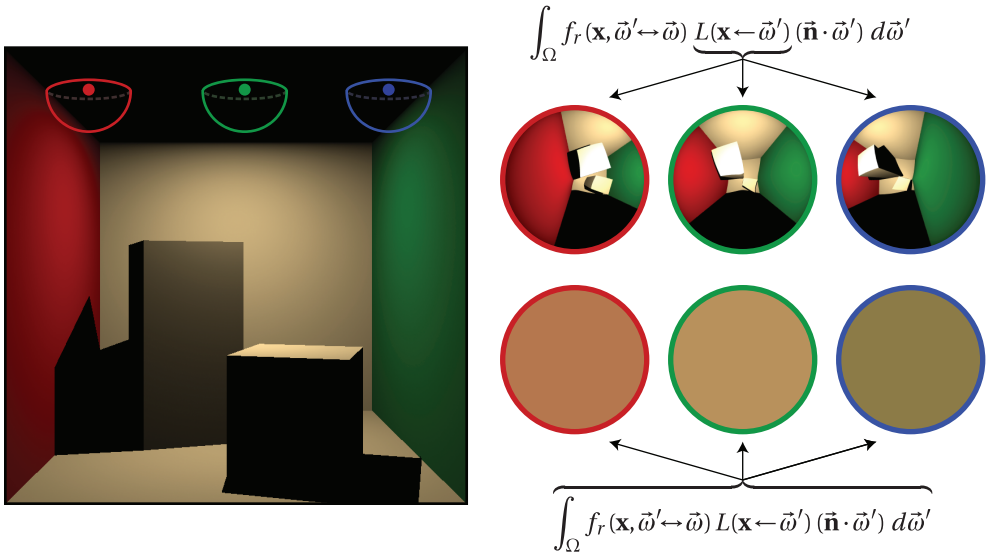


Figure 2.6: The exitant radiance at a point on a surface depends on the incident radiance over the hemisphere above the point. The incident radiance field can be visualized as a hypothetical "fisheye" view from the point (top) and the exitant radiance is the integral of this value over the whole hemisphere.

it receives. By examining Equation 2.14, this can be expressed as the following constraint:

$$\int_\Omega f_r(\mathbf{x}, \vec{\omega}' \leftrightarrow \vec{\omega})\, (\vec{\mathbf{n}} \cdot \vec{\omega}')\, d\vec{\omega}' \leq 1, \quad \forall \vec{\omega}. \tag{2.15}$$

2.3.2 The Rendering Equation

At a high level, the illumination at each surface point is based on the "brightness" of what that point "sees" within its hemisphere of directions. This could be light sources or other surfaces. The brightness of these other surfaces is in turn evaluated in the same way, introducing a recursive definition for calculating illumination. This intuitive explanation of lighting is illustrated in Figure 2.6. In order to compute the lighting at a surface we must solve for this recursive light energy distribution.

Hemispherical Formulation

The equation which mathematically describes this equilibrium is called the rendering equation [Kajiya, 1986]. The rendering equation expresses the outgoing radiance, $L(\mathbf{x} \rightarrow \vec{\omega})$, of any

point in a scene as the sum of the emitted radiance of the underlying surface, L_e, and the reflected radiance, L_r.

$$\underbrace{L(\mathbf{x}\rightarrow\vec{\omega})}_{\text{outgoing}} = \underbrace{L_e(\mathbf{x}\rightarrow\vec{\omega})}_{\text{emitted}} + \underbrace{L_r(\mathbf{x}\rightarrow\vec{\omega})}_{\text{reflected}}. \tag{2.16}$$

The local illumination model in Equation 2.14 expresses the reflected radiance on a surface. Substituting this into Equation 2.16 results in the hemispherical form of the rendering equation:

$$\underbrace{L(\mathbf{x}\rightarrow\vec{\omega})}_{\text{outgoing}} = \underbrace{L_e(\mathbf{x}\rightarrow\vec{\omega})}_{\text{emitted}} + \underbrace{\int_\Omega f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, L(\mathbf{x}\leftarrow\vec{\omega}')\,(\vec{\mathbf{n}}\cdot\vec{\omega}')\, d\vec{\omega}'.}_{\text{reflected}} \tag{2.17}$$

The combination of Equation 2.17 with Equation 2.10 (which relates the exitant radiance on the left hand side to the incident radiance on the right hand side) forms the recursive nature of global illumination. This arises from the fact that outgoing radiance at one point is dependent on the outgoing radiance at all other points in the scene. Hence, the radiance L appears on both sides of the equation.

Area Formulation

Sometimes it is more convenient to express the rendering equation as an integration over points on the surfaces of the scene geometry. This can be done by performing a change of variable using the relation,[1]

$$d\vec{\omega}'(\mathbf{x}) = \frac{(\vec{\mathbf{n}}'\cdot -\vec{\omega}')}{\|\mathbf{x}'-\mathbf{x}\|^2}\, d\mathcal{A}(\mathbf{x}'), \tag{2.18}$$

where $\mathbf{x}'$ is a point on a surface with surface normal $\vec{\mathbf{n}}'$.

In the hemispherical formulation visibility is implicitly accounted for using the ray casting function. In order to change the integration from the hemisphere of directions to surface area we must explicitly take into account the visibility between surface points. To accomplish this we

[1] This expression is obtained by computing the Jacobian determinant of the transformation matrix mapping directions on the hemisphere to points on the scene geometry.

introduce a visibility function V:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathcal{A} : V(\mathbf{x} \leftrightarrow \mathbf{x}') = \begin{cases} 1 & \text{if } \mathbf{x} \text{ and } \mathbf{x}' \text{ are mutually visible,} \\ 0 & \text{otherwise.} \end{cases} \tag{2.19}$$

Table 2.2: Notation used in this dissertation.

Symbol	Units	Description	Page
$\mathbf{x}$		Position	
$\vec{\mathbf{n}}$		Surface normal at $\mathbf{x}$ (always normalized: $\|\vec{\mathbf{n}}\| = 1$)	
$d\mathcal{A}(\mathbf{x})$		Differential surface area at $\mathbf{x}$	
$d\mathcal{A}^{\perp}(\mathbf{x}, \vec{\omega})$		Differential surface area perpendicular to $\vec{\omega}$ at $\mathbf{x}$	10
$\vec{\omega}$		Normalized direction (away from surface)	
$d\vec{\omega}$		Differential solid angle ($d\vec{\omega} = \sin\theta \, d\theta \, d\phi$)	
$d\vec{\omega}^{\perp}$		Differential projected solid angle	11
Ω	sr	Hemisphere of directions	11
$\Omega_{4\pi}$	sr	Sphere of directions	61
$\mathcal{A}$	m^2	Surface area	131
$\mathcal{V}$	m^3	Medium volume	131
$L(\mathbf{x}, \vec{\omega})$	$W \cdot m^{-2} \cdot sr^{-1}$	Radiance at $\mathbf{x}$ in direction $\vec{\omega}$	10
$L(\mathbf{x} \leftarrow \vec{\omega})$	$W \cdot m^{-2} \cdot sr^{-1}$	Incident radiance at $\mathbf{x}$ from $\vec{\omega}$	12
$L(\mathbf{x} \rightarrow \vec{\omega})$	$W \cdot m^{-2} \cdot sr^{-1}$	Outgoing radiance leaving $\mathbf{x}$ in direction $\vec{\omega}$	12
$L(\mathbf{x} \leftarrow \mathbf{x}')$	$W \cdot m^{-2} \cdot sr^{-1}$	Incident radiance at $\mathbf{x}$ from $\mathbf{x}'$	
$L(\mathbf{x} \rightarrow \mathbf{x}')$	$W \cdot m^{-2} \cdot sr^{-1}$	Outgoing radiance leaving $\mathbf{x}$ towards $\mathbf{x}'$	
L_e	$W \cdot m^{-2} \cdot sr^{-1}$	Emitted radiance	17
L_r	$W \cdot m^{-2} \cdot sr^{-1}$	Reflected radiance	15
L_i	$W \cdot m^{-2} \cdot sr^{-1}$	In-scattered radiance	61
$f_r(\mathbf{x}, \vec{\omega} \leftrightarrow \vec{\omega}')$	sr^{-1}	BRDF	14
$p(\mathbf{x}, \vec{\omega} \leftrightarrow \vec{\omega}')$	sr^{-1}	Phase function (normalized)	61
$\sigma_s(\mathbf{x})$	m^{-1}	Scattering coefficient at $\mathbf{x}$	57
$\sigma_a(\mathbf{x})$	m^{-1}	Absorption coefficient at $\mathbf{x}$	57
$\sigma_t(\mathbf{x})$	m^{-1}	Extinction coefficient at $\mathbf{x}$	57
$\tau(\mathbf{x} \leftrightarrow \mathbf{x}')$	unitless	Optical thickness: $\int_{\mathbf{x}'}^{\mathbf{x}} \sigma_t(x) \, dx$	59
$T_r(\mathbf{x} \leftrightarrow \mathbf{x}')$	unitless	Transmittance: $e^{-\tau(\mathbf{x} \leftrightarrow \mathbf{x}')}$	58
$V(\mathbf{x} \leftrightarrow \mathbf{x}')$	unitless	Visibility function	18, 77
$G(\mathbf{x} \leftrightarrow \mathbf{x}')$	unitless	Geometric coupling term (surfaces)	19
$H(\mathbf{x} \rightarrow \mathbf{x}')$	unitless	Geometric coupling term (media)	77
$\hat{G}(\mathbf{x} \leftrightarrow \mathbf{x}')$	unitless	Generalized geometric coupling term (surfaces and media)	132
$W_e(\mathbf{x} \rightarrow \vec{\omega})$	$m^{-3} \cdot sr^{-1}$	Importance at $\mathbf{x}$ towards $\vec{\omega}$	133
$W_e(\mathbf{x} \rightarrow \mathbf{x}')$	$m^{-3} \cdot sr^{-1}$	Importance at $\mathbf{x}$ towards $\mathbf{x}'$	133
α_i	unitless	Weight for photon i	134
$\delta(x)$	unitless	Dirac delta distribution	136
δ_{ij}	unitless	Kronecker delta function	172
ξ	unitless	Uniformly distributed random number between 0 and 1	82

We can now transform Equation 2.17 into an integration over surface area:

$$L(\mathbf{x}\rightarrow\vec{\omega}) = L_e(\mathbf{x}\rightarrow\vec{\omega}) + \int_{\mathcal{A}} f_r(\mathbf{x},\mathbf{x}'\rightarrow\mathbf{x},\vec{\omega})\, L(\mathbf{x}\leftarrow\mathbf{x}')\, V(\mathbf{x}\leftrightarrow\mathbf{x}')\, \frac{(\vec{\mathbf{n}}\cdot\vec{\omega}')\,(\vec{\mathbf{n}}'\cdot-\vec{\omega}')}{\|\mathbf{x}'-\mathbf{x}\|^2}\, d\mathcal{A}(\mathbf{x}'). \tag{2.20}$$

Typically, the last terms involving the surface positions and normals are folded into a symmetric geometric coupling term

$$G(\mathbf{x}\leftrightarrow\mathbf{x}') = \frac{(\vec{\mathbf{n}}\cdot\vec{\omega}')\,(\vec{\mathbf{n}}'\cdot-\vec{\omega}')}{\|\mathbf{x}'-\mathbf{x}\|^2}, \tag{2.21}$$

which results in

$$L(\mathbf{x}\rightarrow\vec{\omega}) = L_e(\mathbf{x}\rightarrow\vec{\omega}) + \int_{\mathcal{A}} f_r(\mathbf{x},\mathbf{x}'\rightarrow\mathbf{x},\vec{\omega})\, L(\mathbf{x}\leftarrow\mathbf{x}')\, V(\mathbf{x}\leftrightarrow\mathbf{x}')\, G(\mathbf{x}\leftrightarrow\mathbf{x}')\, d\mathcal{A}. \tag{2.22}$$

2.4 Methods for Solving the Rendering Equation

The rendering equation is very costly to compute and is far too complex to solve analytically in the general case. Because of this, it is typically approximated using numerical integration. Several algorithms have been proposed in the literature to fully or partially solve the rendering equation. These global illumination algorithms can generally be classified into two categories: finite element methods and Monte Carlo ray tracing methods.

2.4.1 Finite Element Methods

Finite element methods in computer graphics were originally adapted from the radiative heat transfer literature [Siegel and Howell, 2002]. In the compute graphics field, finite element techniques have collectively been called *radiosity methods*. Radiosity computes indirect illumination by discretizing the scene geometry into a collection of small patches. These patches form the basis over which the global illumination solution is computed. The lighting computation is solved by expressing the rendering equation as a linear system of equations describing the light

exchange between patches in the scene.

Initial work in radiosity placed heavy restrictions on the complexity of the light sources, geometry and materials of the scene. These early approaches only worked for scenes with large area light sources and Lambertian, or perfectly diffuse, surfaces where the BRDF is a simple constant [Goral et al., 1984; Cohen and Greenberg, 1985; Nishita and Nakamae, 1985]. With these restrictions, the radiosity algorithm is simple to implement and the lighting solutions computed are *view-independent*. This allows for a single solution to be reused for rendering images from arbitrary viewpoints. Due to this, radiosity techniques were well suited for walk-through animations of simple diffuse scenes. Unfortunately, precomputation for walk-through animations requires significant computation time as well as storage space for the final solution. Many researchers tried to address some of these drawbacks by adding view-dependent computation [Smits et al., 1992], support for more complex reflections [Wallace et al., 1987; Immel et al., 1986; Sillion et al., 1991; Christensen et al., 1996; Gortler et al., 1993], as well as clustered and hierarchical computation [Stamminger et al., 1998; Hanrahan et al., 1991; Smits et al., 1994]. Support for high-frequency lighting can be obtained through discontinuity meshing where the tessellation of the solution is carefully subdivided to better resolve sharp discontinuities such as hard shadow boundaries. These additions unfortunately make the radiosity algorithm much more complex. Furthermore, radiosity cannot efficiently handle scenes with complex geometry since the cost of the lighting simulation is directly tied to the geometric complexity of the scene.

2.4.2 Monte Carlo Ray Tracing Methods

Monte Carlo methods rely on the use of random sampling in order to compute their results. The first real use of Monte Carlo methods stems from work on the atomic bomb during the second world war in the 1940s and has been used extensively in the neutron transport field ever since [Metropolis and Ulam, 1949; Metropolis, 1987]. Monte Carlo methods developed independently in computer graphics. They were introduced to computer graphics by Appel [1968], who first suggested the use of ray casting to generate images by intersecting the scene geometry with a ray for each pixel. In 1980, Whitted suggested the recursive evaluation of illumination by tracing specular reflection and refraction rays, and additionally, shadow rays for direct Lambertian

illumination. Whitted also suggested the use of random perturbation of the surface normal in order to simulate rough or glossy specular reflections.

Ray tracing methods can also naturally account for complex surfaces and light interactions. By further combining ray tracing with Monte Carlo techniques, researchers were able to simulate a vast array of effects including depth of field, motion blur, glossy reflections, and global illumination [Cook, 1986; Cook et al., 1984; Kajiya, 1986]. Monte Carlo ray tracing methods such as path tracing or bidirectional path tracing [Kajiya, 1986; Lafortune and Willems, 1993; Veach and Guibas, 1994] can simulate virtually all effects described by the rendering equation. The downside of Monte Carlo techniques is variance, which manifests itself as high-frequency noise in the rendered image. Though several researchers have explored ways to minimize this noise [Shirley, 1990, 1991; Lafortune, 1996; Veach and Guibas, 1995], these unbiased methods still typically involve tracing hundreds of rays at each pixel in order to converge to solutions of reasonable quality.

One of the main advantages of radiosity over unbiased ray tracing is that for a specified simulation quality, computation time is independent of the actual resolution of the rendered image. Radiosity can effectively reuse computation over large regions of the image. In contrast, unbiased ray tracing techniques require independent evaluation at each pixel, which precludes resolution independence. Biased Monte Carlo ray tracing relaxes this restriction and has produced some of the most efficient methods to date. Irradiance caching [Ward et al., 1988] and photon mapping [Jensen, 2001] methods are two such approaches. We review these two techniques in more detail in Chapter 3 and Chapter 7, respectively. We also provide a more detailed introduction to Monte Carlo integration in Appendix A.

2.4.3 Hybrid Methods

Several methods have also been proposed which combine features of Monte Carlo and finite element techniques in order to get the best of both worlds. These typically take a *multi-pass* approach by augmenting finite element methods with a Monte Carlo pass to account for more general reflection models [Wallace et al., 1987; Sillion and Puech, 1989; Chen et al., 1991]. Another motivation for combining the techniques is to reduce discretization artifacts in finite element methods. A common optimization technique is to precompute a low-quality solution

using radiosity techniques, and then to simulate one additional bounce of illumination using a higher quality *final gather* pass [Reichert, 1992]. This final gather pass typically takes the form of a Monte Carlo evaluation and can effectively remove many of the artifacts present in the low-quality solution. In addition to being a full global illumination algorithm on its own, the irradiance caching method described in the next chapter can also be effectively used as a final gather optimization technique.

3

Irradiance Caching and Derived Methods

"Originality is merely an illusion."

—M.C. Escher, 1898–1972

SINCE its inception in 1988, irradiance caching [Ward et al., 1988] has become a widely-used technique for computing global illumination. In this chapter we explore this influential technique and the numerous extensions and improvements which have followed. We discuss the strengths of the algorithm that have led to its widespread use in academia as well as industry [Tabellion and Lamorlette, 2004; Kato, 2002]. We also present derivations needed to fully understand the approach, which are currently missing in the literature. Finally, we conclude by highlighting some shortcomings of the algorithm, which provides motivation for the contributions presented in Chapters 5 and 6 of this dissertation.

3.1 Algorithm Overview

Ray tracing techniques solve the rendering equation by means of recursive point-sampling. At each level of recursion, shading is evaluated by (1) computing the direct lighting from light sources, (2) computing specular reflection and refraction, and (3) computing diffuse indirect reflections. The lighting integral is effectively split up into disjoint components, each of which is approximated using the point-sampling processing of ray tracing. The cost of performing each of these operations is directly related to the number of rays that need to be traced. Contributions from (1) and (2) typically subtend a small solid angle and can therefore be effectively integrated using a relatively small number of sample rays. In contrast, (3) involves a lighting integral over

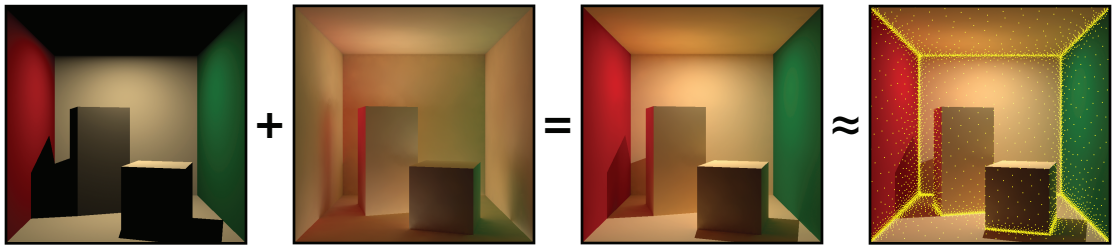


Figure 3.1: Irradiance caching decomposes the incident lighting into a *direct* component (far left) and an *indirect* component (left). The total lighting is obtained by summing these components (right). Even though the direct component may have sharp discontinuities, the indirect illumination tends to vary smoothly across Lambertian surfaces. Irradiance caching exploits this property by only computing this costly value at sparse locations (far right, shown in yellow) and using interpolation whenever possible.

the whole visible hemisphere, the approximation of which typically requires tracing hundreds or thousands of rays.

Irradiance caching is an acceleration technique for computing this diffuse indirect illumination component. Direct lighting tends to have sharp illumination discontinuities, for instance, near hard shadow boundaries. However, Ward et al. made the observation that, although diffuse indirect illumination is expensive to compute, the integration over the whole hemisphere causes this component to be extremely low frequency. This causes indirect illumination to change slowly across Lambertian surfaces. Irradiance caching exploits this property: instead of evaluating the costly illumination integral at each pixel on the image, a high-quality computation is performed only at sparse locations. These irradiance computations are cached and, when possible, reused through interpolation. This approach is illustrated in Figure 3.1.

Irradiance caching incorporates the beneficial properties of radiosity methods while retaining the generality of Monte Carlo approaches. If the cost of interpolating irradiance is negligible compared to a full integration, then the resolution of the rendered image has little impact on the computation time needed to simulate global illumination. However, unlike radiosity, irradiance caching gains further efficiency by concentrating computational effort only in regions of the scene which are directly visible by the camera. Furthermore, the algorithm uses a separate data structure to store the irradiance values. Hence, it does not artificially impose a coupling between the lighting simulation and the geometric representation of the scene. This is a major advantage, as there is no restriction on the kind of geometry that can be simulated with this technique. The

algorithm originally described by Ward et al. was used to accelerate the computation of all indirect diffuse illumination within the RADIANCE Monte Carlo ray tracer [Ward, 1994]; however, since that time, irradiance caching has proven popular also as a high-quality final gather pass for other methods such as photon mapping, which we discuss in Chapter 7.

In the irradiance caching algorithm, indirect lighting values are cached in the following manner:

Algorithm 3.1: COMPUTEINDIRECTIRRADIANCE(**x**)

```
1 if there is at least one stored irradiance value near x then
2     Interpolate irradiance from the stored value(s).
3 else
4     Compute and store a new irradiance value at x.
```

The above procedure computes irradiance in a lazy, on-demand fashion. In order to implement this algorithm, a number of outstanding issues must first be addressed:

- How do we compute the irradiance values?
- What criterion is used to determine whether a cache point is “near”?
- How do we interpolate the nearby cached irradiance values?
- What data structure should be used to provide for efficient storage and query of the cache points?

We answer these questions in the following sections.

3.2 Computing Irradiance

In this section we describe the method used to compute an accurate irradiance estimate. This method is employed when no nearby cache points are found.

The irradiance at a point **x** can be expressed in terms of the incident radiance using

Equation 2.6:

$$E(\mathbf{x}) = \int_{\Omega} L(\mathbf{x} \leftarrow \vec{\omega})\,(\vec{\mathbf{n}} \cdot \vec{\omega})\; d\vec{\omega}. \tag{3.1}$$

This integral can be approximated using ray tracing by evaluating the Monte Carlo estimator:

$$E(\mathbf{x}) \approx \frac{1}{N} \sum_{i=0}^{N-1} \frac{L(\mathbf{x} \leftarrow \vec{\omega})\,(\vec{\mathbf{n}} \cdot \vec{\omega})}{pdf(\vec{\omega})}. \tag{3.2}$$

Ward et al. use two variance reduction techniques to more efficiently evaluate this estimate. Choosing to distribute the sample rays with a cosine probability distribution, $pdf(\vec{\omega}) = (\vec{\mathbf{n}} \cdot \vec{\omega})/\pi$, provides variance-reduction through importance sampling by allowing the foreshortening term to cancel out:

$$E(\mathbf{x}) \approx \frac{1}{N} \sum_{i=0}^{N-1} \frac{L(\mathbf{x} \leftarrow \vec{\omega})(\vec{\mathbf{n}} \cdot \vec{\omega})}{(\vec{\mathbf{n}} \cdot \vec{\omega})/\pi} \Rightarrow \frac{\pi}{N} \sum_{i=0}^{N-1} L(\mathbf{x} \leftarrow \vec{\omega}). \tag{3.3}$$

The evaluation is also stratified by converting to polar coordinates:

$$E(\mathbf{x}) \approx \frac{\pi}{MN} \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} L(\mathbf{x}, \theta_j, \phi_k), \tag{3.4}$$

where

$$\theta_j = \sin^{-1}\left(\sqrt{\frac{j+\xi_1}{M}}\right), \qquad \phi_i = 2\pi \frac{i+\xi_2}{N}, \tag{3.5}$$

$(\xi_1, \xi_2) \in [0,1)^2$ are uniformly distributed random numbers, and the angles (θ_j, ϕ_k) are expressed with respect to the local coordinate frame at the evaluation location $\mathbf{x}$. The above expression uses a total of $M \times N$ samples[1]. Monte Carlo integration is covered in more detail in Appendix A.

[1] Generally M is chosen to be approximately πN

3.3 Interpolating Irradiance

For typical scenes, an accurate computation of Equation 3.4 requires hundreds or even thousands of sample rays. This is obviously a very costly operation; however, the goal of the irradiance caching algorithm is to compute these very accurate irradiance estimates at only a few locations within the scene and to perform interpolation or extrapolation whenever possible. Where the irradiance changes slowly we can compute the costly estimates very sparsely, whereas regions which exhibit sharp irradiance changes require more irradiance samples to faithfully capture the indirect illumination. Ward et al. developed a simplified heuristic which estimates an upper bound for the change of the irradiance $\varepsilon_i(\mathbf{x}, \vec{\mathbf{n}})$. The inverse of this change in irradiance serves as a weight for interpolation between nearby cached irradiance values:

$$w_i(\mathbf{x}, \vec{\mathbf{n}}) = \frac{1}{\varepsilon_i(\mathbf{x}, \vec{\mathbf{n}})}. \tag{3.6}$$

At any shading location, all previously stored cache points whose weight is above a user-specified threshold are used in the averaging. If no cache points have a weight above this threshold, then a new full irradiance estimate needs to be computed using Equation 3.4.

3.3.1 The "Split-Sphere" Model

In order to estimate an upper-bound on the maximal possible irradiance gradient, Ward et al. developed the "split-sphere" heuristic. In this model, a sample point is positioned at the center of a hypothetical spherical environment, which is half black and half white. The sample point is oriented to face the boundary between these two regions as illustrated in Figure 3.2. If we assume that concentrated light sources have been eliminated from the irradiance integral by separately accounting for direct lighting, then this *hypothetical* environment provides the largest possible irradiance gradient configuration.

In the split-sphere, irradiance is a function of the position, $\mathbf{x}$, and surface normal, $\vec{\mathbf{n}}$, of the evaluation location. Therefore, the maximum change in irradiance in the split-sphere is composed of the differential change with respect to change in both $\mathbf{x}$ and $\vec{\mathbf{n}}$. Ward et al. approximated this change by taking a first-order Taylor expansion of irradiance at $\mathbf{x}_0$. From this model, the estimated

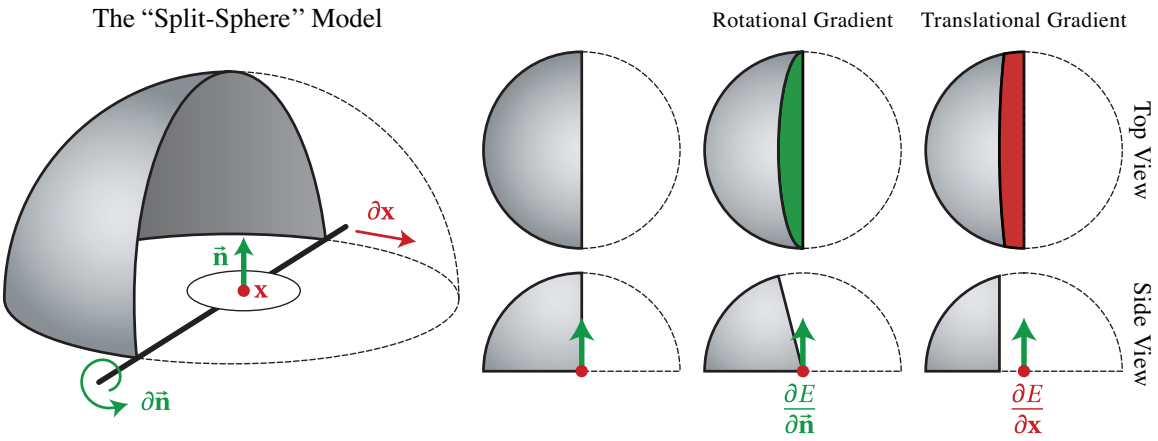


Figure 3.2: The "split-sphere" model. A surface element is positioned at the center of a hypothetical half-dark sphere. The induced change in irradiance is calculated with respect to change in translating the $\mathbf{x}$ (red) and rotating the orientation $\vec{\mathbf{n}}$ (green). This serves as an estimate of the maximum change in irradiance for *any* scene.

change in irradiance can be computed as:

$$\varepsilon_i(\mathbf{x},\vec{\mathbf{n}}) = E_i\left(\frac{4}{\pi}\frac{\|\mathbf{x}-\mathbf{x}_i\|}{R_i} + \sqrt{1-(\vec{\mathbf{n}}\cdot\vec{\mathbf{n}}_i)}\right), \tag{3.7}$$

where:

$\mathbf{x}_i$ = irradiance sample location,

$\vec{\mathbf{n}}_i$ = surface normal at $\mathbf{x}_i$,

E_i = irradiance at $\mathbf{x}_i$,

R_i = harmonic mean distance to surfaces from $\mathbf{x}_i$.

The R_i quantity represents the "average" distance to visible surfaces. This value is computed using the harmonic mean of the sample distances $r_{j,k}$ as:

$$R_i = \frac{MN}{\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}\frac{1}{r_{j,k}}}, \tag{3.8}$$

where each $r_{j,k}$ is attained from the lengths of the rays traced during the irradiance integration. A complete derivation of how this expression is obtained is provided in the next section.

The weight associated with each cache point is equal to the inverse of the estimated error for that cache point at the query location. The irradiance weighted averaging is therefore performed as follows:

$$E(\mathbf{x},\vec{\mathbf{n}}) \approx \frac{\sum_{i\in S} \mathrm{w}_i(\mathbf{x},\vec{\mathbf{n}})\, E_i}{\sum_{i\in S} \mathrm{w}_i(\mathbf{x},\vec{\mathbf{n}})}, \tag{3.9}$$

where:

$$a = \text{user selected error threshold parameter}$$
$$S = \left\{ i : \mathrm{w}_i(\mathbf{x},\vec{\mathbf{n}}) > \frac{1}{a} \right\}$$

In the final weighted interpolation, Ward et al. use a simplified expression to compute the weight by eliminating the constant terms in Equation 3.7

$$\mathrm{w}_i(\mathbf{x},\vec{\mathbf{n}}) = \frac{1}{\frac{\|\mathbf{x}-\mathbf{x}_i\|}{R_i} + \sqrt{1-\vec{\mathbf{n}}\cdot\vec{\mathbf{n}}_i}}. \tag{3.10}$$

Given this formulation of irradiance averaging, it is easy to compute the maximum distance at which the weight of a cache point will fall below the threshold $\frac{1}{a}$ by ignoring the change in surface normal and examining w_i as a function of only distance. This observation imposes a maximum possible valid radius of aR_i for each cache point. The finite spatial extent of cache points allowed Ward et al. to store them in an octree. The octree is used to efficiently find all cache points which overlap with a given shading location $\mathbf{x}$ in order to evaluate Equation 3.9. Since the valid radius depends on the harmonic mean distance R_i, cache points which "see" nearby geometry will have small valid radii, and cache points far away from visible geometry will have larger valid radii.

3.3.2 Derivation of the "Split-Sphere" Model

We now present a more rigorous derivation of the split-sphere irradiance gradient from Equation 3.7 than is currently available in the literature and illustrate the process in Figure 3.2. The change in irradiance depends on the translation of x and the rotation of the surface normal φ. This change can be approximated using a first order Taylor expansion about x_i:

$$\varepsilon(x,\varphi) = E_i\left(\frac{\partial E}{\partial x}(x-x_i) + \frac{\partial E}{\partial \varphi}(\varphi-\varphi_i)\right). \tag{3.11}$$

In the hypothetical split-sphere environment, the irradiance at a sample point is proportional to the projected area of the bright half of the hemisphere. The change in irradiance is equal to the change in ratio of the bright and dark parts. For the translational component $\frac{\partial E}{\partial x}$, the change in projected area of the bright half is shown in red in Figure 3.2. When Δx is small, the projection of the new region can be well approximated by a thin rectangle of size Δx by $2R$. Hence,

$$\frac{\partial E}{\partial x}\Delta x \approx \frac{2R\Delta x}{\frac{1}{2}\pi R^2} \Rightarrow \frac{4\Delta x}{\pi R}. \tag{3.12}$$

In order to examine the change in irradiance due to surface normal, we consider a rotation of the surface normal by a small angle φ. The axis of rotation that induces the most change in irradiance is parallel to the split plane between the bright and dark half of the environment. We can see in Figure 3.2 that as the sample point's orientation is rotated, the change in the projected bright region can be represented by half of an expanding ellipse (shown in green). The area of an ellipse with axes r_1 and r_2 is given by the formula $A_e = \pi r_1 r_2$. In our case, $r_1 = R$, and r_2 is dependent on the rotational offset $\Delta\varphi$. Examining the geometry in the bottom row of Figure 3.2(b), we see that $r_2 = R\sin\left(\Delta\varphi\right)$. This gives us

$$\frac{\partial E}{\partial \varphi}\Delta\varphi \approx \frac{\frac{1}{2}\pi R^2 \sin\left(\Delta\varphi\right)}{\frac{1}{2}\pi R^2} \Rightarrow \sin\left(\Delta\varphi\right). \tag{3.13}$$

The final Taylor expansion for the change in irradiance expands Equation 3.11 to:

$$\varepsilon(x,\varphi) = E_i\left(\frac{4}{\pi}\frac{|x-x_i|}{R} + \sin\left(\varphi-\varphi_0\right)\right). \tag{3.14}$$

In computing the upper bound on the irradiance gradient, we had assumed a fixed, imaginary split-sphere environment a constant distance R away from the sample location. However, we now need to apply this gradient estimate to the actual environment that is sampled using Equation 3.4. Generalizing the above gradient computation into vector-based quantities involves replacing the change in x into a distance between two points and the change in φ into the angle between two surface normals. This brings us to Equation 3.7.

3.4 Irradiance Gradients

Ward and Heckbert [1992] realized that it is possible to improve the quality of the irradiance estimate significantly by better utilizing the information provided during the sampling process. Specifically, it is possible to compute the *actual* gradient of the irradiance function from the information available in a sampled hemisphere. This is in contrast to the split-sphere derivation in the previous section which only computes a loose upper-bound on the gradient. If both the irradiance and its gradient are available, it is possible to perform higher-order interpolation of the irradiance samples.

The sampling of rays over the hemisphere (Equation 3.2) tells us not only the final approximation for irradiance, but also the brightness, direction, and distance for each individual sample ray. This extra data provides us with the information necessary to deduce a gradient of the sampled irradiance function. As with the split-sphere model, the irradiance gradient can be decomposed into two components: $\nabla_t E$, representing the gradient with respect to translation, and $\nabla_r E$, the gradient with respect to rotation. We present a full derivation of the irradiance gradient computation in the next section.

To take advantage of the additional gradient information in a higher-order interpolation,

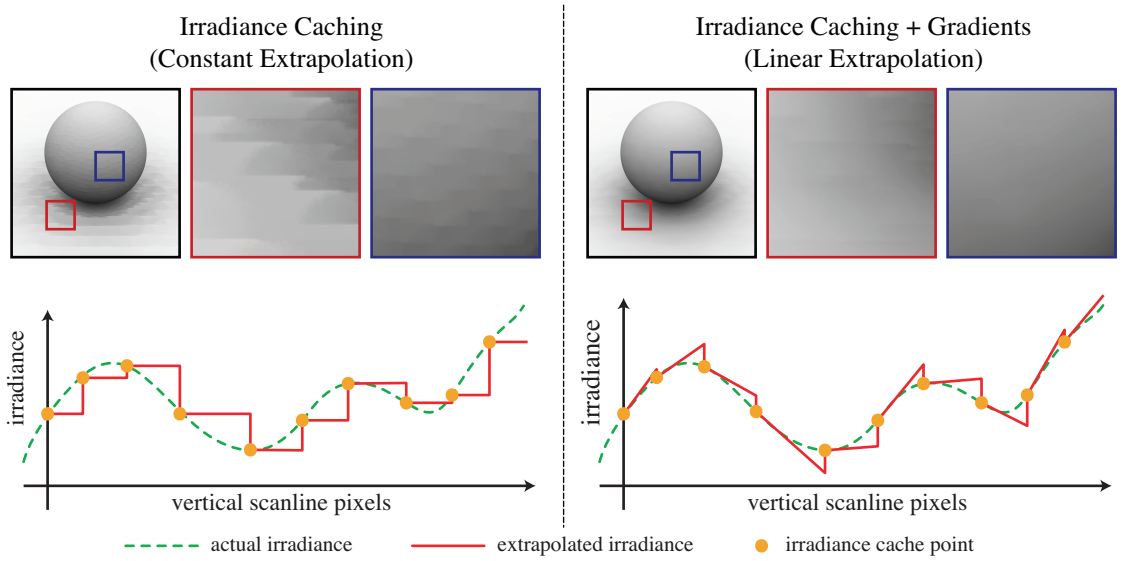


Figure 3.3: Render quality comparison between irradiance caching (left) and irradiance caching with gradients (right). The top row shows a simple scene rendered with both techniques. The two zoomed insets highlight the benefit of the translational (red) and rotational (blue) gradients. The graphs in the bottom image compare the extrapolated irradiance estimate for a single vertical scanline. New cache points are added on demand when the previous cache point's weight falls below a threshold. This results in constant extrapolation for irradiance caching (left), which produces significant errors. Incorporating gradients (right) results in a piecewise linear approximation, which follows the actual irradiance function more closely.

Equation 3.9 can be extended as

$$E(\mathbf{x},\vec{\mathbf{n}}) \approx \frac{\sum_{i\in S} w_i(\mathbf{x},\vec{\mathbf{n}})\,(E_i + (\vec{\mathbf{n}}_i \times \vec{\mathbf{n}})\cdot\nabla_r E_i + (\mathbf{x}-\mathbf{x}_i)\cdot\nabla_t E_i)}{\sum_{i\in S} w_i(\mathbf{x},\vec{\mathbf{n}})}. \tag{3.15}$$

This modification adds contribution from the translational gradient as we *move* away from a cache point and contribution from the rotational gradient as we *rotate* away from a cache point. This can be interpreted as performing a linear extrapolation from one cache point to the next, as opposed to a piecewise constant extrapolation (see Figure 3.3 for a one-dimensional analogy).

The incorporation of gradients into the irradiance caching scheme requires little extra computation but can provide vastly superior interpolation quality. A comparison rendering is provided in Figure 3.3.

3.5 Radiance Caching

One of the underlying observations justifying interpolation for irradiance caching is that reflected radiance due to indirect illumination changes slowly across Lambertian surfaces. Furthermore, since lighting on Lambertian surfaces is view-independent, a single irradiance value is sufficient for interpolating lighting across these surfaces. However, this compact representation is not possible with general surfaces since reflected light is, in general, view-dependent. In a series of papers, Křivánek et al. [2005b,a] extended the irradiance caching method to work with glossy surfaces. They observed that indirect illumination also changes slowly on surfaces with general low-frequency BRDFs. In order to interpolate indirect illumination on view-dependent glossy materials, Křivánek et al. cache the full directional *radiance* function instead of just a scalar irradiance value, and call their extension *radiance caching*.

3.5.1 Radiance Computation

Spherical functions, such as a full radiance field, can be efficiently stored using spherical harmonics. Since the radiance at surfaces is really only defined in the upper hemisphere, Křivánek et al. used hemispherical harmonics instead, which were introduced and derived in an earlier paper [Gautron et al., 2004]. Spherical (SH) and hemispherical (HSH) harmonics act in analogous ways, but HSH can more efficiently approximate functions over just a hemisphere. At a high level, these can be thought of as generalizations of a Fourier series applied to the spherical and hemispherical domains, respectively. In the context of radiance caching they can be interchanged except where explicitly noted. A more detailed introduction to spherical harmonics is provided in Appendix B.

Similarly to irradiance caching, the radiance caching algorithm uses a lazy caching scheme. When no cache records are found nearby, the incident lighting is evaluated accurately using Monte Carlo integration. During this process, the incoming radiance, $L(\mathbf{x} \leftarrow \vec{\omega})$, is projected onto (H)SH and stored as a vector of coefficients, $\Lambda = \{\lambda_l^m\}$, within a new cache record for subsequent interpolation. Each coefficient λ_l^m is computed using the Monte Carlo samples by

integrating the product of the incident radiance and the corresponding basis function y_l^m:

$$\lambda_l^m = \frac{2\pi}{N} \sum_{k=0}^{N-1} L(\mathbf{x} \leftarrow \vec{\omega}_k) y_l^m(\vec{\omega}_k). \tag{3.16}$$

3.5.2 Radiance Interpolation

As in irradiance caching, efficiency is gained in radiance caching by attempting to interpolate from nearby cache points before computing a new cache record. Radiance caching uses a similar weighted averaging scheme as irradiance caching but interpolates the coefficient vectors instead of the irradiance values. However, the calculation of HSH coefficient vectors in Equation 3.16 was performed within the local coordinate frame of each cache point. Multiple HSH coefficient vectors, therefore, cannot be directly interpolated because they potentially lie in different coordinate frames. Fortunately, functions represented using (H)SH can be efficiently rotated [Gautron et al., 2004; Ivanic and Ruedenberg, 1996; Blanco et al., 1997; Choi et al., 1999]. Hence, before interpolation, each cache record's coefficient vector is aligned to a common coordinate frame to make interpolation possible[2]. This results in the following modification to Equation 3.9:

$$\Lambda(\mathbf{x}, \vec{\mathbf{n}}) = \frac{\sum_{i \in S} \mathrm{w}_i(\mathbf{x}, \vec{\mathbf{n}}) \mathbf{M}_i(\Lambda_i)}{\sum_{i \in S} \mathrm{w}_i(\mathbf{x}, \vec{\mathbf{n}})}, \tag{3.17}$$

where $\mathbf{M}_i$ is the HSH rotation matrix, which aligns the coordinate frame of cache point $\mathbf{x}_i$ to the frame at the query point $\mathbf{x}$. Computing the interpolated radiance field and the BRDFs as (H)SH allows us to integrate them at any scene location using a simple dot product of their coefficient vectors.

3.5.3 Translational Radiance Gradients

Ward and Heckbert [1992] found that interpolation quality can be substantially improved by calculating gradient information from the hemispherical samples. This same concept can be applied to the radiance caching algorithm. In irradiance caching with gradients, the gradient is

[2] If SH are used instead, this rotation is not necessary, as the radiance fields can be stored in global instead of local space.

composed of a rotational and translational component. Within the radiance caching framework, we already take rotational change into account by analytically rotating the coefficient vectors before interpolation. However, by incorporating a translational gradient we can further improve interpolation quality.

Křivánek et al. proposed two different methods for computing the translational gradient [2005b]. In radiance caching, the translational gradient expresses the change of each HSH coefficient λ_l^m with respect to movement in the local x- or y-axes at the cache location. As with traditional irradiance gradients, the radiance gradients are computed at the same time as the coefficients during the hemispherical sampling.

The first technique calculates the gradient numerically by translating the sample point $\mathbf{x}$ to $\mathbf{x}'$ along the local x-axis, and to $\mathbf{x}''$ along the local y-axis. It estimates the coefficient $\lambda_l^{m\prime}$ and $\lambda_l^{m\prime\prime}$ at these displaced locations and, finally, takes the finite differences:

$$\left[\frac{\partial \lambda_l^m}{\partial x}, \frac{\partial \lambda_l^m}{\partial y}, 0\right] = \left[\frac{\lambda_l^{m\prime} - \lambda_l^m}{\Delta x}, \frac{\lambda_l^{m\prime\prime} - \lambda_l^m}{\Delta y}, 0\right]. \tag{3.18}$$

The second technique proposed by Křivánek et al. [2005b], and also independently by Annen et al. [2004], calculates the gradient by analytically differentiating the terms in Equation 3.16. This calculation involves deriving partial derivatives for the hemispherical harmonics. We omit the full derivation here and refer the interested reader to the original paper [Křivánek et al., 2005b].

The above two gradient computations treat each sample ray independently. Therefore, any distribution of rays can be used, which allows for different sampling strategies such as Quasi-Monte Carlo sampling. However, because each sample is treated independently, neither of the techniques take changing occlusions into account. To address this limitation, Křivánek et al. [2005a] later proposed an improved gradient computation, which accounts for changing occlusions.

The improved gradient calculation takes the approach originally used by Ward and Heckbert [1992] by stratifying the samples and considering the marginal change in incoming radiance with respect to each cell boundary. To account for the fact that in radiance caching the values are projected onto hemispherical harmonic basis functions, Křivánek et al. [2005a] generalize the

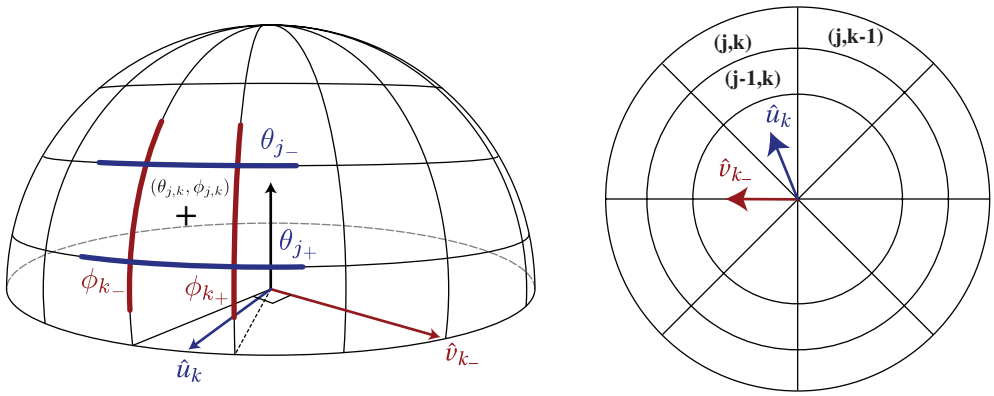


Figure 3.4: The stratified geometry used in the Ward and Heckbert [1992] gradient computation illustrated from the side (left) and above (right). The relevant quantities are described in Table 3.1.

calculations of Ward and Heckbert [1992] to work with an arbitrary weighting function applied to each sample. In Section 3.6 we explore this derivation in more detail and show how it relates to the original irradiance gradient formulation by Ward and Heckbert [1992].

Regardless of the technique used to calculate the gradient, Equation 3.17 can be extended as follows to incorporate the translational gradient information:

$$\Lambda(\mathbf{x},\vec{\mathbf{n}}) = \frac{\sum_{i\in S} \mathrm{w}_i(\mathbf{x},\vec{\mathbf{n}})\,\mathbf{M}_i\left(\Lambda_i + d_x\frac{\partial\Lambda_i}{\partial x} + d_y\frac{\partial\Lambda_i}{\partial y}\right)}{\sum_{i\in S}\mathrm{w}_i(\mathbf{x},\vec{\mathbf{n}})}, \tag{3.19}$$

where d_x and d_y are the displacements of $\mathbf{x}-\mathbf{x}_i$ along the local x- and y-axes.

3.6 Derivation of Irradiance Gradients

Since a complete derivation of the irradiance gradient computation is not included in the literature, we provide one here. The following discussion uses the notation described in Table 3.1 and illustrated in Figure 3.4.

At a high level, the irradiance computation in Equation 3.2 can be thought of as a weighted sum of the radiance. The contribution of each sample is the product of the radiance through the hemispherical cell $L(\mathbf{x}\leftarrow\vec{\omega}_{j,k})$, the solid angle or "area" of the cell $A_{j,k}$, and the cosine term

$(\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k})$:

$$E(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} A_{j,k} L(\mathbf{x} \leftarrow \vec{\omega}_{j,k}) (\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k}). \tag{3.20}$$

The cell area is the integral over the bounds of each cell:

$$A_{j,k} = \int_{\phi_{k_-}}^{\phi_{k_+}} \int_{\theta_{j_-}}^{\theta_{j_+}} \sin\theta d\theta d\phi \implies (\cos\theta_{j_-} - \cos\theta_{j_+})(\phi_{k_+} - \phi_{k_-}). \tag{3.21}$$

The irradiance gradient considers how these terms change when translating the center of projection or rotating the surface normal by an infinitesimal amount. The rotational and translational gradients are obtained by differentiating Equation 3.20:

$$\nabla_r E(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} \nabla_r (A_{j,k} L_{j,k} \cos\theta_j)$$
$$\nabla_t E(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} \nabla_t (A_{j,k} L_{j,k} \cos\theta_j),$$

where we have used the shorthand $L_{j,k} = L(\mathbf{x} \leftarrow \vec{\omega}_{j,k})$, and $\cos\theta_j = (\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k})$. Since in irradiance caching surfaces are assumed to be Lambertian, the radiance in each cell remains constant, which

Table 3.1: Definitions of quantities used in the irradiance gradients derivation.

Symbol	Description
N	Number of divisions in azimuthal angle ϕ
M	Number of divisions in elevation angle θ
$\vec{\omega}_{j,k}$	Sample direction for cell (j,k), i.e.: $(\theta_{j,k}, \phi_{j,k})$
$A_{j,k}$	Area of cell (j,k), i.e.: $(MN)^{-1} pdf(\vec{\omega}_{j,k})^{-1}$
θ_j	Elevation angle at center of cells $(j,\cdot)$
ϕ_k	Azimuthal angle at center of cells $(\cdot,k)$
$(j_-,k),(j_+,k)$	Cell boundaries ($j \leftrightarrow j$-1, k) and ($j \leftrightarrow j$+1, k)
$(j,k_-),(j,k_+)$	Cell boundaries ($j, k \leftrightarrow k$-1) and ($j, k \leftrightarrow k$+1)
θ_{j_+}	Elevation angle at boundary $(j_+,\cdot)$
θ_{j_-}	Elevation angle at boundary $(j_-,\cdot)$
θ_{j_+}	Elevation angle at boundary $(j_+,\cdot)$
ϕ_{k_-}	Azimuthal angle at boundary $(\cdot,k_-)$
ϕ_{k_+}	Elevation angle at boundary $(\cdot,k_+)$
$\hat{u}_k$	Tangent vector in the ϕ_k direction
$\hat{v}_k$	Tangent vector in the $\phi_k + \frac{\pi}{2}$ direction
$\hat{v}_{k_-}$	Tangent vector in the $\phi_{k_-} + \frac{\pi}{2}$ direction

simplifies the above expressions to:

$$\nabla_r E(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} L_{j,k} \nabla_r (A_{j,k} \cos\theta_j) \tag{3.22}$$

$$\nabla_t E(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} L_{j,k} \nabla_t (A_{j,k} \cos\theta_j). \tag{3.23}$$

In the following sections we derive expressions for $\nabla_r E(\mathbf{x})$ and $\nabla_t E(\mathbf{x})$.

3.6.1 The Rotational Gradient

The rotational gradient needs to express the change in the projection solid angle $A_{j,k} \cos\theta_j$ of each hemisphere sample cell as it is rotated in any direction. We make the observation that under a rotation of the surface normal, the stratified cell areas $A_{j,k}$ do not change. Hence,

$$\nabla_r (A_{j,k} \cos\theta_j) = A_{j,k} \nabla_r \cos\theta_j, \tag{3.24}$$

and Equation 3.22 reduces to

$$\nabla_r E(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} L_{j,k} A_{j,k} \nabla_r \cos\theta_j. \tag{3.25}$$

The only factor that influences the rotational irradiance gradient is the cosine foreshortening term $\cos\theta$, and $\frac{d}{d\theta} \cos\theta_j = -\sin\theta_j$. In order to compute the gradient of the whole stratified hemisphere, we need to sum up all of the marginal cell differentials multiplied by their corresponding rotation axes:

$$\nabla_r E = -\sum_{k=0}^{N-1} \left\{ \hat{v}_k \sum_{j=0}^{M-1} L_{j,k} A_{j,k} \sin\theta_j \right\}. \tag{3.26}$$

In the irradiance caching algorithm the hemispherical samples are constructed using a cosine-weighted distribution. This simplifies Equation 3.21 to

$$A_{j,k} = \frac{\pi}{MN \cos\theta_j}, \tag{3.27}$$

which reduces Equation 3.26 to

$$\nabla_r E = -\frac{\pi}{MN}\sum_{k=0}^{N-1}\left\{\hat{v}_k \sum_{j=0}^{M-1} L_{j,k}\frac{\sin\theta_j}{\cos\theta_j}\right\}, \tag{3.28}$$

$$= -\frac{\pi}{MN}\sum_{k=0}^{N-1}\left\{\hat{v}_k \sum_{j=0}^{M-1} L_{j,k}\tan\theta_j\right\}. \tag{3.29}$$

3.6.2 The Translational Gradient

In order to compute the gradient due to translation, we use a derivation similar to that provided by Křivánek et al. [2005a]. We observe that the cosine foreshortening term does not change under a translation, therefore, $\nabla_t E(\mathbf{x})$ simplifies to computing the change in cell area:

$$\nabla_t E(\mathbf{x}) \approx \sum_{j=0}^{M-1}\sum_{k=0}^{N-1} \nabla_t A_{j,k}\, L_{j,k}\cos\theta_j. \tag{3.30}$$

Since we are only interested in translational gradients along the surface, the change in cell area can be reduced to computing the marginal change in the cell walls with respect to translation along the base plane. For each cell (j,k), we decompose the gradient into directional derivatives of the four neighboring cell walls:

$$\nabla_t A_{j,k} = \hat{u}_k \cdot \nabla_{\hat{u}_k} A_{j_-,k} - \hat{u}_k \cdot \nabla_{\hat{u}_k} A_{j_+,k} + \hat{v}_{k_-} \cdot \nabla_{\hat{v}_{k_-}} A_{j,k_-} - \hat{v}_{k_+} \cdot \nabla_{\hat{v}_{k_+}} A_{j,k_+}, \tag{3.31}$$

where $\nabla_{\hat{u}_k} A_{j_-,k}$ computes the scalar directional derivative of area due to movement of boundary (j_-,k) along the $\hat{u}_k$ direction, and the other quantities are defined analogously (see Table 3.1). These directional derivatives can be interpreted as computing the length of the cell wall multiplied by the rate of motion of the wall due to movement in the perpendicular direction. This is illustrated in Figure 3.5.

Since neighboring cells share cell boundaries, it is more convenient to express the translational gradient in Equation 3.30 as a summation over all the cell boundaries instead of the cells. The movement of a cell wall will increase the contribution from one adjacent cell and induce a corresponding decrease from the other adjacent cell. Because of this, we consider the difference

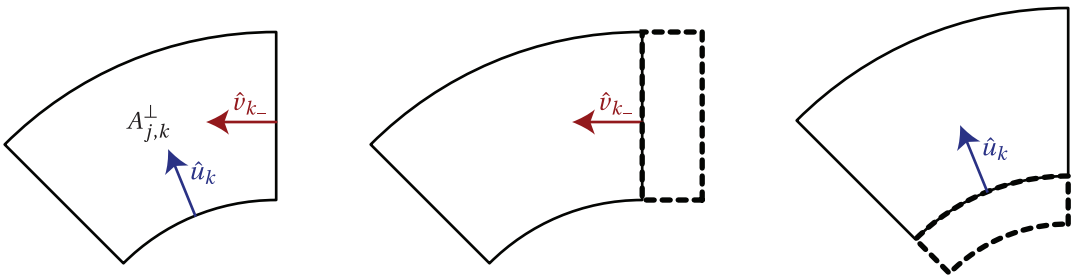


Figure 3.5: The change in cell area due to translation is decomposed into the movement of each cell wall. For both the azimuthal and polar directions, the induced change in cell area is equal to the length of the wall multiplied by the rate of motion of the wall.

in radiance between the two adjacent cells to determine the effect on the overall irradiance sum[3]:

$$\nabla_t E(\mathbf{x}) = \sum_{k=0}^{N-1} \left(\hat{u}_k \sum_{j=1}^{M-1} \nabla_{\hat{u}_k} A_{j_-,k} (L_{j,k} - L_{j\text{-}1,k}) \cos\theta_{j_-} + \hat{v}_{k_-} \sum_{j=0}^{M-1} \nabla_{\hat{v}_{k_-}} A_{j,k_-} (L_{j,k} - L_{j,k\text{-}1}) \cos\theta_j \right). \quad (3.32)$$

This formulation is similar to that provided by Křivánek et al. [2005a] but with the correction that the first term is weighted by the cosine at the cell boundary, $\cos\theta_{j_-}$, instead of at the cell center, $\cos\theta_j$. We derive the rate of motion of the two types of cell walls in the following sections.

Displacement Along $\hat{u}_k$

The induced change in cell area by moving cell wall (j_-, k) along the base plane vector $\hat{u}_k$ is:

$$\nabla_{\hat{u}_k} A_{j_-,k} = \nabla_{\hat{u}_k} \theta_{j_-} \cdot \frac{\partial A_{j,k}}{\partial \theta_{j_-}}, \quad (3.33)$$

$$= \nabla_{\hat{u}_k} \theta_{j_-} \cdot (\phi_{k_+} - \phi_{k_-}) \sin\theta_{j_-}, \quad (3.34)$$

where the $(\phi_{k_+} - \phi_{k_-}) \sin\theta_{j_-}$ arises directly from differentiating the definition of $A_{j,k}$ in Equation 3.21. To determine the change in elevation angle as we move along direction $\hat{u}_k$, we instead

[3]Note that the degenerate cell wall at the pole (where $\theta = 0$) is excluded in the loop by starting at $j = 1$.

consider the analogous problem within the canonical hemispherical parametrization:

$$x = r\cos\phi\sin\theta, \tag{3.35}$$

$$y = r\sin\phi\sin\theta, \tag{3.36}$$

$$z = r\cos\theta, \tag{3.37}$$

$$r = \sqrt{x^2 + y^2 + z^2}, \tag{3.38}$$

$$\phi = \tan^{-1}\left(\frac{y}{x}\right), \tag{3.39}$$

$$\theta = \cos^{-1}\left(\frac{z}{r}\right), \tag{3.40}$$

and determine the change in θ as we translate along x:

$$\begin{aligned}
\frac{\partial\theta}{\partial x} &= \frac{zx}{r^3\left(1-\frac{z^2}{r^2}\right)^{\frac{1}{2}}}, && \text{by differentiating Equation 3.40,} \\
&= \frac{\cos\theta x}{r^2\left(1-\cos^2\theta\right)^{\frac{1}{2}}}, && \text{because } \frac{z}{r} = \cos\theta, \\
&= \frac{\cos\theta x}{r^2\sin\theta}, && \text{because } \sin^2\theta + \cos^2\theta = 1, \\
&= \frac{\cos\theta\cos\phi\sin\theta}{r\sin\theta}, && \text{because } \frac{x}{r} = \cos\phi\sin\theta, \\
&= \frac{\cos\theta}{r}, && \text{because we evaluate at } y = 0, \text{ where } \cos\phi = 1.
\end{aligned} \tag{3.41}$$

In order to apply this derivation to the hemispherical samples, Ward and Heckbert [1992] observed that as the sample location moves, the rate of motion of the boundary between two cells is always determined by the closer of the two sample directions. This observation takes changing occlusions into account. Therefore, the radius in the denominator should use the minimum of the distances for the two neighboring cells, resulting in:

$$\nabla_{\hat{u}_k} A_{j_-,k} = \frac{(\phi_{k_+} - \phi_{k_-})\sin\theta_{j_-}\cos\theta_{j_-}}{\min\left(r_{j,k}, r_{j-1,k}\right)}. \tag{3.42}$$

Displacement Along $\hat{v}_{k_-}$

We also consider a displacement along $\hat{v}_{k_-}$ for cell walls of the form (j, k_-). The induced change in area is:

$$\nabla_{\hat{v}_{k_-}} A_{j,k_-} = \nabla_{\hat{v}_{k_-}} \phi_{k_-} \cdot \frac{\partial A_{j,k}}{\partial \phi_{k_-}}, \tag{3.43}$$

$$= \nabla_{\hat{v}_{k_-}} \phi_{k_-} \cdot (\cos\theta_{j_+} - \cos\theta_{j_-}). \tag{3.44}$$

In order to compute $\nabla_{\hat{v}_{k_-}} \phi_{k_-}$ we again turn to the canonical parametrization and consider the change of ϕ as we translate along y:

$$\begin{aligned}
\frac{\partial \phi}{\partial y} &= \frac{x}{x^2 + y^2}, && \text{by differentiating Equation 3.39,} \\
&= \frac{1}{x}, && \text{because we evaluate at } y = 0, \\
&= \frac{1}{r\cos\phi\sin\theta}, && \text{from Equation 3.35,} \\
&= \frac{1}{r\sin\theta}, && \text{because } \cos\phi = 1 \text{ at } y = 0.
\end{aligned} \tag{3.45}$$

By again using the minimum of the neighboring cell distances we arrive at the following induced change by translating along $\hat{v}_{k_-}$:

$$\nabla_{\hat{v}_{k_-}} A_{j,k_-} = \frac{(\cos\theta_{j_+} - \cos\theta_{j_-})}{\sin\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)}. \tag{3.46}$$

Putting it Together

Plugging the values for the cell wall differentials from Equations 3.42 and 3.46 into Equation 3.32, the final translation gradient is:

$$\begin{aligned}
\nabla_t E(\mathbf{x}) = \sum_{k=0}^{N-1} \Bigg(\hat{u}_k \sum_{j=1}^{M-1} \frac{(\phi_{k_+} - \phi_{k_-})\sin\theta_{j_-}\cos\theta_{j_-}}{\min\left(r_{j,k}, r_{j-1,k}\right)} \left(L_{j,k} - L_{j-1,k}\right)\cos\theta_{j_-} + \\
\hat{v}_{k_-} \sum_{j=0}^{M-1} \frac{(\cos\theta_{j_+} - \cos\theta_{j_-})}{\sin\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)} \left(L_{j,k} - L_{j,k-1}\right)\cos\theta_j \Bigg).
\end{aligned} \tag{3.47}$$

3.6.3 Equivalence of the Gradient Formulations

Ward and Heckbert [1992] compute the translational gradient by considering the movement of the *projected* cell boundaries. Their original formulation is:

$$\nabla_t E(\mathbf{x}) = \sum_{k=0}^{N-1} \left(\hat{u}_k \frac{2\pi}{N} \sum_{j=1}^{M-1} \frac{\sin\theta_{j_-} \cos^2\theta_{j_-}}{\min\left(r_{j,k}, r_{j-1,k}\right)} \left(L_{j,k} - L_{j-1,k}\right) + \right.$$
$$\left. \hat{v}_{k_-} \sum_{j=0}^{M-1} \frac{(\sin\theta_{j_+} - \sin\theta_{j_-})}{\min\left(r_{j,k}, r_{j-1,k}\right)} \left(L_{j,k} - L_{j,k-1}\right) \right). \tag{3.48}$$

The translational gradients we derived in the previous section differ from this formulation. We instead followed the approach of Křivánek et al. [2005a] and considered the movement of the *unprojected* cell boundaries weighted by a cosine term. As Křivánek et al. showed, this approach has extra flexibility since it allows the use of any weighting function in place of the cosine term.

Křivánek et al. [2005a] stated that their "formula yields numerically very similar results to that of Ward and Heckbert." Unfortunately, the derivation by Křivánek et al. [2005a] contained a few minor errors which obscured the fact that these gradient formulations are in fact *analytically* equivalent. In this section we prove this equivalence.

Equivalence Along $\hat{u}_k$

We first consider the gradient contribution along $\hat{u}_k$. In our formulation, the cell radiance difference, $\left(L_{j,k} - L_{j-1,k}\right)$, is weighted by:

$$\frac{(\phi_{k_+} - \phi_{k_-}) \sin\theta_{j_-} \cos\theta_{j_-}}{\min\left(r_{j,k}, r_{j-1,k}\right)} \cdot \cos\theta_{j_-}, \tag{3.49}$$

whereas in the original Ward and Heckbert formulation it is:

$$\frac{2\pi}{N} \frac{\sin\theta_{j_-} \cos^2\theta_{j_-}}{\min\left(r_{j,k}, r_{j-1,k}\right)}. \tag{3.50}$$

For a uniform or cosine-weighted stratification, $(\phi_{k_+} - \phi_{k_-}) = \frac{2\pi}{N}$, which makes these two formulations equivalent along $\hat{u}_k$.

Equivalence Along $\hat{v}_{k_-}$

Showing that the gradient formulations along $\hat{v}_{k_-}$ are equivalent is a bit more involved. Our formulation weights each cell radiance difference, $\left(L_{j,k} - L_{j,k-1}\right)$, by:

$$\frac{\cos\theta_{j_+} - \cos\theta_{j_-}}{\sin\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)} \cdot \cos\theta_j \Longrightarrow \frac{\cos\theta_{j_+} - \cos\theta_{j_-}}{\tan\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)} \tag{3.51}$$

and Ward and Heckbert weight it by:

$$\frac{\sin\theta_{j_+} - \sin\theta_{j_-}}{\min\left(r_{j,k}, r_{j-1,k}\right)}. \tag{3.52}$$

To show that these are equivalent, we start by expressing the boundary angles, θ_{j_-} and θ_{j_+}, as offsets from the angle at the cell center, θ_j:

$$\theta_{j_-} = \theta_j - \frac{\Delta\theta}{2}, \tag{3.53}$$

$$\theta_{j_+} = \theta_j + \frac{\Delta\theta}{2}, \tag{3.54}$$

where $\frac{\Delta\theta}{2}$ is half the cell-width along the θ direction. Using this decomposition, we can rewrite Equations 3.51 and 3.52 as:

$$\frac{\cos\left(\theta_j - \frac{\Delta\theta}{2}\right) - \cos\left(\theta_j + \frac{\Delta\theta}{2}\right)}{\tan\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)}, \qquad \text{and} \qquad \frac{\sin\left(\theta_j + \frac{\Delta\theta}{2}\right) - \sin\left(\theta_j - \frac{\Delta\theta}{2}\right)}{\min\left(r_{j,k}, r_{j-1,k}\right)}.$$

By using the angle sum and difference trigonometric identities:

$$\sin\left(\theta_j \pm \frac{\Delta\theta}{2}\right) = \sin\theta_j \cos\frac{\Delta\theta}{2} \pm \cos\theta_j \sin\frac{\Delta\theta}{2}, \tag{3.55}$$

$$\cos\left(\theta_j \pm \frac{\Delta\theta}{2}\right) = \cos\theta_j \cos\frac{\Delta\theta}{2} \mp \sin\theta_j \sin\frac{\Delta\theta}{2}, \tag{3.56}$$

these two formulations simplify to equivalent expressions:

$$\frac{\left(\cos\theta_j \cos\frac{\Delta\theta}{2} + \sin\theta_j \sin\frac{\Delta\theta}{2}\right) - \left(\cos\theta_j \cos\frac{\Delta\theta}{2} - \sin\theta_j \sin\frac{\Delta\theta}{2}\right)}{\tan\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)} \Longrightarrow \frac{2\sin\theta_j \sin\frac{\Delta\theta}{2}}{\tan\theta_j \min\left(r_{j,k}, r_{j-1,k}\right)},$$

$$\Longrightarrow \frac{2\cos\theta_j \sin\frac{\Delta\theta}{2}}{\min\left(r_{j,k}, r_{j-1,k}\right)}, \tag{3.57}$$

$$\frac{\left(\sin\theta_j \cos\frac{\Delta\theta}{2} + \cos\theta_j \sin\frac{\Delta\theta}{2}\right) - \left(\sin\theta_j \cos\frac{\Delta\theta}{2} - \cos\theta_j \sin\frac{\Delta\theta}{2}\right)}{\min\left(r_{j,k}, r_{j-1,k}\right)} \Longrightarrow \frac{2\cos\theta_j \sin\frac{\Delta\theta}{2}}{\min\left(r_{j,k}, r_{j-1,k}\right)}. \tag{3.58}$$

3.7 Other Extensions

Several other extensions to the irradiance caching method have been proposed in the literature. We briefly summarize some of the most significant contributions in this section.

3.7.1 Approximate Global Illumination

Several papers have explored the topic of accelerating the global illumination computation by making physically inaccurate approximations during the irradiance caching algorithm. These methods loosen the physical restrictions of the computation to gain efficiency, while attempting to minimize the approximation errors and artifacts introduced. We will cover two different approaches to this problem. The first uses simplified scene geometry during the indirect lighting calculation, while the second decomposes the indirect light into near and far components in order to exploit their individual advantages.

Using Simplified Geometry

The algorithm proposed in Tabellion and Lamorlette [2004] was developed within the context of a production rendering system for computer animated films. The primary concerns involved were efficiency and artistic control, as opposed to physically accurate simulation. One of the approximations employed to make the global illumination calculation more computationally tractable was the use of simplified scene geometry during the indirect lighting calculation.

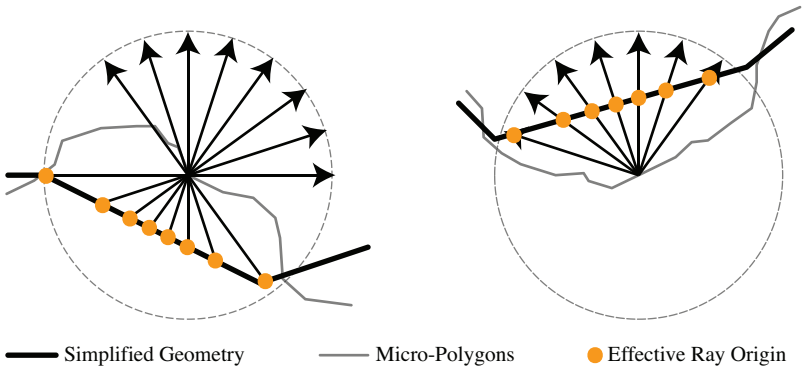


Figure 3.6: Tabellion and Lamorlette adjusted the ray origins in the irradiance estimate in order to compensate for simplified geometry.

The main bottleneck of the irradiance caching algorithm is the time spent tracing rays. To mitigate this, the system proposed by Tabellion and Lamorlette ray traces coarsely tessellated geometry. Since the underlying rendering system was a micro-polygon renderer, coarse representations of the scene geometry were readily available. Using simplified geometry had been proposed in the past [Rushmeier et al., 1994]; however, Tabellion and Lamorlette's system used coarse geometry even near the ray origin. Since rays originate at positions on the fine-scale geometry, problems can occur from incorrect "self-intersections" with the coarse geometry. This problem, and its proposed solution, is illustrated in Figure 3.6.

When tracing an indirect ray, the actual ray origin may need to be adjusted to compensate for the geometric inconsistency. The new ray origin is found by tracing within a user-defined distance both before and after the origin. The closest hit to the ray origin becomes the new ray origin.

Using simplified geometry for the indirect rays allowed Tabellion and Lamorlette to gain an order of magnitude improvement in performance.

Error Metric. Tabellion and Lamorlette also introduced a new error metric used during the weighted averaging of the cached irradiance samples. They argued that the new error metric improves cache point distribution for scenes with high geometric detail by reducing clumping in corners. To compute the error of using a cached sample i at position $\mathbf{x}$, they proposed the

following formula:

$$\varepsilon_i = \kappa \max\left(\varepsilon_{pi}(\mathbf{x}), \varepsilon_{ni}(\vec{\mathbf{n}})\right), \qquad \text{with} \tag{3.59}$$

$$\varepsilon_{pi} = \frac{\|\mathbf{x} - \mathbf{x}_i\|}{\max\left(\min\left(\frac{R_i}{2}, R_+\right), R_-\right)}, \quad \text{and} \tag{3.60}$$

$$\varepsilon_{ni} = \frac{\sqrt{1 - \vec{\mathbf{n}} \cdot \vec{\mathbf{n}}_i}}{\sqrt{1 - \cos(10°)}}. \tag{3.61}$$

Here, κ is the only user-adjustable quality parameter and is by default set to 1. R_i is the closest distance to surfaces seen from $\mathbf{x}_i$. R_- and R_+ are 1.5 and 10 times the square root of the projected pixel area, respectively. These act like minimum and maximum valid radii constraints on the cache points. Their inclusion brings about the reduced clumping properties of the metric.

A weighted average of cache points whose weights are non-negative are included in the interpolated irradiance. The new error metric changes the weighting function used to

$$\mathrm{w}_i(\mathbf{x}, \vec{\mathbf{n}}) = 1 - \varepsilon_i(\mathbf{x}, \vec{\mathbf{n}}). \tag{3.62}$$

Irradiance Decomposition

Arikan et al. [2005] noted that, overall, irradiance caching is an excellent technique for efficiently computing global illumination but suffers from over-sampling around areas of high geometric complexity due to the potentially rapid change in irradiance in these areas. Especially with the trend of increasing geometric complexity, these areas can easily dominate the global illumination computation. The authors attempted to directly attack this problem by decomposing the incoming irradiance into two parts and exploiting the properties of each part individually.

The incident hemisphere Ω is partitioned into Ω_D and Ω_N based on whether a distant or near point is visible in that direction. The distinction of near and far is based on a user-defined distance threshold. This also implies a partitioning of incoming radiance: L_{i_D}, incident radiance from distant surfaces, and L_{i_N}, incident radiance from nearby surfaces.

By construction, L_{i_D} changes very slowly across surfaces and is therefore an excellent candidate for interpolation. In contrast, L_{i_N} varies rapidly and is handled separately.

Arikan et al. cached radiance instead of irradiance to attack the over-sampling problem. The irradiance due to distant surfaces is computed using Monte Carlo integration, and the resulting radiance field is stored as nine SH coefficients. Instead of sampling the hemispherical radiance at specific cache points, Arikan et al. distributed the origin of the hemispherical sample rays over multiple surface points. This allowed them better coverage of the whole radiance field while using fewer cache points. The specifics of this procedure are explained in detail in the original paper [Arikan et al., 2005].

The incident irradiance due to nearby surfaces is calculated using analytic point-to-triangle form factors assuming perfect visibility. This assumption is not always true and is the source of most of the approximation error. Since this quantity can change rapidly it is computed per pixel.

These two irradiance values cannot be combined directly, since the geometry from the nearby irradiance might block the irradiance from distant surfaces. Therefore, when computing the analytic form factors, the distant irradiance is sampled in the direction of the triangle and is assumed to be constant over the whole triangle. These quantities are subtracted to account for the occlusion.

3.7.2 Distributed Global Illumination

The *Kilauea* renderer was designed to handle extreme geometric complexity within a production environment while producing images with high-quality global illumination [Kato, 2002]. Kilauea used photon mapping for global illumination in combination with an irradiance cache-like algorithm for efficient final gather. To satisfy the primary goal of high image quality, the renderer traces $65 \times 65 = 4,225$ rays per irradiance sample. This large amount of gather rays guarantees smooth and accurate results for all but the most grueling scenes, even during animation. Kilauea handles this ray tracing complexity by employing massively parallel rendering. However, as a distributed renderer based on message passing, global illumination within Kilauea has a different performance profile to address. Tracing rays within such a context incurs an even larger overhead than in a traditional ray tracer. Thus, a method which minimizes the total number of final gather rays is essential.

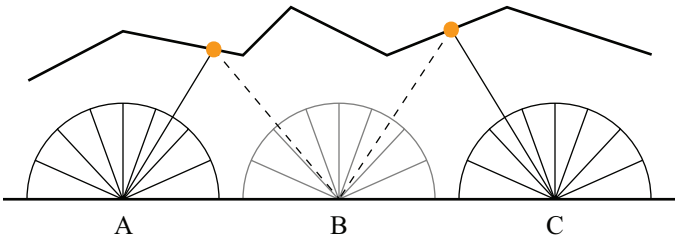


Figure 3.7: Hit point reprojection. We can estimate the radiance strata at location B by reprojecting the hit points from both A and C.

Hit Point Reprojection

The approach taken by Kato [2002] employed final gather reprojection to reduce the number of necessary rays. Instead of simply storing an irradiance value, the whole stratified radiance hemisphere is stored as a 2D grid of 65 × 65 colors and depths. Given this information from neighboring points, we can reproject each strata's hit points onto a different irradiance location, as in Figure 3.7. To minimize the amount of unresolved pixels, four neighboring cache points are used in the reprojection. Nonetheless, gaps could still occur. If a missing stratum is surrounded by valid strata, these values are just interpolated. All remaining gaps are computed by tracing rays. The authors found that this method can drastically reduce the number of necessary final gather rays.

The memory requirements for storing all the hemispherical sample information for each cache point might seem overwhelming, but the pixels in the rendered image are processed in small independent buckets. As a result, this information never needs to be stored for a whole image at once.

Adaptive Sample Placement

In addition to the reprojection, Kilauea employs a hybrid screen/object-space irradiance sample placement algorithm. The pixels in the rendered image are processed in small, independent, rectangular buckets. One bucket at a time, the algorithm first computes the hit location and normal for each pixel. By comparing the location and orientation of neighboring pixel hit points, "critical pixels" are identified. These critical pixels roughly correspond to silhouettes and creases in the scene and, for accuracy, require un-interpolated irradiance computations. For all

remaining “safe pixels,” a precise irradiance estimate is computed (using Equation 3.4 with hit point reprojection) sparsely over the image plane, typically on a grid of every 16×16 pixels. This grid imposes a tiling over which interpolation or subdivision can occur. If no critical pixels overlap a tile, then that tile is a candidate for interpolation. The harmonic mean distance (Equation 3.9) of the corner irradiance samples is evaluated. If the means are found to be large, the irradiance is predicted to vary smoothly, and, consequently, a simple bilinear interpolation of the corner irradiance samples is computed over the entire tile. If critical pixels do overlap the tile, or if the harmonic mean distances are too small, the tile is uniformly subdivided. This process continues down to a user-defined minimum tile size.

Using both adaptive sample placement and hit point reprojection, the authors were able to reduce the number of irradiance gather rays per pixel to the range of $3 - -25$ for typical scenes, in contrast to the $4,225$ samples that would be needed to compute a full 65×65 estimate at each pixel.

3.7.3 Animation

Temporal Aliasing and Coherence

In a naïve extension of irradiance caching to animation, a new irradiance cache would be computed per frame. This new cache would not only have different irradiance values for each cache point, but the location of the irradiance cache points would be different for every frame. Due to the stochastic sampling of the irradiance and the lack of inter-frame coherence of cache point locations, significant differences would arise between frames. The approximation errors due to interpolation and sparse sampling might be unnoticeable for an individual frame, but when these artifacts change drastically from frame to frame, the illusion is ruined. This phenomenon results in what appears as “popping” or “flickering” of the indirect illumination.

Age-based Invalidation. The observation made by Tawara et al. [2004] is that, for many regions of a scene, the irradiance changes slowly and minimally over time when objects are animated. It would be fruitful to exploit this feature, if only these regions could somehow be identified. Instead of explicitly identifying these regions, however, the authors decided to probabilistically

re-evaluate only a small subset of all global illumination rays using a simple age-based heuristic.

In order to accommodate this, the irradiance caching algorithm is extended by storing radiance values instead of irradiance. The full stratified hemisphere of samples is stored, caching the radiance and the distance to the nearest hit point for each stratum. In addition, each stratum is assigned an age, which is initialized to 0. As the animation progresses, the age of each stratum is incremented. A random subset of the oldest radiance strata is selected to be retraced each frame.

Updating a subset of the strata has many advantages. Firstly, by choosing only 10% of all strata per frame, the render speed improves by approximately a factor of 10. Furthermore, in this example, each radiance strata can be expected to be updated, on average, every 10 frames. Moreover, reusing previous radiance cache locations reduces the temporal flickering problem described above.

The authors also suggested an improved heuristic which takes the visibility of dynamic objects into account. A flag is included to indicate whether each stratum intersected a static or dynamic object. When randomly updating strata, more weight is given to strata viewing dynamic objects. This adapts the process to update more vigorously in scene regions near moving objects.

Two issues remain: how do we handle irradiance cache points on dynamic objects, and how do we deal with the fact that the optimal density and distribution of cache point locations changes over time? To address the first issue, the authors simply attached the irradiance samples to dynamic objects. The cache point locations are transformed with their underlying object. This is a gross approximation since the radiance directions will no longer be valid once transformed, but the authors claimed that the artifacts are negligible for slowly moving objects. The second problem can be addressed by searching within the local neighborhood of each cache point and eliminating a point if the number of nearby neighbors is larger than a given threshold (such as 10). This technique showed good results in adapting the cache point distribution to the changing needs of the animated environment.

Separate Static and Dynamic Caches. An alternate approach to age-based invalidation was proposed by Tawara et al. [2002]. Here, the proposed technique separates the irradiance cache into a static and dynamic cache. The motivation behind this is much the same as with irradiance

decomposition by Arikan et al. [2005]. In this case, however, we decompose the irradiance with respect to motion, and not distance, to exploit the properties of the static and dynamic irradiance caches independently.

The static cache contains only irradiance from non-moving objects and hence does not flicker. Static objects can be determined with respect to the whole animation sequence, or the animation can be divided into shorter segments and static objects identified within each time segment. Dynamic objects are handled separately. A distinct dynamic irradiance cache is created which contains contributions from moving objects.

These two caches cannot be directly combined because changing occlusion and inter-reflection from dynamic objects needs to be properly integrated into the final irradiance. The proposed algorithm uses irradiance caching in conjunction with photon mapping [Jensen, 2001]. Changes in occlusions from dynamic objects are handled by photons with negative energy in the same spirit as "shadow photons" [Jensen, 2001]. Utilizing information in this dynamic photon map, the two irradiance caches can be effectively combined.

Exploiting the Human Visual System

The sensitivity of the Human Visual System (HVS) changes with respect to both spatial contrast and temporal effects. For instance, human observers are most sensitive to low frequency spatial patterns and less so to high frequencies. Furthermore, people can discern moving objects with different levels of accuracy with respect to speed. Fast moving objects are more noticeable than static or slow moving objects, but very fast movement can interfere with the eye's tracking ability. Peak temporal sensitivity lies somewhere in between. These spatiotemporal sensitivity disparities allow for an increased level of error tolerance when generating synthetic images.

Using these properties of the HVS, Yee [2000] improved the efficiency of the global illumination computation by computing a spatiotemporal error tolerance map, which determines how important each pixel is for an individual frame in an animation. This map, the Aleph map, is used as an oracle that guides perceptually-driven rendering of the global illumination solution.

The algorithm requires the computation of estimated frames for the animation. These frames can be generated using graphics hardware or by ray tracing with direct illumination only.

The frame estimates serve as input for constructing the spatiotemporal sensitivity map. Though the details of the algorithm are beyond the scope of this paper, the Aleph map computation incorporates the motion of each pixel, through motion estimation, as well as the varying intensity, color, and orientation content of the image. This results in a single grayscale image which identifies areas of increased attention based on the HVS.

The perceptual information is applied to the irradiance caching algorithm by modifying the error threshold "a" in Equation 3.9 based on the Aleph map value of the underlying pixel. Recall that the error threshold parameter determines the search radius for nearby cache locations. This parameter is increased for pixels with a larger error tolerance as governed by the Aleph map. This allows for performing fewer irradiance computations and more irradiance interpolations in areas of low interest.

Yee was able to achieve a 6x to 11x speedup over the base irradiance caching algorithm by using spatiotemporal information. It is important to note that this method also works for static scenes and, by using the spatial importance information alone, a speedup of around 2x was observed.

3.8 Limitations

Over the past 20 years, irradiance caching methods have proven to be a practical solution to the global illumination problem. The many extensions in the literature speak to the versatility of the approach. Irradiance caching and derived methods can be used to effectively simulate lighting in scenes containing diffuse as well as glossy materials, and the algorithm has been successfully applied to animation and the production of computer generated films. However, one underlying assumption limits the applicability of all of these extensions.

All of the previous work has assumed that light travels unobstructed between surfaces. None of the previous work has addressed the issue of volumetric scattering. In the next chapter we extend our theoretical model of light to include effects from scattering media. Light transport in scattering media is a potentially fruitful related problem, which could benefit from the same approach that has worked so well for indirect surface illumination. Though the high-level ideas

could very well be applied within participating media, most of the underlying derivations in irradiance caching explicitly make assumptions which are only valid at surfaces. Therefore, in Chapter 5 we develop a completely new algorithm for caching illumination within participating media, which is inspired by irradiance caching techniques.

Unfortunately, scenes containing participating media also pose a problem for irradiance caching at surfaces since these scenes invalidate many underlying assumptions. Indirect lighting from participating media affects the irradiance gradient at surfaces, invalidating the gradient derivations presented in this chapter. In Chapter 6 we extend the irradiance gradient formulation to properly include effects from participating media.

4

Light Transport in Participating Media

"Thus, if one is to be five times as distant, make it five times bluer."

—Leonardo Da Vinci, 1452–1519

IN previous chapters we assumed that all lighting interactions occurred at surfaces. In particular, the rendering equation from Chapter 2 assumes that the radiance leaving a surface remains unchanged until it hits another surface. The property that radiance stays constant between these two points is expressed in Equation 2.10. However, this relationship is only true when the surfaces are embedded within a vacuum, since photons are able to travel unobstructed as they propagate from one surface to another. In reality, however, most of the scenes we observe from day to day do not satisfy this constraint. The space between objects is typically occupied by some host medium (e.g. air, water) which additionally contains impurities and microscopic suspended particles. In these situations, the medium *participates* in the lighting interactions as light travels between surfaces.

If there are very few particles within the medium, such as over short distances within clean air, the assumption that light travels unobstructed can be a reasonable approximation. However, even clean air scatters light (this is the reason the sky is blue) and as light travels over longer distances, such as in open outdoor scenes, more interactions with the particles are likely to occur.

In this chapter we expand our model for the behavior of light by considering effects caused by participating media. We relax the constraints imposed by Equation 2.9 and 2.10 by deriving expressions for how radiance changes as it travels through participating media. This

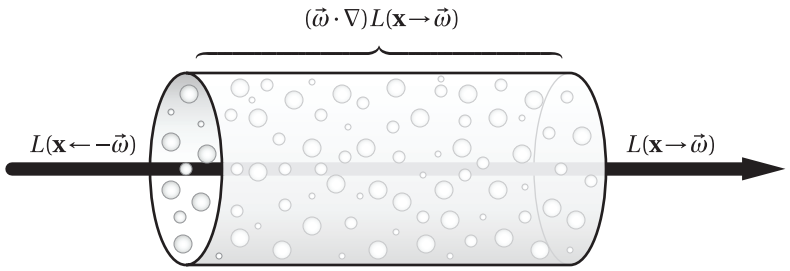


Figure 4.1: We treat participating media as a collection of microscopic scattering particles. When lights travels through the medium, a change of radiance, $(\vec{\omega}\cdot\nabla)L(\mathbf{x},\vec{\omega})$, may occur as the photons interact with the particles.

generalization allows us to simulate lighting within clouds, murky water, fog and any other medium which participates in the lighting interactions.

4.1 Assumptions About Scattering Media

In this dissertation, and most computer graphics applications, assumptions are made about the properties of the scattering media in order to more easily derive expressions about the behavior of light. In particular, we assume that the participating media can be modeled as a collection of microscopic particles (see Figure 4.1). Since the particles are microscopic and randomly positioned, we do not represent each individual particle in the lighting simulation. Instead, we consider the aggregate probabilistic behavior as light travels through the medium. Moreover, these particles are assumed to be spaced far apart relative to the size of an individual particle. This assumption implies that as a photon travels through the medium and interacts at a particle, this interaction is *statistically independent* from the outcome of subsequent interaction events.

4.2 Light Interaction Events

When a photon travels through a collection of microscopic particles, it may either miss all the particles and continue unaffected, or it may interact with some of the particles. The probability that an interaction does occur is related to the extinction coefficient, σ_t (units [1/m]), of the medium. This quantity depends on the density and size of the particles within the medium.

When an interaction occurs, two things may happen: the photon may be absorbed by the particle (by being converted to another form of energy, such as heat), or the photon may be scattered in another direction. The relative probabilities of these two events is given by the absorption, σ_a, and the scattering, σ_s, coefficients, and $\sigma_t = \sigma_a + \sigma_s$ is the extinction coefficient. Either of these two events lead to a change of radiance along the ray.

4.2.1 Extinction

Consider a thin beam of light traveling through a medium in direction $\vec{\omega}$ (see Figure 4.1). The number of photons entering this beam is proportional to the incident radiance $L(\mathbf{x} \to \vec{\omega})$ at the start of the beam $\mathbf{x}$. At each small step Δt along the beam, some fraction of the photons will interact with the medium and become absorbed. If the absorption coefficient within the segment is $\sigma_a(\mathbf{x} + t\vec{\omega})$, then a fraction $\sigma_a(\mathbf{x} + t\vec{\omega})\,\Delta t$ of the photons will be absorbed. Hence, the number of photons exiting this segment can be expressed as:

$$L((\mathbf{x} + t\vec{\omega}) \to \vec{\omega}) = L(\mathbf{x} \to \vec{\omega})\,(1 - \sigma_a(\mathbf{x} + t\vec{\omega})\,\Delta t). \tag{4.1}$$

By rearranging terms, we can determine the change in outgoing radiance between the start and end of this segment:

$$\frac{L((\mathbf{x} + t\vec{\omega}) \to \vec{\omega}) - L(\mathbf{x} \to \vec{\omega})}{\Delta t} = -\sigma_a(\mathbf{x})\,L(\mathbf{x} \to \vec{\omega}). \tag{4.2}$$

Taking the limit as $\Delta t \to 0$ computes the derivative. The differential change in radiance due to *absorption* as light moves along a direction $\vec{\omega}$ is:

$$\lim_{\Delta t \to 0} \left(\frac{L((\mathbf{x} + t\vec{\omega}) \to \vec{\omega}) - L(\mathbf{x} \to \vec{\omega})}{\Delta t} \right) = (\vec{\omega} \cdot \nabla_a)\,L(\mathbf{x} \to \vec{\omega}) = -\sigma_a(\mathbf{x})\,L(\mathbf{x} \to \vec{\omega}), \tag{4.3}$$

where ∇^a denotes the gradient due to absorption, and $(\vec{\omega} \cdot \nabla_a)$ computes the directional derivative along $\vec{\omega}$.

At each step, the radiance may also be reduced due to photons being scattered into other directions. The probability of this happening within the segment is determined by the

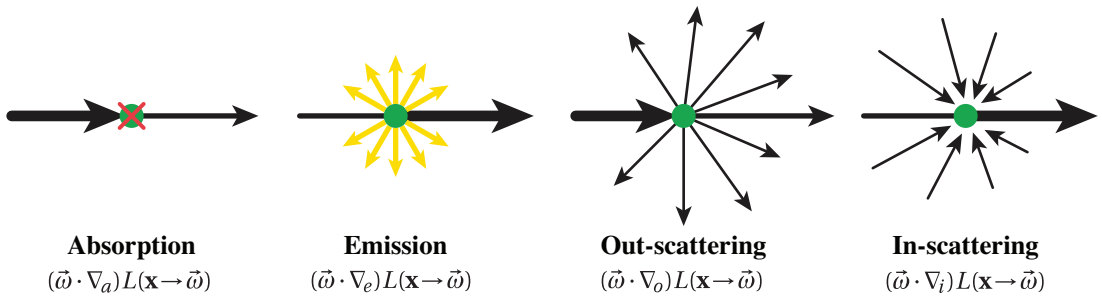


Figure 4.2: As light travels through a participating medium the radiance may change as a result of four different types of interactions: absorption, emission, out-scattering and in-scattering.

scattering coefficient σ_s of the medium. Following the same derivation we can express the change in radiance due to *out-scattering* as:

$$(\vec{\omega}\cdot\nabla_o)L(\mathbf{x}\rightarrow\vec{\omega}) = -\sigma_s(\mathbf{x})\,L(\mathbf{x}\rightarrow\vec{\omega}). \tag{4.4}$$

The net loss of radiance due to either absorption or out-scattering is called extinction, and it can be expressed as:

$$\begin{aligned}(\vec{\omega}\cdot\nabla_t)L(\mathbf{x}\rightarrow\vec{\omega}) &= -(\sigma_a(\mathbf{x})+\sigma_s(\mathbf{x}))\,L(\mathbf{x}\rightarrow\vec{\omega})\\ &= -\sigma_t(\mathbf{x})\,L(\mathbf{x}\rightarrow\vec{\omega}).\end{aligned} \tag{4.5}$$

The differential equation above can be solved to find the *beam transmittance*, or simply transmittance, denoted T_r. Transmittance gives the fraction of photons that can travel unobstructed between two points in the medium along a straight light. The remaining fraction accounts for photons that have either been absorbed or out-scattered in a different direction. By integrating Equation 4.5 we can express the transmittance as:

$$T_r(\mathbf{x}'\leftrightarrow\mathbf{x}) = e^{-\tau(\mathbf{x}'\leftrightarrow\mathbf{x})}. \tag{4.6}$$

The term in the exponent, τ, is called the *optical thickness* or *optical depth*, and comes about by

integrating the effect of extinction along a line segment:

$$\tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \int_0^d \sigma_t(\mathbf{x} + t\vec{\omega})\, dt, \tag{4.7}$$

where $\mathbf{x} + t\vec{\omega}$ with $t \in [0, d]$ parametrizes the line segment between $\mathbf{x}$ and $\mathbf{x}'$. In a homogeneous medium, where σ_t is a constant, τ can be trivially computed as:

$$\tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = d\sigma_t. \tag{4.8}$$

The transmittance is always between zero and one. When only absorption and out-scattering are considered, the transmittance can be used to compute the *reduced radiance*,

$$L(\mathbf{x} \leftarrow \vec{\omega}) = T_r(\mathbf{x}' \leftrightarrow \mathbf{x})\, L(\mathbf{x}' \rightarrow -\vec{\omega}), \tag{4.9}$$

which describes how radiance is reduced as it travels from $\mathbf{x}$ to $\mathbf{x}'$. This relation replaces Equation 2.10 if extinction effects are considered and is also known as *Beer's Law* [Bouguer, 1729].

The transmittance between a point and itself is always one: $T_r(\mathbf{x} \leftrightarrow \mathbf{x}) = 1$. Another useful property is that the transmittance between three points along a line is multiplicative:

$$T_r(\mathbf{x} \leftrightarrow \mathbf{x}'') = T_r(\mathbf{x} \leftrightarrow \mathbf{x}')\, T_r(\mathbf{x}' \leftrightarrow \mathbf{x}''), \tag{4.10}$$

for all points $\mathbf{x}'$ between $\mathbf{x}$ and $\mathbf{x}''$.

4.2.2 In-Scattering and Emission

In addition to loss of radiance, radiance may increase as it travels through the medium. The same process that causes the radiance to reduce along $\vec{\omega}$ due to out-scattering may cause it to increase due to *in-scattering* from other directions. In effect, scattering causes a loss of radiance along the incoming direction and a corresponding increase in radiance into the scattered direction. The change in radiance due to in-scattering can be derived in a similar fashion to

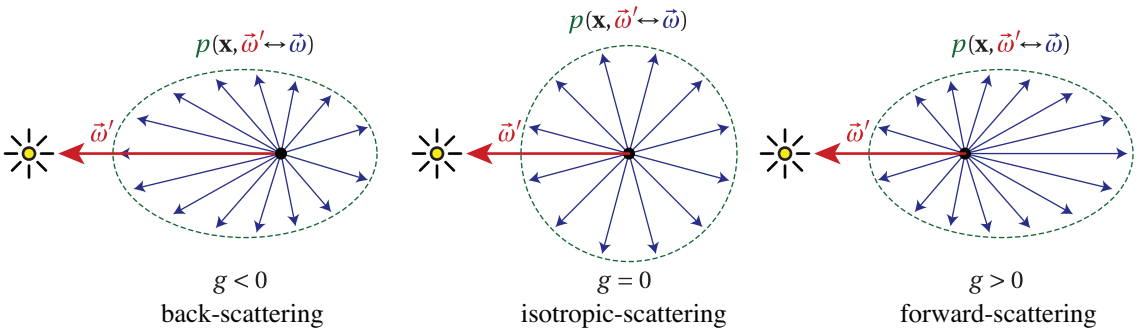


Figure 4.3: The phase function describes the angular distribution of light scattering at any point $\mathbf{x}$ within participating media. In the simplest case, light is scattered equally in all directions (middle). Many natural materials, however, scatter light preferentially in the backward (left) or forward (right) direction.

extinction and out-scattering above, but instead depends on L_i, the in-scattered radiance:

$$(\vec{\omega} \cdot \nabla_i) L(\mathbf{x} \rightarrow \vec{\omega}) = \sigma_s(\mathbf{x})\, L_i(\mathbf{x} \rightarrow \vec{\omega}), \tag{4.11}$$

We explain in-scattered radiance in more detail in the next section.

Media, such as fire, may also emit radiance, L_e, by spontaneously converting other forms of energy into visible light. This emission leads to a gain in radiance expressed as:

$$(\vec{\omega} \cdot \nabla_e) L(\mathbf{x} \rightarrow \vec{\omega}) = \sigma_a(\mathbf{x}) L_e(\mathbf{x} \rightarrow \vec{\omega}). \tag{4.12}$$

4.2.3 Medium Properties

The absorption σ_a and scattering coefficients σ_s, along with the phase function p discussed in the next section, fully define the local behavior of light in a participating medium. If the medium coefficients are constant throughout the medium, then the medium is said to be *homogeneous*. This can be an acceptable approximation in the case of uniform fog for instance. Most media is *heterogeneous*, which means that the absorption and scattering coefficients may vary throughout the medium.

Another descriptive property of the medium can be derived from the *unitless* ratio between the scattering and extinction coefficients, $\frac{\sigma_s}{\sigma_t}$. This is called the scattering *albedo*, and it gives the probability of a photon scattering at a particular location in the medium. An albedo of

zero means that the particles of the medium do not scatter light, while an albedo of one means that the particles do not absorb light. The albedo has a similar meaning as the average reflectivity of a surface.

4.3 In-Scattered Radiance and the Phase Function

4.3.1 In-Scattered Radiance

In-scattered radiance $L_i(\mathbf{x}\rightarrow\vec{\omega})$ represents all the light which is scattered into the thin beam along direction $\vec{\omega}$[1]. The computation of in-scatter radiance involves integrating incident radiance over the whole sphere of directions, $\Omega_{4\pi}$,

$$L_i(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega_{4\pi}} p(\mathbf{x},\vec{\omega}'\rightarrow\vec{\omega})\, L(\mathbf{x}\leftarrow\vec{\omega}')\, d\vec{\omega}', \tag{4.13}$$

where $p(\mathbf{x},\vec{\omega}'\rightarrow\vec{\omega})$ is the phase function describing the angular distribution of light scattering at a point in the medium (see Figure 4.3). Due to this function, in-scattered radiance is usually not isotropic (unlike emission and absorption), but depends on the outgoing direction.

The computation of in-scattered radiance closely parallels the computation of reflected radiance on surfaces. Note the similarity of in-scattered radiance in Equation 4.13 with reflected radiance in Equation 2.14. The distinction is that the BRDF is replaced with the phase function, there is no cosine foreshortening term (since no surface is present), and the integration is performed over the whole sphere of directions instead of the hemisphere.

4.3.2 Properties of the Phase Function

1. **Reciprocity.** Just like the BRDF, the phase function respects Helmholtz's law of reciprocity, so $p(\mathbf{x},\vec{\omega}'\rightarrow\vec{\omega}) = p(\mathbf{x},\vec{\omega}'\leftarrow\vec{\omega})$. To indicate this we use the following notation:

$$p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega}). \tag{4.14}$$

[1] Note that in-scattered radiance represents exitant, not incident, radiance. Hence, the notation $L_i(\mathbf{x}\rightarrow\vec{\omega})$ is used.

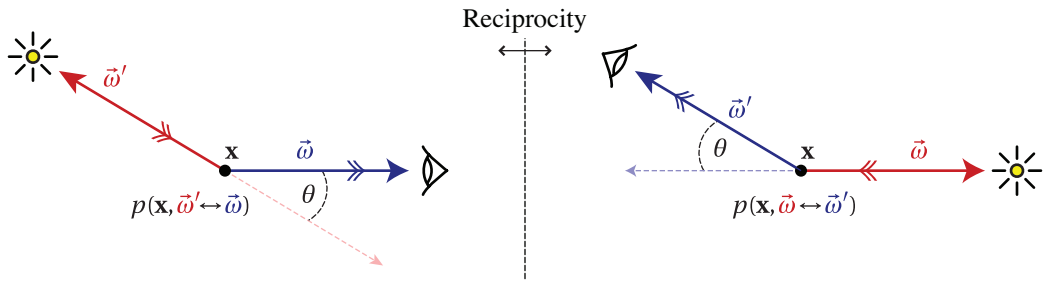


Figure 4.4: The phase function obeys Helmholtz's reciprocity principle: the value of the function remains unchanged if the direction of light is reversed. This is equivalent to swapping the location of the viewer and the light source (right), i.e., $p(\mathbf{x}, \vec{\omega}' \rightarrow \vec{\omega}) = p(\mathbf{x}, \vec{\omega} \rightarrow \vec{\omega}')$.

Note that in this dissertation the incident and outgoing directions for the phase function both point away from the evaluation location $\mathbf{x}$[2]. Additionally, the phase function often depends only on the angle θ between the incoming and outgoing directions, in which case it can be expressed simply as $p(\mathbf{x}, \theta)$, where $\theta = \cos^{-1}(-\vec{\omega} \cdot \vec{\omega}')$. An illustration of these concepts is provided in Figure 4.4.

2. **Normalization.** Unlike the BRDF the phase function has units of [1/sr]. It is also assumed normalized based on the derivations in this dissertation. This means that it always integrates to exactly one over the sphere of directions[3]:

$$\int_{\Omega_{4\pi}} p(\mathbf{x}, \vec{\omega}' \leftrightarrow \vec{\omega}) \, d\vec{\omega}' = 1, \quad \forall \vec{\omega}. \tag{4.15}$$

4.3.3 Examples of Phase Functions

Many different phase functions have been developed in the literature. We briefly explain some of the phase functions commonly used in computer graphics.

The Isotropic Phase Function

The equivalent to diffuse reflection is perfect isotropic scattering. The isotropic phase function is a constant and, based on the constraint in Equation 4.15, it is straightforward to show

[2] Some texts instead follow the convention that one vector points into $\mathbf{x}$ and the other vectors points away from $\mathbf{x}$.

[3] Some texts follow different derivations and instead define the phase function to integrate to the albedo, σ_s/σ_t, or to 4π.

that it equals

$$p_I(\mathbf{x}, \vec{\omega}' \leftrightarrow \vec{\omega}) = \frac{1}{4\pi}. \tag{4.16}$$

Isotropic media scatters light uniformly in all directions regardless of the incoming direction.

The Henyey-Greenstein Phase Function

One of the most commonly used non-isotropic phase functions is the Henyey-Greenstein phase function [1941], which was empirically derived to model the scattering of light by intergalactic dust. Due to its simplicity the Henyey-Greenstein phase function has also been successfully applied for simulating many other scattering materials, such as oceans, clouds, and skin.

The function only depends on the angle θ and is defined as:

$$p_{HG}(\mathbf{x}, \theta) = \frac{1 - g^2}{4\pi(1 + g^2 - 2g\cos\theta)^{1.5}}. \tag{4.17}$$

The single parameter, $g \in [-1, 1]$, determines the relative strength of forward and backward scattering.

The asymmetry parameter g has a physical meaning—it represents the average cosine of the scattered directions. Positive values of g give forward scattering, and negative values give backwards scattering. A value of $g = 0$ results in isotropic scattering. A convenient side-effect of this definition of g is that more complex scattering functions can easily be approximated by the Henyey-Greenstein function by computing g as:

$$g(\mathbf{x}) = \int_{\Omega_{4\pi}} p(\mathbf{x}, \vec{\omega}' \leftrightarrow \vec{\omega}) \cos\theta \, d\vec{\omega}'. \tag{4.18}$$

The Schlick Phase Function

Though the Henyey-Greenstein phase function is intuitive and flexible, the computation of the fractional exponent in the denominator is relatively costly. In computer graphics, the exact shape of the Henyey-Greenstein phase function is usually not essential. This led Blasi and

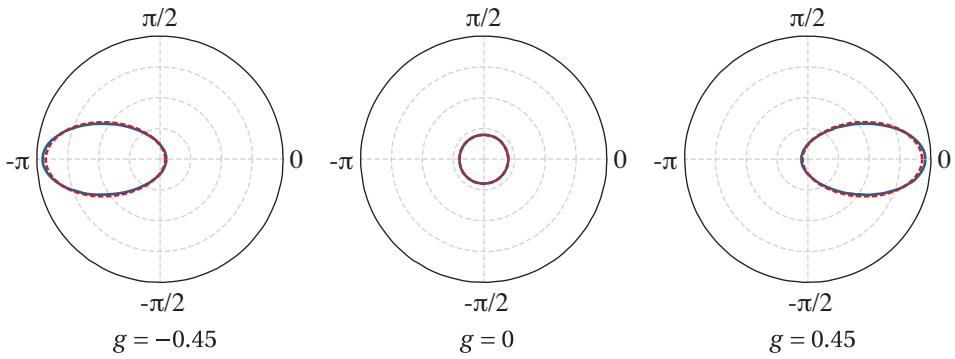


Figure 4.5: Polar plots visualizing the Henyey-Greenstein and Schlick phase functions as functions of θ. The Schlick phase function (red) closely approximates the shape of the Henyey-Greenstein phase function (blue) and is more efficient to evaluate.

colleagues to develop a similar phase function, which is more efficient to evaluate [1993]. The resulting Schlick phase function approximates the shape of the Henyey-Greenstein function as an ellipsoid:

$$p_S(\mathbf{x},\theta) = \frac{1-k^2}{4\pi(1+k\cos\theta)^2}, \tag{4.19}$$

where $k \in [-1,1]$ acts similarly to the g parameter and controls the preferential scattering direction. As with g, total backward scattering is obtained with $k=-1$, $k=1$ gives total forward scattering, and $k=0$ corresponds to isotropic scattering. Pharr and Humphreys [2004] found that for intermediate values of k, the polynomial equation

$$k \approx 1.55g - 0.55g^3 \tag{4.20}$$

can be used to give accurate approximations to the Henyey-Greenstein phase function. The Henyey-Greenstein and Schlick phase functions are shown in Figure 4.5 for three different values of g.

Rayleigh Scattering

In 1871, Rayleigh derived expressions for the scattering of light off molecules of air [1871]. Rayleigh scattering is an approximation for the behavior of light as it scatters in media composed

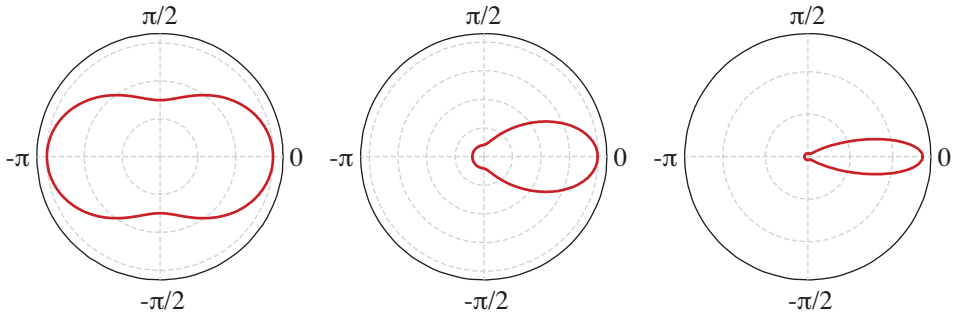


Figure 4.6: The Rayleigh scattering phase function (left) consists of a symmetric forward and backward scattering contribution. The Hazy (middle) and Murky (right) phase functions are approximations to the behavior of scattering in foggy atmospheres dictated by Lorenz-Mie theory.

of particles up to about a tenth of the wavelength of light. The model is derived by assuming a random distribution of very small specular spheres, and the resulting phase function is fairly simple:

$$p_R(\mathbf{x},\theta) = \frac{3}{16\pi}(1+\cos^2\theta). \tag{4.21}$$

The Rayleigh phase function is visualized in Figure 4.6.

Rayleigh scattering is also highly dependent on the wavelength of light. This is captured by a wavelength-dependent scattering coefficient,

$$\sigma_s = \frac{2\pi^5}{3}\frac{d^6}{\lambda^4}\left(\frac{n^2-1}{n^2+2}\right)^2, \tag{4.22}$$

where λ is the wavelength of light, and d and n are the diameter and refractive index of the particles, respectively. This wavelength dependence of Rayleigh scattering is what makes the sky blue and sunsets red.

Lorenz-Mie Theory

Rayleigh scattering is just an approximation that works when the size of the particles is small relative to the wavelength. When the particles are comparable to the wavelength of light, such as water droplets in fog, the more complicated Lorenz-Mie theory of scattering becomes applicable [Lorenz, 1890; Mie, 1908]. Lorenz-Mie theory can be used to derive phase functions

for a homogeneous collection of spherical particles where any ratio of diameter to wavelength is allowed. Mie scattering is based on more general theory derived from Maxwell's equations.

Nishita et al. [1987] provide two empirically derived approximations to the complicated Lorenz-Mie scattering functions for foggy atmospheres, one for "hazy" atmospheres, and one for "murky" atmospheres:

$$p_{MH}(\mathbf{x},\theta) = \frac{1}{4\pi}\left(\frac{1}{2} + \frac{9}{2}\left(\frac{1+\cos\theta}{2}\right)^8\right) \tag{4.23}$$

$$p_{MM}(\mathbf{x},\theta) = \frac{1}{4\pi}\left(\frac{1}{2} + \frac{33}{2}\left(\frac{1+\cos\theta}{2}\right)^{32}\right). \tag{4.24}$$

These phase functions are shown in Figure 4.6.

More recently, generalizations of Lorenz-Mie theory have been presented which can compute the scattering properties of a collection of non-spherical particles within an absorbing host medium [Frisvad et al., 2007].

4.4 The Volume Rendering Equation

4.4.1 The Radiative Transfer Equation

Given the four scattering events described in the previous sections we can begin to form a complete model of how light behaves in a participating medium. By combining Equations 4.5, 4.11, and 4.12, the total change in radiance along the ray at a position $\mathbf{x}$ can be expressed as

$$(\vec{\omega}\cdot\nabla)L(\mathbf{x}\rightarrow\vec{\omega}) = -\underbrace{\underbrace{\underbrace{\sigma_a(\mathbf{x})L(\mathbf{x}\rightarrow\vec{\omega})}_{\text{absorption}} - \underbrace{\sigma_s(\mathbf{x})\,L(\mathbf{x}\rightarrow\vec{\omega})}_{\text{out-scattering}}}_{\text{extinction}} + \underbrace{\sigma_a(\mathbf{x})\,L_e(\mathbf{x}\rightarrow\vec{\omega})}_{\text{emission}} + \underbrace{\sigma_s(\mathbf{x})\,L_i(\mathbf{x}\rightarrow\vec{\omega})}_{\text{in-scattering}}}. \tag{4.25}$$

This equation is known as the integro-differential form of the *radiative transfer*, or *radiative transport equation*, or simply the RTE [Chandrasekhar, 1960], and it incorporates the four possible types of interaction events that can occur within the medium (recall the illustration in Figure 4.2).

By integrating both sides of Equation 4.25 and using the rendering equation (Equa-

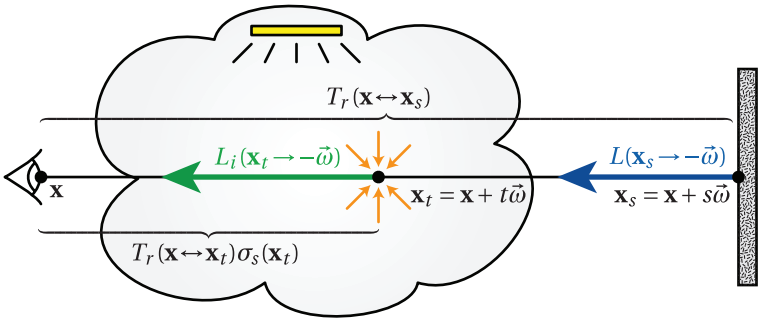


Figure 4.7: The radiance reaching the eye $L(\mathbf{x} \leftarrow \vec{\omega})$ is the sum of the reduced radiance from the nearest visible surface $L(\mathbf{x}_s \rightarrow \vec{\omega})$ and the accumulated in-scattered radiance $L_i(\mathbf{x}_t \rightarrow -\vec{\omega})$ along a ray.

tion 2.17) as a boundary condition, it is possible to obtain a purely integral equation for the radiance in the presence of participating media. The integral form of the RTE expresses the radiance as a sum of three terms:

$$\begin{aligned} L(\mathbf{x} \leftarrow \vec{\omega}) = & \underbrace{T_r(\mathbf{x} \leftrightarrow \mathbf{x}_s)\, L(\mathbf{x}_s \rightarrow -\vec{\omega})}_{\text{reduced surface radiance}} + \\ & \underbrace{\int_0^s T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t)\, \sigma_a(\mathbf{x})\, L_e(\mathbf{x} \rightarrow -\vec{\omega})\, dt}_{\text{accumulated emitted radiance}} + \\ & \underbrace{\int_0^s T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t)\, \sigma_s(\mathbf{x}_t)\, L_i(\mathbf{x}_t \rightarrow -\vec{\omega})\, dt}_{\text{accumulated in-scattered radiance}}, \end{aligned} \tag{4.26}$$

where $L(\mathbf{x} \leftarrow \vec{\omega})$ describes the radiance that reaches $\mathbf{x}$ from direction $\vec{\omega}$, s is the depth of the medium from $\mathbf{x}$ in direction $\vec{\omega}$, $\mathbf{x}_t = \mathbf{x} + t\vec{\omega}$ with $t \in (0, s)$, and $\mathbf{x}_s = \mathbf{x} + s\vec{\omega}$ is a point on a surface past the medium (see Figure 4.7). The transmittance, T_r, models the reduction of radiance as it travels through the medium, as described in Section 4.2.1.

The first term in Equation 4.26 represents surface radiance entering at the backside of the medium, the second term incorporates the emission of radiance along the medium, and the last term is radiance scattered within the medium. The in-scattered radiance is given by Equation 4.13. In the remainder of this dissertation we will ignore the emission term since it is trivial to compute. With this simplification, the relationship between incident and excitant radiance can be expressed

as

$$L(\mathbf{x}\leftarrow\vec{\omega}) = \underbrace{T_r(\mathbf{x}\leftrightarrow\mathbf{x}_s)\,L(\mathbf{x}_s\rightarrow-\vec{\omega})}_{\text{reduced surface radiance}} + \underbrace{\int_0^s T_r(\mathbf{x}\leftrightarrow\mathbf{x}_t)\,\sigma_s(\mathbf{x}_t)\,L_i(\mathbf{x}_t\rightarrow-\vec{\omega})\,dt}_{\text{accumulated in-scattered radiance}}. \tag{4.27}$$

This equation replaces the relation in Equation 2.10 when media scattering and absorption are considered.

In computer graphics, this integral form of the radiative transfer equation is often referred to as the *volume rendering equation* because it is the equation that must be solved in order to render images with participating media. Unfortunately, the volume rendering equation is even more complex than the regular rendering equation. It describes radiance within participating media as a five-dimensional function over 3D positions in the volume and 2D directions over the sphere. The radiance at any point in the medium depends on the radiance of all other points in the medium as well as the radiance of all points on surfaces. In contrast, the regular rendering equation is only a four-dimensional subset of this expression (2D surface locations and 2D directions). This makes the volume rendering equation *extremely* costly to solve.

4.5 Methods for Solving the Volume Rendering Equation

Simulating light transport in participating media has been studied extensively in the literature. Many of the techniques used today were intially developed to study radiative heat transfer and neutron transport in the 1950s and 60s [Kourgnaoff, 1952; Chandrasekhar, 1960]. Mishra1 and Prasad [1998] provide an excellent survey of this early work. It was not until Blinn's pioneering paper in 1982 [Blinn, 1982] that volume effects were considered in computer graphics. Since that time a number of researchers have investigated ways to efficiently simulate volume scattering in the context of computer graphics. We briefly summarize the most significant work here and refer to the excellent survey by Cerezo et al. [2005] for a more comprehensive overview of solving the RTE for computer graphics.

4.5.1 Deterministic Methods

Some of the earliest research for participating media rendering extended discrete radiosity methods to handle isotropic volume scattering. Rushmeier and Torrance [1987] used the *zonal method* [Hottel and Sarofim, 1967] which discretizes the volume rendering equation and solves it over a collection of finite elements or *zones*. This early approach simulated volume-volume, surface-volume, as well as surface-surface interactions but were impractical for all but the simplest scenes due to the high storage costs and $O(n^7)$ computation complexity. As with surface-base radiosity techniques, some of these issues were later addressed by using clustering and hierarchical computation [Bhate, 1993; Sillion, 1995].

Deterministic methods were later augmented to handle anisotropic scattering. P_N methods use a series approximation to replace the integro-differential RTE with a set of partial differential equations, which are expanded using N spherical harmonics terms. This approach was introduce to graphics in 1984 by Kajiya and Herzen but had been used in radiative heat transfer and neutron transport fields much earlier [Chandrasekhar, 1960; Kourgnaoff, 1952] and was originally introduced as early as 1917 by Jeans. Bhate and Tokuta [1992] extended the zonal method with P_N approximations by using spherical harmonics instead of constant basis functions to simulate anisotropic scattering. The approach was even more computationally expensive and required the computation of $O(n^7 + N^2 n^6)$ form factors. An alternate approach is the *discrete ordinates* method, which was first proposed by Chandrasekhar [1960] for simulating stellar and atmospheric radiation. This method subdivides the radiosity leaving each zone into a collection of M discrete directional bins and reduces the complexity to $O(n^7 + Mn^6)$. Unfortunately, discrete ordinates suffer from the "ray effect" [Lathrop, 1968] since light is propagated in discrete directions instead of over the whole solid angle of the bins. Max [1994] reduced the ray effect by approximating the spread through the whole bin and was able to improve efficiency to $O(Mn^3 \log n + M^2 n^3)$ per iteration.

One major advantage of radiosity techniques is that the solution is computed in world-space. This allows the algorithms to exploit spatial coherence by computing the solution sparsely. Unfortunately, volumetric finite element methods share the drawbacks of surface-based radiosity

techniques. Scenes with complex geometry and volume densities cannot easily be handled since the computation of light transport it strictly coupled to the geometric representation of the scene.

4.5.2 Stochastic Methods

Some of the most popular techniques to date are based on stochastic sampling and Monte Carlo integration. Rushmeier [1988] extended the zonal method by using a Monte Carlo pass to account for one bounce of anisotropic scattering. Extensions of path tracing to simulate volumetric scattering in participating media have been suggested by Hanrahan and Krueger [1993] and Pattanaik and Mudur [1993]. Extensions to bidirectional path tracing and Metropolis light transport have also been proposed [Lafortune and Willems, 1996; Pauly et al., 2000]. Monte Carlo approaches are attractive because of their sound underlying theoretical framework and their generality. They are unbiased and guaranteed to converge to the exact solution. In addition, it is straightforward to include heterogeneous media, anisotropic phase functions, and scattering from surfaces. The downside of these approaches is that they suffer from noise that can only be overcome with a huge computational effort.

One strategy to solve this issue is to make simplifying assumptions about the participating media. For example, homogeneous media with a high scattering albedo can be modeled accurately using a diffusion approximation [Stam, 1995; Jensen et al., 2001b], which leads to very efficient rendering algorithms. Premoze et al. [2004], under the assumption that the medium is tenuous and strongly forward scattering, use a path integral formulation to derive efficient rendering algorithms. Sun et al. [2005] render single scattering in real time, but without shadowing effects.

In contrast, photon mapping [Jensen and Christensen, 1998] improves the efficiency of path-tracing without making additional assumptions about the properties of the medium being rendered. Similar to Monte Carlo methods, photon mapping handles isotropic, anisotropic, homogeneous, and heterogeneous media of arbitrary albedo. A disadvantage of photon mapping is that it introduces bias to the solution of the radiative transfer equation. In practice, however, this bias is preferable to the noisy solutions of pure Monte Carlo methods. We cover the photon

mapping method in more detail in Chapter 7.

This dissertation focuses on biased Monte Carlo solutions to the radiative transport equation. These techniques exploit the spatial coherence benefits associated with finite element methods while still retaining the generality of Monte Carlo based approaches.

5

Radiance Caching in Participating Media

"A good action is never lost; it is a treasure laid up and guarded for the doer's need."

—Edwin Markham, 1852–1940

In this chapter we develop a novel method for efficiently rendering participating media using Monte Carlo ray tracing. Our approach is inspired by the irradiance caching techniques described in Chapter 3 but applied to light transport within participating media as described in the previous chapter. Our method handles all types of light scattering, including anisotropic scattering, and it works in both homogeneous and heterogeneous media.

5.1 Contributions

A key contribution of this chapter is a technique for computing gradients of radiance in participating media. Computing illumination gradients is of interest for many applications and has been investigated by many researchers [Ward and Heckbert, 1992; Arvo, 1994; Holzschuch and Sillion, 1995, 1998; Igehy, 1999; Annen et al., 2004; Křivánek et al., 2005b; Durand et al., 2005; Ramamoorthi et al., 2007]. However, no previous work has performed a first-order analysis of light within participating media. We derive analytic expressions for the gradients of the radiative transport equation. These gradients take the full path of the scattered light into account, including the changing properties of the medium in the case of heterogeneous media. Moreover, the gradients can be estimated using Monte Carlo ray tracing simultaneously with the in-scattered radiance with negligible overhead.

Our second contribution is a new radiance caching scheme for participating media. We draw inspiration for our approach from irradiance caching methods for surfaces; however, we compute, cache, and reuse estimates of the radiance *within* participating media. This caching scheme uses the information in the radiance gradients to sparsely sample as well as interpolate radiance within the medium, utilizing a novel, perceptually-based error metric. We compute gradients for single scattering from lights and surfaces and for multiple scattering, and we use a spherical harmonics representation in anisotropic media. We further exploit the information in the gradients to improve the accuracy of interpolation. Volumetric radiance caching provides several orders of magnitude speedup compared to path tracing and produces higher quality results than volumetric photon mapping. Furthermore, it is view-driven and well suited for large scenes where methods such as photon mapping become costly.

5.2 Overview

To compute the in-scattered radiance within participating media we use Monte Carlo ray tracing based on a combination of ray marching and random-walk sampling. To simplify notation we ignore the radiance emitted by the volume as it is trivial to compute. We use ray marching to numerically integrate the radiance value seen directly by the observer. Ray marching solves the volume rendering equation by discretizing the 1D integral along a ray using small steps through the medium. Equation 4.27 is approximated as,

$$L(\mathbf{x}\leftarrow\vec{\omega}) \approx T_r(\mathbf{x}\leftrightarrow\mathbf{x}_s)L(\mathbf{x}_s\rightarrow-\vec{\omega}) + \left(\sum_{t=0}^{S-1} T_r(\mathbf{x}\leftrightarrow\mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_i(\mathbf{x}_t\rightarrow-\vec{\omega})\Delta_t\right), \tag{5.1}$$

where Δ_t is the length of each segment along the ray and $\mathbf{x}_0,\ldots,\mathbf{x}_s$ are the sample points for each segment ($\mathbf{x}_0$ is the point where the ray enters the medium and $\mathbf{x}_s$ is a point on a surface past the medium). This approach is illustrated in Figure 5.1.

The ray marching processes samples the in-scattered radiance at discrete points along the ray. The transmittance at each step can be computed incrementally by exploiting the multi-

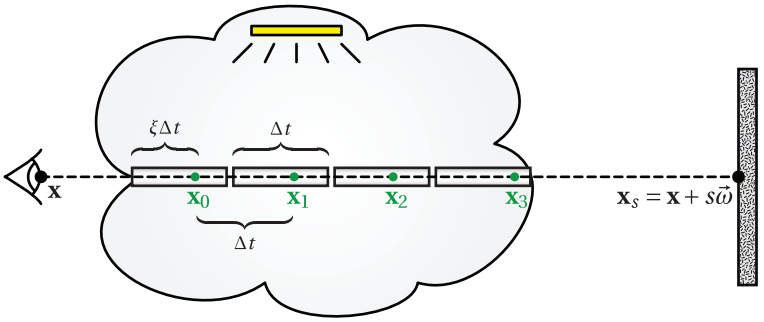


Figure 5.1: Ray marching computes lighting within participating media by dividing the ray into small discrete segments. The lighting and properties of the medium are computed at the sample locations $\mathbf{x}_i$ and assumed constant within each segment. To avoid aliasing, the collection of all samples is offset along the ray by a random amount ξ.

plicative property from Equation 4.10:

$$T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t) = T_r(\mathbf{x} \leftrightarrow \mathbf{x}_{t-1}) T_r(\mathbf{x}_{t-1} \leftrightarrow \mathbf{x}_t). \tag{5.2}$$

To avoid structured aliasing artifacts, the sample points are randomly chosen within the extent of the segments. This can be accomplished using an independent random location within each segment or by randomly offsetting the collection of samples as a whole [Pauly et al., 2000].

The most expensive part to compute in Equation 5.1 is the in-scattered radiance L_i, because it involves integrating over all paths that carry radiance from any other point in the scene to the current sample point $\mathbf{x}_t$ along the eye ray. Our approach therefore focuses on the efficient computation of this term.

It is possible to recursively solve for in-scattered radiance using volumetric path tracing. This approach is, however, extremely costly since it typically requires tracing thousands of paths to obtain noise-free results. Fortunately, the distribution of in-scattered radiance is often smooth in large parts of the medium. We exploit this property by computing in-scattered radiance accurately only at a sparse set of locations and using interpolation whenever possible.

To make the computation more practical, we divide the in-scattered radiance into two components,

$$L_i(\mathbf{x} \rightarrow \vec{\omega}) = L_s(\mathbf{x} \rightarrow \vec{\omega}) + L_m(\mathbf{x} \rightarrow \vec{\omega}), \tag{5.3}$$

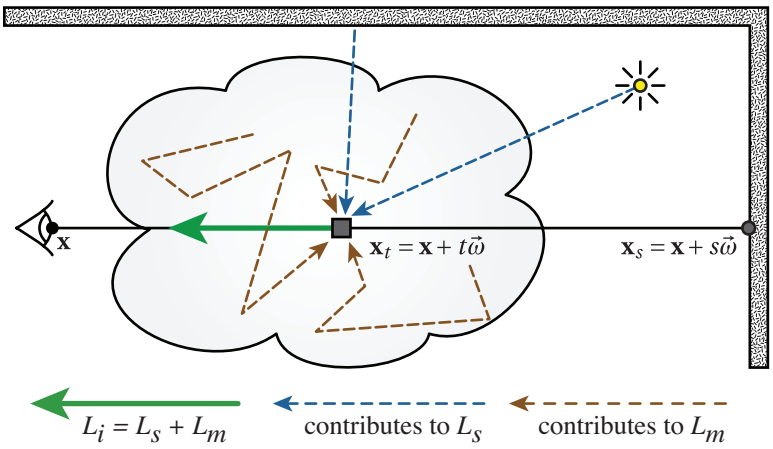


Figure 5.2: Radiance in participating media is computed by our method using a combination of ray marching and random-walk sampling. We split the computation of in-scattered radiance into single scattering, L_s, and multiple scattering, L_m, terms.

where L_s and L_m represent single and multiple scattering, respectively (see Figure 5.2).

The *single scattering term* represents radiance that undergoes a single scattering event along its path from a surface to the eye. Surfaces include light sources and any other surface that reflects light due to direct or indirect illumination. Single scattering may be formulated as an integration over the sphere, or over all surfaces.

The *multiple scattering term* incorporates radiance that scatters at least once in the medium before it reaches **x**. This implies that the path from the surface to the eye includes multiple scattering events. Incident radiance due to multiple scattering is more difficult to evaluate than single scattering, as it leads to a recursive integral.

The distribution of in-scattered radiance is often smooth in large parts of the participating medium. We take advantage of this property by *caching* radiance at a sparse set of locations, and using extrapolation to evaluate the radiance at nearby points in the medium. We show that results without visible errors are achievable with relatively few cached values, resulting in large speedups compared to Monte Carlo ray tracing without caching. Furthermore, the use of caching does not limit the type of media that can be rendered, and our method can easily be incorporated into existing Monte Carlo ray tracers.

We improve the efficiency of the cache by not only storing the value of scattered radiance at each cache point but also its gradient with respect to translation of the point. The gradient is

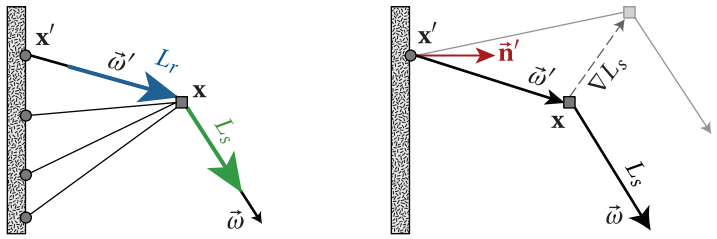


Figure 5.3: Computing the single scattering radiance (left) and gradient (right). For single scattering we distribute samples $\mathbf{x}'$ on the area of the light source. The gradient ∇L_s is computed with respect to translating the evaluation point $\mathbf{x}$ while keeping $\mathbf{x}'$ fixed.

used to estimate the local smoothness and to improve the accuracy of extrapolation of cached values. In the following sections, we explain two separate methods: one to compute the single scattering contribution and its gradient (Section 5.3), and another to compute the multiple scattering contribution and its gradient (Section 5.4). Finally, we detail how to use these quantities in a practical radiance caching algorithm for participating media in Section 5.5.

5.3 Single Scattering

In this section, we describe how to compute the single scattering radiance L_s at a position $\mathbf{x}$ and its gradient ∇L_s with respect to $\mathbf{x}$. The derivation of the gradient is more convenient if we write L_s as an integral over the surfaces of the scene, instead of the standard integral over the sphere:

$$L_s(\mathbf{x}\rightarrow\vec{\omega}) = \int_A p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, L_r(\mathbf{x}\leftarrow\mathbf{x}')\, V(\mathbf{x}\leftrightarrow\mathbf{x}')\, H(\mathbf{x}\leftarrow\mathbf{x}')\, d\mathbf{x}', \tag{5.4}$$

where A denotes surface area and $\vec{\omega}'$ points from $\mathbf{x}$ towards $\mathbf{x}'$. We illustrate our notation in Figure 5.3. The term $L_r(\mathbf{x}\leftarrow\mathbf{x}')$ represents the *reduced radiance*. It is given by the product of the radiance leaving a surface at $\mathbf{x}'$ towards $\mathbf{x}$ and the transmittance through the medium:

$$L_r(\mathbf{x}\leftarrow\mathbf{x}') = T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\, L(\mathbf{x}'\rightarrow\mathbf{x}). \tag{5.5}$$

We also include a *visibility function* $V(\mathbf{x} \leftrightarrow \mathbf{x}')$, whose value is unity if $\mathbf{x}'$ is visible from $\mathbf{x}$; otherwise it is zero. Finally, H is a *geometry term*,

$$H(\mathbf{x} \leftarrow \mathbf{x}') = \frac{\vec{\mathbf{n}}' \cdot \frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}}{||\mathbf{x}-\mathbf{x}'||^2}, \tag{5.6}$$

due to the change of the integration variable, where $\vec{\mathbf{n}}'$ is the surface normal at $\mathbf{x}'$. Note that negative values of the dot product must be clamped to zero, but we omit this in our notation.

The single scattering gradient follows directly from Equation 5.4,

$$\nabla L_s(\mathbf{x} \rightarrow \vec{\omega}) = \int_A \nabla\big(p\, L_r\, V\, H\big)\, d\mathbf{x}' \tag{5.7}$$

$$= \int_A (\nabla p)\, L_r\, V\, H + p\, (\nabla L_r)\, V\, H + p\, L_r\, (\nabla V)\, H + p\, L_r\, V\, (\nabla H)\, d\mathbf{x}', \tag{5.8}$$

where we omit the function arguments for brevity. The symbol ∇ always denotes a gradient with respect to $\mathbf{x}$. Intuitively, the gradient corresponds to the direction of maximum change of the scattered radiance at $\mathbf{x}$ under an infinitesimal motion of $\mathbf{x}$ while keeping $\mathbf{x}'$ fixed. This is illustrated in Figure 5.3.

In the remainder of this section we examine the single scattering computation in more detail. In Section 5.3.1 we discuss how to estimate Equation 5.4 and Equation 5.7 using Monte Carlo integration and describe how to incorporate point light sources in Section 5.3.2. Evaluating the radiance gradient relies on computing the gradient of each individual term in Equation 5.7. We derive the gradient of the geometry term ∇H in Section 5.3.3 and present gradients for common phase functions in Section 5.3.4. Finally, we discuss how to compute the gradient of the reduced radiance term in detail in Section 5.3.5.

5.3.1 Monte Carlo Integration

We compute single scattering radiance values (Equation 5.4) using Monte Carlo ray tracing, which leads to the following formula:

$$L_s(\mathbf{x} \rightarrow \vec{\omega}) \approx \frac{1}{N} \sum_{j=1}^{N} \frac{p\, L_r\, V\, H}{pdf(\mathbf{x}'_j)}. \tag{5.9}$$

Similarly, we use Monte Carlo estimation to evaluate the gradient of single scattering radiance from Equation 5.7:

$$\nabla L_s(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{j=1}^{N}\frac{(\nabla p)\,L_r\,V\,H + p\,(\nabla L_r)\,V\,H + p\,L_r\,V\,(\nabla H)}{pdf(\mathbf{x}'_j)}. \tag{5.10}$$

The radiance and the gradient share many of the same terms and are evaluated simultaneously.

Because the visibility function V is discontinuous, its "gradient" is not taken into account. For the purposes of gradient computation, we currently assume constant visibility and discuss the implications of this decision in Section 5.7.

Monte Carlo approximation relies on a set of samples $\mathbf{x}'_j$ that are distributed on the surfaces of the scene according to a probability density function $pdf(\mathbf{x}')$. Surfaces include both area light sources that emit light and any other surface in the scene that may reflect light.

To gather radiance emitted by area light sources, we place samples on their area. Our method allows the use of arbitrary light source sampling techniques to optimize the sample distribution. To account for radiance emitted by all other surfaces, we distribute samples uniformly over the sphere of directions. Using the standard solid angle to area conversion, uniform sampling on the sphere corresponds to a distribution

$$pdf(\mathbf{x}') = \frac{\vec{\mathbf{n}}'\cdot\frac{\mathbf{x}_0-\mathbf{x}'}{||\mathbf{x}_0-\mathbf{x}'||}}{||\mathbf{x}_0-\mathbf{x}'||^2}, \tag{5.11}$$

where $\vec{\mathbf{n}}'$ is the surface normal at $\mathbf{x}'$, and $\mathbf{x}_0$ is the location at which to compute the spherical distribution. Conceptually, $\mathbf{x}_0$ is independent of the location $\mathbf{x}$ where we evaluate in-scattering. Although in practice $\mathbf{x}_0$ and $\mathbf{x}$ usually coincide, $pdf(\mathbf{x}')$ does not depend on $\mathbf{x}$. Hence, it does not contribute to the gradient.

5.3.2 Point Lights

Even though Equation 5.4 is formulated with respect to surface area, it is straightforward to incorporate point lights. We account for the contribution of point lights to the Monte Carlo

estimate of L_s by adding the following summand to Equation 5.9:

$$L_s(\mathbf{x}\rightarrow\vec{\omega}) \mathrel{+}= p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\,\Phi_p\,T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\,V(\mathbf{x}\leftrightarrow\mathbf{x}')\,H_p(\mathbf{x}'\rightarrow\mathbf{x}), \tag{5.12}$$

where Φ_p is the power of the light source and H_p describes the light's fall-off characteristics. For a uniform point light, we have

$$H_p^u(\mathbf{x}\leftarrow\mathbf{x}') = \frac{1}{||\mathbf{x}-\mathbf{x}'||^2}. \tag{5.13}$$

For a Phong light, the fall-off term is

$$H_p^p(\mathbf{x}\leftarrow\mathbf{x}') = \frac{\left(\vec{\mathbf{n}}'\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)^s}{||\mathbf{x}-\mathbf{x}'||^2}, \tag{5.14}$$

where s is the Phong exponent.

The contribution of a point light to the single scattering gradient is:

$$\nabla L_s(\mathbf{x}\rightarrow\vec{\omega}) \mathrel{+}= \Phi_p\left((\nabla p)\,T_r\,H_p + p\,(\nabla T_r)\,H_p + p\,T_r\,(\nabla H_p)\right). \tag{5.15}$$

5.3.3 Gradient of Geometry Terms

Computing the radiance gradient in Equations 5.10 and 5.15 relies on the gradients of several geometry terms. By differentiating the definitions in Equations 5.6, 5.13, and 5.14 we arrive at the following gradient terms:

$$\nabla H(\mathbf{x}\leftarrow\mathbf{x}') = \nabla\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)\frac{1}{||\mathbf{x}-\mathbf{x}'||^2} + \vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right), \tag{5.16}$$

$$\nabla H_p^u(\mathbf{x}\leftarrow\mathbf{x}') = \nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right), \tag{5.17}$$

$$\nabla H_p^p(\mathbf{x}\leftarrow\mathbf{x}') = s\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)^{s-1}\nabla\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)\frac{1}{||\mathbf{x}-\mathbf{x}'||^2} + \left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)^s\nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right), \tag{5.18}$$

where

$$\nabla\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)=\frac{\vec{\mathbf{n}}}{||\mathbf{x}-\mathbf{x}'||}-\frac{(\vec{\mathbf{n}}\cdot(\mathbf{x}-\mathbf{x}'))(\mathbf{x}-\mathbf{x}')}{||\mathbf{x}-\mathbf{x}'||^3}, \tag{5.19}$$

$$\nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right)=\frac{-2(\mathbf{x}-\mathbf{x}')}{||\mathbf{x}-\mathbf{x}'||^4}. \tag{5.20}$$

5.3.4 Gradient of Phase Function

In isotropic media, the phase function is a constant, and its gradient vanishes. For anisotropic media, we focus on the Henyey-Greenstein phase function p_{HG} and the Schlick phase function p_S defined in Chapter 4.

These phase functions are 1D, only depending on the cosine of the angle between the incident and outgoing directions. We express $\cos(\theta)$ in terms of $\vec{\omega}$, $\mathbf{x}$, and $\mathbf{x}'$ as:

$$\cos(\theta)=\vec{\omega}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}. \tag{5.21}$$

The gradients of the phase functions with respect to $\mathbf{x}$ are then

$$\nabla p_{\mathrm{HG}}(\mathbf{x},\theta)=\frac{3g(1-g^2)}{4\pi(1+g^2-2g\cos\theta)^{2.5}}\nabla\cos\theta \tag{5.22}$$

$$\nabla p_{\mathrm{S}}(\mathbf{x},\theta)=\frac{k(1-k^2)}{2\pi(1+k\cos\theta)^3}\nabla\cos\theta, \tag{5.23}$$

where

$$\nabla\cos(\theta)=\cos\theta\frac{\mathbf{x}-\mathbf{x}'}{\|\mathbf{x}-\mathbf{x}'\|^2}-\frac{\vec{\omega}}{\|\mathbf{x}-\mathbf{x}'\|}. \tag{5.24}$$

5.3.5 Reduced Radiance and Transmittance Gradient

For the sake of computing the gradient of the reduced radiance, we assume diffuse surfaces and light sources. In this case, the gradient of the surface radiance $L(\mathbf{x}' \rightarrow \mathbf{x})$ with respect to $\mathbf{x}$ vanishes. However, it is straightforward to account for general radiance distributions by adding a term for ∇L. With this simplification, the gradient of the reduced radiance is

$$\nabla L_r(\mathbf{x}' \rightarrow \mathbf{x}) = L(\mathbf{x}' \rightarrow \mathbf{x}) \nabla T_r(\mathbf{x}' \leftrightarrow \mathbf{x}),$$

and the gradient of the transmittance is

$$\nabla T_r(\mathbf{x}' \leftrightarrow \mathbf{x}) = -\nabla \tau(\mathbf{x}' \leftrightarrow \mathbf{x}) T_r(\mathbf{x}' \leftrightarrow \mathbf{x}). \tag{5.25}$$

Homogeneous Media. If the medium is homogeneous with constant extinction coefficient σ_t, then the optical thickness is

$$\tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \sigma_t ||\mathbf{x} - \mathbf{x}'||, \tag{5.26}$$

and its gradient is

$$\nabla \tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \sigma_t \nabla (||\mathbf{x} - \mathbf{x}'||). \tag{5.27}$$

Heterogeneous Media. In a heterogeneous medium, to compute optical thickness the extinction needs to be integrated as

$$\tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \int_0^1 \sigma_t(\mathbf{y}(t)) ||\mathbf{x}' - \mathbf{x}|| \, dt, \tag{5.28}$$

where we have parametrized the line segment between $\mathbf{x}'$ and $\mathbf{x}$ as $\mathbf{y}(t) = \mathbf{x} + t(\mathbf{x}' - \mathbf{x})$.

The corresponding gradient is

$$\nabla \tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \int_0^1 \nabla \left(\sigma_t(\mathbf{y}(t)) ||\mathbf{x}' - \mathbf{x}|| \right) dt. \tag{5.29}$$

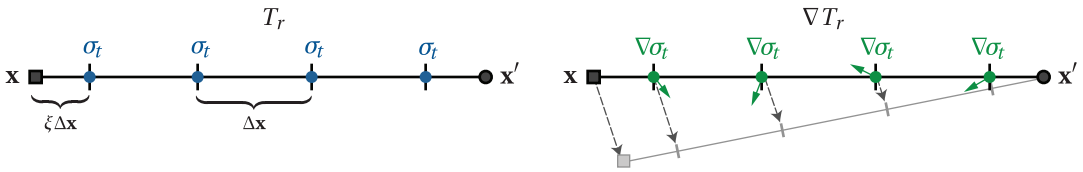


Figure 5.4: In heterogeneous media, the transmittance (left) between two points $\mathbf{x}$ and $\mathbf{x}'$ is evaluated using ray marching with a step size $\Delta\mathbf{x}$ and a random offset ξ. Within each step the extinction coefficient of the medium is assumed to be constant. Our gradient computation not only takes into account the local properties of the medium at $\mathbf{x}$, but also the changing properties along the whole segment between $\mathbf{x}$ and $\mathbf{x}'$. For single scattering, we compute the gradient with respect to translation of $\mathbf{x}$ as $\mathbf{x}'$ is kept fixed.

The optical thickness and its gradient can be expressed using Monte Carlo integration as

$$\tau(\mathbf{x}'\leftrightarrow\mathbf{x}) \approx \frac{1}{N}\sum_j \frac{\sigma_t(\mathbf{y}(t_j))||\mathbf{x}'-\mathbf{x}||}{pdf(t_j)}, \tag{5.30}$$

$$\nabla\tau(\mathbf{x}'\leftrightarrow\mathbf{x}) \approx \frac{1}{N}\sum_j \frac{\nabla\left(\sigma_t(\mathbf{y}(t_j))||\mathbf{x}'-\mathbf{x}||\right)}{pdf(t_j)}. \tag{5.31}$$

We evaluate the two Monte Carlo integrals simultaneously using ray marching with a fixed step size and a single random offset ξ (see Figure 5.4).

At a high level, the evaluation of transmittance aggregates σ_t along a line segment, while its gradient aggregates the contribution of $\nabla\sigma_t$ along the line segment. An important characteristic of this gradient formulation is that it not only takes into account the local properties at $\mathbf{x}$, it incorporates how the properties change along the whole segment between $\mathbf{x}$ and $\mathbf{x}'$. This implies that our gradient computation contains meaningful information about how the transmittance changes even when moving $\mathbf{x}$ out of the line connecting $\mathbf{x}$ and $\mathbf{x}'$.

5.3.6 Isotropic and Anisotropic Scattering

Isotropic Media. In isotropic media, the phase function is a constant $p = 1/4\pi$. Therefore, the scattered radiance and its gradient are independent of the outgoing direction $\vec{\omega}$, and all terms that include the gradient of the phase function ∇p in Equations 5.10 and 5.15 vanish.

Anisotropic Media. For anisotropic media, the scattered radiance is a function of the outgoing direction $\vec{\omega}$, depending on the shape of the phase function. As phase functions are smooth, we can compute a compact representation of the directional radiance distribution. We use a spherical

harmonic expansion, although other spherical functions could be used. For each sample $\mathbf{x}'_j$ with a fixed incident direction $\vec{\omega}'_j$, we compute the spherical harmonic expansion of the phase function as the dot product

$$p(\mathbf{x}, \vec{\omega}'_j \leftrightarrow \vec{\omega}) \approx \sum_{k=1}^{M} y_k(\vec{\omega}) p_{j,k}, \tag{5.32}$$

where y_k are the spherical harmonic functions, and $p_{j,k}$ are the corresponding coefficients. We use a single index for the spherical harmonics to simplify notation.

Applying this formulation to Equation 5.9 gives the following expression for scattered radiance in anisotropic media:

$$L_s(\mathbf{x} \rightarrow \vec{\omega}) \approx \frac{1}{N} \sum_{k=1}^{M} y_k(\vec{\omega}) \sum_{j=1}^{N} \frac{p_{j,k} L_r V H}{pdf(\mathbf{x}'_j)}. \tag{5.33}$$

In order to efficiently evaluate the lighting for arbitrary outgoing directions, we project the entire in-scattered radiance in the above expression onto spherical harmonics. In anisotropic media, we represent the single scattering radiance L_s using a vector of spherical harmonic coefficients, $\Lambda_s = \{\lambda_{s,k}\}$, computed as:

$$\lambda_{s,k} = \frac{1}{N} \sum_{j=1}^{N} \frac{p_{j,k} L_r V H}{pdf(\mathbf{x}'_j)}, \tag{5.34}$$

which allows us to evaluate Equation 5.33 for any direction $\vec{\omega}$ using a simple dot product

$$L_s(\mathbf{x} \rightarrow \vec{\omega}) \approx \sum_{k=1}^{M} y_k(\vec{\omega}) \lambda_{s,k}. \tag{5.35}$$

We apply the same projection mechanism to the gradient computation. We denote the spherical harmonic expansion of the gradient of the phase function as a function over outgoing angle $\vec{\omega}$, by

$$\nabla p(\mathbf{x}, \vec{\omega}'_j \leftrightarrow \vec{\omega}) \approx \sum_{k=1}^{M} p'_{j,k} y_k(\vec{\omega}). \tag{5.36}$$

This leads to the following expression for the gradient:

$$\nabla L_s(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{k=1}^{M} y_k(\vec{\omega}) \sum_{j=1}^{N} \frac{p'_{j,k}\, L_r\, V\, H + p_{j,k}\,(\nabla L_r)\, V\, H + p_{j,k}\, L_r\, V\,(\nabla H)}{pdf(\mathbf{x}'_j)}. \tag{5.37}$$

As with radiance, the entire gradient computation is projected onto spherical harmonics, resulting in a vector of gradient coefficients, $\Lambda'_s = \{\lambda'_{s,k}\}$:

$$\lambda'_{s,k} = \frac{1}{N}\sum_{j=1}^{N} \frac{p'_{j,k}\, L_r\, V\, H + p_{j,k}\,(\nabla L_r)\, V\, H + p_{j,k}\, L_r\, V\,(\nabla H)}{pdf(\mathbf{x}'_j)}. \tag{5.38}$$

The radiance gradient can be evaluated in any direction $\vec{\omega}$ by taking the dot product of the coefficient vector Λ'_s with the basis functions, as in Equation 5.35.

5.4 Multiple Scattering

In this section we describe how we calculate the multiple scattering radiance L_m and its gradient ∇L_m. We evaluate multiple scattering using standard Monte Carlo path tracing and compute a Monte Carlo estimate of the gradient of the path tracing integral.

The radiance due to multiple scattering is an integral of radiance arriving from all other points within the participating medium. We express this as

$$L_m(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega_{4\pi}}\int_0^{\infty} p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, T_r(\mathbf{x}'\leftrightarrow\mathbf{x})\, \sigma_s(\mathbf{x}')\, L_i(\mathbf{x}'\rightarrow\vec{\omega}')\, V(\mathbf{x}'\leftrightarrow\mathbf{x})\, dr' d\vec{\omega}', \tag{5.39}$$

where we parametrize the integration domain using polar coordinates $(r',\vec{\omega}')$ centered at $\mathbf{x}$ so $\mathbf{x}' = \mathbf{x} + r'\vec{\omega}'$. The $L_i(\mathbf{x}'\rightarrow\vec{\omega}')$ term is the in-scattered radiance at $\mathbf{x}'$ towards $\mathbf{x}$ and needs to be evaluated recursively. See Figure 5.5 for an illustration.

The gradient of multiple scattering is computed by differentiating Equation 5.39 with respect to $\mathbf{x}$:

$$\nabla L_m(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega_{4\pi}}\int_0^{\infty} p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, \nabla\big(T_r(\mathbf{x}'\leftrightarrow\mathbf{x})\, \sigma_s(\mathbf{x}')\, L_i(\mathbf{x}'\rightarrow\vec{\omega}')\, V(\mathbf{x}'\leftrightarrow\mathbf{x})\big)\, dr' d\vec{\omega}'.$$

Note that with the chosen parametrization the incoming and outgoing directions do not change

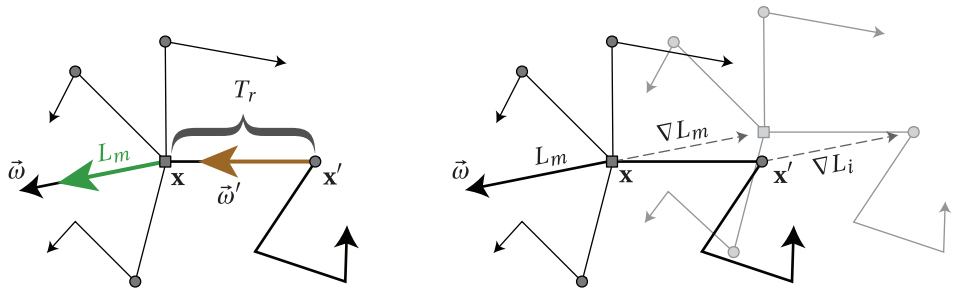


Figure 5.5: Computing the multiple scattering radiance (left) and gradient (right). For multiple scattering we distribute samples $\mathbf{x}'$ by generating random-walk paths starting at $\mathbf{x}$. The gradient ∇L_m is computed by considering the translation of $\mathbf{x}$ and the paths *as a whole*.

under translation of $\mathbf{x}$, so there is no gradient term for the phase function. In homogeneous media, the gradient of the transmittance vanishes. For the general case of heterogeneous media, the gradients of transmittance and optical thickness are computed as in Equation 5.25 and 5.29. However, in the multiple scattering case, $\|\mathbf{x}-\mathbf{x}'\|$ does not change under a translation, and so the $\nabla(\|\mathbf{x}-\mathbf{x}'\|)$ term vanishes. This is illustrated in Figure 5.6.

5.4.1 Monte Carlo Integration

Just as with single scattering, we evaluate the radiance due to multiple scattering and its gradient using Monte Carlo integration. The radiance is estimated by

$$L_m(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{j=1}^{N}\frac{p\,T_r\,\sigma_s\,L_i\,V}{pdf(r'_j)\,pdf(\vec{\omega}'_j)}. \tag{5.40}$$

In practice, at the first scattering event we evaluate this expression using N samples. In order to prevent exponential growth, all subsequent scattering events use only one random sample. Hence, Equation 5.40 forms the start of N random-walk paths through the medium, with the polar coordinates $(r'_j,\vec{\omega}'_j)$ as random variables. We sample the radius according to

$$pdf(r'_j) = \sigma_t(\mathbf{x})e^{-\sigma_t(\mathbf{x})r'_j}. \tag{5.41}$$

We distinguish between isotropic and anisotropic media when sampling directions, discussed in more detail below.

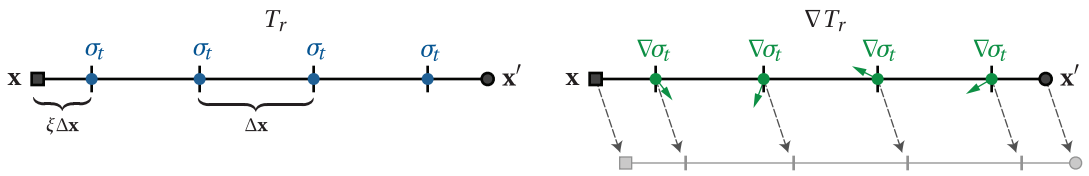


Figure 5.6: For multiple scattering we compute the transmittance (left) using ray marching just like in the single scattering case. The gradient, however, is computed as both endpoints are translated together (right).

The gradient $\nabla L_m(\mathbf{x}\rightarrow\vec{\omega})$ is also found using Monte Carlo sampling:

$$\nabla L_m(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{j=1}^{N} \frac{p\left((\nabla T_r)\sigma_s L_i V + T_r(\nabla\sigma_s)L_i V + T_r\sigma_s(\nabla L_i)V\right)}{pdf(r'_j)\,pdf(\vec{\omega}'_j)}. \tag{5.42}$$

Intuitively, the gradient captures the change in radiance as each random-walk path is translated, as a whole, by an infinitesimal amount. This is illustrated in Figure 5.5. As with single scattering, we currently ignore the change in the visibility term.

5.4.2 Isotropic and Anisotropic Scattering

Isotropic Media. For isotropic media, we distribute the directions $\vec{\omega}'_j$ uniformly, i.e., $pdf(\vec{\omega}'_j) = 1/4\pi$.

Anisotropic Media. For anisotropic media, the multiple scattered radiance depends on the outgoing direction $\vec{\omega}$. As with the single scattering contribution, we use spherical harmonics to represent the phase function $p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})$, which leads to expressions analogous to Equations 5.33 and 5.37. However, there is no term for the gradient of the phase function in multiple scattering, as the angle between path vertices does not change under a translation of the whole path. We project the multiple scattering radiance function and its gradient as vectors of spherical harmonic coefficients $\Lambda_m = \{\lambda_{m,k}\}$ and $\Lambda'_m = \{\lambda'_{m,k}\}$.

We apply importance sampling according to the phase function at all but the first scattering event, since the outgoing direction is given by the direction to the previous path vertex. However, we always sample the direction at the first scattering event uniformly. The reason is that the outgoing direction $\vec{\omega}$ is not known a priori. It will be determined only when the cache point is

evaluated for a certain view direction.

5.5 Algorithm

Using the groundwork laid out in the previous sections, we now describe a radiance caching method for participating media. Our algorithm efficiently computes lighting within participating media by sparsely sampling L_s and L_m and caching them for extrapolation. In addition to computing single and multiple scattering separately as described in Section 5.2, we separate our cache based on the expected frequency content of the radiance values. Specifically, we maintain distinct cache points for single scattering from lights, single scattering from surfaces, and multiple scattering. We call these *single scattering*, *surface scattering*, and *multiple scattering* cache points. This allows our system to sample low-frequency components more sparsely.

The core algorithm for radiance caching in participating media is as follows. For each point in the volume where we need to evaluate in-scattered radiance, we first query the cache for a set of usable cache points. If we find any points, then we use the cached values to extrapolate the radiance to this point, as described in Section 5.5.3. If there are no usable points, then we compute the radiance and its gradient, as described in Sections 5.3 and 5.4, and store these values in the cache. The procedure is performed for all three caches. The total in-scattered radiance is the sum of the three contributions.

5.5.1 Cache Entry Storage

For efficient access to cache points, we use an octree. The data structure and lookup procedure is similar to previous work on surface radiance caching techniques [Ward et al., 1988].

All cache entries store the in-scattered radiance and its gradients at a location in space, as well as a valid radius to later determine if a cache entry is valid for extrapolation at a given query location. Note that the in-scattered radiance is the amount of light that *leaves* a point in a particular direction; thus, we must store the radiance over the full sphere of directions.

Isotropic Media. Since the in-scattered radiance does not depend on direction, each cache entry stores it using three scalar values, one for each color channel. In addition, we compute and

store the gradient for each channel separately for greater accuracy. The total storage requirement for each isotropic entry is 16 floating point numbers. In total, the size of an isotropic cache entry is 48 bytes, though this could easily be reduced using an RGBE [Ward, 1991] or 16-bit floating point representation [Kainz et al., 2006].

Anisotropic Media. In anisotropic media, each Monte Carlo sample of the radiance has a different directional contribution that depends on the shape of the phase function. Because phase functions are generally smooth, we capture this directional dependence using spherical harmonics by caching the spherical harmonic coefficient vectors Λ_s and Λ_m, as well as their corresponding gradient coefficient vectors Λ'_s and Λ'_m.

Our system supports storing spherical harmonic expansions to an arbitrary number of coefficients. However, for scenes with area lights and moderately anisotropic media (e.g., a Henyey-Greenstein phase function with $|g| < 0.6$), we have found that using $M = 9$ coefficients in Equation 5.32 is sufficient to accurately represent the in-scattered radiance. Using nine coefficients, the full RGB representation of the in-scattered radiance requires 27 coefficients. Including the cache sample location and valid radius, this gives 28 floating point values, or 112 bytes per entry, assuming 32-bit values.

5.5.2 Valid Radius and Error Tolerance

Each cache point has a valid radius within which it is a candidate for extrapolation. To compute this radius we have developed a perceptually-based error metric, which adapts to the local smoothness of the radiance field.

To estimate a valid radius for cache points we define the notion of an "optimal radius." The optimal radius for a cache point is the largest radius such that the total relative L_2 error over the extrapolated region is below some error threshold ε. More formally, if $\hat{L}_{x_0}(x)$ is the radiance extrapolated from location x_0 to x, then the optimal radius is given by

$$R_{opt}(\mathbf{x}) = \max_r \left(\frac{\int_{\mathbf{x}\in\Omega_r} \left| L(\mathbf{x}) - \hat{L}_{\mathbf{x}_0}(\mathbf{x}) \right| d\mathbf{x}}{\int_{\mathbf{x}\in\Omega_r} L(\mathbf{x})\, d\mathbf{x}} \leq \varepsilon \right), \tag{5.43}$$

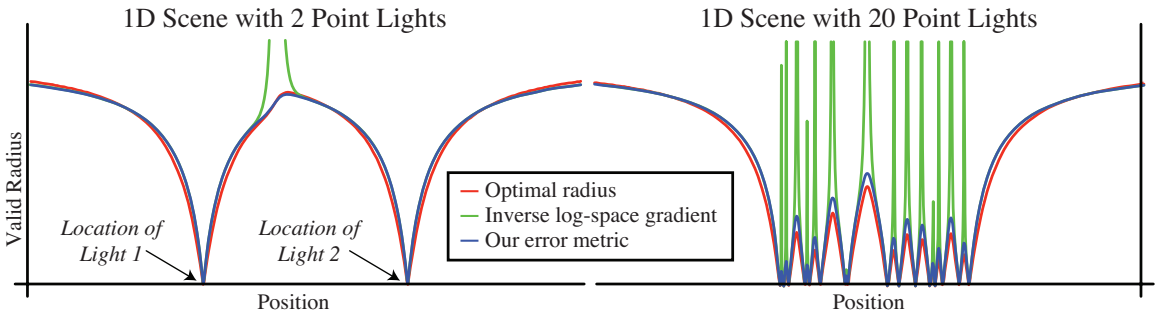


Figure 5.7: Experimental validation of our error metric as compared to a numerically computed optimal radius for a 1D scene with two point lights of differing strengths (left) and 20 equal strength point lights (right). Though the log-space gradient matches the optimal radius in most situations, it produces infinite radii where the gradient vanishes. These singularities become even more prominent with many light sources. In both cases, our perceptually-based error metric robustly approximates the optimal radius even in the presence of such saddle points in the radiance field.

where Ω_r is the spherical domain of radius r around $\mathbf{x}_0$. We use the relative error because the human visual system is more sensitive to contrast than absolute differences in intensity [Glassner, 1995]. Solving for $R_{opt}(\mathbf{x})$ during rendering would be impractical since it requires knowledge of the true radiance within the valid radius of any cache point. Our goal is therefore to devise an error metric which approximates the optimal radius as closely as possible using only information already available from the Monte Carlo samples at $\mathbf{x}$.

The ratio of the radiance gradient to the radiance is a local measure of the contrast. It is also equivalent to the log-space gradient of the radiance:

$$\nabla \ln(L) = \frac{\nabla L}{L}. \tag{5.44}$$

Making the valid radius inversely proportional to the magnitude of the log-space gradient at $\mathbf{x}$ makes sense at an intuitive level: areas where the radiance field has high contrast will be sampled more densely than smoothly varying regions. Furthermore, using the log-space gradient instead of the regular gradient means that the valid radius computation is independent of the absolute intensity of the lighting in the scene.

In our 1D test cases this approach matches the behavior of the optimal radius well at many locations (see Figure 5.7). Unfortunately, the log-space gradient is a completely local quantity and can be a bad predictor of the contrast in the surrounding area. Specifically, the magnitude of the

gradient can drop to zero, causing an infinite valid radius. A zero gradient, or saddle-point, can naturally occur in many situations, e.g., between two lights.

Using the Monte Carlo samples, the reciprocal magnitude of the log-space gradient can be expressed as:

$$\frac{L(\mathbf{x})}{\|\nabla L(\mathbf{x})\|} = \frac{\sum_j L_j(\mathbf{x})}{\left\|\sum_j \nabla L_j(\mathbf{x})\right\|}, \tag{5.45}$$

where j represents the Monte Carlo samples over light sources (for single scattering), surfaces (for surface scattering), and random-walk paths (for multiple scattering). However, instead of using the log-space gradient directly, we use a modified metric designed to eliminate saddle-points. The valid radius of cache points is computed as:

$$R(\mathbf{x}) = \varepsilon \frac{\sum_j L_j(\mathbf{x})}{\sum_j \left\|\nabla L_j(\mathbf{x})\right\|}, \tag{5.46}$$

where ε is a user controlled global error tolerance parameter. By summing over gradient magnitudes instead of gradient directions in the denominator, we prevent the introduction of infinite radii. In our 1D test cases this error metric matches the optimal radius remarkably well (see Figure 5.7) and in practice produces very good cache point distributions.

For colored media, we compute the valid radius for the three color channels independently and use the minimum. In anisotropic media, we use the zero-order spherical harmonic coefficients to compute the radius.

In our implementation, the amount of error in the extrapolation is controlled by the tolerance ε. A tolerance of 0 indicates that values should never be cached or reused. Setting $\varepsilon = 1$ indicates that the extrapolated radiance will be usable until it differs by about a factor of two from the cached radiance value.

In areas that are occluded from all light sources, the radiance due to direct illumination and its gradient are zero. This means that the valid radius of the direct cache is undefined. To remedy this problem, we also compute an *unshadowed* gradient for the single scattering cache by assuming that all light sources are visible. We then choose the smaller of the regular and the unshadowed radii as the valid radius.

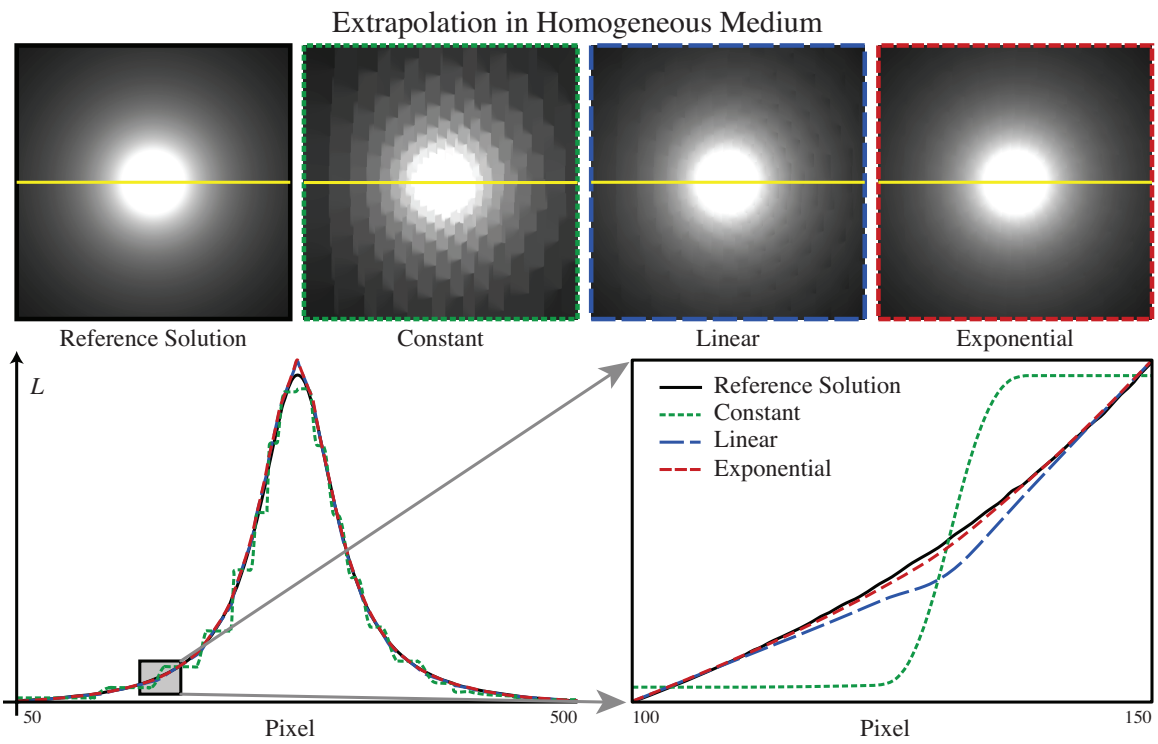


Figure 5.8: A comparison of extrapolation methods for a point-light scene in a homogeneous medium. The top row shows renderings using three different extrapolation methods. The scanlines highlighted in yellow are plotted in the graphs below. Our proposed exponential extrapolation scheme reconstructs the sampled function more accurately than the other methods.

Finally, we set minimum and maximum radii for the cache points in both world and screen space, and clamp gradients according to standard techniques when using radiance caching methods [Křivánek et al., 2006].

5.5.3 The Extrapolated Radiance Estimate

Previous caching techniques for surfaces use constant or linear extrapolation to approximate the radiance at a given location from a set of cached points. Constant extrapolation reuses the value at a cache entry for the new location, while linear extrapolation uses a constant slope to extrapolate the radiance with distance. In participating media, however, light intensity falls off exponentially with distance. We exploit this property in extrapolating the in-scattered radiance from cache points by exponentially extrapolating the cached values. We have found that exponential extrapolation provides a smoother reconstruction of the radiance field than linear extrapolation.

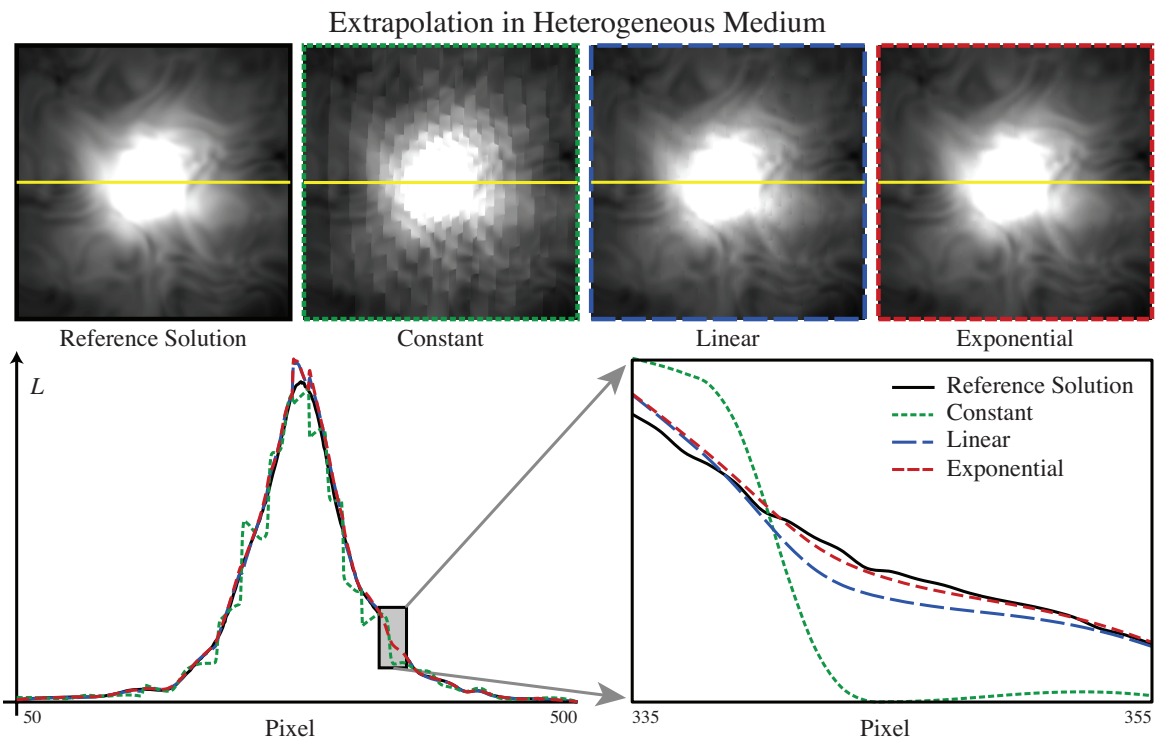


Figure 5.9: A comparison of extrapolation methods for a point-light scene in a heterogeneous medium. The top row shows renderings using three different extrapolation methods. The scanlines highlighted in yellow are plotted in the graphs below. Our proposed exponential extrapolation scheme reconstructs the sampled function more accurately than the other methods.

Exponential Extrapolation. Exponential extrapolation is equivalent to linear extrapolation in log-space. The radiance at a point $\mathbf{x}$ is approximated from a set of cache points C whose valid radii contain our query location by a weighted sum of extrapolated radiance values:

$$L(\mathbf{x}) \approx \exp\left(\frac{\sum_{k\in C}\left(\ln(L_k)+\frac{\nabla L_k}{L_k}\cdot(\mathbf{x}-\mathbf{x}_k)\right)w(d_k)}{\sum_{k\in C} w(d_k)}\right) \tag{5.47}$$

where L_k, ∇L_k, $\mathbf{x}_k$, and r_k are the radiance, gradient, position, and valid radius of cache point k, respectively. The weighting function w is a smooth cubic, $w(d) = 3d^2 - 2d^3$. We define $d_k = 1 - ||\mathbf{x}_k - \mathbf{x}||/r_k$ so that the weighting function has the most influence at the cache location, and this influence falls off smoothly to zero at the valid radius.

Figures 5.8 and 5.9 compare constant, linear, and exponential extrapolation in a 2D slice of a scene containing a point source in a participating medium. The images were rendered using

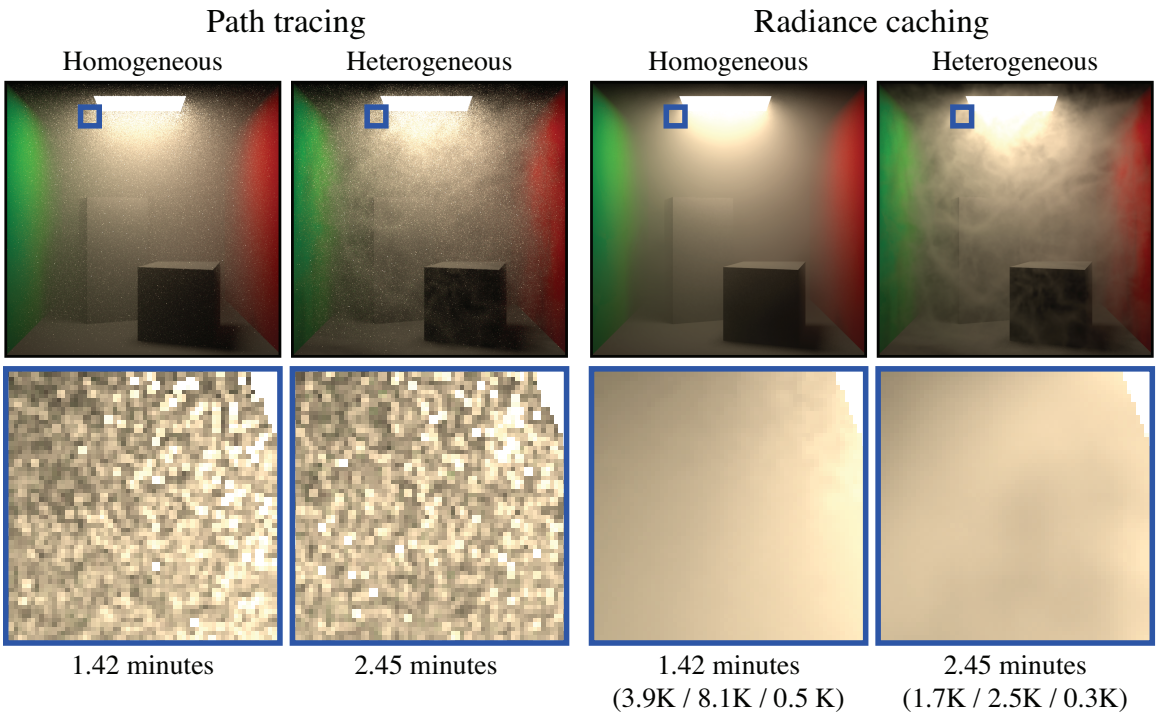


Figure 5.10: A Cornell box filled with isotropic smoke. We perform an equal-time comparison of our method to path tracing and perform a full lighting simulation including single scattering from lights, single scattering from surfaces, and multiple scattering. Render times are reported underneath each image, with the number of cache points in the single, surface, and multiple scattering caches in parentheses. A visualization of the cache points used to render the far left image is shown in Figure 5.12.

the same cache point locations; the only difference is the method used to extrapolate the radiance. Constant extrapolation is clearly a poor choice for extrapolation in participating media, as it gives strong discontinuity artifacts from the jumps between points, while linear extrapolation does a somewhat better job at approximating the change in radiance. Exponential extrapolation most precisely matches a reference solution with almost no visible artifacts, as it more closely approximates the real change in radiance over distance.

5.6 Results

We implemented the volumetric radiance caching algorithm in a Monte Carlo ray tracer, and all our timings were made on an Intel Core 2 Duo 2.4 GHz machine using only one core. We performed comparisons between path tracing, radiance caching, and photon mapping. Since each rendering technique has its own set of parameters, we performed equal-time comparisons

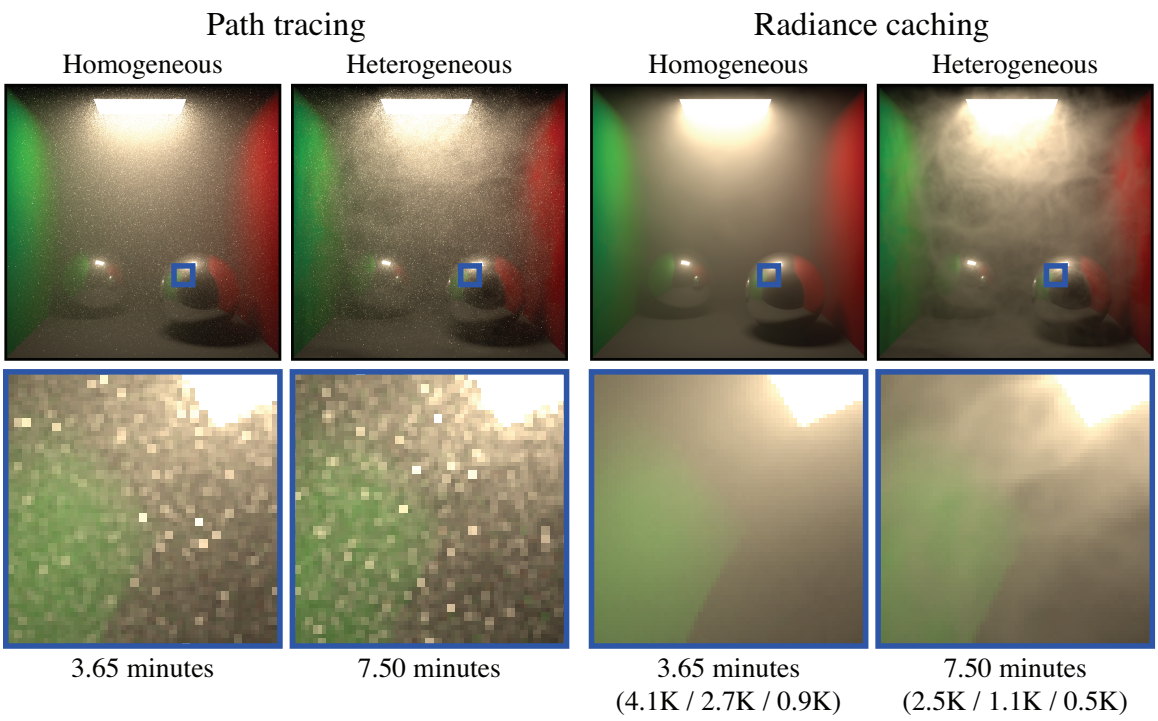


Figure 5.11: A Cornell box filled with anisotropic smoke. We perform an equal-time comparison of our method to path tracing and perform a full lighting simulation including single scattering from lights, single scattering from surfaces, and multiple scattering. We use a Henyey-Greenstein phase function with $g = 0.4$. Render times are reported underneath each image, with the number of cache points in the single, surface, and multiple scattering caches in parentheses.

while hand-picking the best parameters for each algorithm. All three methods make full use of standard optimization techniques including Russian roulette, stratification, and importance sampling, where applicable. All images were rendered at 1K horizontal resolution with up to 16 samples per pixel.

Figures 5.10 and 5.11 show a series of images of a Cornell box filled with smoke. We compare our method to path tracing for a variety of participating media types. Render times range from around 1.5 minutes for homogeneous media with isotropic scattering, to 7.5 minutes in heterogeneous media with anisotropic scattering. In all four examples, path tracing exhibits significant noise for the same render time. In the heterogeneous images, the scattering properties are defined procedurally using several octaves of noise [Ebert et al., 2002]. For anisotropic scattering, by projecting the full directional radiance field onto spherical harmonics, we are able to correctly reuse cached values even for reflection rays on the mirror spheres. A visualization of the radiance cache point locations from the homogeneous/isotropic image is shown in Figure 5.12.

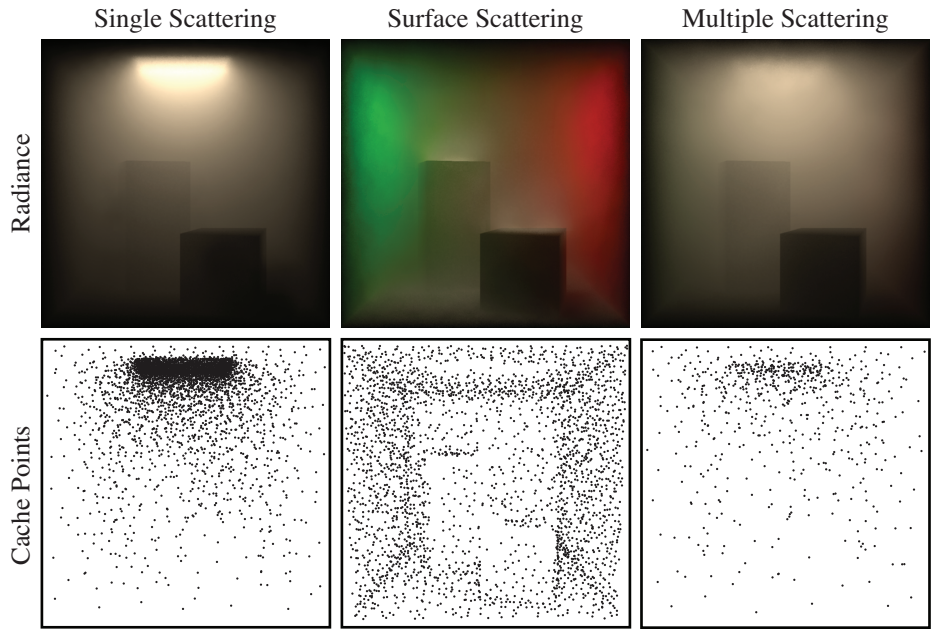


Figure 5.12: The bottom row shows a visualization of the single, surface, and multiple scattering cache points using the radiance cache from the homogeneous image in Figure 5.10. The top row shows the corresponding extrapolated radiance components (the radiance from surfaces past the medium is not included). For display purposes, the multiple and surface scattering images have been raised by one and three f-stops respectively.

Figure 5.13 shows a frame from an animation moving through light beams in the Sponza atrium. In this scene, all lighting is provided using a single point light, so we do not cache single-scattering but compute it directly. Computing multiple scattering is particularly challenging in this scene and requires tracing thousands of paths to estimate the in-scattered radiance. Our caching method is able to produce a noise-free result in 19 minutes. For walk-through style animations where only the camera position changes, we are able to reuse the radiance cache for subsequent frames, achieving an even greater speedup. Sponza is an extremely difficult scene for photon mapping. We tried performing an equal-time comparison to photon mapping. However, even by using projection maps, tracing photons for over 40 minutes produced less than 1K usable photons.

Figure 5.14 shows a frame from an animation of evolving smoke, which took less than six minutes to generate. In this scene, each frame is rendered using a separate cache. Our method is able to produce flicker-free animations even when the cache is recomputed at each frame. With the same render time the path tracing solution exhibits visible noise, which flickers significantly

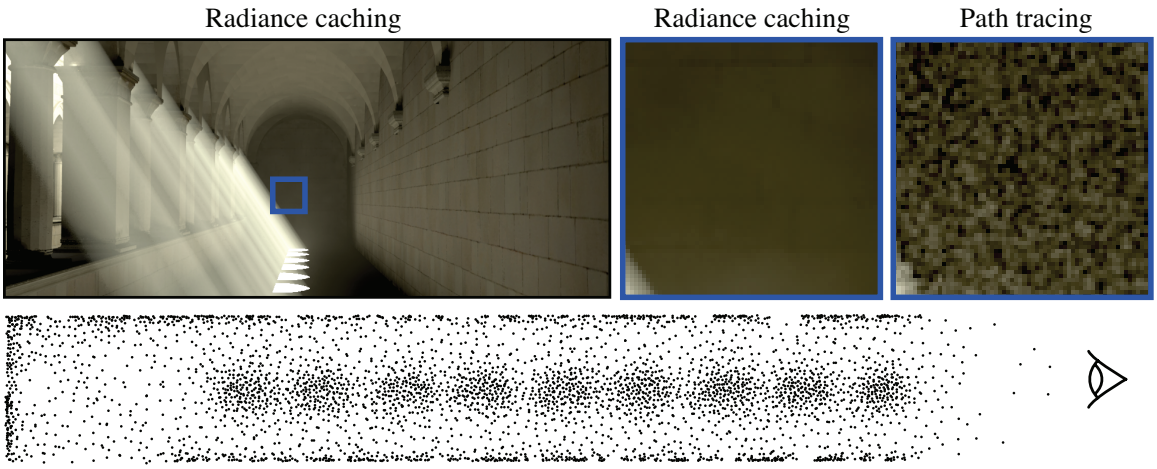


Figure 5.13: The Sponza atrium with beams of light and multiple scattering. With volumetric radiance caching, this scene renders in 19 minutes using about 46K cache points and 79M cache queries. The zoomed insets on the right provide an equal-time comparison to path tracing. The bottom image visualizes the surface cache using a horizontal cross section near the floor of the hallway. The camera is located on the right looking towards the left. Note the concentration of cache points near the brightly illuminated regions of the floor. The radiance caching image was rendered with error tolerances set at 0.5 and 0.01 for the surface and multiple scattering caches, respectively. For multiple scattering, our method traces 512 random-walk paths per cache point.

when animated. For this scene, we store about 9K cache points.

Figures 5.15 and 5.16 show a scene of two cars on a foggy road. This scene has a large extent and is costly to render with methods such as photon mapping. Here, our caching method fully exploits the smoothness in the radiance across the scene. In the same render time, we are able to attain significant noise reduction compared to path tracing and a smoother reconstruction than photon mapping using over eight million photons.

For all example scenes in this chapter, a large portion of render time is spent querying for cache points within the octree (typically between one third and one half of total render time). This is due to the large number of queries performed while ray marching through the medium (in the millions) compared to the number of cache points created (in the thousands). However, for the multiple scattering computation in our example scenes, we trace between 2K and 8K random-walk paths. Due to this, it is important to note that computing a new cache point takes about 100 to 1,000 times longer than querying within the octree.

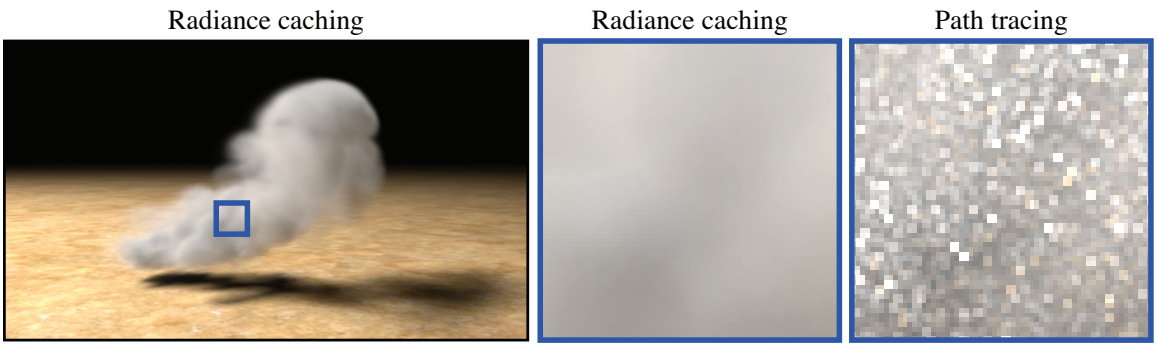


Figure 5.14: This still frame from an animation of heterogeneous smoke renders in 5.8 minutes using 9K cache points and 8.2M cache queries. Our method is able to produce flicker-free animation even when recomputing a new cache for each frame. The highlighted region is shown zoomed-in on the right, providing an equal-time comparison to path tracing. Error tolerances of 0.05, 1.0, and 1.0 were used for the single, surface, and multiple scattering caches with 2K multiple scattering paths per cache point.

5.7 Summary and Discussion

Many promising avenues remain for combining the strengths of photon mapping and radiance caching. Since photon mapping does not attempt to accelerate the computation of single scattering, radiance caching could effectively accelerate this computation within a photon mapping renderer. Another promising extension is to use our caching technique as a final gather pass for photon mapping. Currently, in highly scattering scenes with many occluders, a large number of multiple scattering rays are necessary using our method. The density estimation in photon mapping provides a natural smoothing, which could reduce the number of multiple scattering rays necessary and improve the cached result. Furthermore, photon mapping is able to efficiently simulate multiple scattering in a variety of scenes because it is well suited for detecting light paths difficult to sample from the eye. However, for smooth reconstruction a large number of photons are required. Performing a final gather would also reduce the necessary number of stored photons and improve the reconstruction quality as compared to regular photon mapping. For isotropic media, final gather optimizations developed for surfaces [Christensen, 1999] could be extended for volumetric final gather. Information from the photon map could also be used in other ways – for instance, to provide an importance sampling function for computing the direct and indirect in-scattered illumination.

One of the main challenges for our technique stems from ignoring the visibility gradient.

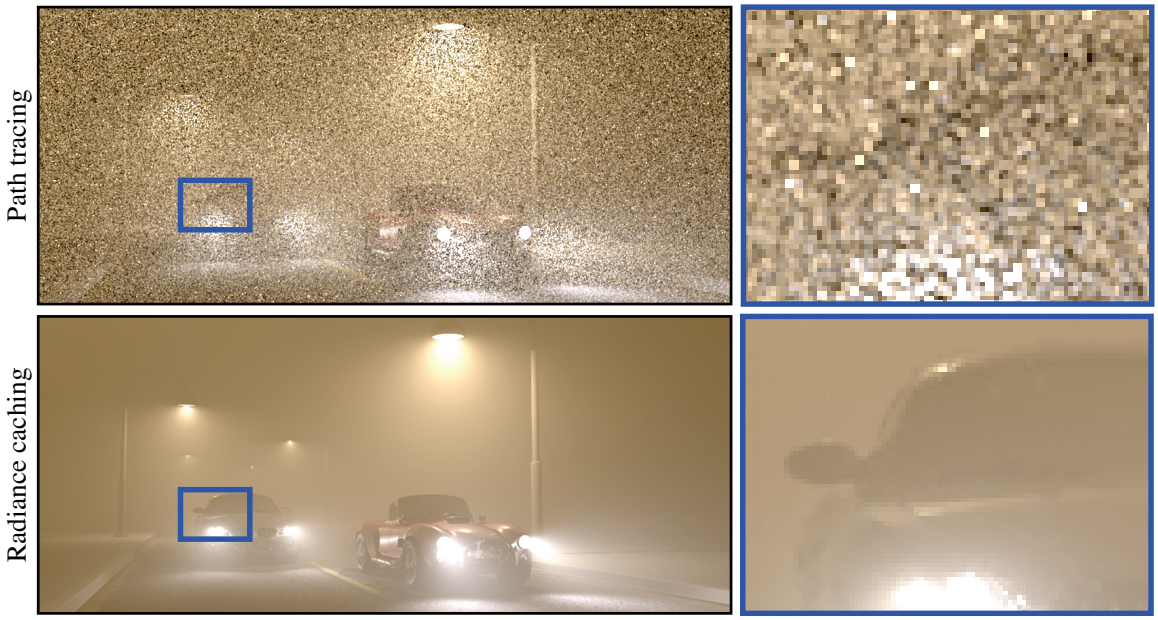


Figure 5.15: Two cars in a dense fog on a road illuminated by 60 lights. Our radiance caching method renders the cars scene in 20.35 minutes using only 175K cache points and 27M cache queries. The multiple scattering computation traces 4K random-walk paths per cache point. The highlighted region is shown zoomed-in on the right, providing an equal-time quality comparison between radiance caching and path tracing. The error tolerances were set at 0.25 and 0.75 for the single and multiple scattering caches, respectively.

Due to this simplification, our error metric is unable to adapt sample density at volumetric shadow boundaries, forcing the user to set a lower global error tolerance to produce artifact-free renderings. Incorporating the visibility gradient, both for extrapolation and in the error metric, could significantly increase quality while reducing render times. We plan to address this limitation in future work.

Caching the radiance and gradient computation using spherical harmonics is effective for moderately anisotropic scattering. For highly directional scattering, this approach becomes less beneficial. Since the cache points can potentially be reused from any viewing direction, our error metric does not account for the outgoing direction when computing the valid radius. An alternate approach could be to store radiance values in a 5D cache with associated outgoing directions. Since, in this case, cache points would only be reused for similar outgoing directions, importance sampling could be used even at the first bounce for multiple and surface scattering. Such an approach would benefit significantly from a full 5D error metric that computes a valid radius independently for the spatial and angular dimensions.

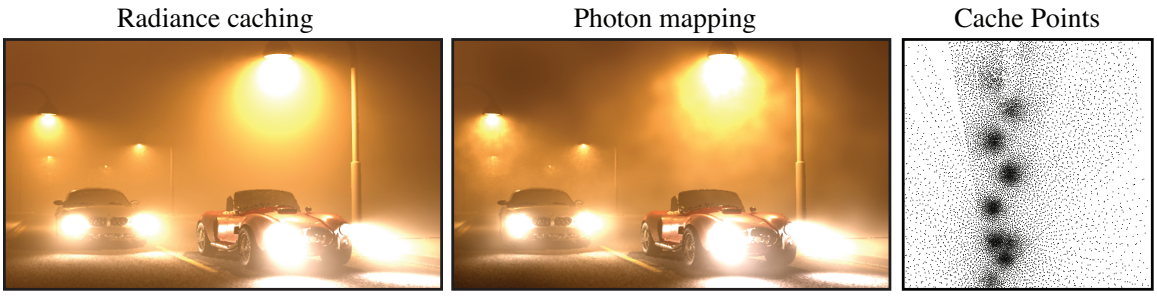


Figure 5.16: A equal-time, contrast-enhanced comparison between radiance caching and photon mapping in the cars scene from Figure 5.15. The scene has a large extent, making it difficult to render with methods such as photon mapping that compute scattering in the entire medium. Artifacts remain in the *homogeneous* medium even after storing 8M photons. On the right we show a horizontal cross section of the single scattering cache points as seen from above. In this visualization, the camera is located at the bottom looking up. Note the concentration of cache points around the street lights and headlights near the camera. The error tolerances were set at 0.25 and 0.75 for the single and multiple scattering caches, respectively.

Although our method caches the results of integrating single and multiple scattering, other expensive quantities could also be cached to further improve performance. For scenes with a large visible extent and large image resolutions, cache queries dominate the render time due to the repeated lookups necessary for integrating radiance along eye rays. Caching the radiance integration along eye rays could provide for efficient reuse of the *whole* radiance computation from neighboring pixels, completely eliminating the eye integration for a majority of the image.

Finally, the original surface irradiance caching algorithm could be extended to account for the presence of participating media and coupled with our method. Since, however, there is no distance to surfaces associated with volumetric cache points, this would be a challenging undertaking. Also, the irradiance gradients would have to account for the change in optical properties along a path as the point moves on the surface, similar to our scattering gradients.

5.8 Conclusion

In this chapter we presented a new method for caching radiance in the presence of participating media. The method is general and adapts to scenes with arbitrary scattering and absorbing properties. It works directly in both homogeneous and heterogeneous media, and can handle both isotropic or anisotropic scattering. When rendering anisotropic media, we compactly store the directional distribution of the radiance using spherical harmonics. Our method caches

both single and multiple scattered radiance from light sources and surfaces while providing significant gains in speed over traditional Monte Carlo ray tracing techniques.

Volumetric radiance caching is founded on using the gradients of a Monte Carlo formulation of radiative transport to accurately predict the change in radiance throughout the scene. We have shown how to compute these gradients and also how to use them to extrapolate the radiance to new locations. This, combined with an exponential extrapolation technique that exploits the properties of light in a medium, allow us to render high-quality images of scenes with a relatively small number of cache points. Caches produced by our method handle complex illumination and can be later reused for animations.

5.9 Acknowledgements

The material in this chapter is, in part, a reproduction of the material published in Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. “Radiance Caching for Participating Media.” In *ACM Transactions on Graphics*, 27(1):1–11, 2008. The dissertation author was the primary investigator and author of this paper. This work was supported in part by NSF grant CPA 0701992.

6

Irradiance Gradients in the Presence of Participating Media and Occlusions

"Write a wise saying and your name will live forever."

—Unknown

THE irradiance caching [Ward et al., 1988] algorithm described in Chapter 3 is one of the most successful acceleration strategies for solving the rendering equation using Monte Carlo ray tracing. In the previous chapter we developed a complementary radiance caching algorithm to efficiently solve the *volume* rendering equation *within* participating media by accounting for surface-volume and volume-volume interactions of light. These two approaches can easily be coupled to simulate all light transport by caching both on surfaces and within the medium. However, irradiance caching and volumetric radiance caching both rely on accurate gradient computations to improve the accuracy of interpolation. Participating media, unfortunately, affects not only the irradiance but also the irradiance gradient on surfaces embedded within media.

6.1 Contributions

In Table 6.1 we summarize the most important previous techniques for computing illumination gradients for irradiance caching. Unfortunately, none of these methods take into account the full radiative transport equation, which leads to increased interpolation artifacts in the presence of participating media. In this chapter we present extensions to these techniques to reduce these problems. We develop novel gradients that consider absorption, emission, and scattering in volumetric media, and the effect of surfaces occluding media. In particular, we

present the following contributions:

1. We derive gradients to handle indirect illumination from surfaces in the presence of **absorbing** media and occlusions. This generalization also allows us to handle non-Lambertian reflectors.

2. We derive gradients of irradiance in the presence of **scattering** or **emissive** media and account for surface-medium occlusion changes.

These two contributions enable, for the first time, the accurate computation of illumination gradients in the context of the full radiative transport equation. Without tracing any additional rays, the gradients described in this chapter are more accurate and contain significantly less noise than gradients computed using expensive numerical techniques such as finite differencing. The resulting gradient computations are easy to implement and straightforward to incorporate into an irradiance caching framework. We show that the new gradients produce higher-quality interpolation than previous techniques, which do not take into account full radiative transport.

Table 6.1: A comparison of the capabilities of illumination gradient techniques. The gradients derived in this chapter take into account the full radiative transport equation, including the effects of absorbing, emissive, and scattering participating media (PM). We support cache points on glossy surfaces (GCP), and we consider effects of glossy indirect reflectors (GIR), visibility changes (V), indirect illumination (II), and curved objects (CO).

Method	PM	GCP	GIR	V	II	CO
Ward and Heckbert [1992]				✓	✓	✓
Arvo [1994]				✓	✓	
Křivánek et al. [2005b]		✓			✓	✓
Křivánek et al. [2005a]		✓		✓	✓	✓
Ramamoorthi et al. [2007]		✓	✓	✓		✓
Chapter 5	✓	✓	✓		✓	✓
This Chapter	✓	✓	✓	✓	✓	✓

6.2 Overview

As we saw in Chapter 2, irradiance E at a surface location $\mathbf{x}$ can be written as the cosine-weighted integral of incident radiance,

$$E(\mathbf{x}) = \int_{\Omega} L(\mathbf{x} \leftarrow \vec{\omega}) \, (\vec{\mathbf{n}} \cdot \vec{\omega}) \, d\vec{\omega}. \tag{6.1}$$

We are interested in computing the gradient $\nabla E(\mathbf{x})$ with respect to a translation of $\mathbf{x}$.

In the presence of participating media, light transport obeys the *radiative transfer equation* from Equation 4.27. In this case we can interpret the incident radiance $L(\mathbf{x} \leftarrow \vec{\omega})$ as the sum of two separate contributions: the radiance L_s coming from surfaces that reflect illumination, and the radiance L_m contributed by the participating medium, i.e.,

$$L(\mathbf{x} \leftarrow \vec{\omega}) = \underbrace{T_r(\mathbf{x} \leftrightarrow \mathbf{x}_s) \, L(\mathbf{x}_s \rightarrow -\vec{\omega})}_{L_s(\mathbf{x} \leftarrow \vec{\omega})} + \underbrace{\int_0^s T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t) \, \sigma_s(\mathbf{x}_t) \, L_i(\mathbf{x}_t \rightarrow -\vec{\omega}) \, dt}_{L_m(\mathbf{x} \leftarrow \vec{\omega})}, \tag{6.2}$$

where $\mathbf{x}_t = \mathbf{x} + t\vec{\omega}$ with $t \in (0, s)$, and s is the distance through the medium to the nearest surface at $\mathbf{x}_s = \mathbf{x} + s\vec{\omega}$. The outgoing radiance $L(\mathbf{x}_s \rightarrow -\vec{\omega})$ is computed by integrating the lighting and BRDF at $\mathbf{x}_s$, and $T_r(\mathbf{x} \leftrightarrow \mathbf{x}_s)$ is the transmittance of the medium between $\mathbf{x}$ and $\mathbf{x}_s$.

To improve the clarity of our derivations, we split up the total irradiance in Equation 6.1 into irradiance from surfaces E_s and irradiance from the medium E_m such that

$$E_s(\mathbf{x}) = \int_{\Omega} L_s(\mathbf{x} \leftarrow \vec{\omega}) \, (\vec{\mathbf{n}} \cdot \vec{\omega}) \, d\vec{\omega}, \tag{6.3}$$

$$E_m(\mathbf{x}) = \int_{\Omega} L_m(\mathbf{x} \leftarrow \vec{\omega}) \, (\vec{\mathbf{n}} \cdot \vec{\omega}) \, d\vec{\omega}, \tag{6.4}$$

and $E(\mathbf{x}) = E_s(\mathbf{x}) + E_m(\mathbf{x})$. The total gradient is simply $\nabla E = \nabla E_s + \nabla E_m$. We derive gradients for the irradiance from surfaces in Section 6.3 and for irradiance from participating media in Section 6.4. We illustrate this process in Figure 6.1.

Note that in the following sections we only consider irradiance gradients for brevity, but all of these computations could easily be projected onto spherical harmonics to obtain full radiance gradients, as done in the previous chapter. Also, for clarity, we do not include media emission in

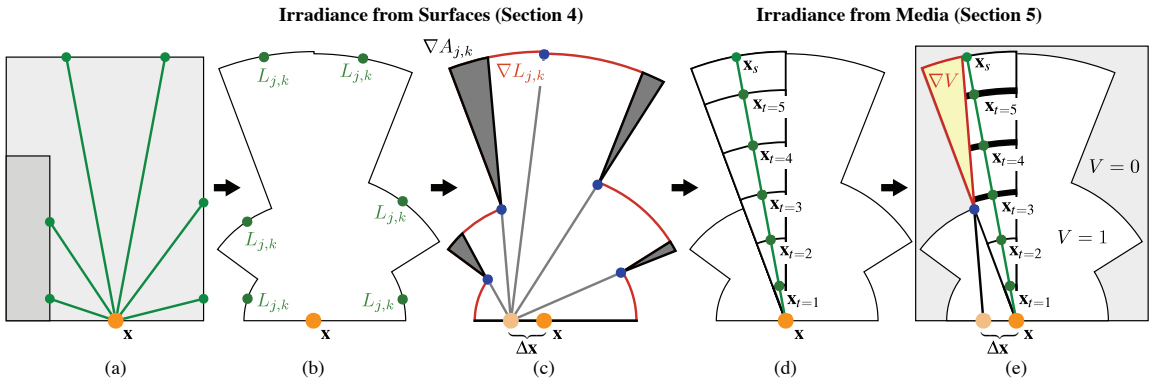


Figure 6.1: Irradiance caching approximates irradiance by tracing rays in a hemisphere (a) and calculating the radiance and distance to each hitpoint (b). Ward and Heckbert express irradiance gradients (c) by computing $\nabla A_{j,k}$ (grey) of each cell under a translation of $\mathbf{x}$, where the rate of motion of a cell boundary is determined by the distance (blue) to the closer surface. Our method accounts for absorbing media by additionally considering $\nabla L_{j,k}$ (red). We use ray marching (d) by sampling the in-scattered radiance at discrete points $\mathbf{x}_t$. For each discrete "shell" we compute both a radiance gradient and a visibility gradient (e) to account for surfaces occluding the medium. Conceptually, ∇V reduces the contribution of radiance from shells (highlighted) beyond the occluder (blue).

our derivations. This can be trivially included since the emission and in-scattered terms are nearly identical. We also restrict our derivations to translational gradients. We account for gradients on curved surfaces by including a rotational gradient computed as described by Ward and Heckbert [1992]. Unlike the translational gradient, participating media does not influence the computation of the rotational gradient.

6.3 Irradiance Gradients for Surfaces

In this section we present a generalization of Ward and Heckbert [1992] and Křivánek et al. [2005a] irradiance gradients. Our generalization relaxes the restriction that radiance arriving at $\mathbf{x}$ from an indirect light source at $\mathbf{x}'$ is constant under translation of $\mathbf{x}$. This generalization allows us to correctly handle scenes where the radiance may change due either to absorption by participating media or glossy reflectors.

6.3.1 Irradiance Gradients

In the irradiance gradient computations of Ward and Heckbert [1992] and Křivánek et al. [2005a], the irradiance integral from Equation 6.3 is estimated using the stratified Monte Carlo

estimator introduced in Chapter 3,

$$E_s(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} A_{j,k} L_s(\mathbf{x} \leftarrow \vec{\omega}_{j,k}) (\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k}), \tag{6.5}$$

where we defined the relevant quantities in Table 3.1 and illustrated the stratified geometry in Figure 3.4.

The contribution of each sample is the product of the radiance through the cell, $L_s(\mathbf{x} \leftarrow \vec{\omega}_{j,k})$, and the area of the cell, $A_{j,k}$, and this contribution is weighted by the cosine term $(\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k})$. The irradiance gradient considers how these terms change when translating the center of projection by an infinitesimal amount. By differentiating Equation 6.5 we get:

$$\begin{aligned} \nabla E_s(\mathbf{x}) &\approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} \nabla (A_{j,k} L_s(\mathbf{x} \leftarrow \vec{\omega}_{j,k}) (\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k})) \\ &= \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} (\nabla A_{j,k} L^s_{j,k} + A_{j,k} \nabla L^s_{j,k}) (\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k}), \end{aligned} \tag{6.6}$$

where we have used $L^s_{j,k} = L_s(\mathbf{x} \leftarrow \vec{\omega}_{j,k})$ for brevity. Translation in the tangent plane may induce both a change in the area of a cell $\nabla A_{j,k}$ and a change in the radiance seen through a cell $\nabla L^s_{j,k}$. Note that the cosine term does not change under translation, so the $\nabla(\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k})$ term drops out.

The total surface irradiance gradient can be expressed as a sum of two terms,

$$\nabla E_s(\mathbf{x}) \approx \nabla_A E_s(\mathbf{x}) + \nabla_L E_s(\mathbf{x}), \tag{6.7}$$

where $\nabla_A E_s(\mathbf{x})$ includes the gradient terms containing changing cell area $\nabla A_{j,k}$, and $\nabla_L E_s(\mathbf{x})$ incorporates the terms with changing cell radiance $\nabla L^s_{j,k}$.

Previous work on irradiance and radiance gradients only considers the rate of change of each cell area, $\nabla A_{j,k}$, but assumes that $\nabla L^s_{j,k} = 0$. In Section 3.6.2 we provided detailed derivations of the cell area gradient contribution. In this chapter we consider the effects of $\nabla_L E_s(\mathbf{x})$, for which we derive expressions in the following section.

6.3.2 Gradient of Cell Radiance

By allowing the cell radiances to change with translation, we can include attenuation due to participating media as well as glossy reflectors in the gradients. We additionally compute a gradient $\nabla_L E_s$ that incorporates changes in cell radiance,

$$\nabla_L E_s(\mathbf{x}) \approx \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} A_{j,k} \nabla L^s_{j,k} (\vec{\mathbf{n}} \cdot \vec{\omega}_{j,k}), \tag{6.8}$$

where the area of each cell is

$$A_{j,k} = \int_{\phi_{k_-}}^{\phi_{k_+}} \int_{\theta_{j_-}}^{\theta_{j_+}} \sin\theta d\theta d\phi \implies (\cos\theta_{j_-} - \cos\theta_{j_+})(\phi_{k_+} - \phi_{k_-}). \tag{6.9}$$

For a cosine-weighted distribution, this gradient term reduces to the average of the cell gradients,

$$\nabla_L E_s(\mathbf{x}) \approx \frac{\pi}{MN} \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} \nabla L^s_{j,k}. \tag{6.10}$$

Using the definition of surface radiance from Equation 6.2, the gradient $\nabla L^s_{j,k}$ is

$$\nabla L^s_{j,k} = \nabla T_r(\mathbf{x} \leftrightarrow \mathbf{x}_s)\, L(\mathbf{x}_s, -\vec{\omega}_{j,k}) + T_r(\mathbf{x} \leftrightarrow \mathbf{x}_s)\, \nabla L(\mathbf{x}_s, -\vec{\omega}_{j,k}). \tag{6.11}$$

We compute the gradient of the transmittance term using the techniques described in the previous chapter. Though we do not demonstrate this in our results, glossy reflectors can easily be handled in our framework by computing the radiance gradient using the gradient of the BRDF at $\mathbf{x}_s$.

The final irradiance gradient is simply the sum of Ward and Heckbert's original gradient $\nabla_A E_s(\mathbf{x})$ and our additional cell radiance gradient term $\nabla_L E_s(\mathbf{x})$ as expressed in Equation 6.7. In Figure 6.2 we demonstrate the impact of adding the gradients of cell radiance. The resulting irradiance gradients are more accurate, and the quality of interpolation in irradiance caching is significantly improved.

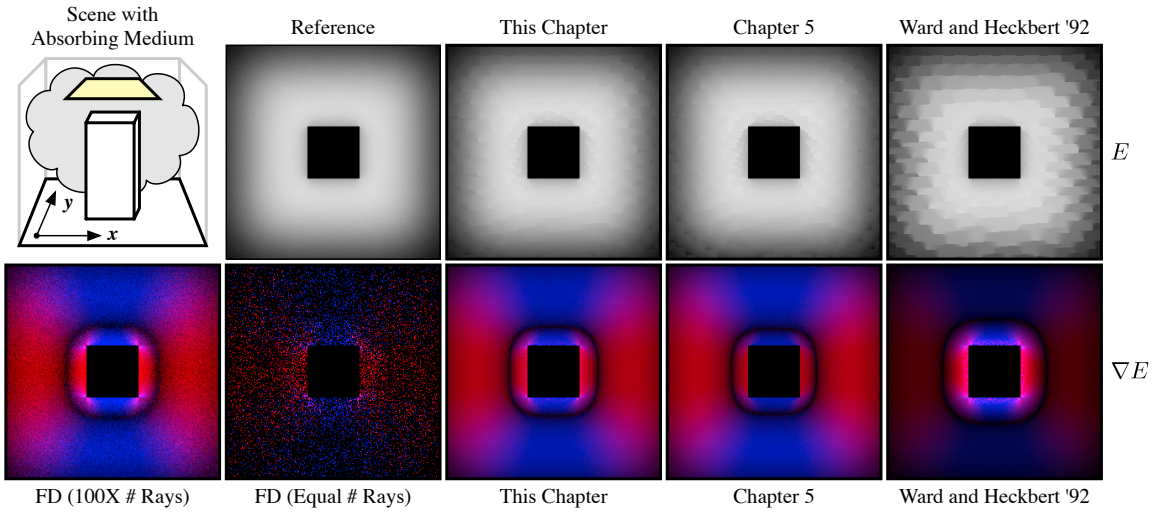


Figure 6.2: We compare irradiance caching and extrapolation using different gradient computation techniques in a scene with an **absorbing** medium. The top row shows irradiance *extrapolated* over the ground plane from a few cache points. The improved gradients from this chapter consider changes of visibility and absorption by the medium. Our method from Chapter 5 ignores visibility and Ward and Heckbert [1992] ignore absorption by the medium, which leads to increased artifacts. In the bottom row we visualize the irradiance gradients computed *per-pixel*. The red and blue color coding shows absolute values of the x and y components of the gradient, respectively. We computed gradients for each technique using 1.8K gather rays per pixel and also a "ground truth" gradient using finite differences (FD). Since finite difference gradients are extremely noisy, our reference uses 100 times more rays (180K rays/pixel). Our method produces accurate gradients with few gather rays and converges to the reference solution using finite differencing.

6.4 Irradiance Gradients for Media

In this section we consider the contribution of scattering media to the irradiance gradient. Our derivation is based on a reformulation of the radiance from the medium, L_m, in Equation 6.2. Instead of integrating to the closest visible surface, we introduce a visibility function and integrate to infinity. We rewrite the media irradiance, Equation 6.4, as

$$E_m(\mathbf{x}) = \int_\Omega \int_0^\infty T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t)\, \sigma_s(\mathbf{x}_t)\, L_i(\mathbf{x}_t \rightarrow -\vec{\omega})\, V(\mathbf{x} \leftrightarrow \mathbf{x}_t)\, (\vec{\mathbf{n}} \cdot \vec{\omega})\, dt\, d\vec{\omega}, \tag{6.12}$$

where we ignore the emission terms. The visibility function V returns one if the two arguments are mutually visible, and zero otherwise. If we collect all factors except visibility in a new term $L_V(\mathbf{x}, \mathbf{x}_t, \vec{\omega}) = T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t) \sigma_s(\mathbf{x}_t) L_i(\mathbf{x}_t, -\vec{\omega})(\vec{\mathbf{n}} \cdot \vec{\omega})$, which we call *unshadowed radiance*, the media

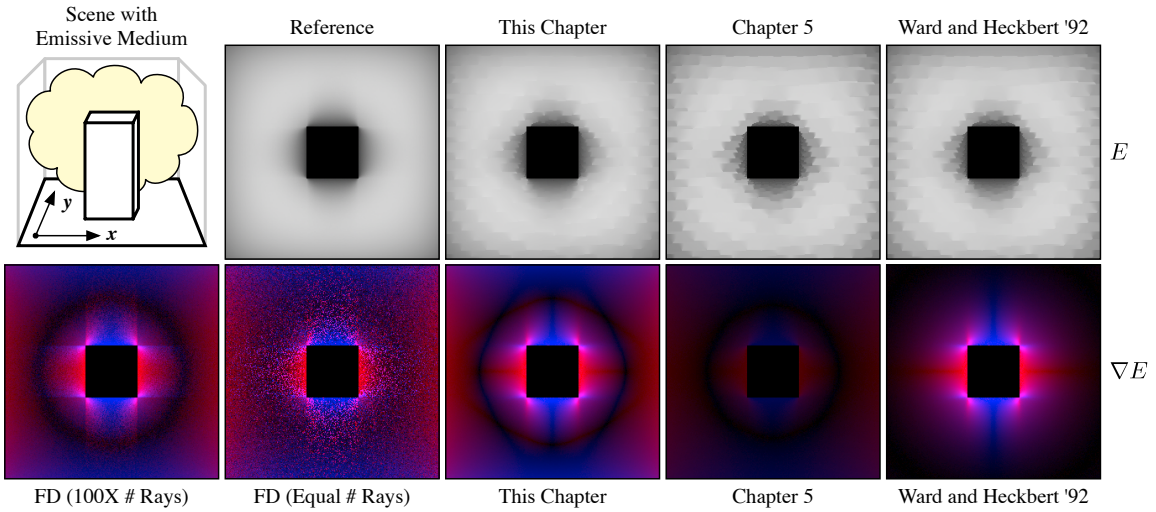


Figure 6.3: This modification of the scene from Figure 6.2 contains an **emissive** medium. The improved gradient formulation obtains accurate results by taking into account the medium and surface-medium occlusions, whereas previous methods ignore these effects and suffer from artifacts.

irradiance reduces to a product of unshadowed radiance and visibility,

$$E_m(\mathbf{x}) = \int_\Omega \int_0^\infty L_V(\mathbf{x}, \mathbf{x}_t, \vec{\omega})\, V(\mathbf{x} \leftrightarrow \mathbf{x}_t)\, dt\, d\vec{\omega}. \tag{6.13}$$

Differentiation using the product rule leads to the gradient of media irradiance,

$$\nabla E_m(\mathbf{x}) = \int_\Omega \int_0^\infty \nabla L_V\, V + L_V\, \nabla V\, dt\, d\vec{\omega}, \tag{6.14}$$

where we have omitted the function arguments for brevity.

In the previous chapter, we derived gradients of multiple scattering within participating media in a similar manner. In those derivations, however, we assumed visibility was constant, and the gradient of V was ignored. In effect, we only considered the first term of Equation 6.14. In order to obtain the full gradient, we now also compute the gradient of visibility, as described in the following section.

6.4.1 Visibility Gradient

In order to account for the visibility gradient, we need to compute the following quantity:

$$\nabla_V E_m(\mathbf{x}) = \int_0^\infty \int_\Omega L_V \nabla V \, d\vec{\omega} \, dt, \tag{6.15}$$

$$= \int_0^\infty \tilde{V}_\Omega(\mathbf{x}, t) \, dt, \tag{6.16}$$

where we use ∇_V to denote that this is a gradient only of the visibility component, and we swap the order of integration to make the derivation more convenient. Additionally, we introduce the shorthand $\tilde{V}_\Omega$ for the hemispherical integral of $L_V \nabla V$.

If we consider a fixed t, $\tilde{V}_\Omega$ computes the gradient of the weighted visibility integral over the hemisphere. This is similar to Ward and Heckbert, except they compute the gradient of a weighted *radiance* integral over the hemisphere. Therefore, by substituting the radiance function with visibility, and replacing the weighting functions, we can compute this weighted visibility gradient using Ward and Heckbert's formulation. Intuitively, we are computing an irradiance gradient in a scene where the radiance function encodes V as black occluders in front of a distant environment map, L_V. Applying these modifications to Equation 3.32 results in

$$\tilde{V}_\Omega \approx \sum_{k=0}^{N-1} \left(\hat{u}_k \sum_{j=1}^{M-1} \nabla_{\hat{u}_k} A_{j\text{-},k} (V_{j,k} - V_{j\text{-}1,k}) L_V(j\text{-}, k) + \hat{v}_{k\text{-}} \sum_{j=0}^{M-1} \nabla_{\hat{v}_{k\text{-}}} A_{j,k\text{-}} (V_{j,k} - V_{j,k\text{-}1}) L_V(j, k\text{-}) \right),$$

where the directional derivatives $\nabla_{\hat{u}_k} A_{j\text{-},k}$ and $\nabla_{\hat{v}_{k\text{-}}} A_{j,k\text{-}}$ are computed using Equations 3.42 and 3.46 and utilize the distance to occluders in their denominators. $V_{j,k}$ is a binary function indicating for each direction $\vec{\omega}_{j,k}$ whether the distance s to the nearest surface is less than or greater than t:

$$V_{j,k}(t, s) = \begin{cases} 0 & \text{if } t \geq s, \\ 1 & \text{if } t < s. \end{cases}. \tag{6.17}$$

In order to integrate over t we perform ray marching, which discretizes the medium into "shells" as illustrated in Figure 6.1(d,e). Conceptually, we consider radiance from the medium as coming from expanding shells of radius t about the evaluation point $\mathbf{x}$ and compute the visibility

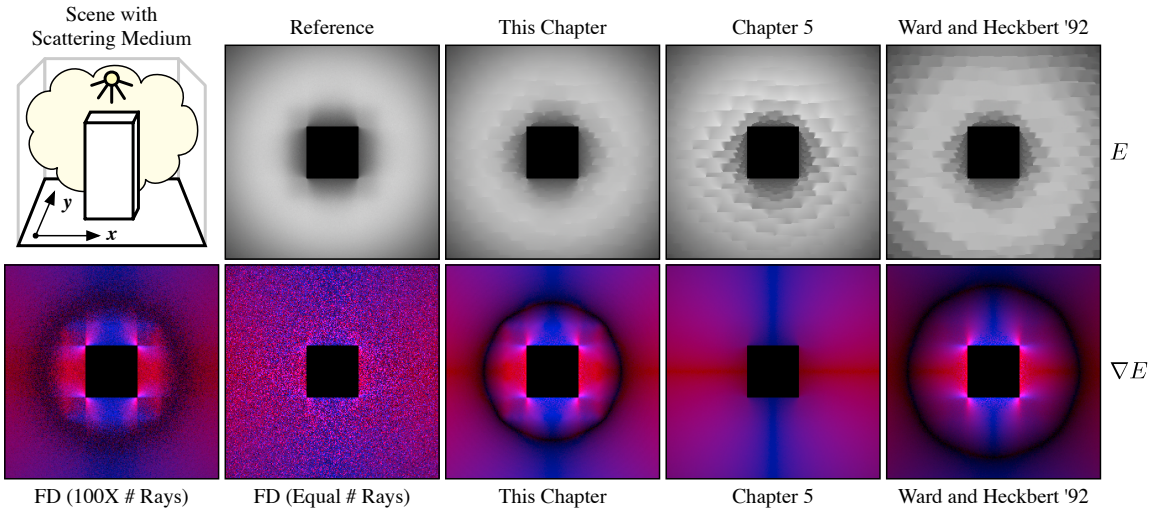


Figure 6.4: This modification of the scene from Figure 6.2 contains a spotlight and a **scattering** medium. The improved gradient formulation obtains accurate results by taking into account the medium and surface-medium occlusions, whereas previous methods ignore these effects.

gradient in Equation 6.16 as

$$\nabla_V E_m(\mathbf{x}) \approx \sum_{t=0}^{S-1} \tilde{V}_\Omega \Delta t. \tag{6.18}$$

In practice, however, ray marching is performed independently for each hemispherical direction, which swaps the order of the summations back again.

In summary, for a fixed distance t, we compute L_V, ∇L_V as in Chapter 5, and we compute a Ward and Heckbert-style gradient for $L_V \nabla V$ by evaluating the unshadowed radiance along visibility boundaries.

6.5 Implementation

Our new irradiance gradients can easily be added to a Monte Carlo renderer, which uses irradiance caching and supports participating media. The irradiance caching algorithm stays the same; just the gradient computation needs to be modified to account for the medium. Our procedure is illustrated in Figure 6.1. We provide pseudo-code for computing E_s and E_m and the gradients ∇E_s and ∇E_m in Algorithm 6.1 and describe the procedure in the following sections.

Algorithm 6.1: COMPUTEGRADIENT($\mathbf{x}, \vec{\mathbf{n}}$)

Data: $\mathbf{x}$ is the position and $\vec{\mathbf{n}}$ is the normal at the evaluation location.
Result: E_s, E_m, ∇E_s, and ∇E_m

begin
  foreach *cell* (j,k) **do**
    Determine distance `s` by tracing ray towards $\vec{\omega}_{j,k}$;
    Compute $L(\mathbf{x}_s, -\vec{\omega}_{j,k})$ and $\nabla L(\mathbf{x}_s, -\vec{\omega}_{j,k})$;
    `Ls(j,k)` $= L(\mathbf{x}_s, -\vec{\omega}_{j,k})$;
    `gLs(j,k)` $= \nabla L(\mathbf{x}_s, -\vec{\omega}_{j,k})$;
  $A \Leftarrow \frac{\pi}{M \cdot N}$;
  foreach *cell* (j,k) **do**
    $T_r \Leftarrow 1$;
    $\nabla T_r \Leftarrow 0$;
    foreach *ray-marching step through the medium* **do**
      Update T_r and ∇T_r;
      Compute L_V, and ∇L_V as in Chapter 5;
      Compute $L_V \nabla V$ using Equation 6.17;
      `Lm(j,k)` $+= L_V A \Delta t$;
      `gLm(j,k)` $+= (L_V \nabla V + \nabla L_V A)\Delta t$;
    `gLs(j,k)` $=$ `gLs(j,k)` $T_r +$ `Ls(j,k)` ∇T_r;
    `Ls(j,k)` $*= T_r$;
  return $E_s = A \sum_{j,k}$ `Ls(j,k)`;
  return $\nabla E_s =$ Equation 3.47 $+ A \sum_{j,k}$ `gLs(j,k)`;
  return $E_m = \sum_{j,k}$ `Lm(j,k)`;
  return $\nabla E_m = \sum_{j,k}$ `gLm(j,k)`;
end

Initialization. The irradiance and gradient computation starts by creating a $M \times N$ array, where each element stores a `HemiSample`. This array is used to store the hemispherical samples needed to compute the irradiance and the irradiance gradient. Each `HemiSample` stores the distance `s`, the surface radiance and gradient `Ls` and `gLs`, and the media radiance and gradient `Lm` and `gLm`. All these values are initialized to zero.

Sampling Surfaces. The algorithm starts by looping over each cell (j,k) and tracing a ray in the $\vec{\omega}_{j,k}$ direction, generated using a cosine-weighted distribution [Ward and Heckbert, 1992]. We save the hit distance to the nearest surface, `s`, within the `HemiSample` and store the radiance $L(\mathbf{x}_s, -\vec{\omega}_{j,k})$ in `Ls`. If the intersected surface at $\mathbf{x}_s$ is glossy, we also compute a gradient $\nabla L(\mathbf{x}_s, -\vec{\omega}_{j,k})$ and store this in `gLs`. At the end of this stage we have a full hemispherical discretization of the

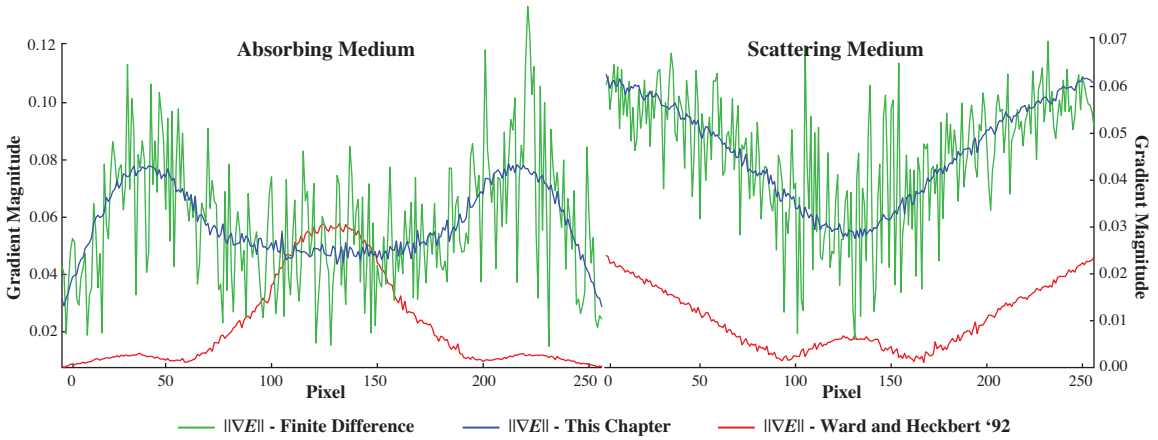


Figure 6.5: Visualizations of the gradient magnitude, (left) along a scanline in Figure 6.2, and (right) along a scanline in Figure 6.4. Our new gradient formulation correctly matches the profile of the finite difference gradient but with significantly less noise. Since it ignores participating media, the original Ward and Heckbert formulation does not produce the correct gradient in either of these situations.

scene, with a radiance, gradient, and distance to the nearest surface in each cell. The cell radiances and gradients do not yet take into account participating media.

Sampling Media. In order to account for the media, we perform ray marching individually for each cell (j,k). As mentioned in Section 6.4, the integration over the hemisphere is the outer loop. For each cell, at each step in the medium we compute $L_V(\mathbf{x}_t)$ and $\nabla L_V(\mathbf{x}_t)$ using the techniques developed in Chapter 5. We also compute $L_V(\mathbf{x}_t)\nabla V(\mathbf{x}_t)$ by evaluating Equation 6.17 for the (up to) four boundaries of the current cell: (j_-,k_-), (j_-,k_+), (j_+,k_-), and (j_+,k_+). We then multiply the terms by the cell area $A_{j,k}$ and accumulate their contributions into `Lm` and `gLm`.

At each step through the medium we maintain the current value of the transmittance T_r and its gradient ∇T_r. Once ray marching terminates, these values correspond to the transmittance $T_r(\mathbf{x}\leftrightarrow\mathbf{x}_s)$ and transmittance gradient $\nabla T_r(\mathbf{x}\leftrightarrow\mathbf{x}_s)$ from the evaluation location $\mathbf{x}$ to the surface at $\mathbf{x}_s$. We use these transmittance values to augment the `Ls` and `gLs` values computed in the previous stage in order to properly account for absorption by applying Equations 6.2 and 6.11.

Integrating. At the end of this process we integrate the irradiance and its gradient by summing over all the `HemiSamples`. Additionally, we add Equation 3.47 to the surface irradiance gradient to account for the changing cell areas.

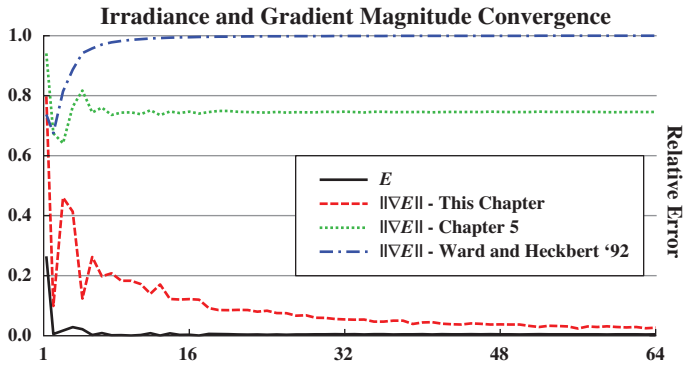


Figure 6.6: Relative errors for computing the irradiance and irradiance gradient for a single pixel in the scene from Figure 6.3. The x-axis plots M, the number of divisions in elevation angle, and N is set to $2M$. Previous gradient methods converge to incorrect values, leading to high error even with many samples.

6.6 Results

We implemented our irradiance gradients within a global illumination renderer written in C++. We compare our gradient method to Ward and Heckbert [1992] for a number of scenes. We also compare to a modification of Chapter 5's gradients, where we compute irradiance gradients on surfaces by integrating over the hemisphere (instead of the whole sphere) and consider the cosine-weighted BRDF (instead of the phase function). In all of our results comparing irradiance caching, we only change the gradient method used. All other parameters, including the number of evaluation rays and the resulting cache point locations, are kept exactly the same. All timings are for an Intel Core 2 Duo 2.4 GHz machine using one core.

In Figures 6.2, 6.3, and 6.4 we demonstrate the importance of considering effects from the medium for variations of a scene containing absorbing, emitting, and scattering media, respectively. We directly visualize the x, y gradient components in these figures and compare them to "reference" solutions computed using finite differences. These renderings were computed at a resolution of 256×256 using $30 \times 60 = 1,800$ rays per evaluation. Our gradients match the reference solutions more faithfully than previous techniques and contain less noise than finite difference gradients computed using 180,000 evaluation rays per pixel. We demonstrate the accuracy of our gradients in Figures 6.5 and 6.6. Figure 6.5 plots the gradient magnitude along a scanline in the scenes from Figures 6.2 and 6.4. In Figure 6.6 we visualize the convergence rate for the irradiance and gradient computed at a single pixel in the scene from Figure 6.3. Previous

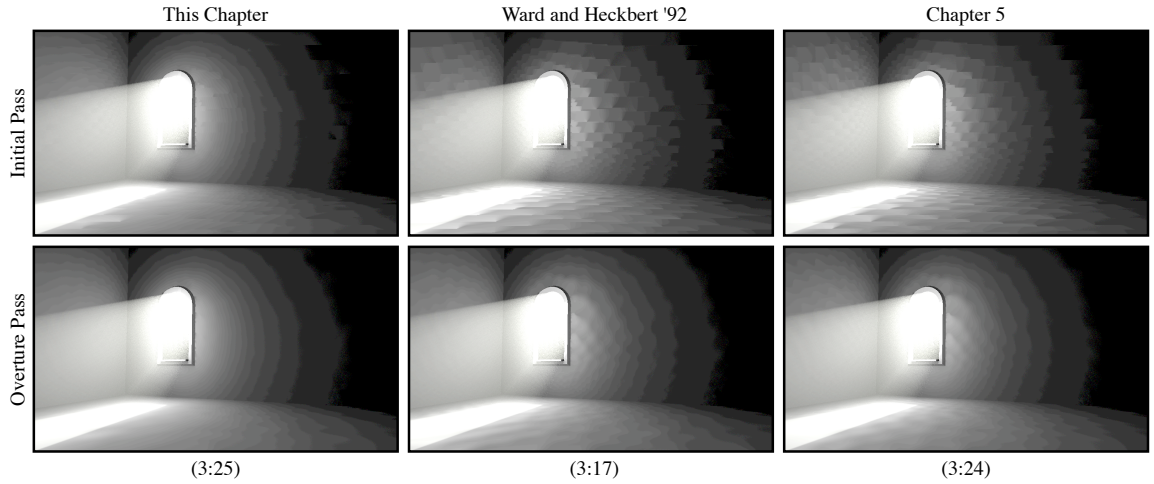


Figure 6.7: All the illumination on the walls of this room has first scattered off the floor or the bright beam of light through the window. Our improved gradient computation significantly reduces extrapolation artifacts (top). Using an "overture" pass, where the scene is re-rendered once all cache points are computed, previous techniques still suffer from interpolation artifacts (bottom), while our method produces smooth reconstruction.

techniques do not correctly capture the true gradient. The technique described in this chapter matches the finite difference gradient and quickly converges to the correct solution.

Figure 6.7 features a room with a strong volumetric light beam entering an open window. The walls and most of the floor in this scene are indirectly illuminated by this single beam of light. Using Ward and Heckbert's gradient computation, this scene renders in 3:17 minutes at a horizontal resolution of 1K. Since this gradient method ignores the media, all light is incorrectly assumed to come from surfaces during gradient computation. This results in significant artifacts. Performing an "overture" re-rendering of the image is a common technique for improving the quality of irradiance cache renderings. This allows for interpolation, instead of extrapolation, to be used for the entire image. Even though the overture pass improves the quality of the rendering by reducing sudden discontinuities, the inaccurate gradients result in distracting "ripples" on the walls and floor. The gradients from Chapter 5 do account for the medium but fail to handle visibility changes, which also leads to significant artifacts. The gradients from this chapter, on the other hand, can correctly handle this scene and produce noticeably smoother reconstruction of the indirect illumination. The overhead of the improved gradients is fairly negligible, and we are able to render the same image in 3:25 minutes. The overture pass takes a fraction of the total render time for each method, and completes in under four seconds.

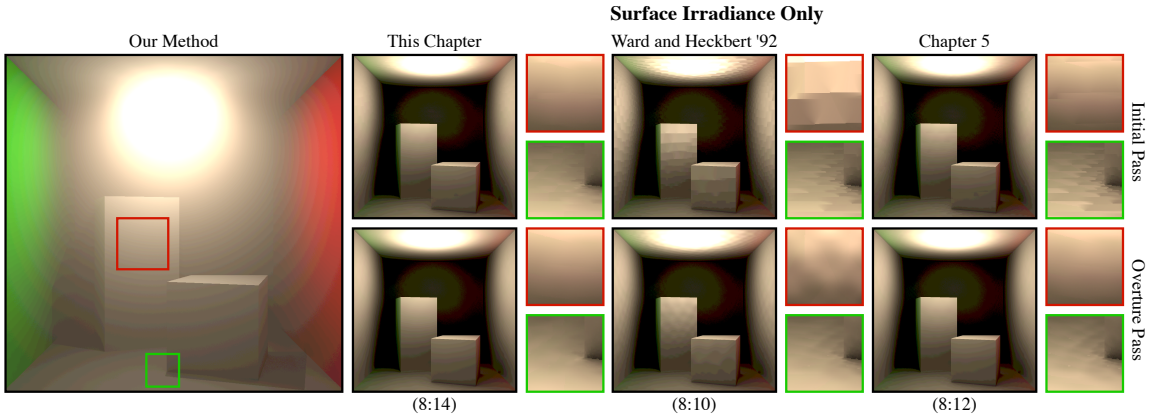


Figure 6.8: The classic Cornell box scene with a scattering medium. We compare the quality of irradiance caching and provide two zoomed-in regions for each result. Ward's gradient formulation does not consider the medium, which results in inaccurate gradients and significant extrapolation artifacts. Our participating media gradients from Chapter 5 do a much better job, but still suffer from artifacts due to occlusion changes. The gradients from this chapter obtain the smoothest results by taking into account the media and occlusions.

Figure 6.8 contains a modification of the classic Cornell box rendered at a resolution of 1K. Ward's method produces distracting artifacts even after an overture pass. Gradients computed using the techniques from Chapter 5 do a much better job in this scene, but still contain artifacts in areas with occlusion changes. The gradients derived in this chapter produce smooth results even in the initial pass. The overture pass takes less than five seconds for each method.

The disco light scene in Figure 6.9 presents a particular challenge for conventional gradient methods, since all lighting on surfaces is indirect lighting from the scattering medium. Due to this, Ward and Heckbert's gradients suffer from significant artifacts both on the surface of the sphere and the ground plane. The gradients from Chapter 5, since they ignore visibility changes, also suffer from artifacts. This is particularly noticeable on the ground near the base of the disco light, where visibility changes are most prominent. The improved gradients are able to obtain a smooth reconstruction of the irradiance on the first pass and we can render this scene at 1K resolution in 10:33 minutes. This is only a slight overhead on top of the 10:30 minute render time using Ward and Heckbert's gradients. The overture pass takes less than seven seconds to compute for each method.

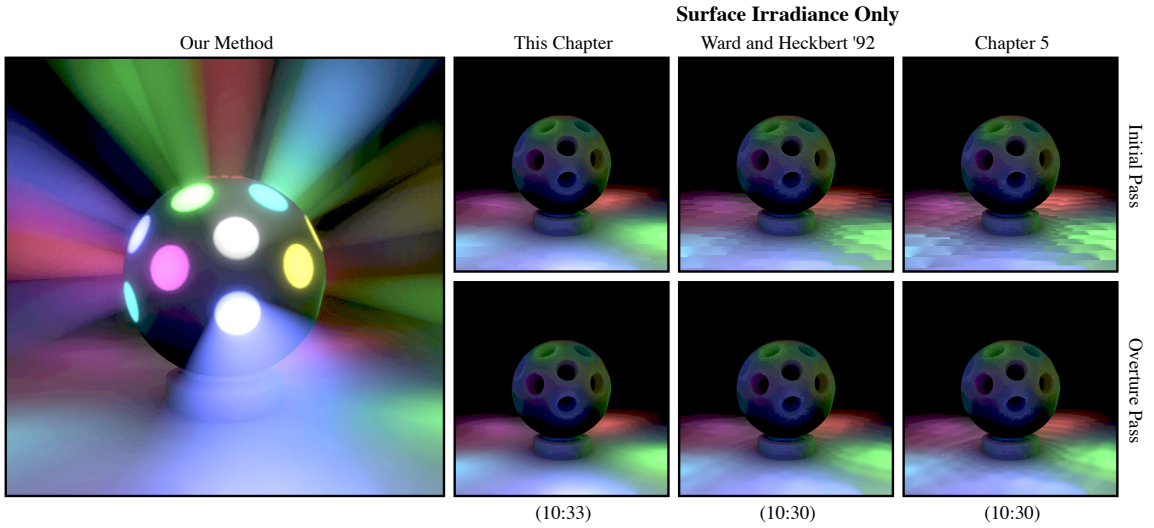


Figure 6.9: This disco light scene contains 21 light sources, but all lighting on surfaces is indirect. Ward's gradient formulation assumes all lighting (even from the media) arrives from surfaces, which results in inaccurate gradients and significant extrapolation artifacts. The participating media gradients from Chapter 5, however, do not take the visibility term into account. The gradients developed in this chapter obtain the smoothest results by taking into account the media and occlusions.

6.7 Summary and Discussion

Our work on computing accurate irradiance gradients in scenes containing participating media exposes a number of limitations of current techniques and, in the process, suggests several exciting possibilities for future work.

Error Metric. In this chapter we were only concerned with improving the quality of interpolation by computing more accurate gradients. However, another significant contributor to the efficiency and quality of the irradiance caching method is the error metric used to compute valid radii of cache points. The split-sphere model, which drives the error metric, is geometry-driven and completely ignores lighting and all effects from participating media. This can lead to suboptimal cache point distributions. A more general error metric is desirable, but it is not immediately clear how to extend the split-sphere model to incorporate these effects. In Chapter 5 we derived an error metric specifically for participating media, and Křivánek et al. [2006] introduced a neighbor clamping technique. Creating a robust error metric that takes into account the local geometry as well as lighting from surfaces and media is a difficult problem which warrants further work.

Radiance Gradients. Though we presented all of our derivations within the context of irradiance, it is trivial to apply our approach to radiance caching on surfaces. Křivánek et al. [2005a] showed that, to compute radiance gradients, the cosine terms in Equations 6.5 and 6.6 can be replaced by a set of basis functions. This projection enables efficiently storing the full radiance function as a vector of coefficients and the radiance gradient as a corresponding set of gradient coefficients.

Radiance Gradients in Participating Media. In Chapter 5 we compute radiance gradients of single scattering from lights, single scattering from surfaces, and multiple scattering, but ignore visibility in all of these computations. This chapter provides the first step in computing visibility-aware radiance gradients for caching *within* participating media.

Computing single scattering from surfaces is very similar to the surface irradiance presented in Section 6.3. Ward and Heckbert, however, only consider gradients in 2D along the tangent plane. Our surface irradiance gradients could be extended to work in participating media by additionally deriving expressions for the gradients of cell area with respect to motion along a third axis.

Similarly, the definition of media irradiance in Equation 6.12 is nearly identical to the computation of multiple scattering. Extending Ward and Heckbert's stratified gradients to consider motion along all three dimensions would also enable the use of a visibility gradient within the multiple scattering gradient.

Gradients of single scattering from light sources cannot be handled using Ward and Heckbert's stratified gradients since these effects are typically not computed using hemispherical integration. For point light sources, however, visibility gradients can be efficiently approximated without tracing additional rays by using shadow maps and performing finite differencing. We, in fact, implemented this extension and use it when computing the gradient of single-scattered radiance embedded within the ∇L_V term of Equation 6.14.

Visibility Gradient. Ward and Heckbert's stratified gradient formulation is intuitive and works well in practice; however, it is difficult to quantify its mathematical correctness since the gradient is performed after discretization. More recently, Ramamoorthi et al. [2007] presented an elegant

new visibility gradient formulation. Their approach is more mathematically rigorous since it presents an analytic expression for the gradients and only then performs discretization of this analytic expression. In Section 6.4.1 we modify Ward and Heckbert's gradients to estimate visibility gradients; however, an obvious alternative to this approach would be to directly use the gradients presented by Ramamoorthi et al. This approach would allow for a completely analytic expression for the media irradiance gradient. We explored this avenue, but Ramamoorthi's visibility integration suffers from a weak singularity. Though this singularity can be avoided by using adaptive sampling over the hemisphere, it makes it more cumbersome to integrate into the stratified ray marching process needed within our context. Nevertheless, we still believe this is a promising alternative and plan to investigate this approach further.

6.8 Conclusion

In this chapter we presented a method for accurately computing irradiance gradients on surfaces in the presence of participating media and occlusions. We applied our gradient derivations to the irradiance caching method and demonstrated that incorporating participating media and visibility information within the gradient is important for high quality irradiance interpolation in scenes containing these effects.

6.9 Acknowledgements

The material in this chapter is, in part, a reproduction of the material published in Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "Irradiance Gradients in the Presence of Participating Media." In *Computer Graphics Forum (Proceedings of EGSR 2008)*, 27(4), 2008. The dissertation author was the primary investigator and author of this paper. This work was supported in part by NSF grant CPA 0701992 and the UCSD FWGrid Project, NSF Research Infrastructure Grant Number EIA-0303622.

7

The Photon Mapping Method

"I get by with a little help from my friends."

—John Lennon, 1940–1980

PHOTON mapping is a practical approach for computing global illumination within complex environments. Much like irradiance caching methods, photon mapping caches and reuses illumination information in the scene for efficiency. Photon mapping has also been successfully applied for computing lighting within, and in the presence of, participating media. In this chapter we briefly introduce the photon mapping technique. This sets the foundation for our contributions in the next chapter, which make volumetric photon mapping practical.

7.1 Algorithm Overview

Photon mapping, introduced by Jensen [1995; 1996; 1997; 1998; 2001], is a practical approach for computing global illumination. At a high level, the algorithm consists of two main steps:

Algorithm 7.1: PHOTONMAPPING()

```
1 PHOTONTRACING();
2 RENDERUSINGPHOTONMAP();
```

In the first step, a lighting simulation is performed by tracing packets of energy, or *photons*, from light sources and storing these photons as they scatter within the scene. This processes

Algorithm 7.2: PHOTONTRACING()

```
1 n_e = 0;
2 repeat
3     (l, pdf(l)) = CHOOSELIGHT();
4     (x_p, ω⃗_p, Φ_p) = GENERATEPHOTON(l);
5     TRACEPHOTON(x_p, ω⃗_p, Φ_p / pdf(l));
6     n_e += 1;
7 until photon map full ;
8 Scale power of all photons by 1/n_e;
```

results in a set of *photon maps*, which can be used to efficiently query lighting information. In the second pass, the final image is rendered using Monte Carlo ray tracing. This rendering step is made more efficient by exploiting the lighting information cached in the photon map. Radiance can be evaluated at arbitrary points in the scene by locally computing the photon density within the photon map.

7.2 Photon Tracing

In the first pass, photons are emitted from light sources and traced through the scene just as rays are in ray tracing. The photon tracing pass is summarized in Algorithm 7.2. We describe this procedure in the following sections.

7.2.1 Photon Emission

Intuitively, photon mapping works by splitting the energy emitted by each light source into discrete packets called photons[1]. A number of photons are emitted from a light source, and each photon represents a fraction of the total power of the light. Photon tracing starts by generating a random photon at a light source. Each photon p includes a position and direction $\mathbf{x}_p, \vec{\omega}_p$, as well as an associated power, Φ_p. Any type of light source can be used to emit photons by providing an appropriate GENERATEPHOTON procedure. Jensen provides details on how to generate photons for emission from a variety of commonly used light sources [2001]. If multiple

[1] A "photon" in the photon mapping technique actually corresponds to a collection of physical photons.

light sources are present, each time a photon is emitted, Russian roulette is used to choose the emitting light source. More photons can be shot from brighter light sources by basing the roulette probabilities on the power of the lights.

7.2.2 Photon Scattering

After a photon is generated for emission at a light source it is traced through the scene. This process is encompassed in the TRACEPHOTON function and proceeds analogously to tracing rays. When a photon intersects a surface it is either scattered or absorbed. Russian roulette is used to probabilistically choose which of these events occurs, and the surface's material properties determine the corresponding probabilities. The TRACEPHOTON function terminates once the photon is absorbed or it propagates out of the scene. If reflection or transmission occur, the photon is scattered according to the BRDF. For perfect mirror surfaces, the scattered direction is the mirror reflection direction, whereas for perfect Lambertian surfaces the direction is computed according to a cosine distribution. It is possible to importance sample a scattering direction from a wide range of other BRDFs as well. If importance sampling the BRDF is not possible then a random scattering direction can be chosen, and the photon's power then needs to be scaled according to the BRDF.

7.2.3 Photon Storage

Whenever a photon hits a non-specular surface, it is stored in a global data structure called a photon map. Photons hitting perfectly specular surfaces are not stored since this form of illumination can be more effectively computed using Monte Carlo ray tracing during the rendering pass. During rendering, illumination information is queried by performing range searches within the photon map. The photons are inserted into a balanced kd-tree acceleration structure in order to perform these range searches efficiently.

7.2.4 Importance-Driven Photon Mapping

The photon tracing procedure described so far is view-independent. This means that, if the camera is aimed at a relatively small portion of the scene, many photons may be wasted

in “unimportant” regions of the scene. Peter and Pietrek [1998] used the concept of importance to concentrate photons in the parts of the scene where they contribute most to the final image. Before tracing photons, they emitted importance particles (or “importons”) from the camera and stored these in an importance map as they scattered at non-specular surfaces. The importance map can be used to guide the reflection of photons [Peter and Pietrek, 1998], or to determine the probability of storing each photon [Suykens and Willems, 2000; Keller and Wald, 2000]. These approaches can dramatically reduce the required number of stored photons; however, the number of emitted photons remains high. Peter and Pietrek [1998] and Christensen [Jensen et al., 2001a] suggested ways to address this by using a set of “test” photons to determine important emission directions; however, the robustness of these techniques has not been well tested in practice. Christensen [2003] provides an in-depth survey of importance within computer graphics.

7.3 Radiance Estimation

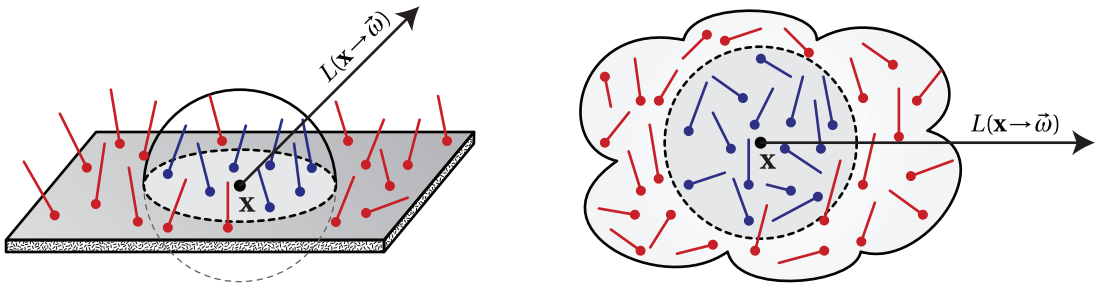
After photon tracing finishes, the collection of stored photons represents the incident illumination (flux) on surfaces. Each photon represents a fraction of the flux emitted either directly by a light source or indirectly through intermediate scattering events.

Reflected radiance at a surface is computed used Equation 2.14:

$$L(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega} f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, L(\mathbf{x}\leftarrow\vec{\omega}')\, (\vec{\mathbf{n}}\cdot\vec{\omega}')\, d\vec{\omega}'. \tag{7.1}$$

We can relate this to the flux stored in the photon map using Equation 2.2, which expresses the relationship between radiance and flux:

$$L(\mathbf{x},\vec{\omega}) = \frac{d^2\Phi(\mathbf{x},\vec{\omega})}{(\vec{\mathbf{n}}\cdot\vec{\omega})\, d\vec{\omega}\, d\mathcal{A}(\mathbf{x})}. \tag{7.2}$$



Radiance Estimate Volumetric Radiance Estimate

Figure 7.1: The radiance estimate (Equation 7.5) and the volumetric radiance estimate (Equation 7.10) compute outgoing radiance using density estimation by finding the nearest k photons to the query location $\mathbf{x}$. On surfaces, the contribution from each photon is weighted by the BRDF, accumulated, and divided by the *projected surface area* of the bounding sphere. Within participating media, each photon is instead weighted by the phase function, accumulated, and the result is divided by the *volume* of the bounding sphere.

By combining these expressions we arrive at:

$$L(\mathbf{x}\rightarrow\vec{\omega}) = \int_\Omega f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\,\frac{d^2\Phi(\mathbf{x},\vec{\omega}')}{(\vec{\mathbf{n}}\cdot\vec{\omega}')\,d\vec{\omega}'\,d\mathcal{A}(\mathbf{x})}\,(\vec{\mathbf{n}}\cdot\vec{\omega}')\,d\vec{\omega}', \tag{7.3}$$

$$= \int_\Omega f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\,\frac{d^2\Phi(\mathbf{x},\vec{\omega}')}{d\vec{\omega}'\,d\mathcal{A}(\mathbf{x})}\,d\vec{\omega}'. \tag{7.4}$$

This integral can be approximated using the photon map. Since each photon p carries flux Φ_p, the local density of the photons can be used to estimate the above integral. The density is computed by adding up the flux of the nearest photons in a small local region and dividing by the projected area $\mathcal{A}$ of the sphere containing these photons (see Figure 7.1). The reflected radiance at a point $\mathbf{x}$ in direction $\vec{\omega}$ can be estimated using the *radiance estimate* as

$$L(\mathbf{x}\rightarrow\vec{\omega}) \approx \sum_{p=1}^{k} f_r(\mathbf{x},-\vec{\omega}_p\leftrightarrow\vec{\omega})\,\frac{\Phi_p(\mathbf{x}\leftarrow-\vec{\omega}_p)}{\mathcal{A}(\mathbf{x})}, \tag{7.5}$$

where $\mathcal{A}(\mathbf{x}) = \pi r(\mathbf{x})^2$ is a surface area of a small region around $\mathbf{x}$.

This density estimation can be computed in a number of ways. The simplest approach sets r to a constant value over the whole scene. This approach would involve finding all photons p within a constant radius r sphere from the shading location $\mathbf{x}$ and applying the above expression. Since photons lie on surfaces, the surface area $\mathcal{A}(\mathbf{x})$ corresponds to projecting the sphere onto the surface and using the area of the resulting circle, $\mathcal{A}(\mathbf{x}) = \pi r^2$. The drawback of this simple

approach is that a single radius may not be well suited for all regions of the scene. Jensen instead proposed using the nearest neighbor method to estimate the local density. The nearest neighbor method finds the nearest k photons from the query location $\mathbf{x}$ and sets the radius equal to the distance to the k^{th} photon from $\mathbf{x}$, $r(\mathbf{x}) = d_k(\mathbf{x})$. With this approach the search radius is allowed to adapt to the local density of photons.

The radiance estimate in Equation 7.5 can be further improved in a number of ways. A typical improvement is to weight the contribution of photons differently within the search radius. Weighting the photon contributions with a smooth filtering kernel smooths the results while reducing overblurring. This technique is called the generalized nearest neighbor method within the density estimation literature [Silverman, 1986]. We review density estimation in more detail in Appendix C.

7.4 Participating Media

In reality, light interacts not only with surfaces but also scatters and gets absorbed by the medium between surfaces. Jensen and Christensen extended the photon mapping method to simulate global illumination within participating media [1998]. This extension requires modification to both the photon tracing and the rendering portion of the photon mapping algorithm.

7.4.1 Photon Tracing

In the presence of participating media, when a photon is emitted or reflected off a surface, it travels through the medium for some distance until it is either scattered or absorbed. As we saw in Chapter 4, the transmittance T_r describes the reduction of radiance as it travels through the medium due to absorption and out-scattering. Photon tracing can be augmented to propagate and store photons within the participating medium by probabilistically choosing the distance to the next interaction. The photon is advanced through the medium by this distance, at which point an interaction with the medium is simulated. At each interaction location, the photon is stored in a volume photon map.

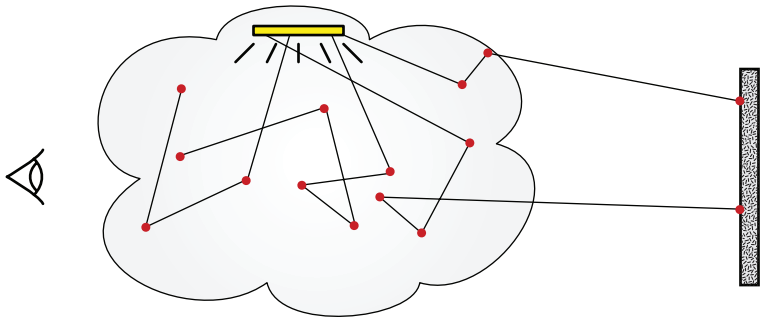


Figure 7.2: Photon mapping starts by emitting photons from light sources. In the presence of participating media, photons are stored not only on surfaces, but also within the medium.

Sampling Propagation Distance

Recall from Chapter 4 that the transmittance T_r actually gives the *probability* that a photon can travel unobstructed between two points in a medium. The propagation distance should therefore be computed by importance sampling the transmittance term. Given a random number $\xi \in (0,1]$, this can be accomplished with the following expression for d:

$$d(\mathbf{x}_p) = -\frac{\log(\xi)}{\bar{\sigma}_t(\mathbf{x}_p)}, \tag{7.6}$$

where the corresponding probability density is

$$pdf(d) = e^{-\bar{\sigma}_t(\mathbf{x}_p)d}. \tag{7.7}$$

In defining the above two equations, we introduced the term $\bar{\sigma}_t(\mathbf{x}_p)$. We have the freedom to define this parameter as any positive value; however, our goal is to use a value which results in a distance distribution proportional to the transmittance term T_r.

Homogeneous Media. For homogeneous media, if we set $\bar{\sigma}_t(\mathbf{x}_p)$ to the medium's extinction coefficient, $\bar{\sigma}_t(\mathbf{x}_p) = \sigma_t$, then the PDF above will be exactly proportional to the transmittance.

Heterogeneous Media. Distributing the distance according to transmittance is much more difficult in heterogeneous media and in general can only be approximated. The optimal solution would set $\bar{\sigma}_t(\mathbf{x}_p)$ to the mean extinction coefficient between $\mathbf{x}_p$ and the next interaction event

Algorithm 7.3: VOLUMETRICPHOTONTRACING($\mathbf{x}_p, \vec{\omega}_p, \Phi_p$)

```
d = -log(ξ) / σ̄_t(x_p);
pdf(d) = e^{-σ̄_t(x_p) d};
while no surface between x_p and x_p + d ω⃗_p do
    Φ_p *= e^{-σ_t(x_p) d} / pdf(d);
    x_p += d ω⃗_p;
    STOREPHOTON(x_p, ω⃗_p, Φ_p);
    if ξ < σ_s / σ_t then
        return;
    ω⃗_p = COMPUTESCATTEREDDIRECTION(x_p, ω⃗_p);
    d = -log(ξ) / σ̄_t(x_p);
    pdf(d) = e^{-σ̄_t(x_p) d};
return;
```

at $\mathbf{x}_p + d\vec{\omega}$. Clearly this is not a practical approach since it requires knowledge of the distance d, which we are trying to compute in the first place! One approximate solution is to set the parameter to the extinction coefficient at the current scattering event, $\bar{\sigma}_t(\mathbf{x}_p) = \sigma_t(\mathbf{x}_p)$. In highly heterogeneous media this can lead to fairly erratic distance values, and care must be taken to ensure that $\bar{\sigma}_t > 0$. Another option is to compute the average extinction coefficient over the entire medium and use this as the value of $\bar{\sigma}_t(\mathbf{x}_p)$ for all $\mathbf{x}_p$. A more accurate, but potentially costly, approach is to choose the random value ξ and then incrementally compute T_r by taking small steps along $\vec{\omega}$ using ray marching. The next scattering event will occur where $T_r = \xi$.

Volumetric Scattering

At an interaction location, the photon can be either absorbed or scattered, and Russian roulette is used to choose between these two events. The ratio between the absorption and extinction coefficients describes the probability of absorption:

$$Pr\{\text{absorb}\} = \frac{\sigma_a}{\sigma_t}. \tag{7.8}$$

If the photon is not absorbed, it will be scattered into another direction through the medium. This direction can be computed by importance sampling the phase function.

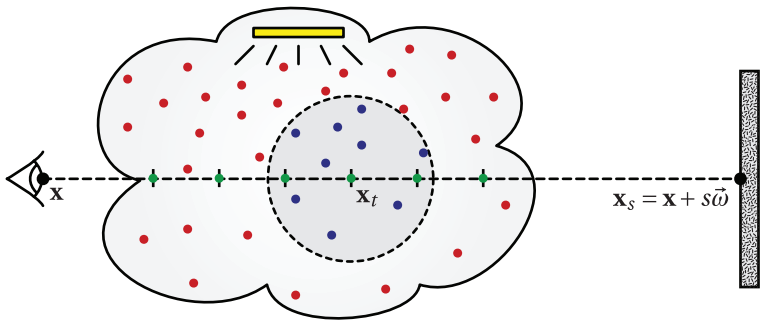


Figure 7.3: Volumetric photon mapping uses ray marching to accumulate in-scattered radiance along the length of an eye ray through the medium. At each discrete sample point the nearest photons are gathered, and the in-scattered radiance is approximated using the volumetric radiance estimate.

The volumetric photon tracing process is illustrated in Figure 7.2 and summarized in Algorithm 7.3.

7.4.2 Ray Marching and the Volumetric Radiance Estimate

During rendering, volumetric photon mapping uses ray marching (Equation 5.1) to numerically integrate the volume rendering equation for radiance seen directly by the observer:

$$L(\mathbf{x} \leftarrow \vec{\omega}) \approx T_r(\mathbf{x} \leftrightarrow \mathbf{x}_s) L(\mathbf{x}_s \rightarrow -\vec{\omega}) + \left(\sum_{t=0}^{S-1} T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t) \sigma_s(\mathbf{x}_t) L_i(\mathbf{x}_t \rightarrow -\vec{\omega}) \Delta_t \right). \tag{7.9}$$

This is illustrated in Figure 7.3. The most expensive part to compute in Equation 7.9 is the in-scattered radiance L_i, because it involves accounting for all light arriving at each point $\mathbf{x}_t$ along the ray from any other point in the scene. Instead of computing these values independently for each location, volumetric photon mapping gains efficiency by reusing the computation performed during the photon tracing stage. The in-scattered radiance is approximated using the *volumetric radiance estimate* by gathering photons within a small spherical neighborhood of radius r around each sample location $\mathbf{x}_t$,

$$L_i(\mathbf{x}_t, \vec{\omega}) \approx \sum_{p=1}^{k} p(\mathbf{x}_t, \vec{\omega} \leftrightarrow \vec{\omega}_p) \frac{\Phi_p(\mathbf{x} \rightarrow -\vec{\omega}_p)}{\mathcal{V}(\mathbf{x})}, \tag{7.10}$$

where $\mathcal{V}(\mathbf{x}) = \frac{4}{3}\pi d_k(\mathbf{x})^3$ is the volume of the sphere containing the nearest k photons. This is known as the *volume radiance estimate* and is illustrated in Figure 7.1.

8

The Beam Radiance Estimate

"Scattering is easier than gathering."

—Irish Proverb

THE volumetric photon mapping [Jensen and Christensen, 1998] technique described in the previous chapter can efficiently simulate scattering in participating media without making simplifying assumptions about the properties of the medium being rendered. Similar to the volumetric radiance caching technique developed in Chapter 5, photon mapping handles isotropic, anisotropic, homogeneous, and heterogeneous media of arbitrary albedo. Photon mapping gains efficiency by reusing a small collection of photons to estimate in-scattered radiance at all locations in the scene using density estimation.

Just like in volumetric radiance caching, a ray marching process is used to integrate the contribution of radiance directly seen by the camera. In volumetric photon mapping, however, the radiance estimate, which is evaluated at each step during ray marching, requires costly range queries within the photon map. Minimizing the number of queries is desirable; however, if not enough sample points are used, the result is likely to be noisy. On the other hand, increasing the number of sample points is very costly and can slow down rendering significantly. Even with a fixed number of samples, finding an optimal distribution of sample points along the ray is difficult. Moreover, the ray marching formulation is suboptimal, firstly because it may gather the same photons more than once if the spherical neighborhoods overlap and, secondly, because it can lead to noise if the step size is too large and photons are omitted (see Figure 8.1).

In this chapter we develop a novel radiance estimate technique for participating me-

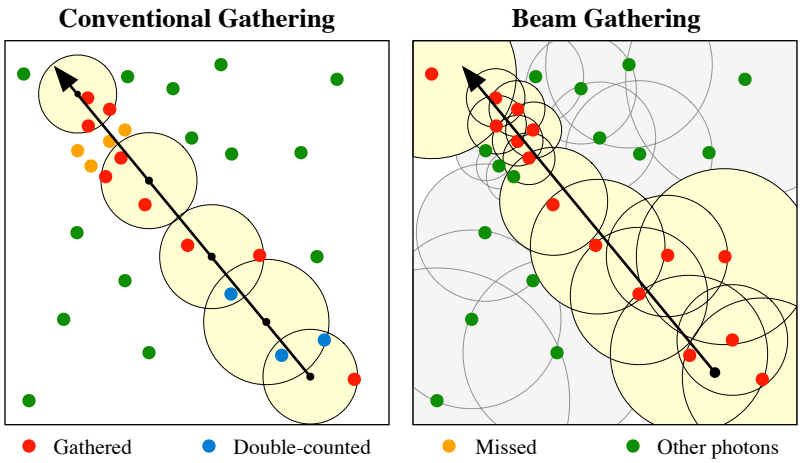


Figure 8.1: Conventional gathering (left) searches for photons in a sphere around numerous samples along the ray. This is inefficient because it can double-count photons if the searches overlap (blue) and it can miss important photons (orange) if the step-size becomes too large. The method described in this chapter (right) assigns a radius to each photon and performs a single range search to find all photons along the length of the entire ray.

dia, which eliminates this problem. To accomplish this, we use a theoretical reformulation of volumetric photon mapping. The technique developed in this chapter replaces the multiple point-queries performed during ray marching with a single beam-query, which explicitly gathers all photons along the length of an entire ray. These photons are used to estimate the accumulated in-scattered radiance arriving from a particular direction and need to be gathered only once per ray. This method handles both fixed and adaptive kernels, is significantly faster than conventional volumetric photon mapping, and produces images with less noise.

8.1 Contributions

In this chapter, we propose a novel approach for computing the contribution of in-scattered radiance. We gather photons along viewing rays and analytically compute their contributions, without point sampling. We present the following contributions:

- We combine the theory from Veach [1997] and Pauly et al. [2000] to derive a reformulation of volumetric photon mapping as a solution to the measurement equation. This theory allows for arbitrary *measurements* of radiance to be computed within participating media, where a measurement is simply an integral of the radiance multiplied with a weighting

function.

- Using this new theory, we present an improved radiance estimate for volumetric photon mapping based on "beam gathering." This technique eliminates the need for stepping through the medium to find photons. Instead, it gathers all photons along a ray. We show how to efficiently implement this new gathering technique for both fixed and adaptive smoothing kernels and demonstrate that our method produces images with less noise than conventional photon mapping.

The rest of this chapter is organized as follows. In Section 8.2, we reformulate volumetric photon mapping in terms of the measurement equation and show how the photon map can be used to estimate *any* measurement of radiance within the scene. In Section 8.3, we present our new beam radiance estimate using this theory and describe the data structures needed to evaluate it efficiently. Finally, we show comparisons of our approach to conventional photon mapping in Section 8.4 and discuss avenues of future work in Section 8.5.

8.2 Reformulation of Volumetric Photon Mapping

Our technique queries once for photons along the length of an entire ray, instead of multiple times near points along the ray (see Figure 8.1). More formally, whereas regular photon mapping estimates L_i at discrete points using Equation 7.10, our main contribution is to directly estimate

$$\int_0^s T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t)\, \sigma_s(\mathbf{x}_t)\, L_i(\mathbf{x}_t \rightarrow \vec{\omega})\, dt \tag{8.1}$$

along rays.

Though the explanation of photon mapping from the previous chapter is appealing at an intuitive level, it does not rigorously present the algorithm as a numerical solution to the RTE. Furthermore, this explanation is heavily tied to the geometric interpretation of gathering photons within a disc (on surfaces) or within a sphere (in participating media). In order to avoid these limitations and use the photon map to estimate general radiometric quantities in the volume,

such as Equation 8.1, we use a more flexible derivation of particle tracing methods presented by Veach [1997]. We extend this derivation to handle participating media by combining it with the generalized path integral formulation of the radiative transport equation [Pauly et al., 2000] (Section 8.2.1) and show how to represent particle tracing algorithms like volumetric photon mapping in terms of the measurement equation (Sections 8.2.2 and 8.2.3). Finally, we show how to use the same photon maps to estimate more general quantities of radiance (Section 8.2.4).

8.2.1 Generalized Path Integral Formulation

We use the path integral formulation of the RTE, which arises by recursively expanding the right hand side of Equation 4.27. Instead of expressing the radiance equilibrium *recursively*, the resulting path integral formulation is a *sum* over light-carrying paths of different lengths. In order to do this, we define the path space, the corresponding differential measure, and generalized radiometric terms using notation inspired by Pauly et al. [2000].

A light path $\bar{\mathbf{x}}_k^l$ is a set of $k+1$ vertices $\mathbf{x}_i$. Each path is classified according to its *path characteristic* $l \in \mathbb{N}$, which determines for each vertex whether it is in the volume or on a surface. This allows us to integrate over different measures for scattering events at surfaces and within the volume. We define the path characteristic l of a path $\bar{\mathbf{x}}_k^l$ such that the i$^{\text{th}}$ bit of l, $b_i(l)$, equals 1 if vertex $\mathbf{x}_i$ is on a surface and $b_i(l) = 0$ if it is in the volume. The space of all paths of length k with characteristic l is therefore:

$$\mathbf{X}_k^l = \left\{ \bar{\mathbf{x}}_k^l = \mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k \,\middle|\, \mathbf{x}_i \in \begin{cases} \mathcal{V}, & \text{if } b_i(l) = 0 \\ \mathcal{A}, & \text{if } b_i(l) = 1 \end{cases} \right\}, \tag{8.2}$$

for $1 \le k < \infty$ and $0 \le l < 2^{k+1}$, and where $\mathcal{V}$ and $\mathcal{A}$ are the media volume and surface area, respectively (see Figure 8.2 for an illustration of this notation). We define the corresponding differential measure at a path vertex $\mathbf{x}_i$ as:

$$d\mu^l(\mathbf{x}_i) = \begin{cases} d\mathcal{V}(\mathbf{x}_i), & \text{if } \mathbf{x}_i \in \mathcal{V}, \text{ i.e. when } b_i(l) = 0 \\ d\mathcal{A}(\mathbf{x}_i), & \text{if } \mathbf{x}_i \in \mathcal{A}, \text{ i.e. when } b_i(l) = 1 \end{cases}. \tag{8.3}$$

Additionally, in order to express Equation 4.27 in terms of paths we need to transform the

integration domain from solid angle (Ω and $\Omega_{4\pi}$), to area or volume ($\mathcal{A}$ and $\mathcal{V}$) depending on the type of scattering event. This transformation is achieved using the generalized geometry term:

$$\hat{G}(\mathbf{x}\leftrightarrow\mathbf{y}) = \frac{V(\mathbf{x}\leftrightarrow\mathbf{y})D_{\mathbf{x}}(\mathbf{y})D_{\mathbf{y}}(\mathbf{x})\sigma(\mathbf{x})T_r(\mathbf{x}\leftrightarrow\mathbf{y})}{\left\|\mathbf{x}-\mathbf{y}\right\|^2} \tag{8.4}$$

where

$$D_{\mathbf{x}}(\mathbf{y}) = \begin{cases} 1, & \text{if } \mathbf{x}\in\mathcal{V} \\ \vec{\mathbf{n}}(\mathbf{x})\cdot\vec{\omega}_{\mathbf{xy}}, & \text{if } \mathbf{x}\in\mathcal{A} \end{cases}. \tag{8.5}$$

We define $\vec{\omega}_{\mathbf{xy}}$ to be the unit direction vector from $\mathbf{x}$ to $\mathbf{y}$, and $D_{\mathbf{y}}(\mathbf{x})$ is defined symmetrically to $D_{\mathbf{x}}(\mathbf{y})$ for both cases. The visibility function, V, is defined in Equation 2.19[1]. The $\sigma(\mathbf{x})$ function returns the scattering coefficient $\sigma_s(\mathbf{x})$ if $\mathbf{x}\in\mathcal{V}$ and 1 otherwise. Note that Equation 8.4 simplifies to the regular geometry term if no participating media is present.

Similarly, we generalize scattering events and emitted radiance. We define $\hat{f}$ to be the generalized scattering function combining the phase function and the surface BRDF

$$\hat{f}(\mathbf{x}_i) = \begin{cases} p(\mathbf{x}_{i+1}\rightarrow\mathbf{x}_i\rightarrow\mathbf{x}_{i-1}), & \text{if } \mathbf{x}_i\in\mathcal{V} \\ f_r(\mathbf{x}_{i+1}\rightarrow\mathbf{x}_i\rightarrow\mathbf{x}_{i-1}), & \text{if } \mathbf{x}_i\in\mathcal{A} \end{cases}, \tag{8.6}$$

and $\hat{L}_e$ is the generalized emitted radiance

$$\hat{L}_e(\mathbf{x}_i\rightarrow\mathbf{x}_{i-1}) = \begin{cases} \frac{\sigma_a(\mathbf{x}_i)}{\sigma_s(\mathbf{x}_i)}L_e(\mathbf{x}_i\rightarrow\mathbf{x}_{i-1}), & \mathbf{x}_i\in\mathcal{V} \\ L_e(\mathbf{x}_i\rightarrow\mathbf{x}_{i-1}), & \mathbf{x}_i\in\mathcal{A} \end{cases}, \tag{8.7}$$

where we multiply by $\frac{\sigma_a}{\sigma_s}$ because the emitted radiance in a volume needs to be multiplied by the absorption, not the scattering, coefficient.

Given this notation, the generalized path integral formulation expresses the outgoing radiance at $\mathbf{x}_1$ towards $\mathbf{x}_0$ as a sum over all possible light paths arriving at $\mathbf{x}_1$. This includes light paths of all lengths k, as well as all possible characteristics l for each length

$$L(\mathbf{x}_1\rightarrow\mathbf{x}_0) = \sum_{k=1}^{\infty}\sum_{l=0}^{2^{k+1}-1}\bar{L}(\bar{\mathbf{x}}_k^l). \tag{8.8}$$

[1] Note that unlike G, to simplify notation we include the visibility function in $\hat{G}$.

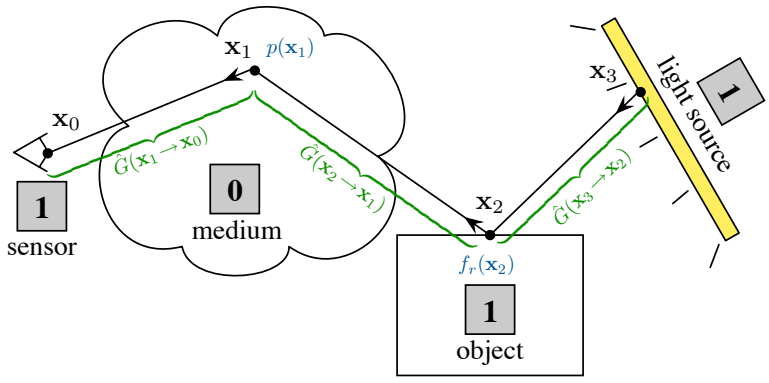


Figure 8.2: An example path, $\bar{\mathbf{x}}_3^{13}$, with length $k = 3$. The path characteristic, $l = 1101_b = 13$, concatenates the type of scattering event from each path vertex. The radiance transported along the path $\bar{L}(\bar{\mathbf{x}}_3^{13})$ is the emitted radiance at the light multiplied by a series of scattering events (blue) and geometry terms (green).

$\bar{L}(\bar{\mathbf{x}}_k^l)$ measures the amount of radiance transported along a path $\bar{\mathbf{x}}_k^l$ and is defined as

$$\bar{L}(\bar{\mathbf{x}}_k^l) = \int_{\mu^l(\mathbf{x}_k)} \cdots \int_{\mu^l(\mathbf{x}_2)} \hat{L}_e(\mathbf{x}_k \rightarrow \mathbf{x}_{k-1}) \left(\prod_{j=1}^{k-1} \hat{f}(\mathbf{x}_j) \hat{G}(\mathbf{x}_{j+1} \leftrightarrow \mathbf{x}_j) \right) d\mu^l(\mathbf{x}_2) \cdots d\mu^l(\mathbf{x}_k). \tag{8.9}$$

The radiance transported along an example path is shown in Figure 8.2.

8.2.2 The Measurement Equation

Many global illumination algorithms can be described in terms of the *measurement equation*. The measurement equation describes an abstract measurement of incident radiance taken over some set of rays in a scene:

$$I = \langle W_e, L \rangle = \int_{\mathcal{V}} \int_{\Omega_{4\pi}} W_e(\mathbf{x} \rightarrow \vec{\omega}) \, L(\mathbf{x} \leftarrow \vec{\omega}) \, d\vec{\omega} \, d\mathcal{V}(\mathbf{x}). \tag{8.10}$$

The importance function W_e represents an abstract measuring sensor and is defined over the whole ray space $\mathcal{V} \times \Omega_{4\pi}$ (though typically W_e is non-zero for only a small subset of this domain).

Path tracing, for instance, measures the contribution of radiance arriving over the area of a pixel. Radiosity algorithms integrate the contribution of radiance over basis functions defined on the scene geometry. Both of these approaches can be described using Equation 8.10 with an appropriate importance function.

In his dissertation, Veach [1997] showed how particle tracing methods for surface illu-

mination can also be expressed as a solution to the measurement equation by using the path integral form of the rendering equation. We extend this idea and use the generalized path integral formulation to describe volumetric photon tracing in the same way.

8.2.3 Volumetric Photon Tracing

Photon tracing methods can be thought of as a way of generating samples from the scene's equilibrium radiance distribution and then using this single collection of samples to render the entire image. The photon tracing stage generates N weighted sample rays, or photons, $(\alpha_i, \mathbf{x}_i, \vec{\omega}_i)$, where each $(\mathbf{x}_i, \vec{\omega}_i)$ is a ray and α_i is a corresponding weight. Our goal is to use these samples to take *unbiased* estimates of the radiance as a weighted sum,

$$E\left[\frac{1}{N}\sum_{i=1}^{N} W_e(\mathbf{x}_i \rightarrow \vec{\omega}_i)\,\alpha_i\right] = \langle W_e, L\rangle, \tag{8.11}$$

for an arbitrary importance function W_e. We must therefore determine the proper distribution of samples for Equation 8.11 to hold.

We start by manipulating the measurement equation on the right-hand side. To express the measurement equation using paths, we expand Equation 8.10 in terms of the *outgoing* radiance and convert to area form, introducing an additional geometry term,

$$\langle W_e, L\rangle = \int_{\mu^l(\mathbf{x}_1)} \int_{\mu^l(\mathbf{x}_0)} W_e(\mathbf{x}_0 \rightarrow \mathbf{x}_1)\, L(\mathbf{x}_1 \rightarrow \mathbf{x}_0)\, \hat{G}(\mathbf{x}_1 \leftrightarrow \mathbf{x}_0)\, d\mu^l(\mathbf{x}_0)\, d\mu^l(\mathbf{x}_1). \tag{8.12}$$

By combining with Equations 8.8 and 8.9 and moving the summations outside the integrals, this becomes

$$\sum_{k=1}^{\infty} \sum_{l=0}^{2^{k+1}-1} \left[\int_{\mu^l(\mathbf{x}_k)} \cdots \int_{\mu^l(\mathbf{x}_0)} W_e(\mathbf{x}_0 \rightarrow \mathbf{x}_1)\, \hat{L}_e(\mathbf{x}_k \rightarrow \mathbf{x}_{k-1}) \left(\prod_{j=1}^{k-1} \hat{f}(\mathbf{x}_j)\, \hat{G}(\mathbf{x}_{j+1} \leftrightarrow \mathbf{x}_j) \right) \hat{G}(\mathbf{x}_1 \leftrightarrow \mathbf{x}_0) d\mu^l(\mathbf{x}_0) \cdots d\mu^l(\mathbf{x}_k) \right].$$

The Monte Carlo estimator for this expression using N samples is

$$E\left[\frac{1}{N}\sum_{i=1}^{N} W_e(\mathbf{x}_{i,0} \rightarrow \mathbf{x}_{i,1}) R_i\right], \tag{8.13}$$

where each R_i is a random-walk path of length k_i generated using Russian roulette,

$$R_i = \frac{\hat{L}_e(\mathbf{x}_{i,k_i} \rightarrow \mathbf{x}_{i,k_{i-1}})}{pdf(\mathbf{x}_{i,k_i}, \mathbf{x}_{i,k_i-1})} \prod_{j=1}^{k_i-1} \left(\frac{1}{q_{i,j}} \frac{\hat{f}(\mathbf{x}_{i,j})\, \hat{G}(\mathbf{x}_{i,j+1} \leftrightarrow \mathbf{x}_{i,j})}{pdf(\mathbf{x}_{i,j-1})} \right) \hat{G}(\mathbf{x}_{i,1} \leftrightarrow \mathbf{x}_{i,0}), \tag{8.14}$$

and $q_{i,j}$ was the probability of terminating R_i at the j^{th} vertex. Comparing Equation 8.13 with 8.11 we see that in order to satisfy the requirements we need to set $\alpha_i = R_i$.

Connection to Conventional Photon Tracing. Though derived in a different fashion, Equation 8.14 is exactly how conventional photon mapping distributes photons within the scene. For instance, for a diffuse area light, photons are emitted using a cosine distribution with the power of the light source. In Equation 8.14, photons are emitted with the radiance of the light source divided by the pdf of choosing a position and direction on the light. These quantities are equivalent. Hence the particles generated above represent differential flux. The correspondence between the photon powers [Jensen and Christensen, 1998] and the sample weights is $\Delta\Phi_i = \frac{\alpha_i}{N}$.

8.2.4 Radiance Estimation Using the Measurement Equation

The main advantage of the reformulation in Section 8.2.3 is that it naturally accommodates computation of any measurement of radiance within the scene simply by using an appropriately defined importance function W_e. In this section, we first show how the conventional estimate for in-scattered radiance can be expressed as a measurement. We then go one step further and show how to derive a beam radiance estimate which approximates Equation 8.1 along rays directly.

Conventional Radiance Estimate. The conventional radiance estimate approximates the value of the in-scattered radiance L_i at fixed points within the scene. To express this using the theory from Section 8.2, we need to transform Equation 4.13 into the measurement equation. Since the measurement equation is an integral over all of ray space ($\mathcal{V} \times \Omega_{4\pi}$), we artificially expand L_i to

also integrate over the volume

$$L_i(\mathbf{x}_t \to \vec{\omega}) = \int_{\Omega_{4\pi}} p(\mathbf{x}_t, \vec{\omega} \leftrightarrow \vec{\omega}_t) L(\mathbf{x}_t \leftarrow \vec{\omega}_t) d\vec{\omega}_t \tag{8.15}$$
$$= \int_{\mathcal{V}} \int_{\Omega_{4\pi}} \delta(\|\mathbf{x}' - \mathbf{x}_t\|)\, p(\mathbf{x}', \vec{\omega} \leftrightarrow \vec{\omega}_t)\, L(\mathbf{x}_t \leftarrow \vec{\omega}_t) d\vec{\omega}_t d\mathcal{V}(\mathbf{x}').$$

In order to keep the expressions equivalent when we add the integration over volume, we also introduce a Dirac delta function δ.

The bottom row of the above equation is now the measurement equation, where $W_e = \delta(\|\mathbf{x}' - \mathbf{x}_t\|)\, p(\mathbf{x}', \vec{\omega} \leftrightarrow \vec{\omega}_t)$. Hence we can compute an *unbiased* estimate using the photon map by evaluating Equation 8.11 with this importance function. However, in order to obtain a useful estimate of radiance at all points in the scene, a normalized kernel function is used in place of the delta function. This is where bias is introduced in the photon mapping method. Another interpretation is that by replacing the delta function with a kernel, photon mapping computes an *unbiased* estimate of *blurred* radiance. Jensen and Christensen [1998] use a constant three-dimensional kernel with a radius based on the k^{th} nearest neighbor. This results in the following radiance estimate by applying Equation 8.11

$$L_i(\mathbf{x}_t \to \vec{\omega}) \approx \int_{\mathcal{V}} \int_{\Omega_{4\pi}} K_t(\|\mathbf{x}' - \mathbf{x}_t\|)\, p(\mathbf{x}', \vec{\omega} \leftrightarrow \vec{\omega}_t)\, L(\mathbf{x}_t \leftarrow \vec{\omega}_t) d\vec{\omega}_t d\mathcal{V}(\mathbf{x}'), \tag{8.16}$$
$$\approx \frac{1}{N} \sum_{i=1}^{N} K_t(\|\mathbf{x}_i - \mathbf{x}_t\|)\, p(\mathbf{x}_i, \vec{\omega} \leftrightarrow \vec{\omega}_i) \alpha_i \tag{8.17}$$

where the kernel K_t is defined as

$$K_t(r) = \begin{cases} \frac{3}{4\pi d_k(\mathbf{x}_t)^3} & \text{if } r \in [0, d_k(\mathbf{x}_t)] \\ 0 & \text{otherwise} \end{cases}, \tag{8.18}$$

and $d_k(\mathbf{x}_t)$ is the distance from $\mathbf{x}_t$ to the k^{th} nearest photon. Note that this is equivalent to the conventional volumetric radiance estimate in Equation 7.10.

Beam Radiance Estimate. A similar procedure can be used to derive an estimate for the accumulated in-scattered radiance along an entire ray. To accomplish this, we first expand out L_i

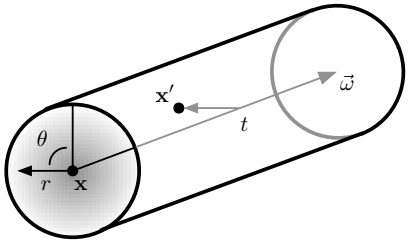


Figure 8.3: In the beam radiance estimate, $\mathbf{x}'$ is parametrized in cylindrical coordinates, (t,θ,r), about the ray $(\mathbf{x},\vec{\omega})$. An unbiased estimate would only consider points directly on the ray, while a biased version uses a kernel (shown in grey) to blur the radiance within a beam.

in Equation 8.1 and then artificially inflate the resulting expression to integrate over the whole volume:

$$\int_0^s \int_{\Omega_{4\pi}} T_r(\mathbf{x}\leftrightarrow\mathbf{x}_t)\,\sigma_s(\mathbf{x}_t)\,p(\mathbf{x}_t,\vec{\omega}\leftrightarrow\vec{\omega}_t)\,L(\mathbf{x}_t\leftarrow\vec{\omega}_t)\,d\vec{\omega}_t\,dt = \tag{8.19}$$

$$\int_{\mathbb{R}}\int_0^{2\pi}\int_{\mathbb{R}}\int_{\Omega_{4\pi}} \delta(r)\,(H(t)-H(t-s))\,T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\,\sigma_s(\mathbf{x}')\,p(\mathbf{x}',\vec{\omega}\leftrightarrow\vec{\omega}_t)\,L(\mathbf{x}'\leftarrow\vec{\omega}_t)\,d\vec{\omega}_t\,dr\,d\theta\,dt. \tag{8.20}$$

$\mathbb{R}$ is the set of real numbers and $\mathbf{x}'$ is expressed in cylindrical coordinates, (t,θ,r), about $(\mathbf{x},\vec{\omega})$, where r is the radius to the ray (see Figure 8.3). We have added a Dirac delta function δ as before, and the Heaviside step functions ($H(x) = 1$ when $x > 0$ and 0 otherwise) constrain the computation to $t \in (0,s)$. Equation 8.20 is now equivalent to the measurement equation, where the integral over volume has been converted into cylindrical coordinates and where $W_e = \delta(r)(H(t) - H(t-s))\,T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\sigma_s(\mathbf{x}')\,p(\mathbf{x}',\vec{\omega}\leftrightarrow\vec{\omega}_t)$.

Since the probability of photons landing exactly on the ray $(\mathbf{x},\vec{\omega})$ is zero, we introduce bias by blurring the radiance and replacing the delta and step functions with a smooth kernel, K. This integral can then be estimated with the measurement equation using the photons as

$$\int_{\mathbb{R}}\int_0^{2\pi}\int_{\mathbb{R}}\int_{\Omega_{4\pi}} K(t,\theta,r)\,T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\,\sigma_s(\mathbf{x}')\,p(\mathbf{x}',\vec{\omega}\leftrightarrow\vec{\omega}_t)\,L(\mathbf{x}'\leftarrow\vec{\omega}_t)\,d\vec{\omega}_t\,dr\,d\theta\,dt =$$
$$\frac{1}{N}\sum_{i=1}^{N} K(t_i,\theta_i,r_i)\,T_r(\mathbf{x}\leftrightarrow\mathbf{x}_i)\,\sigma_s(\mathbf{x}_i)\,p(\mathbf{x}_i,\vec{\omega}\leftrightarrow\vec{\omega}_i)\,\alpha_i, \tag{8.21}$$

where $(t_i,\theta_i,r_i) = \mathbf{x}_i$ are the cylindrical coordinates of photon i about the ray.

The blurring in the conventional radiance estimate is spherical, and so the kernel needs

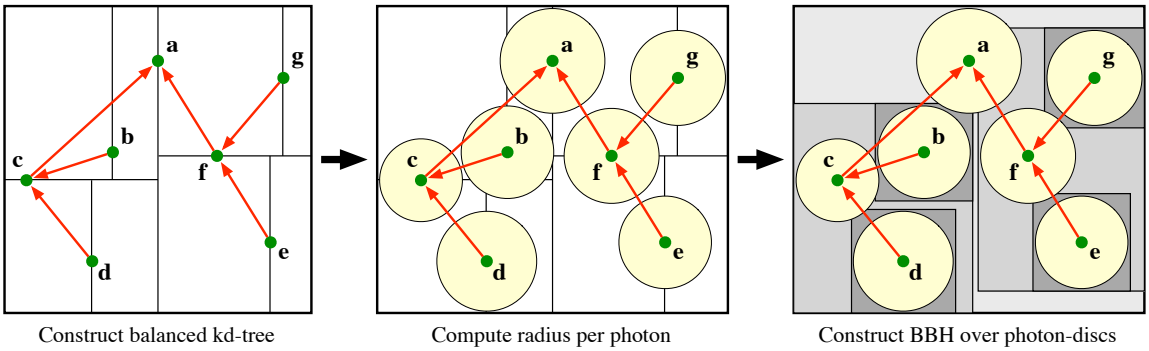


Figure 8.4: After photons have been distributed in the scene, our algorithm constructs a balanced kd-tree (left). We assign a valid radius to each photon by querying in the kd-tree (middle). Finally, we rapidly construct a bounding-box hierarchy over the photon-discs (right) by reusing the same hierarchical structure (shown in red) as the kd-tree.

to be normalized for 3D. However, with the beam radiance estimate, we blur in two dimensions (perpendicular to the ray) since the radiance we are computing already includes the integration along the ray itself. Therefore, the kernel in the beam estimate is normalized for 2D.

8.2.5 Kernel Radiance Estimation

For both the conventional and beam radiance estimates, the characteristics of the bias and blur are determined by the smoothing function chosen. Several options exist for applying a smoothing kernel to the photon map data.

The *kernel method* uses a fixed-radius smoothing kernel and results in a uniform blur of radiance within the scene. In practice, using a fixed-width circular kernel implies that in order to evaluate the beam radiance estimate (Equation 8.21) using the photon map (Equation 8.11) we only need to consider photons which are located within a fixed-radius cylinder about the ray $(\mathbf{x}, \vec{\omega})$. Alternatively, the equivalent dual interpretation considers each photon as the center of an oriented disc *facing the ray* and all *photon-discs* that intersect the ray need to be found.

If the density of photons varies significantly it can be difficult to choose a single radius that works well for all regions of the scene. This can be solved by allowing the size and shape of the blurring kernel to vary spatially. In conventional photon mapping, the k^{th} *nearest neighbor method* (k-NN) is used to adapt the kernel width to the local density. Generalizing point-based k-NN to a visually comparable range search along rays is challenging. However, spatial variation

can easily be applied to the dual *photon-discs* interpretation using the *variable kernel method* (VK) [Breiman et al., 1977]. A smoothing kernel is "attached" to each photon, and the radius of the kernel is allowed to vary from one photon to another, based on the local density. In contrast to k-NN estimation, where the kernel widths vary based on the distance from the *evaluation* location to the data points, in the VK method the kernel widths only depend on the data points themselves. In order to facilitate this, the method relies on a *pilot estimate* of the local density at each data point in order to assign the kernel widths. We review these and other density estimation methods in more detail in Appendix C. This is the approach we take.

8.3 Algorithm

In order to use the dual interpretation to evaluate the beam radiance estimate (Equation 8.21), we need an efficient way of locating all photon-discs that intersect an arbitrary ray. Additionally, to use variable width kernels we need to efficiently compute a radius for each photon in the photon map. At a high level, our volumetric photon mapping technique performs the five steps in Algorithm 8.1. Steps 1 and 2 are identical to conventional photon mapping, while 3–5 are unique to our approach and are explained in more detail below.

Photon Radius Computation. We augment the traditional photon map by associating a radius with each photon. For fixed width kernels the radius is a constant for all photons and does not need to be explicitly stored. For variable width kernels using the VK method, we perform a density estimation at each photon to assign a radius. At each photon we compute the local density by estimating the distance to the k^{th} nearest photon and use this as the photon-disc's radius. This pilot estimate is performed using the photon map kd-tree. The value k plays the same role as in the conventional radiance estimate: it controls the amount of blur.

As an optimization, we only search for the nearest $n \ll k$ photons and estimate the necessary radius for k photons. By assuming locally uniform photon density, if $d_n(\mathbf{x}_i)$ is the distance to the n^{th} photon from photon i, we estimate the radius as $r_i = d_n(\mathbf{x}_i)\sqrt[3]{\frac{k}{n}}$. The n parameter controls the sensitivity of the computed radius to the local variation in density. Very small values of n, $n < 5$, can produce noisy radii, which change drastically between neighboring

Algorithm 8.1: Beam photon mapping.

1 Shoot photons from light sources.
2 Construct a balanced kd-tree for the photons.
3 Assign a radius for each photon.
4 Construct a bounding-box hierarchy over the photon-discs.
5 Use the photon BBH to render the image.

photons, while large values are more expensive to compute. In practice, we have found that $n = \sqrt{k}$ works well as a default value, and this value was used for all our scenes, significantly accelerating the preprocessing step.

Bounding Box Hierarchy Construction. In order to efficiently locate all photons-discs which intersect a ray, we construct a bounding box hierarchy. Heuristics for constructing efficient BBHs have been extensively studied within the context of ray tracing [Wald et al., 2007]. However, the performance characteristics of our ray intersections are different than for regular ray tracing since we are interested in all intersections, not just the first intersection along a ray. Furthermore, the best heuristics tend to induce long construction times. We employ a rapid construction scheme by exploiting the information in the photon map kd-tree and reusing that hierarchy for our BBH.

For each photon in the photon map, we compute the bounding box of all descendant photon-discs. The bounding box of each node encloses the node's photon radius and the bounding boxes of its two child nodes. The computation starts at the leaves and propagates upwards through the tree. This procedure results in a balanced median-split-style BBH, but unlike traditional BBHs our hierarchy contains photons at interior nodes, not just at the leaves. Figure 8.4 illustrates the relationship between the kd-tree and the BBH. The BBH can be constructed by passing the root of the photon map kd-tree to Algorithm 8.2.

Given a balanced kd-tree, this linear-time construction procedure is extremely fast and produces BBHs which can be efficiently traversed for nearby photons during rendering. After the BBH is constructed the photon map kd-tree is no longer used and can be freed from memory. Using a BBH and a per-photon radius, an additional seven floating-point values need to be stored, increasing the size of each photon from 20 bytes to 46 bytes.

Algorithm 8.2: CONSTRUCTBBH(p)

```
Data: p is a node in a balanced photon map.
Result: The subtree at p contains a valid BBH.
1  begin
2      B ⇐ BOUNDINGBOX(p.position, p.radius);
3      if p.leftChild then
4          B ⇐ B ∪ CONSTRUCTBBH(p.leftChild);
5      end
6      if p.rightChild then
7          B ⇐ B ∪ CONSTRUCTBBH(p.rightChild);
8      end
9      p.bbox ⇐ B;
10     return B
11 end
```

The Beam Radiance Estimate. During rendering we estimate the accumulated in-scattered radiance (Equation 8.1) along viewing rays by locating all photons whose bounding spheres intersect the ray. These photons are found using a standard ray-BBH intersection traversal. The contribution from each photon $(\alpha_i, \mathbf{x}_i, \vec{\omega}_i)$ is accumulated using Equation 8.21; however, with the variable kernel method, a kernel K_i is defined per photon. This leads to the following radiance estimate

$$\frac{1}{N}\sum_{i=1}^{N} K_i(\mathbf{x}, \vec{\omega}, s, \mathbf{x}_i, r_i)\, T_r(\mathbf{x} \leftrightarrow \mathbf{x}'_i)\, \sigma_s(\mathbf{x}'_i)\, p(\mathbf{x}_i, \vec{\omega} \leftrightarrow \vec{\omega}_i)\, \alpha_i, \tag{8.22}$$

Table 8.1: Rendering parameters and timings, in seconds, (s), and minutes, (m), for all example scenes. Statistics relating to the photon tracing preprocess are shown in the first set of columns. We compare our beam radiance estimate method (B) to conventional photon mapping (C) with both a fixed width kernel and an adaptive width kernel. The r column represents the fixed-width kernel radius, while r_+ is the maximum search radius and the number of nearest neighbors is k.

	Photon Tracing Preprocess				**Fixed Radius Render**				**Adaptive Radius Render**				
Scene	N	Shoot (s)	Balance (s)	Radius (s)	r	Δt	C (m)	B (m)	r_+	k	Δt	C (m)	B (m)
Cornell	0.4M	1.50	0.30	2.0	0.4	0.40	3:19	3:00	0.6	1.5K	0.80	4:03	3:35
Stage	1M	3.25	0.76	5.0	0.3	0.20	4:21	4:15	0.5	0.5K	0.70	6:38	6:22
Cars	2M	19.0	1.50	2.0	0.4	1.25	1:30	1:30	0.5	1K	1.25	2:02	1:53
Lighthouse	1M	2.83	0.78	6.0	0.4	0.25	1:07	0:59	0.5	0.4K	1.00	1:12	1:05

where $\mathbf{x}_i' = \mathbf{x} + t_i\vec{\omega}$ is the projection of the photon location $\mathbf{x}_i$ onto the ray's direction $\vec{\omega}$, and $t_i = (\mathbf{x}_i - \mathbf{x}) \cdot \vec{\omega}$. We define the kernel as

$$K_i(\mathbf{x}, \vec{\omega}, s, \mathbf{x}_i, r_i) = \begin{cases} r_i^{-2} K_2\left(\frac{d_i}{r_i}\right) & \text{if } d_i \in [0, r_i] \\ 0 & \text{otherwise} \end{cases}, \tag{8.23}$$

where r_i is the pre-computed radius for photon i. We use Silverman's two-dimensional biweight kernel [Silverman, 1986] along the ray, $K_2(x) = 3\pi^{-1}(1-x^2)^2$, where d_i is the shortest distance from photon i to the ray. We chose this kernel because it is smooth, efficient to evaluate, and has local support. Equation 8.22 is the beam radiance estimate, and it replaces the ray marching computation from conventional photon mapping (second term in Equation 7.9).

Heterogeneous Media. For homogeneous media, the transmission terms, $T_r(\mathbf{x} \leftrightarrow \mathbf{x}_i')$, can be computed explicitly for each photon during gathering. Beam gathering in heterogeneous media can also be handled efficiently by first marching along the ray and saving the transmission values in a 1D lookup table. Then, during gathering, each photon's transmission is computed by interpolating within the lookup table. This preprocess needs to be performed independently for each ray, just prior to gathering, so the lookup table can be reused. Furthermore, if single scattering is simulated separately by directly sampling light sources, the lookup table can be populated during this marching step.

8.4 Results

We compared our beam gathering technique against conventional volumetric photon mapping using ray marching. In order to isolate just the performance of the photon gathering methods, we use the photon map for both single and multiple scattering. We compared results on four test scenes: Cars, Lighthouse, Stage, and a Cornell box. For each scene we compare using a fixed gathering radius for both types of estimates, and we also compare the conventional estimate using k-NN to the beam estimate using the VK method. The images were all rendered with a maximum dimension of 1024 pixels with up to four samples per pixels on an Intel Core 2 Duo 2.4 GHz machine using one core.

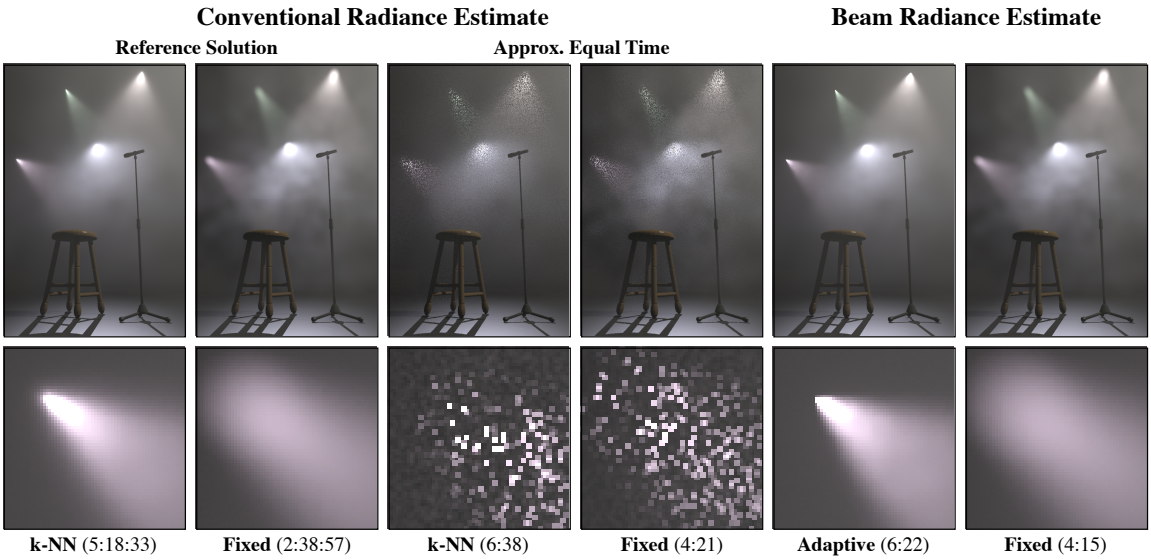


Figure 8.5: A comparison between the convention radiance estimate and our beam radiance estimate on the Stage scene with render times provided as (hours:minutes:seconds). Our method (right) produces images with much less noise than an equal time rendering using conventional volumetric photon mapping (middle) for both a fixed radius and an adaptive radius gathering approach. Our method does not require stepping but matches the quality of conventional photon mapping if a very small step size is used (left).

In our experimental setup, we first choose suitable gathering parameters (search radius and number of nearest neighbors k) and render the scenes using our method. We then use the same parameters using conventional photon mapping but tune the minimum step-size Δt to obtain approximately equal render times. Note that Δt is the *minimum* step-size and that exponential stepping is used to sample the ray according to transmission. Finally, we render a high-quality result with conventional photon mapping using a very small step size as a "reference."

We show visual comparisons of the methods in Figures 8.5 and 8.6. All images of each scene are rendered using the *same photon map*. The only differences in quality and performance are due to the gathering method used. We used the Henyey-Greenstein phase function for all

Table 8.2: Medium scattering properties and photon tracing statistics for the four example scenes. N is the total number of photons stored.

	Medium Parameters				**Photon Tracing**		
Scene	σ_s	σ_a	g	N	Shoot (s)	Balance (s)	Radius (s)
Cornell	0.225	0.225	0.00	0.4M	1.50	0.30	2.0
Stage	0.225	0.225	0.75	1M	3.25	0.76	5.0
Cars	0.06	0.015	0.00	2M	19.0	1.50	2.0
Lighthouse	0.24	0.010	0.75	1M	2.83	0.78	6.0

scenes. The render times, gathering parameters and timings for constructing the photon maps are listed in Table 8.1. We report the scattering parameters for all the example scenes in Table 8.2.

In all cases, our method produces significantly higher-quality images than conventional photon mapping. This is because querying once for all photons along a ray is more efficient than repeatedly querying for photons near numerous samples along the ray. Not surprisingly, we see that the reduced blurring of the adaptive kernel gathering methods is essential for scenes like the Stage and Lighthouse, where concentrated beams of light are visible. However, at the same render time, this advantage is difficult to discern in the conventional photon mapping images.

Though the k-NN and VK methods both adapt the width of the kernel to the local photon density, they are distinct approaches which result in similar, but not identical, density estimates. This is what accounts for the small differences in blurring between our adaptive results and the k-NN "reference" images. However, as our results show, the same value of k produces visually comparable renderings using the two methods.

8.5 Summary and Discussion

In this chapter, we showed how volumetric photon mapping can be expressed as a solution to the measurement equation. This formulation showed that *any* measurement of radiance within participating media can be estimated using the photon map. We applied this formulation by using the photon map to directly estimate accumulated in-scattered radiance along rays. This approach was implemented using an efficient beam gathering method, which can be used for both fixed and adaptive width kernels. The resulting algorithm produces images with significantly less noise than conventional volumetric photon mapping using the same render time.

8.6 Acknowledgements

This chapter is, in part, a reproduction of the material published in Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. "The Beam Radiance Estimate for Volumetric Photon Mapping." In *Computer Graphics Forum (Proceedings of Eurographics 2008)*, 27(2):557–566, 2008; as well as the extended technical report Wojciech Jarosz, Matthias Zwicker, and Henrik

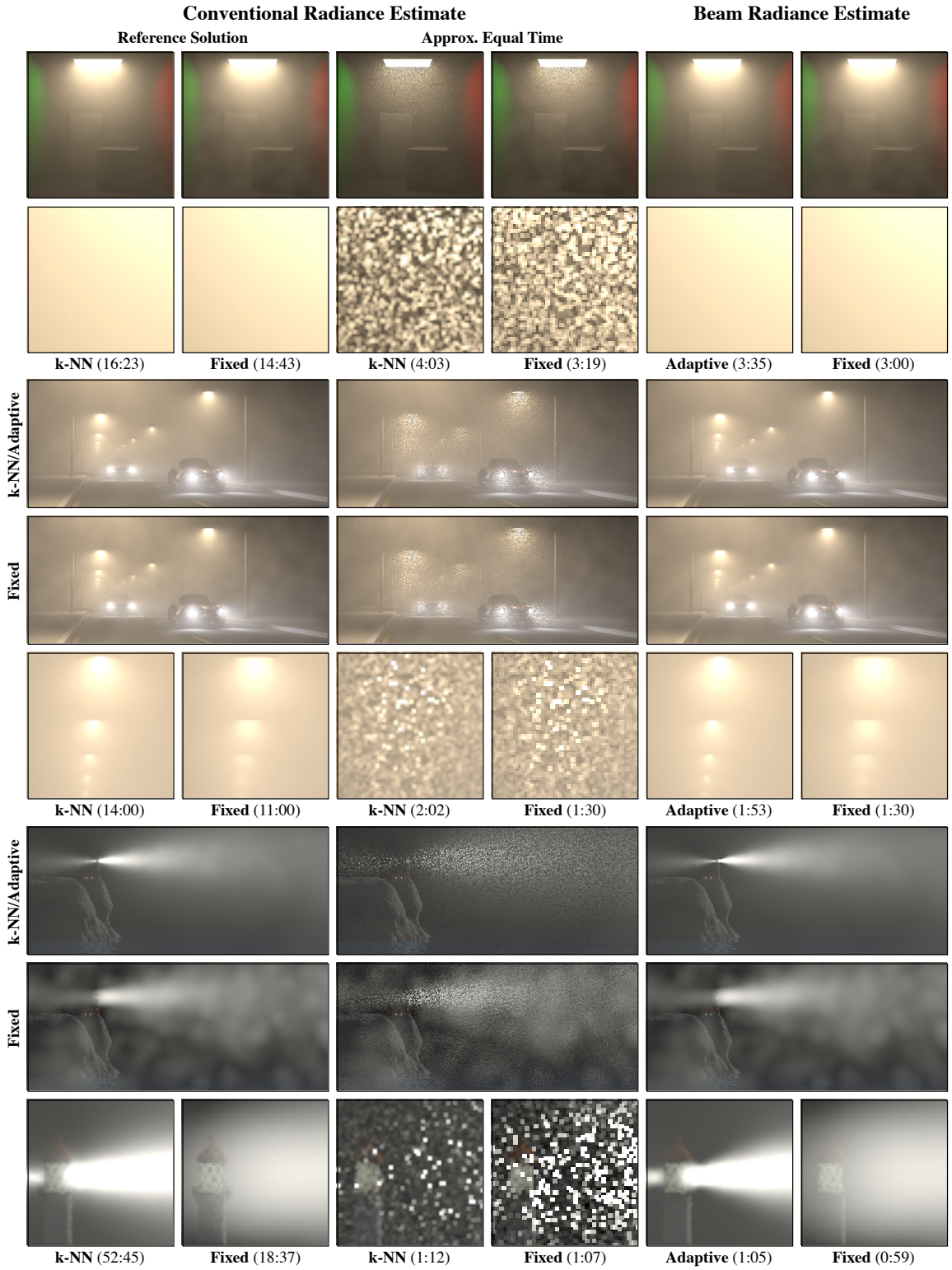


Figure 8.6: Visual comparison for the Cornell box, Cars, and Lighthouse scenes.

Wann Jensen. “The Beam Radiance Estimate for Volumetric Photon Mapping.” Technical Report CS2008-0914, University of California, San Diego, 2008. The dissertation author was the primary investigator and author of both papers. This work was supported in part by NSF grant CPA 0701992.

9

Conclusions

"This is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning."

—Winston Churchill, 1874–1965

IN this dissertation we sought to develop efficient algorithms for simulating light transport within arbitrary participating media. We provided missing derivations for the split-sphere model, which is central to the success of the irradiance caching algorithm. We also provided full derivations of the irradiance gradient computation, extended them to participating media, and have developed a reformulation of the volumetric photon mapping method. These theoretical examinations have provided insights which have directly led to several significant steps towards our goal.

We introduced two concrete approaches for efficient light transport within participating media: volumetric radiance caching and volumetric photon mapping using beam radiance estimation. These techniques share many similarities. Both approaches are based around a Monte Carlo ray tracing framework. This grants the algorithms a level of generality which is not attainable with finite element or analytic methods. Both methods also exploit the smooth nature of illumination within participating media by reusing illumination computations across the scene. This property is key to the efficiency of the methods. The techniques impose no restrictions on the properties and representation of the medium being rendered. They can efficiently compute single and multiple scattering in heterogeneous media with anisotropic phase functions.

We have also introduced improvements to the irradiance caching method to properly account for participating media. We derived gradients of the radiative transfer equation which

correctly account for scatter, absorption, and emission, and which incorporate changes in occlusion. This allows irradiance caching to be used effectively even in scenes containing participating media.

Volumetric radiance caching and photon mapping also contain significant differences. Radiance caching is a view-dependent approach which only computes lighting within the visible portion of the scene. This makes the algorithm well suited for scenes with a large extent or scenes where only a very small portion of the scene is visible to the camera. In contrast, photon mapping simulates lighting within the whole scene. This can be a significant drawback if there are many light sources that do not contribute much to the chosen view. On the other hand, photon mapping is the only available method which can efficiently simulate volumetric caustics. These differences actually provide a great benefit by making the approaches complementary. We believe a more general and robust algorithm can be achieved by using volumetric radiance caching as a final gather pass for photon mapping. The photon mapping reformulation we presented also suggests interesting possibilities for other custom-designed radiance estimates. A final gather radiance estimate could be designed which collects photons along all hemispherical beams instead of just along the beam of a single ray.

A

Monte Carlo Integration

THE techniques developed in this dissertation are all Monte Carlo methods. Monte Carlo methods are numerical techniques which rely on random sampling to *approximate* their results. Monte Carlo integration applies this process to the numerical estimation of integrals. In this appendix we review the fundamental concepts of Monte Carlo integration upon which our methods are based. From this discussion we will see why Monte Carlo methods are a particularly attractive choice for the multidimensional integration problems common in computer graphics. Good references for Monte Carlo integration in the context of computer graphics include Pharr and Humphreys [2004], Dutré et al. [2006], and Veach [1997].

The term "Monte Carlo methods" originated at the Los Alamos National Laboratory in the late 1940s during the development of the atomic bomb [Metropolis and Ulam, 1949]. Not surprisingly, the development of these methods also corresponds with the invention of the first electronic computers, which greatly accelerated the computation of repetitive numerical tasks. Metropolis [1987] provides a detailed account of the origins of the Monte Carlo method.

Las Vegas algorithms are another class of method which rely on randomization to compute their results. However, in contrast to Las Vegas algorithms, which always produce the exact or correct solution, the accuracy of Monte Carlo methods can only be analyzed from a statistical viewpoint. Because of this, we first review some basic principles from probability theory before formally describing Monte Carlo integration.

A.1 Probability Background

In order to define Monte Carlo integration, we start by reviewing some basic ideas from probability.

A.1.1 Random Variables

A *random variable* X is a function that maps outcomes of a random process to numbers. A random variable can be either discrete (e.g., the roll of a six-sided die where a fixed set of outcomes is possible, $X = \{1,2,3,4,5,6\}$), or continuous (e.g., a person's height, which can take on real values $\mathbb{R}$). In computer graphics we more commonly deal with continuous random variables which take on values over ranges of continuous domains (e.g., the real numbers $\mathbb{R}$ or the sphere of directions Ω).

A.1.2 Cumulative Distributions and Density Functions

The *cumulative distribution function*, or CDF, of a random variable X is the probability that a value chosen from the variable's distribution is less than or equal to some thresold x:

$$cdf(x) = Pr\{X \leq x\}. \tag{A.1}$$

The corresponding *probability density function*, or PDF, is the derivative of the CDF:

$$pdf(x) = \frac{d}{dx} cdf(x). \tag{A.2}$$

CDFs are always monotonically increasing, which means that the PDF is always non-negative. An important relationship arises from the above two equations, which allows us to compute the probability that a random variable lies within an interval:

$$Pr\{a \leq X \leq b\} = \int_a^b pdf(x)\, dx. \tag{A.3}$$

From this expression it is clear that the PDF must always integrate to one over the full extent of its domain.

A.1.3 Expected Values and Variance

The *expected value* or *expectation* of a random variable $Y = f(X)$ over a domain $\mu(x)$ is defined as

$$E[Y] = \int_{\mu(x)} f(x)\, pdf(x)\, d\mu(x), \tag{A.4}$$

while its *variance* is

$$\sigma^2[Y] = E\left[(Y - E[Y])^2\right], \tag{A.5}$$

where σ, the *standard deviation*, is the square root of the variance. From these definitions it is easy to show that for any constant a,

$$E[a\,Y] = a\,E[Y], \tag{A.6}$$

$$\sigma^2[a\,Y] = a^2\,\sigma^2[Y]. \tag{A.7}$$

Furthermore, the expected value of a sum of random variables Y_i is the sum of their expected values:

$$E\left[\sum_i Y_i\right] = \sum_i E[Y_i]. \tag{A.8}$$

From these properties it is possible to derive a simpler expression for the variance:

$$\sigma^2[Y] = E\left[Y^2\right] - E[Y]^2. \tag{A.9}$$

Additionally, if the random variables are *uncorrelated*, a summation property also holds for the variance[1]:

$$\sigma^2\left[\sum_i Y_i\right] = \sum_i \sigma^2[Y_i]. \tag{A.10}$$

A.2 The Monte Carlo Estimator

The Basic Estimator. Monte Carlo integration uses random sampling of a function to numerically compute an estimate of its integral. Suppose that we want to integrate the one-dimensional function $f(x)$ from a to b:

$$F = \int_a^b f(x)\,dx. \tag{A.11}$$

We can approximate this integral by averaging samples of the function f at uniform random points within the interval. Given a set of N uniform random variables $X_i \in [a,b)$ with a corresponding PDF of $1/(b-a)$, the Monte Carlo estimator for computing F is

$$\left\langle F^N \right\rangle = (b-a)\frac{1}{N-1}\sum_{i=0}^{N} f(X_i), \tag{A.12}$$

The random variable $X_i \in [a,b)$ can be constructed by warping a canonical random number uniformly distributed between zero and one, $\xi_i \in [0,1)$:

$$X_i = a + \xi_i(b-a). \tag{A.13}$$

Using this construction, we can expand the estimator to:

$$\left\langle F^N \right\rangle = (b-a)\frac{1}{N}\sum_{i=0}^{N-1} f\left(a+\xi_i(b-a)\right). \tag{A.14}$$

[1] This property is often made with the stronger condition that the variables are *independent*; however, it suffices for them to be uncorrelated.

Since $\langle F^N \rangle$ is a function of X_i, it is itself a random variable. We use this notation to clarify that $\langle F^N \rangle$ is an approximation of F using N samples.

Intuitively, the Monte Carlo estimator in Equation A.12 computes the mean value of the function $f(x)$ over the interval a to b, and then multiplies this mean by the length of the interval $(b-a)$. By moving $(b-a)$ into the summation, the estimator can be thought of as choosing a height at a random evaluation of the function and averaging a set of rectangular areas computed by multiplying this height by the interval length $(b-a)$. These two interpretations are illustrated in Figure A.1.

A.2.1 Expected Value and Convergence

It is easy to show that the expected value of $\langle F^N \rangle$ is in fact F:

$$\begin{aligned}
E\left[\langle F^N \rangle\right] &= E\left[(b-a)\frac{1}{N}\sum_{i=0}^{N-1} f(X_i)\right], \\
&= (b-a)\frac{1}{N}\sum_{i=0}^{N-1} E\left[f(X_i)\right], && \text{from Equations A.8 and A.6} \\
&= (b-a)\frac{1}{N}\sum_{i=0}^{N-1} \int_a^b f(x)\,pdf(x)\,dx, && \text{from Equation A.4} \\
&= \frac{1}{N}\sum_{i=0}^{N-1} \int_a^b f(x)\,dx, && \text{since } pdf(x) = 1/(b-a) \\
&= \int_a^b f(x)\,dx, \\
&= F. && \text{(A.15)}
\end{aligned}$$

Furthermore, as we increase the number of samples N, the estimator $\langle F^N \rangle$ becomes a closer and closer approximation of F. Due to the *Strong Law of Large Numbers*, in the limit we can guarantee that we have the exact solution:

$$Pr\left\{\lim_{N\to\infty} \langle F^N \rangle = F\right\} = 1. \tag{A.16}$$

In practice we are interested in knowing just how quickly this estimate converges to a

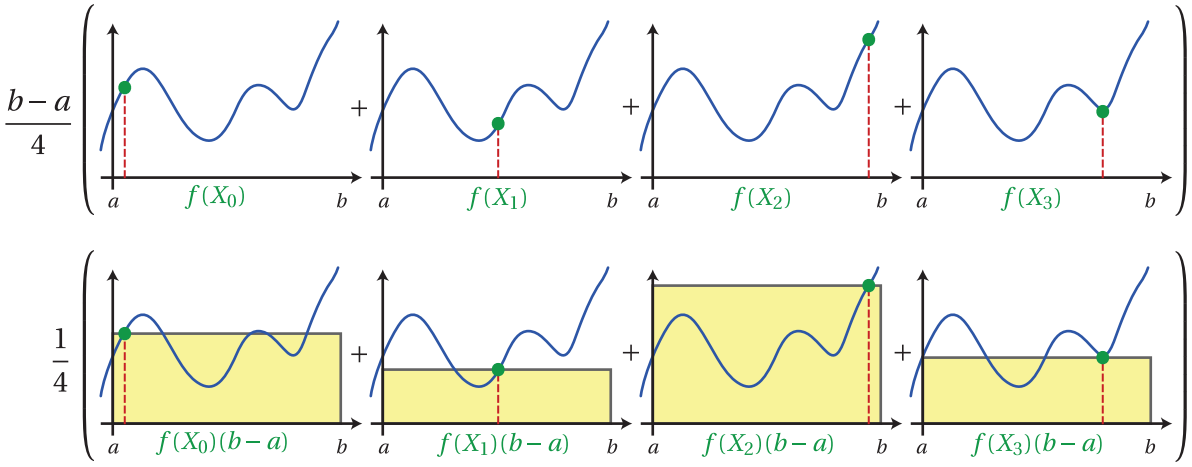


Figure A.1: An illustration of the two interpretations of the basic Monte Carlo estimator in Equation A.12 using four samples: computing the mean value, or height, of the function and multiplying by the interval length (top), or computing the average of several rectangular areas (bottom).

sufficiently accurate solution. This can be analyzed by determining the *convergence rate* of the estimators variance. In the next section we will show that the standard deviation is proportional to:

$$\sigma\left[\langle F^N \rangle\right] \propto \frac{1}{\sqrt{N}}. \tag{A.17}$$

Unfortunately, this means that we must quadruple the number of samples in order to reduce the error by half!

Standard integration techniques exist which converge much faster in one dimension; however, these techniques suffer from the *curse of dimensionality*, where the convergence rate becomes *exponentially* worse with increased dimensions. The basic Monte Carlo estimator above can easily be extended to multiple dimensions, and, in contrast to deterministic quadrature techniques, the convergence rate for Monte Carlo is independent of the number of dimensions in the integral. This makes Monte Carlo integration the only practical technique for many high dimensional integration problems, such as those encountered when computing global illumination.

A.2.2 Multidimensional Integration

Monte Carlo integration can be generalized to use random variables drawn from arbitrary PDFs and to compute multidimensional integrals, such as

$$F = \int_{\mu(x)} f(x)\, d\mu(x), \tag{A.18}$$

with the following modification to Equation A.12:

$$\left\langle F^N \right\rangle = \frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i)}{pdf(X_i)}. \tag{A.19}$$

It is similarly easy to show that this generalized estimator also has the correct expected value:

$$\begin{aligned}
E\left[\left\langle F^N \right\rangle\right] &= E\left[\frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i)}{pdf(X_i)}\right], \\
&= \frac{1}{N} \sum_{i=0}^{N-1} E\left[\frac{f(X_i)}{pdf(X_i)}\right], \\
&= \frac{1}{N} \sum_{i=0}^{N-1} \int_\Omega \frac{f(x)}{pdf(x)} pdf(x)\, dx, \\
&= \frac{1}{N} \sum_{i=0}^{N-1} \int_\Omega f(x)\, dx, \\
&= \int_\Omega f(x)\, dx, \\
&= F.
\end{aligned} \tag{A.20}$$

In addition to the convergence rate, a secondary benefit of Monte Carlo integration over traditional numerical integration techniques is the ease of extending it to multiple dimensions. Deterministic quadrature techniques require using N^d samples for a d-dimensional integral. In contrast, Monte Carlo techniques provide the freedom of choosing any arbitrary number of samples.

As mentioned previously, the Monte Carlo estimator has a constant $O(\sqrt{N})$ convergence rate in any dimension. However, in many situations it is possible to do much better. The efficiency of Monte Carlo integration can be significantly improved using a variety of techniques, which we

discuss in the next section.

A.3 Variance Reduction

Sources of Variance. In order to improve the quality of Monte Carlo integration we need to reduce variance. Since the samples in Monte Carlo integration are independent, using Equation A.10, the variance of $\langle F^N \rangle$ can be simplified to:

$$\begin{aligned}
\sigma^2 \left[\langle F^N \rangle \right] &= \sigma^2 \left[\frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i)}{pdf(X_i)} \right] \\
&= \frac{1}{N^2} \sum_{i=0}^{N-1} \sigma^2 \left[\frac{f(X_i)}{pdf(X_i)} \right] \\
&= \frac{1}{N^2} \sum_{i=0}^{N-1} \sigma^2 \left[Y_i \right] \\
&= \frac{1}{N} \sigma^2 \left[Y \right], \qquad \text{(A.21)}
\end{aligned}$$

and hence,

$$\sigma \left[\langle F^N \rangle \right] = \frac{1}{\sqrt{N}} \sigma \left[Y \right], \qquad \text{(A.22)}$$

where $Y_i = f(X_i)/pdf(X_i)$ and Y represents the evaluation of any specific Y_i, e.g., $Y = Y_1$.

This derivation proves our earlier statement that the standard deviation converges with $O(\sqrt{N})$. Moreover, this expression shows that by reducing the variance of each Y_i we can reduce the overall variance of $\langle F^N \rangle$.

Variance-reduction techniques try to make each Y_i as constant as possible in order to reduce the overall error of the estimator. A significant amount of research has been put into this area, leading to a number of complementary techniques. Most of these methods rely on exploiting some previous knowledge of the function being integrated.

A.3.1 Importance Sampling

Importance sampling reduces variance by observing that we have the freedom to choose the PDF used during integration. By choosing samples from a distribution $pdf(x)$, which has a *similar shape* as the function $f(x)$ being integrated, variance is reduced. Intuitively, importance sampling attempts to place more samples where the contribution of the integrand is high, or "important." If we can properly guess the important regions during integration, the variance of the standard Monte Carlo estimator can be significantly reduced.

The Perfect Estimator

To demonstrate the effect of importance sampling, consider a PDF which is exactly proportional to the function being integrated, $pdf(x) = c\,f(x)$ for some normalization constant c. Since c is a constant, if we apply this PDF to the Monte Carlo estimator in Equation A.19, each sample X_i would have the same value,

$$Y_i = \frac{f(X_i)}{pdf(X_i)} = \frac{\cancel{f(X_i)}}{c\cancel{f(X_i)}} = \frac{1}{c}. \tag{A.23}$$

By using this PDF we have succeeded at reducing the variance of each Y_i. In fact, since each Y_i returns the same value, the overall variance is zero!

Since the PDF must integrate to one, it is easy to derive the value of c:

$$c = \frac{1}{\int f(x)\,dx}. \tag{A.24}$$

This unfortunately shows us that determining the normalization constant c in the PDF involves solving the integral we are interested in estimating in the first place. This best case is therefore not a realistic situation. In practice, all validly constructed PDFs will still produce a convergence rate of $\sigma \propto \sqrt{N}$; however, by choosing a PDF that is similar to $f(x)$, we can make the variance arbitrarily low.

The only restriction on the PDF, in order to maintain the correct expected value from Equation A.20, is that it must be non-zero everywhere where the function $f(x)$ is non-zero.

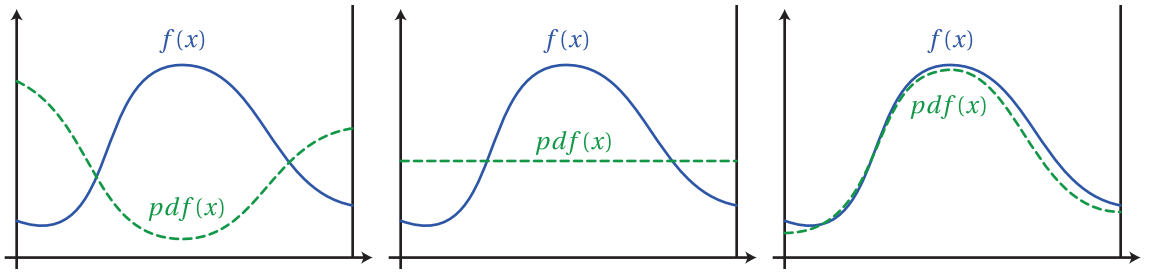


Figure A.2: Comparison of three probability density functions. The PDF on the right provides variance reduction over the uniform PDF in the center. However, using the PDF on the left would significantly increase variance over simple uniform sampling.

Importance sampling should not be used carelessly however. Though we can decrease the variance by using a "good" importance function, we can also make the variance arbitrarily large by choosing an importance function that increases the variance of Y. Figure A.2 shows a uniform PDF, as well as examples of PDFs which reduce and increase the overall variance.

Importance Sampling Complex Functions

Most of the time, the integrand $f(x)$ is very complicated, and we cannot guess its full behavior ahead of time. However, we may know something about its general structure. For instance, the integrand function $f(x)$ may in fact be the combination of more than one function, e.g., $f(x) = g(x)\,h(x)$. In these situations, it may not be possible to create a PDF exactly proportional to $f(x)$, but, if we know one of the functions in advance, we may be able to construct a PDF proportional to a portion of $f(x)$, e.g., $pdf_g(x) \propto g(x)$. In this situation, the Monte Carlo estimator simplifies to:

$$\begin{aligned}\langle F^N \rangle &= \frac{1}{N}\sum_{i=0}^{N-1}\frac{f(X_i)}{pdf_g(X_i)},\\ &= \frac{1}{N}\sum_{i=0}^{N-1}\frac{g(X_i)\,h(X_i)}{c\,g(X_i)},\\ &= \frac{1}{cN}\sum_{i=0}^{N-1} h(X_i).\end{aligned} \tag{A.25}$$

Since $h(x)$ will in general be smoother than the full $g(x)\,h(x)$, the variance of the Monte Carlo estimator is reduced.

Multiple Importance Sampling

What if we could also create a PDF which is proportional to $h(x)$, $pdf_h \propto h(x)$? Should we sample according to pdf_h or pdf_g? As it turns out, we don't have to choose. We can benefit from distributing samples independently from each PDF and combining the results using *multiple importance sampling* [Veach and Guibas, 1995; Veach, 1997].

A.3.2 Control Variates

Control variates is another variance-reduction technique which relies on some prior knowledge of the behavior of f. The idea behind control variates is to find a function g which can be analytically integrated and subtract it from the integral of f:

$$\begin{aligned} F &= \int_a^b f(x)\,dx, \\ &= \int_a^b f(x) - g(x)\,dx + \int_a^b g(x)\,dx, \\ &= \int_a^b f(x) - g(x)\,dx + G. \end{aligned} \tag{A.26}$$

We can then apply Monte Carlo integration to the modified integrand $f(x) - g(x)$:

$$\left\langle F_c^N \right\rangle = \left(\frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i) - g(X_i)}{pdf(X_i)} \right) + G, \tag{A.27}$$

where the value of the integral G is known exactly.

If $f(X_i)$ and $g(X_i)$ are correlated, then the variance of $\left\langle F^N \right\rangle$ is reduced. For control variates, g should be chosen so that $f - g$ is nearly constant. In contrast, with importance sampling we tried to choose a PDF, $pdf = g$, so that f/g was nearly constant.

As with importance sampling, we can obtain optimal results by choosing a g such that

$$Y_i = \frac{f(X_i) - g(X_i)}{pdf(X_i)} \tag{A.28}$$

is a constant. Unfortunately, this again requires knowledge of the true integral F.

Another way to think about control variates is by rewriting Equation A.27 as

$$\begin{aligned}\left\langle F_c^N \right\rangle &= \left(\frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i)}{pdf(X_i)} \right) + \left(G - \frac{1}{N} \sum_{i=0}^{N-1} \frac{g(X_i)}{pdf(X_i)} \right), \\ &= \left\langle F^N \right\rangle + G - \left\langle G^N \right\rangle, \end{aligned} \tag{A.29}$$

where the same set of random samples X_i are used in both summations. Using this formulation, control variates can be interpreted as computing the difference between the exact integral G and its Monte Carlo approximation $\left\langle G^N \right\rangle$, and then adding this difference to our original Monte Carlo estimator $\left\langle F^N \right\rangle$. The idea is to detect what type of errors are introduced during Monte Carlo sampling of a known integral. If g and f are correlated, then we can expect the same errors to be present in the integral of f, so we can remove them.

A.3.3 Uniform Sample Placement

Importance sampling probabilistically places more samples where the function is deemed more important. The problem with this is that it does not eliminate sample "clumping," as the PDF only dictates the expected number, not the actual number, of samples in any given region. Intuitively, sample clumping is wasteful, since a sample that is very close to another does not provide much new information about the function being integrated. In order to obtain the best solution, we want to maximize the amount of information we gain from each sample.

Stratified Sampling

One powerful variance-reduction technique that addresses this problem is called stratified sampling. Stratified sampling works by splitting up the original integral into a sum of integrals over sub-domains. In its simplest form, stratified sampling divides the domain $[a,b]$ into N sub-domains (or *strata*) and places a random sample within each of these intervals. If using a

uniform PDF, with $\xi_i \in [0,1)$, this can be expressed as

$$\begin{aligned}\left\langle F_s^N \right\rangle &= \frac{(b-a)}{N} \sum_{i=0}^{N-1} f\left(X_i\right), \\ &= \frac{(b-a)}{N} \sum_{i=0}^{N-1} f\left(a + \tfrac{i+\xi_i}{N}(b-a)\right) \\ &= \frac{(b-a)}{N} \sum_{i=0}^{N-1} f\left(a + \xi_i^N (b-a)\right). \end{aligned} \tag{A.30}$$

Compare this estimator to the expanded basic estimator in Equation A.14. Most of the time, stratification can be added by simply replacing a canonical random number $\xi_i \in [0,1)$ with a sequence of stratified random numbers in the same range $\xi_i^N = \frac{i+\xi_i}{N}$.

It can be shown that stratified sampling can never result in higher variance than pure random sampling. In fact, stratified sampling is asymptotically better, since the error reduces linearly with the number of samples, $\sigma \propto 1/N$. Therefore, whenever possible, stratified sampling *should* be used.

Stratified sampling does place more restrictions on the sampling process. This can become problematic when, for instance, the number of samples is not known in advance. Furthermore, though it is possible to extend stratified sampling to higher dimensions, in general, N^d strata would need to be created for a d dimensional domain. This explosion of samples can make fine-grained control of simulation times difficult.

Comparison to Deterministic Quadrature. Stratified sampling incorporates ideas from deterministic quadrature, while still retaining the statistical properties of Monte Carlo integration. It is informative to compare Equation A.30 to a simple Riemann summation of the integral:

$$F = \int_a^b f(x)\, dx \approx \sum_{i=0}^{N-1} f(x_i)\, \Delta x, \tag{A.31}$$

$$\approx \sum_{i=0}^{N-1} f\left(a + \tfrac{i}{N}(b-a)\right) \frac{(b-a)}{N}, \tag{A.32}$$

$$\approx \frac{(b-a)}{N} \sum_{i=0}^{N-1} f\left(a + \tfrac{i}{N}(b-a)\right). \tag{A.33}$$

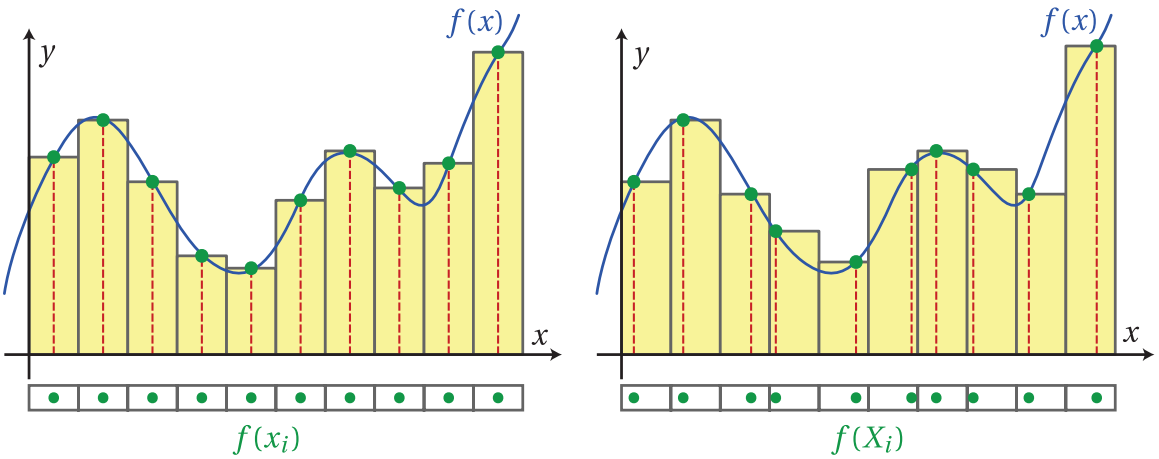


Figure A.3: Deterministic quadrature techniques such as a Riemann summation (left) sample the function at *regular* intervals. Conceptually, stratified Monte Carlo integration (right) performs a very similar summation, but instead evaluates the function at a *random* location within each of the strata.

In effect, the only difference between the Riemann sum and the stratified estimator in Equation A.30 is that Riemann summation evaluates the function at deterministic locations within the domain (the start of each stratum in the above example) whereas stratified Monte Carlo places the samples randomly within each stratum. This distinction is illustrated in Figure A.3.

Other Approaches

There is a wide range of other methods to more uniformly distribute samples. We illustrate some of these approaches in Figure A.4. Other stratification approaches developed include N-Rooks [Shirley, 1990] or Latin hypercube sampling [McKay et al., 1979], multi-jittered [Chiu et al., 1994] and orthogonal array sampling [Owen, 1992], and multi-stage N-Rooks sampling [Wang and Sung, 1999]. Veach [1997] provides a good overview of these approaches.

Several advanced techniques go beyond analyzing just the variance of the Monte Carlo estimator. One approach is to measure the quality of the sample distribution based on frequency analysis of the variance. This analysis shows that, to minimize variance while avoiding aliasing artifacts beyond the Nyquist limit, a sample distribution conforming to a blue-noise frequency spectrum is desirable [Cook, 1986; Mitchell, 1991]. In fact, this distribution of samples has been observed in the placement of photoreceptors in the eye of the rhesus monkey [Yellott, 1983].

Another technique is to compute the *discrepancy* [Shirley, 1991] of the points, which

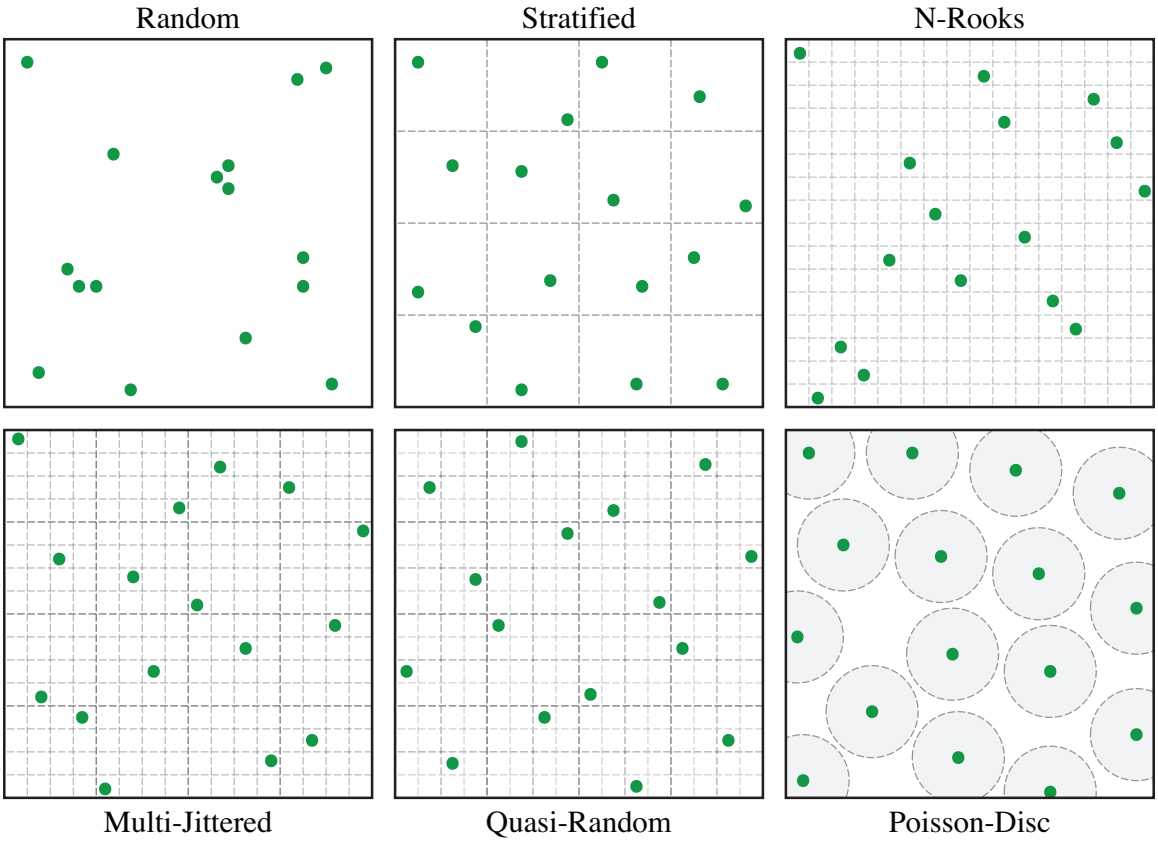


Figure A.4: An illustration comparing several 2D sampling approaches each using 16 samples. Purely random sampling (top left) can suffer from clumping, which increases variance by undersampling other regions of the integrand. All of the other approaches illustrated try to minimize this clumping to reduce variance.

measures how much the density of the points varies from one location to the next. Quasi-Monte Carlo techniques take this a step further and completely replace the use of pseudo-random numbers with judiciously tailored deterministic sequences. These quasi-random numbers are constructed explicitly to minimize clumping by reducing the discrepancy of the point set. Quasi-Monte Carlo sampling is therefore also referred to as low-discrepency sampling. Niederreiter [1992] provides an excellent theoretical introduction to Quasi-Monte Carlo techniques. Quasi-Monte Carlo techniques were introduced to computer graphics by Keller. See his dissertation and recent technical report for a thorough introduction [1998; 2003].

A.3.4 Adaptive Sampling

Like importance sampling, adaptive sampling is another technique which tries to intelligently place more samples where they will provide the most benefit. However, importance sampling requires some *a priori* knowledge of the function being integrated, whereas adaptive sampling relaxes this requirement by inferring this information *on the fly* during the sampling process.

Adaptive sampling tries to reduce variance by making adaptive decisions based on the samples taken so far and placing future samples in areas of high variance. The motivation for this approach is that regions of the integrand which are smooth can be well approximated using only a few samples, whereas regions with rapid variation require more samples to faithfully capture the behavior of the integrand.

A number of adaptive sampling techniques have been developed in computer graphics, most of which deal exclusively with adaptively sampling the image plane. In addition to introducting ray tracing, Whitted [1980] also proposed the use of a hierarchical adaptive sampling technique to increase the sample density around image discontinuities. Several other researchers have proposed stochastic adaptive sampling of the image plane [Dippé and Wold, 1985; Mitchell, 1987; Painter and Sloan, 1989; Mitchell, 1991].

Though adaptive sampling can be a very effective optimization technique, a number of issues complicate its use in practice. Firstly, choosing a robust adaptive refinement criterion can be very difficult and application specific. This can be particularly challenging in the context of Monte Carlo ray tracing since each Monte Carlo sample contains significant variance, making features and noise indistinguishable during adaptive sample refinement. Nevertheless, successful applications of Monte Carlo adaptive sampling in computer graphics do exist. For instance, the irradiance caching algorithm described in Chapter 3 and our volumetric radiance caching scheme developed in Chapter 6 are in essence approaches for adaptively sampling the indirect illumination on surfaces and within media, respectively. Another consideration with adaptive sampling is that, unlike importance sampling, it can introduce unbounded systematic errors unless special precautions are taken [Kirk and Arvo, 1991]. This systematic error, called *bias*, is

discussed in the next section.

A.3.5 Biased Monte Carlo

Estimators that always produce the correct expected value (like the ones we have considered so far) are called *unbiased* estimators. By using unbiased estimators, due to Equation A.15, we can be confident that the expected value of the estimator, using any number of samples, will always be correct. The only error in unbiased estimators is variance.

However, it is also possible to construct estimators which do not satisfy Equation A.15. These estimators are called *biased*, since, in addition to variance, they also contain a systematic bias:

$$\beta\left[\left\langle F^N\right\rangle\right] = E\left[\left\langle F^N\right\rangle\right] - F, \tag{A.34}$$

which is the difference in the expected value of the estimator and the actual value of the integral.

A biased estimator is called *consistent* if it converges to the correct result with more samples. Hence, consistent biased estimators satisfy Equation A.16 but not Equation A.15. A simple example of a consistent but biased estimator for I would be

$$F \approx (b-a)\frac{1}{N+1}\sum_{i=0}^{N-1} f(X_i). \tag{A.35}$$

It is easy to see that this estimator does not have the correct expected value, but as we increase the number of samples it does converge to the solution.

Sometimes it is useful to use biased estimators if they result in significantly reduced variance. All the techniques developed in this dissertation are consistent biased Monte Carlo methods, which introduce bias for exactly this purpose.

Sometimes, a more informative way to think about biased estimators is that they compute *unbiased* estimates of the *wrong* answer! In effect, biased estimators replace the integral I with a different integral $\bar{I}$, which has some desirable properties. For instance, it could be easier to integrate, or it could have lower variance. In a consistent estimator, as we increase the number of samples, the properties of $\bar{I}$ start to resemble the true function I more closely.

This alternative interpretation can be useful because it provides higher level knowledge about the exact form of the bias introduced. For instance, in Chapter 8 we proved that photon mapping actually computes an *unbiased* estimate of the *blurred* radiance. The amount of blur is directly controlled by the size and number of photons in the radiance estimate. This also makes it clear that photon mapping is consistent, since as we decrease the amount of blur to zero, the bias disappears.

B

Spherical Harmonics

SPHERICAL harmonics are a frequency-space basis for representing functions defined over the sphere. They are the spherical analogue of the 1D Fourier series. Spherical harmonics arise in many physical problems ranging from the computation of atomic electron configurations to the representation of gravitational and magnetic fields of planetary bodies. They also appear in the solutions of the Schrödinger equation in spherical coordinates. Spherical harmonics are therefore often covered in textbooks from these fields [MacRobert and Sneddon, 1967; Tinkham, 2003].

Spherical harmonics also have direct applicability in computer graphics. Light transport involves many quantities defined over the spherical and hemispherical domains, making spherical harmonics a natural basis for representing these functions. Early applications of spherical harmonics to computer graphics include the work by Cabral et al. [1987] and Sillion et al. [1991]. More recently, several in-depth introductions have appeared in the graphics literature [Ramamoorthi, 2002; Green, 2003; Wyman, 2004; Sloan, 2008].

In this appendix, we briefly review the spherical harmonics as they relate to computer graphics. We define the basis and examine several important properties arising from this definition.

B.1 Definition

A *harmonic* is a function that satisfies Laplace's equation:

$$\nabla^2 f = 0. \tag{B.1}$$

As their name suggests, the spherical harmonics are an infinite set of harmonic functions defined on the sphere. They arise from solving the angular portion of Laplace's equation in spherical coordinates using separation of variables. The spherical harmonic basis functions derived in this fashion take on complex values, but a complementary, strictly real-valued, set of harmonics can also be defined. Since in computer graphics we typically only encounter real-valued functions, we restrict our discussion to the real-valued basis.

If we represent a direction vector $\vec{\omega}$ using the standard spherical parameterization,

$$\vec{\omega} = \left(\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta\right), \tag{B.2}$$

then the real spherical harmonic basis functions are defined as:

$$y_l^m(\theta,\phi) = \begin{cases} \sqrt{2}K_l^m \cos(m\phi)P_l^m(\cos\theta) & \text{if } m > 0, \\ K_l^0 P_l^0(\cos\theta) & \text{if } m = 0, \\ \sqrt{2}K_l^m \sin(-m\phi)P_l^{-m}(\cos\theta) & \text{if } m < 0. \end{cases} \tag{B.3}$$

where K_l^m are the normalization constants

$$K_l^m = \sqrt{\frac{(2l+1)}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}}, \tag{B.4}$$

and P_l^m are the associated Legendre polynomials. There are many ways to define the associated Legendre polynomials but the most numerically robust way to evaluate them is using a set of

recurrence relations [Press et al., 1992][1]:

$$P_0^0(z) = 1, \tag{B.5}$$

$$P_m^m(z) = (2m-1)!!(1-z^2)^{m/2}, \tag{B.6}$$

$$P_{m+1}^m(z) = z(2m+1)P_m^m(z), \tag{B.7}$$

$$P_l^m(z) = \frac{z(2l-1)}{l-m}P_{l-1}^m(z) - \frac{(l+m-1)}{l-m}P_{l-2}^m(z). \tag{B.8}$$

The basis functions are indexed according to two integer constants, the *order*, l, and the *degree*, m[2]. These satisfy the constraint that $l \in \mathbb{N}$ and $-l \leq m \leq l$; thus, there are $2l+1$ basis functions of order l.

The order l determines the frequency of the basis functions over the sphere. The spherical harmonics may be written either as trigonometric functions of the spherical coordinates θ and ϕ as above, or alternately as polynomials of the cartesian coordinates x, y, and z. Using the cartesian representation, each y_l^m for a fixed l corresponds to a polynomial of maximum order l in x, y, and z.

[1] We omit the Condon-Shortley phase factor [Condon and Shortley, 1951] of $(-1)^m$ which is sometimes includes in the definition of P_l^m or y_l^m since this simplifies our notation.

[2] The order l is also sometimes refered to as the *band* index, and in quantum mechanics, l and m are refered to as "quantum numbers" and the spherical harmonics "states."

The first few spherical harmonics, in both spherical and cartesian coordinates, expand to:

		Spherical	**Cartesian**
$l=0$	$y_0^0(\theta,\phi) =$	$\sqrt{\frac{1}{4\pi}}$	$\sqrt{\frac{1}{4\pi}}$,
$l=1$	$y_1^{-1}(\theta,\phi) =$	$\sqrt{\frac{3}{4\pi}}\sin\phi\sin\theta$	$\sqrt{\frac{3}{4\pi}}x$,
	$y_1^0(\theta,\phi) =$	$\sqrt{\frac{3}{4\pi}}\cos\theta$	$\sqrt{\frac{3}{4\pi}}z$,
	$y_1^1(\theta,\phi) =$	$\sqrt{\frac{3}{4\pi}}\cos\phi\sin\theta$	$\sqrt{\frac{3}{4\pi}}y$,
$l=2$	$y_2^{-2}(\theta,\phi) =$	$\sqrt{\frac{15}{4\pi}}\sin\phi\cos\phi\sin^2\theta$	$\sqrt{\frac{15}{4\pi}}xy$,
	$y_2^{-1}(\theta,\phi) =$	$\sqrt{\frac{15}{4\pi}}\sin\phi\sin\theta\cos\theta$	$\sqrt{\frac{15}{4\pi}}yz$,
	$y_2^0(\theta,\phi) =$	$\sqrt{\frac{5}{16\pi}}(3\cos^2\theta-1)$	$\sqrt{\frac{5}{16\pi}}(3z^2-1)$,
	$y_2^1(\theta,\phi) =$	$\sqrt{\frac{15}{4\pi}}\cos\phi\sin\theta\cos\theta$	$\sqrt{\frac{15}{8\pi}}xz$,
	$y_2^2(\theta,\phi) =$	$\sqrt{\frac{15}{16\pi}}(\cos^2\phi-\sin^2\phi)\sin^2\theta$	$\sqrt{\frac{15}{32\pi}}(x^2-y^2)$.

We illustrate the first basis functions in Figure B.1.

B.2 Projection and Expansion

The spherical harmonics define a complete basis over the sphere. Thus, any real-valued spherical function f may be *expanded* as a linear combination of the basis functions

$$f(\vec{\omega}) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} y_l^m(\vec{\omega})\, f_l^m, \tag{B.9}$$

where the coefficients f_l^m are computed by *projecting* f onto each basis function y_l^m

$$f_l^m = \int_{\Omega_{4\pi}} y_l^m(\vec{\omega}) f(\vec{\omega}) d\vec{\omega}. \tag{B.10}$$

Just as with the Fourier series, this expansion is exact as long as l goes to infinity; however, this requires an infinite number of coefficients. By limiting the number of bands to $l = n-1$ we retain

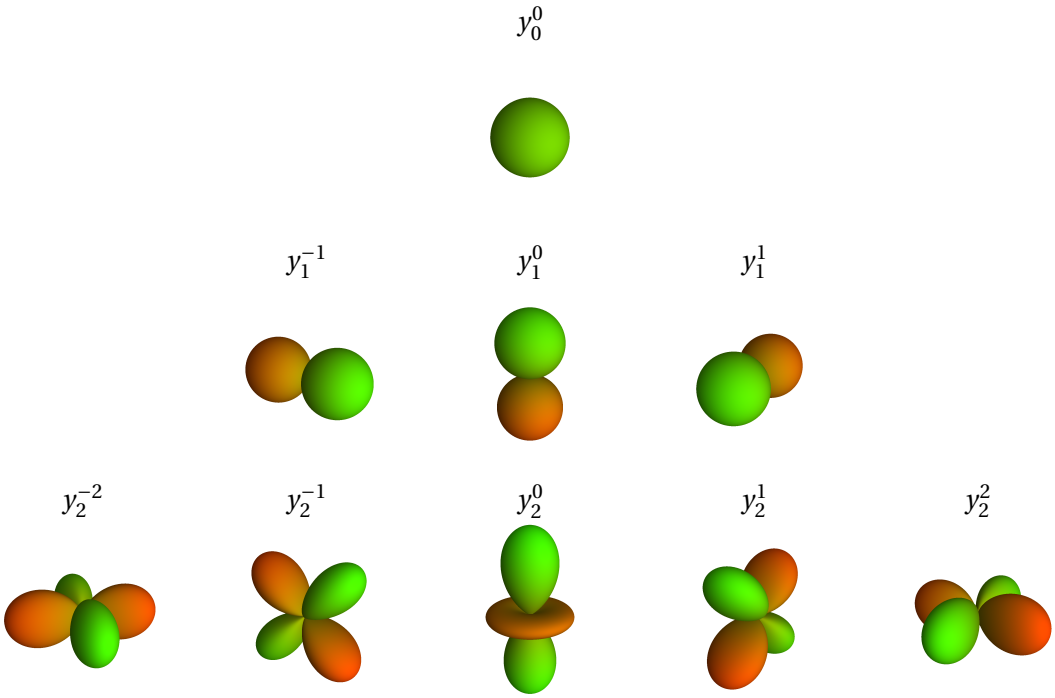


Figure B.1: Plots of the real-valued spherical harmonic basis functions. Green indicates positive values and red indicates negative values.

only the frequencies of the function up to some threshold, obtaining an n^{th} order band-limited approximation $\tilde{f}$ of the original function f:

$$\tilde{f}(\vec{\omega}) = \sum_{l=0}^{n-1} \sum_{m=-l}^{l} y_l^m(\vec{\omega})\, f_l^m. \tag{B.11}$$

Low-frequency functions can be well approximated using only a few bands, and as the number of coefficients increases, higher frequency signals can be approximated more accurately.

It is often convenient to reformulate the indexing scheme to use a single parameter $i = l(l+1)+m$. With this convension it is easy to see that an n^{th} order approximation can be reconstructed using n^2 coefficients,

$$\tilde{f}(\vec{\omega}) = \sum_{i=0}^{n^2-1} y_i(\vec{\omega})\, f_i. \tag{B.12}$$

B.3 Properties

The spherical harmonic functions have many basic properties that make them particularly convenient for use in computer graphics.

1. **Convolution.** Since the spherical harmonic basis is effectively a Fourier domain basis defined over the sphere, it inherits a similar frequency space convolution property. If $h(z)$ is a circularly symmetric kernel, then the *convolution* $h \star f$ is equivalent to weighted *multiplication* in the SH domain

$$(h \star f)_l^m = \sqrt{\frac{4\pi}{2l+1}} h_l^0 f_l^m. \tag{B.13}$$

   The convolution property allows for efficient computation of prefiltered environment maps and irradiance environment maps [Ramamoorthi and Hanrahan, 2001].

2. **Orthonormality.** The spherical harmonics are orthogonal for different l and different m. This means that the inner product of any two distinct basis functions is zero. Furthermore, the normalization constant K_l^m ensures that the inner product of a basis function with itself is one. This can be expressed mathematically as

$$\int_{\Omega_{4\pi}} y_i(\vec{\omega})\, y_j(\vec{\omega})\, d\vec{\omega} = \delta_{ij}, \tag{B.14}$$

   where δ_{ij} is the Kronecker delta function.

   The efficient projection and expansion operations described above are made possible by the fact that the SH basis is orthonormal. Many other useful and efficient operations also result from this important property.

3. **Double Product Integral.** The orthonormality property provides for a very simple expression to compute the integrated product of two functions represented in the SH basis. The

integral product of two SH functions $\tilde{a}(\vec{\omega})$ and $\tilde{b}(\vec{\omega})$ can be expanded as

$$\int_{\Omega_{4\pi}} \tilde{a}(\vec{\omega})\,\tilde{b}(\vec{\omega})\,d\vec{\omega} = \int_{\Omega_{4\pi}} \left(\sum_i a_i\, y_i(\vec{\omega})\right)\left(\sum_j b_j\, y_j(\vec{\omega})\right) d\vec{\omega}, \tag{B.15}$$

$$= \sum_i \sum_j a_i\, b_j \underbrace{\int_{\Omega_{4\pi}} y_i(\vec{\omega})\, y_j(\vec{\omega})\, d\vec{\omega}}_{C_{ij}}, \tag{B.16}$$

where C_{ij} are called the **coupling coefficients**, which, due to the definition of orthonormality in Equation B.14, are simply $C_{ij} = \delta_{ij}$. This simple form for the coupling coefficients introduces significant sparsity in the expression above, leading to the simplification:

$$\int_{\Omega_{4\pi}} \tilde{a}(\vec{\omega})\,\tilde{b}(\vec{\omega})\,d\vec{\omega} = \sum_i \sum_j a_i\, b_j\, C_{ij}, \tag{B.17}$$

$$= \sum_i \sum_j a_i\, b_j\, \delta_{ij}, \tag{B.18}$$

$$= \sum_i a_i\, b_i. \tag{B.19}$$

This expression states that the integrated product of two SH functions is simply the dot product of their coefficient vectors. The double product integral is of particular interest in computer graphics since it means lighting can be computed very efficiently in the frequency domain. If both the lighting and the cosine-weighted BRDF are represented in the SH basis, then the lighting integral can be computed using a simple dot product. This property is exploited by many PRT techniques [Sloan et al., 2002; Kautz et al., 2002].

4. **Triple Product Integral.** In many applications, the product integral of not just two, but three SH functions is of particular interest. This product can be expanded as

$$\int_{\Omega_{4\pi}} \tilde{a}(\vec{\omega})\,\tilde{b}(\vec{\omega})\,\tilde{c}(\vec{\omega})\,d\vec{\omega} = \int_{\Omega_{4\pi}} \left(\sum_i a_i\, y_i(\vec{\omega})\right)\left(\sum_j b_j\, y_j(\vec{\omega})\right)\left(\sum_k c_k\, y_k(\vec{\omega})\right) d\vec{\omega}, \tag{B.20}$$

$$= \sum_i \sum_j \sum_k a_i\, b_j\, c_k \int_{\Omega_{4\pi}} y_i(\vec{\omega})\, y_j(\vec{\omega})\, y_k(\vec{\omega})\, d\vec{\omega}, \tag{B.21}$$

$$= \sum_i \sum_j \sum_k a_i\, b_j\, c_k\, C_{ijk}, \tag{B.22}$$

which gives rise to the **tripling coefficients**, C_{ijk}. Unlike double product integrals, which have an incredibly simple form and reduce to a single dot product, the triple product integral is seemingly much more complicated. Fortunately, the set of tripling coefficients is also sparse, so not all the individual coefficients need to be explicitly computed and stored. For spherical harmonics, the C_{ijk} correspond to Clebsch-Gordan coefficients, whose analytic values and properties are well studied [Tinkham, 2003].

5. **Double Product Projection.** The tripling coefficients also arise when computing the product of two spherical harmonic functions directly in the SH basis. We can compute the i^{th} coefficient of the SH projection of the product $c(\vec{\omega}) = a(\vec{\omega})b(\vec{\omega})$ as

$$c_i = \int_{\Omega_{4\pi}} y_i(\vec{\omega})\, c(\vec{\omega}) d\vec{\omega}, \tag{B.23}$$

$$= \int_{\Omega_{4\pi}} y_i(\vec{\omega})\, a(\vec{\omega})\, b(\vec{\omega}) d\vec{\omega}, \tag{B.24}$$

$$= \int_{\Omega_{4\pi}} y_i(\vec{\omega}) \left(\sum_j a_j\, y_j(\vec{\omega})\right) \left(\sum_k b_k\, y_k(\vec{\omega})\right) d\vec{\omega}, \tag{B.25}$$

$$= \sum_j \sum_k a_j\, b_k \int_{\Omega_{4\pi}} y_i(\vec{\omega})\, y_j(\vec{\omega})\, y_k(\vec{\omega}) d\vec{\omega}, \tag{B.26}$$

$$= \sum_j \sum_k a_j\, b_k\, C_{ijk}. \tag{B.27}$$

This expression states that the i^{th} coefficient of c is a linear combination of the, up to, $j \times k$ coefficients from a and b. The weighting of these terms is determined by the triping coefficients, which are *independent* of the particular choice of a and b. In effect, if we wish to efficiently compute the product projection of many pairs of functions, we only need to compute the tripling coefficients once.

The product projection simplifies further if we know one of the functions beforehand. For instance, if $b(\vec{\omega})$ is fixed, then we can construct a *transfer matrix*, $\mathbf{M}$, which directly transforms coefficients of any arbitrary function $\tilde{a}$ into the coefficients of the product $\tilde{c}$

using a single vector-matrix multiplication:

$$c_i = \int_{\Omega_{4\pi}} y_i(\vec{\omega})\, a(\vec{\omega})\, b(\vec{\omega}) d\vec{\omega}, \tag{B.28}$$

$$= \int_{\Omega_{4\pi}} y_i(\vec{\omega}) \left(\sum_j a_j\, y_j(\vec{\omega}) \right) b(\vec{\omega}) d\vec{\omega}, \tag{B.29}$$

$$= \sum_j a_j \int_{\Omega_{4\pi}} y_i(\vec{\omega})\, y_j(\vec{\omega})\, b(\vec{\omega}) d\vec{\omega}, \tag{B.30}$$

$$= \sum_j a_j\, \mathbf{M}_{ij}. \tag{B.31}$$

6. **Rotational Invariance.** The SH basis functions are *rotationally invariant*, which means that if g is a rotated copy of f, i.e.,

$$g(\vec{\omega}) = f(\mathbf{R}\vec{\omega}), \tag{B.32}$$

for any 3×3 rotation matrix $\mathbf{R}$, then

$$\tilde{g}(\vec{\omega}) = \tilde{f}(\mathbf{R}\vec{\omega}). \tag{B.33}$$

This property means that, in order to evaluate a rotated SH function $\tilde{g}$, we can either rotate the *lookup* into the unrotated approximation $\tilde{f}$ or lookup directly into the rotated approximation $\tilde{g}$. This property implies that spherical harmonic projection produces no aliasing.

7. **Rotation.** Spherical harmonics also support efficient rotation. This means that if we know $\tilde{f}$, we can compute the SH coefficients of the rotated function $\tilde{g}$ *exactly* by just applying a linear transformation to the projection coefficients of $\tilde{f}$. This linear transformation is itself a higher-dimensional rotation matrix, $\tilde{\mathbf{R}}$, where the i^{th} coefficient of the rotated function g

is simply:

$$g_i = \sum_j f_j \tilde{\mathbf{R}}_{ij}. \tag{B.34}$$

Due to the rotation invariance property, the coefficients in one band of f only influence the same band of coefficients in the rotated representation g. This leads to the following block-sparse structure:

$$\tilde{\mathbf{R}} = \left[\begin{array}{c|ccc|ccccc|c}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
\hline
0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & \cdots \\
\hline
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
0 & 0 & 0 & 0 & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \cdots \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}\right]. \tag{B.35}$$

Several methods have been proposed for efficiently computing this coefficient transformation. Analytic forms can be derived for axis-aligned rotations, which can then be combined using Euler angle decomposition to construct rotations about arbitrary axes [Kautz et al., 2002]. This approach is only practical for low-order coefficients, and rotations of higher order SH functions are more efficiently implemented using recurrence relations [Ivanic and Ruedenberg, 1996, 1998; Blanco et al., 1997; Choi et al., 1999; Pinchon and Hoggan, 2007]. Křivánek et al. [2005c] instead proposed a fast approximate SH rotation method for small angles, which uses a truncated Taylor expansion of the rotation matrix. Green [2003] provides a more detailed summary of available methods.

If the function f has circular symmetry about the z-axis, then its projection consists only of zonal harmonics (only the $m = 0$ SH basis functions). Zonal harmonics (ZH) are an important subset of the full set of SH basis functions since they often lead to more efficient

operations. Furthermore, circularly symmetric functions are common in graphics. Most phase functions discussed in Chapter 4, for instance, exhibit circular symmetry.

Since ZH functions only have one non-zero coefficient per band, they can be rotated much more efficiently. This means that only one column of each band-matrix is needed:

$$\tilde{\mathbf{R}} = \left[\begin{array}{c|cccc|ccccc|c} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \hline 0 & 0 & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \mathbf{X} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{X} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{X} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{X} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{X} & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{X} & 0 & 0 & \cdots \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{array}\right]. \tag{B.36}$$

The non-zero elements in this matrix have an easy analytic form. To rotate a zonal harmonic function into direction $\vec{d}$ we simply need to apply the formula:

$$g_l^m = \sqrt{\frac{4\pi}{2l+1}} f_l^0 y_l^m(\vec{d}). \tag{B.37}$$

Note the similarity of this equation to the convolution in Equation B.13. In effect, rotation of a circularly symmetric function is the same as convolving a kernel with a delta function at the desired rotation axis.

C

Density Estimation

DENSITY estimation is a common problem that occurs in many different fields. We may, for instance, want to determine the likelihood of heart attack for a particular age group given a large collection of medical reports. We may also be interested in the geographical distribution of a particular group of people, using census reports or a telephone survey. Another application could be in predicting the outcome of a natural phenomenon based on observations of its past behavior, such as the eruption interval of a geyser. In the context of computer graphics, we may also wish to determine the intensity of light in a medium based on the distribution of photons in the scene. All of these problems are concerned with the same basic problem of density estimation.

Density estimation is a research area in statistics and has been studied extensively in this field. Density estimation has also been used in the field of computer graphics [Zareski et al., 1995; Shirley et al., 1995; Walter et al., 1997; Walter, 1998]. The volumetric photon mapping technique developed in Chapter 8, as well as the original photon mapping method upon which it is based [Jensen, 1996, 2001; Jensen and Christensen, 1998], can be thought of as a density estimation process and relies on tools developed in statistical modeling.

In this chapter we provide a brief overview of the theory of density estimation as well as review some of the available approaches. More in-depth introduction to these concepts can be found in the classical text by Silverman [1986] as well as Scott [1992]. Other references in the computer graphics literature include Dutré et al. [2006] and Walter [1998].

C.1 Introduction

In density estimation we are interested in determining an unknown function f, given only random samples or observations distributed according to this function. More formally, the goal of density estimation is to infer the probability density function, or PDF, from observations of a random variable. We have already discussed the use of PDFs and random variables in Appendix A with the goal of numerically integrating functions. Density estimation is concerned with a related, but inverse, problem: given a set of random samples, determine what PDF was used to create them.

Density estimation approaches can be broadly classified into two groups: parametric density estimation and non-parametric density estimation.

Parametric Methods. Parametric methods make strict a priori assumptions about the form of the underlying density function. For instance, a parametric approach may assume the random variables have a Gaussian distribution or the PDF is a polynomial of a particular degree. Such assumptions significantly simplify the problem, since only the parameters of the chosen family of functions need to be determined. In the case of a normal distribution, the density estimation process reduces to determining the mean μ and standard deviation σ of the sample points.

Non-Parametric Methods. Oftentimes it is not possible to make such strict assumptions about the form of the underlying density function. Non-parametric approaches are more appropriate in these situations. These techniques make few assumptions about the density function and allow the data to drive the estimation process more directly. In the context of computer graphics, non-parametric approaches are typically the most appropriate, and we will focus on these in the remaining sections.

C.2 Histograms

The simplest form of density estimation is the histogram method. This approach subdivides the domain into bins and counts the number of samples n_b which fall into each bin. The

local probability density is obtained by dividing the number of samples in each bin by the total number of samples N and the bin width h. This can be expressed as

$$\hat{f}(x) = \frac{n_b}{Nh} \quad \text{for } x_b \le x < x_{b+1}, \tag{C.1}$$

where x_b and x_{b+1} are the extents of bin b, and $h = x_{b+1} - x_b$. We use $\hat{f}$ to denote a density estimate of the probability density function f.

The histogram method has a number of advantages. It is easy to implement and provides results which are straightforward to visualize and intuitive to interpret. Also, it is easy to show that $\hat{f}$ is a valid PDF since it is always non-negative and integrates to one over the entire domain. However, histograms have many problems, which motivated the developed of more advanced methods.

One issue with histograms is that the resulting density function is not smooth. In fact, it has zero derivatives everywhere, except at the bin transitions, where its derivative is infinite. This issue can be catastrophic in applications, such as clustering, which rely on following the derivative to find local maxima. In computer graphics, this issue is not as extreme, but the derivative discontinuities can lead to objectionable artifacts, which can be avoided with more advanced techniques.

Another issue with histograms is that the choice of bin transition locations, even when keeping h fixed, can significantly affect the resulting PDF. This extra degree of freedom is completely independent of the underlying data and is simply a side effect of the estimation method itself.

Several approaches have been developed to address some of these concerns. Orthogonal series estimation, discussed in the next section, directly addresses the issue of discontinuity. The naïve estimator and all its generalizations come about by attempting to remove the choice of absolute bin positions.

C.3 Orthogonal Series Estimation

Histograms yield a single average value within each bin, which leads to discontinuities. In order to construct smoother approximations of the underlying PDF, it is possible to directly estimate a higher-order function within each bin. By choosing the functions at neighboring bins to match at the transitions, we can further construct approximations of the PDF which are differentiable everywhere.

To compute higher-order approximations, we decompose the PDF within each bin b as a weighted sum of *basis functions* $\Psi_{b,j}(x)$:

$$\hat{f}(x) = \sum_j f_{b,j}\,\Psi_{b,j}(x). \tag{C.2}$$

The $f_{b,j}$ terms are coefficients which scale the contribution of each basis function j within each bin b. These are defined as the inner product of the density function f and the *dual* basis functions $\tilde{\Psi}_{b,j}$:

$$f_{b,j} = \int f(x)\,\tilde{\Psi}_{b,j}\,dx. \tag{C.3}$$

The dual basis function $\tilde{\Psi}_{b,j}$ are obtained using an orthogonality constraint. Specifically, for any fixed bin b the inner product of $\tilde{\Psi}_{b,j}$ with any of the basis functions $\Psi_{b,j}$ should be:

$$\int \tilde{\Psi}_{b,j}(x)\,\Psi_{b,k}(x)\,dx = \delta_{j,k}. \tag{C.4}$$

The coefficients can be estimated from the random samples using a Monte Carlo estimate of Equation C.3:

$$f_{b,j} \approx \frac{1}{N}\sum_{i=0}^{N-1}\frac{f(x_i)\,\tilde{\Psi}_{b,j}(x_s)}{pdf(x_i)} \tag{C.5}$$

$$\approx \frac{1}{N}\sum_{i=0}^{N-1}\tilde{\Psi}_{b,j}(x_i). \tag{C.6}$$

Relation to Histograms. In the case of a constant basis function within each bin, the orthogonal series estimator reverts to the histogram method. In this case the single dual basis function for bin b is

$$\tilde{\Psi}_b(x) = \begin{cases} \frac{1}{h} & \text{if } x_b \le x < x_{b+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{C.7}$$

It is easy to see that this basis function results in the histogram estimator in Equation C.1.

C.4 Naïve Estimator

Another generalization of the histogram method, which Silverman [1986] calls the *naïve estimator*, addresses the choice of bin locations. The main idea behind this method is to use the estimation point to adaptively determine the bin locations, thereby eliminating it as an extra parameter. This estimator can be written as:

$$\hat{f}(x) = \frac{n_x}{N2h}, \tag{C.8}$$

where n_x is the number of sample points which fall within the interval $[x-h, x+h)$. Comparing the above estimator to Equation C.1 we see that it is equivalent to a histogram where the estimation point x is used as the center of the bin, and the bin width is $2h$. Therefore, the naïve estimator is always globally a valid PDF, i.e., it is non-negative and integrates to one.

It can be informative to rewrite Equation C.8 by introducting the weighting function w:

$$w(t) = \begin{cases} \frac{1}{2} & \text{if } |t| < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{C.9}$$

Using this notation, we can express the naïve estimator as

$$\hat{f}(x) = \frac{1}{Nh} \sum_{i=0}^{N-1} w\left(\frac{x - x_i}{h}\right), \tag{C.10}$$

where x_i are the data samples. In this form, it is easy to see that the naïve estimator places a "box" of width $2h$ and height $(2hN)^{-1}$ at each data point and sums up the contributions. This

interpretation is useful in deriving the kernel estimator, which we discuss in the next section.

C.5 Kernel Estimator

Though the naïve estimator eliminates the problem of choosing the bin locations, it does not address some of the other limitations of the histogram method. The kernel method generalizes the naïve estimator to eliminate the discontinuous nature of the resulting PDF.

By examining Equation C.10 we observe that the reason for discontinuities is due to our particular choice of weighting function w, which has zero derivatives and discontinuities at $|x| = 1$. In fact, it is easy to show that the naïve estimator inherits *all* the differential properties of the weighting function since it is a simple sum. We can therefore improve the smoothness of the estimator by improving the smoothness of the weighting function. We accomplish this by replacing w with a smooth *kernel* function K. Our only restriction is that K must integrate to one:

$$\int_{-\infty}^{\infty} K(t)\, dt = 1. \tag{C.11}$$

With this restriction in place, we can define the kernel estimator as

$$\hat{f}(x) = \frac{1}{Nh} \sum_{i=0}^{N-1} K\Big(\frac{x - x_i}{h}\Big), \tag{C.12}$$

where h controls the amount of smoothing and is called the *window width* or *bandwidth* of the kernel. To make the notation more concise, it is often convenient to set $K_h(t) = (h^{-1})K(h^{-1}t)$ and express the kernel estimator simply as

$$\hat{f}(x) = \frac{1}{N} \sum_{i=0}^{N-1} K_h(x - x_i). \tag{C.13}$$

Typically we will choose K to be a smooth, symmetric function, which has a strong influence at $x = x_i$ and decreased influence as the distance $x - x_i$ increases. Some common examples include the normalized Gaussian kernel, the bi-weight kernel, and the Epanechnikov kernel [Silverman, 1986]. Analogous to our interpretation of the naïve estimator as a sum of

"boxes" at the data point, the kernel estimator is a sum of smooth "bumps" at the data points. Also note that if we use a constant kernel, then the kernel method reverts to the naïve estimator.

The kernel method can also be interpreted as a blurring process, or a convolution operation on the data points. If we consider the data points as Dirac delta impulses, δ, then the kernel method can be written as

$$\hat{f}(x) = K_h(x) \star \left(\frac{1}{N} \sum_{i=0}^{N-1} \delta(x - x_i) \right) \tag{C.14}$$

$$= \int_{-\infty}^{\infty} K_h(x) \left(\frac{1}{N} \sum_{i=0}^{N-1} \delta(x - x_i) \right) dx, \tag{C.15}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} K_h(x - x_i). \tag{C.16}$$

C.6 Locally Adaptive Estimators

In developing all the estimators so far, we have ignored one important consideration. The window width, or, in the case of the histogram the bin width, plays an important role in the behavior of the resulting density function.

At a high level, the bandwidth determines the amount of smoothing that is performed on the sample data. It is important to set this parameter properly. Choosing a value that is too small will result in under-smoothing and erratic fluctuations in the estimator which are not present in the original PDF. On the other hand, using a very large value may over-smooth the data, potentially eliminating important features of the underlying PDF. In practice it is very hard to choose an optimal bandwidth without knowing something about the density function itself.

More formally, this tradeoff can be expressed as trying to minimize the sum of the global bias and variance of the estimator. As we increase the bandwidth, we are smoothing the true PDF, which reduces variance but introduces bias. In contrast, if we decrease the bandwidth, we decrease the bias but increase variance. When we need to choose a bandwidth parameter for the whole domain of the data, a common technique is to try to minimize the *mean integrated squared*

error:

$$\text{MISE}(\hat{f}) = E\int \left[\hat{f}(x) - f(x)\right]^2 \, dx, \tag{C.17}$$

$$= \int V\left[\hat{f}(x)\right] + \beta\left[\hat{f}(x)\right]^2 \, dx, \tag{C.18}$$

which is the sum of the integrated variance and the integrated squared bias. Computing the MISE exactly requires information about the unknown true density f. However, since the density estimator $\hat{f}$ is a function of a random variable, it is itself a random variable. We can therefore apply the tools from Appendix A to estimate the variance and the bias of $\hat{f}$.

Unfortunately, a single bandwidth may not be optimal for all regions of the domain. For instance, a single bandwidth value may over-smooth features in high density regions and, at the same time, under-smooth regions in the "tails" of the distribution where few samples are present. Locally adaptive methods attempt to address this issue by allowing the bandwidth to change across the domain of the PDF. We will consider two different approaches for adaptively modify the kernel bandwidth [Jones, 1990]. The first class of techniques, called *balloon estimators*, vary the bandwidth based on the evaluation point x. The second set of techniques, called *sample-point estimators*, vary the bandwidth based on the data points x_i.

C.6.1 Balloon Estimator

The general form of the balloon estimator is given by

$$\hat{f}(x) = \frac{1}{Nh(x)} \sum_{i=0}^{N-1} K\left(\frac{x - x_i}{h(x)}\right), \tag{C.19}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} K_{h_x}(x - x_i), \tag{C.20}$$

where $h(x)$ is the bandwidth as a function of x, the evaluation location.

The balloon estimator was introduced by Loftsgaarden and Quesenberry [1965] in the form of the k^{th} nearest neighbor estimator, which can be written in the form of Equation C.19 by using a constant kernel and setting $h(x) = d_k(x)$ where $d_k(x)$ returns the distance to the k^{th} nearest data point to x. In fact, Silverman [1986] refers to (a slightly more restricted version of)

Equation C.19 as the *generalized k^{th} nearest neighbor estimator*.

The balloon estimator unfortunately suffers from a number of inefficiencies, especially in the univariate case. Firstly, the PDF derived using a balloon estimator will not, in general, integrate to one over the entire domain. If a *global* estimate of the PDF is needed, then this can be problematic. In computer graphics, however, this is not generally a major problem since we typically only care about the *pointwise* behavior of the estimated density. Another problem with the nearest neighbor estimator is that the bandwidth is a discontinuous function, and these discontinuities manifest themselves directly in the resulting PDF. This is true even if the chosen kernel is itself smooth. In computer graphics this can lead to mach banding and other visual artifacts.

C.6.2 Sample-point Estimator

The second type of local bandwidth estimator is called the *sample-point estimator*. The general form of the sample-point estimator is given by

$$\hat{f}(x) = \frac{1}{N}\sum_{i=0}^{N-1}\frac{1}{h(x_i)}K\left(\frac{x-x_i}{h(x_i)}\right), \tag{C.21}$$

$$= \frac{1}{N}\sum_{i=0}^{N-1}K_{h_{x_i}}(x-x_i). \tag{C.22}$$

Note that the only difference to the balloon estimator in Equation C.19 is that the bandwidth $h(x_i)$ is a function of the sample points x_i and not the evaluation point x. The sample-point estimator was first introduced by Breiman et al. [1977] under the name of the *variable kernel method*.

Just as in the regular kernel estimator, the sample-point estimator places a kernel at each data point, but these kernels are allowed to vary in size from one data point to another. Typically, the bandwidth of a data point is chosen based on the local density of samples in the vicinity. This requires performing a *pilot estimate* to compute the local density at each data point and assign the bandwidths.

Sample-point estimators do have a number of benefits over balloon estimators. Firstly, since each kernel is normalized, the estimator itself is a valid PDF, which integrates to one. Furthermore, unlike the balloon estimator, the sample-point estimator inherits all the differential

properties of the kernel functions and can therefore be made fully continuious. Another practical advantage of sample-point estimators is that the extent of each sample point is known *before* density evaluation begins. This often allows for simpler and more efficient data structures to be used during density evaluation. The beam radiance estimate developed in Chapter 8 exploits exactly this property of sample-point estimators.

Bibliography

Thomas Annen, Jan Kautz, Frédo Durand, and Hans-Peter Seidel. Spherical harmonic gradients for mid-range illumination. In Alexander Keller and Henrik Wann Jensen, editors, *Eurographics Symposium on Rendering*, pages 331–336, Norrköping, Sweden, 2004. Eurographics Association. ISBN 3-905673-12-6. 35, 72

Arthur Appel. Some techniques for shading machine renderings of solids. In *AFIPS 1968 Spring Joint Computer Conference*, volume 32, pages 37–45, 1968. 20

Okan Arikan, David A. Forsyth, and James F. O'Brien. Fast and detailed approximate global illumination by irradiance decomposition. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, pages 1108–1114. ACM Press, July 2005. doi: 10.1145/1073204.1073319. URL `http://doi.acm.org/10.1145/1073204.1073319`. 47, 48, 52

James Arvo. The irradiance jacobian for partially occluded polyhedral sources. In *Computer Graphics (Proceedings of SIGGRAPH 94)*, pages 343–350. ACM Press, 1994. ISBN 0-89791-667-0. doi: 10.1145/192161.192250. URL `http://dx.doi.org/10.1145/192161.192250`. 72, 102

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. 69

Neeta Bhate and A. Tokuta. Photorealistic volume rendering of media with directional scattering. In Alan Chalmers, Derek Paddon, and François Sillion, editors, *Rendering Techniques '92*, Eurographics, pages 227–246. Consolidation Express Bristol, 1992. 69

Miguel A. Blanco, M. Florez, and M. Bermejo. Evaluation of the rotation matrices in the basis of real spherical harmonics. *Journal Molecular Structure (Theochem)*, 419:19–27, December 1997. doi: 10.1016/S0166-1280(97)00185-1. URL `http://dx.doi.org/10.1016/S0166-1280(97)00185-1`. 34, 176

Philippe Blasi, Bertrand Le Saëc, and Christophe Schlick. A rendering algorithm for discrete volume density objects. *Computer Graphics Forum (Eurographics '93)*, 12(3):201–210, 1993. 64

James F. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. *Computer Graphics (Proceedings of SIGGRAPH 82)*, 16(3):21–29, 1982. ISSN 0097-8930. doi: 10.1145/965145.801255. URL `http://doi.acm.org/10.1145/965145.801255`. 3, 68

Pierre Bouguer. Essai d'optique sur la gradation de la lumiere. *Jombert, Paris, reprinted in: Les maıtres de la pensee scientifique, Paris*, 1729. 59

Leo Breiman, William Meisel, and Edward Purcell. Variable kernel estimates of multivariate densities. *Technometrics*, 19(2):135–144, May 1977. 139, 186

Brian Cabral, Nelson Max, and Rebecca Springmeyer. Bidirectional reflection functions from surface bump maps. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, pages 273–281. ACM Press, July 1987. ISBN 0-89791-227-6. doi: 10.1145/37401.37434. URL `http://doi.acm.org/10.1145/37401.37434`. 167

Eva Cerezo, Frederic Pérez, Xavier Pueyo, Francisco J. Seron, and François X. Sillion. A survey on participating media rendering techniques. *The Visual Computer*, 21(5):303–328, June 2005. doi: 10.1007/s00371-005-0287-1. URL `http://dx.doi.org/10.1007/s00371-005-0287-1`. 68

Subrahmanyan Chandrasekhar. *Radiative Transfer*. Dover Publications, New York, 1960. 66, 68, 69

Shenchang Eric Chen, Holly E. Rushmeier, Gavin Miller, and Douglass Turner. A progressive multi-pass method for global illumination. In Thomas W. Sederberg, editor, *Computer Graphics (Proceedings of SIGGRAPH 91)*, volume 25, pages 165–174. ACM Press, July 1991. ISBN 0-89791-436-8. doi: 10.1145/122718.122737. URL `http://doi.acm.org/10.1145/122718.122737`. 21

Kenneth Chiu, Peter Shirley, and Changyaw Wang. *Multi-jittered Sampling*, pages 370–374. Graphics Gems Series. Academic Press Professional, Inc., San Diego, CA, USA, 1994. ISBN 0-12-336155-9. 162

Cheol Ho Choi, Joseph Ivanic, Mark S. Gordon, and Klaus Ruedenberg. Rapid and stable determination of rotation matrices between spherical harmonics by direct recursion. *The Journal of Chemical Physics*, 111(19):8825–8831, 1999. doi: 10.1063/1.480229. URL `http://dx.doi.org/10.1063/1.480229`. 34, 176

Per H. Christensen. Faster photon map global illumination. *Journal of Graphics Tools*, 4(3):1–10, 1999. 97

Per H. Christensen. Adjoints and importance in rendering: an overview. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):329–340, July 2003. ISSN 1077-2626. doi: 10.1109/TVCG.2003.1207441. URL `http://dx.doi.org/10.1109/TVCG.2003.1207441`. 122

Per H. Christensen, Eric J. Stollnitz, and David H. Salesin. Global illumination of glossy environments using wavelets and importance. *ACM Transactions on Graphics*, 15(1):37–71, 1996. ISSN 0730-0301. doi: 10.1145/226150.226153. URL `http://doi.acm.org/10.1145/226150.226153`. 20

Michael F. Cohen and Donald P. Greenberg. The hemi-cube; a radiosity solution for complex environments. *Computer Graphics (Proceedings of SIGGRAPH 85)*, 19(3):31–40, August 1985. ISSN 0097-8930. doi: 10.1145/325165.325171. URL `http://doi.acm.org/10.1145/325165.325171`. 20

Edward Uhler Condon and George Shortley. *The Theory of Atomic Spectra*. Cambridge University Press, 1951. 169

Robert L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, 5(1): 51–72, January 1986. ISSN 0730-0301. doi: 10.1145/7529.8927. URL http://doi.acm.org/10.1145/7529.8927. 21, 162

Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *Computer Graphics (Proceedings of SIGGRAPH 84)*, pages 137–145, July 1984. ISSN 0097-8930. doi: 10.1145/964965.808590. URL http://doi.acm.org/10.1145/964965.808590. 21

Leonardo Da Vinci. *A Treatise on Painting*. 1651. 1

Mark A. Z. Dippé and Erling Henry Wold. Antialiasing through stochastic sampling. *Computer Graphics (Proceedings of SIGGRAPH 85)*, 19(3):69–78, 1985. ISSN 0097-8930. doi: 10.1145/325165.325182. URL http://doi.acm.org/10.1145/325165.325182. 164

Frédo Durand, Nicolas Holzschuch, Cyril Soler, Eric Chan, and François Sillion. A frequency analysis of light transport. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, 24 (3):1115–1126, August 2005. ISSN 0730-0301. doi: http://doi.acm.org/10.1145/1073204.1073320. URL 10.1145/1073204.1073320. 72

Philip Dutré, Philippe Bekaert, and Kavita Bala. *Advanced Global Illumination*. AK Peters, Ltd., second edition, 2006. 149, 178

David S. Ebert, Kenton F. Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. *Texturing & Modeling: A Procedural Approach, Third Edition (The Morgan Kaufmann Series in Computer Graphics)*. Morgan Kaufmann, December 2002. ISBN 1558608486. 94

Jeppe Revall Frisvad, Niels Jørgen Christensen, and Henrik Wann Jensen. Computing the scattering properties of participating media using lorenz-mie theory. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, page 60, New York, NY, USA, 2007. ACM. doi: 10.1145/1275808.1276452. URL http://dx.doi.org/10.1145/1275808.1276452. 66

Pascal Gautron, Jaroslav Křivánek, Sumanta N. Pattanaik, and Kadi Bouatouch. A novel hemispherical basis for accurate and efficient rendering. In Alexander Keller and Henrik Wann Jensen, editors, *Eurographics Symposium on Rendering*, pages 321–330, Norrköping, Sweden, 2004. Eurographics Association. ISBN 3-905673-12-6. 33, 34

Andrew S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann, 1995. 89

Samuel Glasstone and Alexander Sesonske. *Nuclear Reactor Engineering*. Van Nostrand Company, 1955. 3

Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modeling the interaction of light between diffuse surfaces. In *Computer Graphics (Proceedings of ACM SIGGEAPH 84)*, pages 213–222, New York, NY, USA, 1984. ACM Press. ISBN 0-89791-138-5. doi: 10.1145/800031.808601. URL http://doi.acm.org/10.1145/800031.808601. 20

Steven J. Gortler, Peter Schröder, Michael F. Cohen, and Pat Hanrahan. Wavelet radiosity. In *Computer Graphics (Proceedings of SIGGRAPH 93)*, pages 221–230, New York, NY, USA, 1993. ACM. ISBN 0-89791-601-8. doi: 10.1145/166117.166146. URL http://doi.acm.org/10.1145/166117.166146. 20

Robin Green. Spherical harmonic lighting: The gritty details. *Archives of the Game Developers Conference*, March 2003. 167, 176

Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. In *Computer Graphics (Proceedings of SIGGRAPH 93)*, pages 165–174, New York, NY, USA, 1993. ACM. ISBN 0-89791-601-8. doi: 10.1145/166117.166139. URL `http://dx.doi.org/10.1145/166117.166139`. 70

Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):197–206, 1991. ISSN 0097-8930. doi: 10.1145/127719.122740. URL `http://doi.acm.org/10.1145/127719.122740`. 20

Louis George Henyey and Jesse Leonard Greenstein. Diffuse radiation in the galaxy. *Astrophysics*, 93:70–83, 1941. 63

Nicolas Holzschuch and François Sillion. Accurate computation of the radiosity gradient with constant and linear emitters. In Patrick M. Hanrahan and Werner Purgathofer, editors, *Rendering Techniques '95*, Eurographics, pages 186–195. Springer-Verlag Wien New York, 1995. 72

Nicolas Holzschuch and François Sillion. An exhaustive error-bounding algorithm for hierarchical radiosity. *Computer Graphics Forum*, 17(4), 1998. 72

Hoyt C. Hottel and Adel F. Sarofim. *Radiative Transfer*. McGraw Hill, New York, 1967. 69

Homan Igehy. Tracing ray differentials. In *Computer Graphics (Proceedings of SIGGRAPH 99)*, pages 179–186, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-48560-5. doi: 10.1145/311535.311555. URL `http://doi.acm.org/10.1145/311535.311555`. 72

David S. Immel, Michael F. Cohen, and Donald P. Greenberg. A radiosity method for non-diffuse environments. *Computer Graphics (Proceedings of SIGGRAPH 86)*, 20(4):133–142, 1986. ISSN 0097-8930. doi: 10.1145/15886.15901. URL `http://doi.acm.org/10.1145/15886.15901`. 20

Joseph Ivanic and Klaus Ruedenberg. Rotation matrices for real spherical harmonics. direct determination by recursion. *Journal of Physical Chemistry*, 100(15):6342–6347, 1996. ISSN 0022-3654. 34, 176

Joseph Ivanic and Klaus Ruedenberg. Additions and corrections : Rotation matrices for real spherical harmonics. *J. Phys. Chem. A*, 102(45):9099–9100, 1998. 176

James Hopwood Jeans. The equations of radiative transfer of energy. *Monthly Notices of the Royal Astronomical Society*, 78:28–36, 1917. 69

Henrik W. Jensen, Frank Suykens, and Per H. Christensen. A practical guide to global illumination using photon mapping. In *SIGGRAPH 2001 Course Notes # 38*. ACM, August 2001a. 122

Henrik Wann Jensen. Global illumination using photon maps. In Xavier Pueyo and Peter Schröder, editors, *Rendering Techniques '96*, Eurographics, pages 21–30. Springer-Verlag Wien New York, 1996. 119, 178

Henrik Wann Jensen. Rendering caustics on non-Lambertian surfaces. *Computer Graphics Forum*, 16(1):57–64, March 1997. ISSN 0167-7055 (print), 1467-8659 (electronic). 119

Henrik Wann Jensen. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Ltd., Natick, MA, USA, 2001. ISBN 1-56881-147-0. 21, 52, 119, 120, 124, 178

Henrik Wann Jensen and Niels Jørgen Christensen. Photon maps in bidirectional Monte Carlo ray tracing of complex objects. *Computers & Graphics*, 19(2):215–224, 1995. doi: 10.1016/0097-8493(94)00145-O. URL http://dx.doi.org/10.1016/0097-8493(94)00145-O. 119

Henrik Wann Jensen and Per H. Christensen. Efficient simulation of light transport in scences with participating media using photon maps. In *Computer Graphics (Proceedings of SIGGRAPH 98)*, pages 311–320, New York, NY, USA, 1998. ACM Press. ISBN 0-89791-999-8. doi: 10.1145/280814.280925. URL http://dx.doi.org/10.1145/280814.280925. 70, 119, 124, 128, 135, 136, 178

Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *Computer Graphics (Proceedings of SIGGRAPH 2001)*, pages 511–518, New York, NY, USA, 2001b. ACM Press. ISBN 1-58113-374-X. doi: 10.1145/383259.383319. URL http://dx.doi.org/10.1145/383259.383319. 3, 70

M.C. Jones. Variable kernel density estimates and variable kernel density estimates. *Australian & New Zealand Journal of Statistics*, 32(3):361–371, 1990. doi: 10.1111/j.1467-842X.1990.tb01031.x. URL http://dx.doi.org/10.1111/j.1467-842X.1990.tb01031.x. 185

Florian Kainz, Wojciech Jarosz, and Rod Bogart. *Technical Introduction to OpenEXR*. Lucas Digital Ltd. LLC., 2006. URL http://www.openexr.com. 88

James T. Kajiya. The rendering equation. In *Computer Graphics (Proceedings of SIGGRAPH 86)*, pages 143–150, New York, NY, USA, 1986. ACM Press. ISBN 0-89791-196-2. doi: 10.1145/15922.15902. URL http://dx.doi.org/10.1145/15922.15902. 16, 21

James T. Kajiya and Brian P. Von Herzen. Ray tracing volume densities. In *Computer Graphics (Proceedings of SIGGRAPH 84)*, pages 165–174, New York, NY, USA, 1984. ACM Press. ISBN 0-89791-138-5. doi: 10.1145/800031.808594. URL http://dx.doi.org/10.1145/800031.808594. 69

Toshiaki Kato. Photon mapping in kilauea. In *Siggraph 2002, Course Notes No. 43, A Practical Guide to Global Illumination using Photon Mapping organized by Jensen, H.W.*, pages 159–191, 2002. 23, 48, 49

Jan Kautz, Peter-Pike Sloan, and John Snyder. Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In *Proceedings of the 13th Eurographics workshop on Rendering*, pages 291–296. Eurographics Association, 2002. ISBN 1-58113-534-3. 173, 176

Alexander Keller. *Quasi-Monte Carlo Methods for Photorealistic Image Synthesis*. Ph.D. thesis, Shaker Verlag Aachen, 1998. 163

Alexander Keller. Strictly deterministic sampling methods in computer graphics. *SIGGRAPH 2003 Course Notes, Course# 44: Monte Carlo Ray Tracing*, 2003. 163

Alexander Keller and Ingo Wald. Efficient importance sampling techniques for the photon map. In *Proceedings of the Fifth Fall Workshop on Vision, Modeling, and Visualization*, pages 271–279. IEEE, November 2000. 122

David Kirk and James Arvo. Unbiased sampling techniques for image synthesis. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, pages 153–156, New York, NY, USA, 1991. ACM. ISBN 0-89791-436-8. doi: 10.1145/122718.122735. URL http://doi.acm.org/10.1145/122718.122735. 164

Vladimir Kourgnaoff. *Basic Methods in Transfer Problems*. Oxford University Press, London, 1952. 68, 69

Jaroslav Křivánek, Pascal Gautron, Kadi Bouatouch, and Sumanta Pattanaik. Improved radiance gradient computation. In *SCCG '05: Proceedings of the 21th spring conference on Computer graphics*, pages 155–159, New York, NY, USA, 2005a. ACM Press. ISBN 1-59593-203-6. 33, 35, 39, 40, 43, 102, 104, 117

Jaroslav Křivánek, Pascal Gautron, Sumanta Pattanaik, and Kadi Bouatouch. Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics*, 11(5):550–561, 2005b. ISSN 1077-2626. doi: 10.1109/TVCG.2005.83. URL http://dx.doi.org/10.1109/TVCG.2005.83. 33, 35, 72, 102

Jaroslav Křivánek, Jaakko Konttinen, Kadi Bouatouch, Sumanta Pattanaik, and Jiří Žára. Fast approximation to spherical harmonic rotation. In *SCCG '06: Proceedings of the 22nd spring conference on Computer graphics*, New York, NY, USA, 2005c. ACM Press. 176

Jaroslav Křivánek, Kadi Bouatouch, Sumanta N. Pattanaik, and Jiří Žára. Making radiance and irradiance caching practical: Adaptive caching and neighbor clamping. In Tomas Akenine-Möller and Wolfgang Heidrich, editors, *Eurographics Workshop/ Symposium on Rendering*, pages 127–138, Nicosia, Cyprus, 2006. Eurographics Association. ISBN 3-905673-35-5. doi: 10.2312/EGWR/EGSR06/127-138. URL http://dx.doi.org/10.2312/EGWR/EGSR06/127-138. 91, 116

Eric Lafortune. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*. Ph.D. thesis, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, February 1996. 21

Eric P. Lafortune and Yves D. Willems. Bi-directional path tracing. In H. P. Santo, editor, *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93)*, pages 145–153, Alvor, Portugal, 1993. 21

Eric P. Lafortune and Yves D. Willems. Rendering participating media with bidirectional path tracing. In Xavier Pueyo and Peter Schröder, editors, *Rendering Techniques '96*, Eurographics, pages 91–100. Springer-Verlag Wien New York, 1996. 70

K. D. Lathrop. Ray effects in discrete ordinates equations. *Nuclear Science and Engineering*, 32:357–369, 1968. 69

D. O. Loftsgaarden and C. P. Quesenberry. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3):1049–1051, 1965. ISSN 00034851. doi: 10.1214/aoms/1177700079. URL http://www.jstor.org/stable/2238216. 185

Ludvig Lorenz. Lysbevægelser i og uden for en af plane lysbølger belyst kugle. *Videnskabernes Selskabs Skrifter*, 6:2–62, 1890. 65

Thomas Murray MacRobert and Ian Naismith Sneddon. *Spherical harmonics: an elementary treatise on harmonic functions, with applications*. Pergamon Press, Oxford, England, third edition, 1967. 167

Nelson Max. Efficient light propagation for multiple anisotropic volume scattering. In Georgios Sakas, Peter Shirley, and Stefan Müller, editors, *Rendering Techniques '94*, Eurographics, pages 87–104. Springer-Verlag Berlin Heidelberg New York, 1994. 69

Michael D. McKay, Richard J. Beckman, and William Jay Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979. 162

Nicholas Metropolis. The beginning of the Monte Carlo method. *Los Alamos Science*, (15):125–130, 1987. URL http://library.lanl.gov/cgi-bin/getfile?number15.htm. 20, 149

Nicholas Metropolis and Stanisław Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949. ISSN 01621459. doi: 10.2307/2280232. URL http://dx.doi.org/10.2307/2280232. 20, 149

Gustav Mie. Beiträge zur optik trüber medien, speziell kolloidaler metallösungen. *Annalen der Physik*, 330:377–445, 1908. doi: 10.1002/andp.19083300302. URL http://dx.doi.org/10.1002/andp.19083300302. 65

Subhash C. Mishra1 and Manohar Prasad. Radiative heat transfer in participating media – a review. *Sadhana*, 23(2):213–232, April 1998. ISSN 0256-2499. doi: 10.1007/BF02745682. URL http://dx.doi.org/10.1007/BF02745682. 68

Don P. Mitchell. Generating antialiased images at low sampling densities. *Computer Graphics (Proceedings of SIGGRAPH 87)*, 21(4):65–72, 1987. ISSN 0097-8930. doi: 10.1145/37402.37410. URL http://doi.acm.org/10.1145/37402.37410. 164

Don P. Mitchell. Spectrally optimal sampling for distribution ray tracing. *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):157–164, 1991. ISSN 0097-8930. doi: 10.1145/127719.122736. URL http://doi.acm.org/10.1145/127719.122736. 162, 164

Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial Mathematics, 1992. 163

Tomoyuki Nishita and Eihachiro Nakamae. Continuous tone representation of three-dimensional objects taking account of shadows and interreflection. *Computer Graphics (Proceedings of SIGGRAPH 85)*, 19(3):23–30, 1985. ISSN 0097-8930. doi: 10.1145/325165.325169. URL http://doi.acm.org/10.1145/325165.325169. 20

Tomoyuki Nishita, Yasuhiro Miyawaki, and Eihachiro Nakamae. A shading model for atmospheric scattering considering luminous intensity distribution of light sources. *Computer Graphics (Proceedings of SIGGRAPH 87)*, 21(4):303–310, 1987. ISSN 0097-8930. doi: 10.1145/37402.37437. URL http://doi.acm.org/10.1145/37402.37437. 3, 66

Art B. Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, 2(2):439–452, 1992. 162

James Painter and Kenneth R. Sloan. Antialiased ray tracing by adaptive progressive refinement. *Computer Graphics (Proceedings of SIGGRAPH 89)*, 23(3):281–288, 1989. ISSN 0097-8930. doi: 10.1145/74334.74362. URL http://doi.acm.org/10.1145/74334.74362. 164

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. ISSN 1049-8907. 70

Mark Pauly, Thomas Kollig, and Alexander Keller. Metropolis light transport for participating media. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, pages 11–22, London, UK, 2000. Springer-Verlag. ISBN 3-211-83535-0. 70, 74, 129, 131

Ingmar Peter and Georg Pietrek. Importance driven construction of photon maps. In *Rendering Techniques '98 (Proceedings of the Ninth Eurographics Workshop on Rendering)*, pages 269–280. Springer-Verlag, 1998. 122

Matt Pharr and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. ISBN 012553180X. 64, 149

Didier Pinchon and Philip E Hoggan. Rotation matrices for real spherical harmonics: general rotations of atomic orbitals in space-fixed axes. *Journal of Physics A: Mathematical and Theoretical*, 40(7):1597–1610, 2007. doi: 10.1088/1751-8113/40/7/011. URL http://dx.doi.org/10.1088/1751-8113/40/7/011. 176

Simon Premoze, Michael Ashikhmin, Ravi Ramamoorthi, and Shree K. Nayar. Practical rendering of multiple scattering effects in participating media. In Alexander Keller and Henrik Wann Jensen, editors, *Eurographics Symposium on Rendering*, pages 363–374, Norrköping, Sweden, 2004. Eurographics Association. ISBN 3-905673-12-6. 70

William Press, Saul Teukolsky, William Vetterling, and Brian Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, UK, 2nd edition, 1992. 169

Ravi Ramamoorthi. *A Signal-Processing Framework for Forward and Inverse Rendering*. PhD thesis, Stanford University, 2002. 167

Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Computer Graphics (Proceedings of SIGGRAPH 2001)*, pages 497–500, New York, NY, USA, 2001. ACM Press. ISBN 1-58113-374-X. doi: 10.1145/383259.383317. URL http://doi.acm.org/10.1145/383259.383317. 172

Ravi Ramamoorthi, Dhruv Mahajan, and Peter Belhumeur. A first-order analysis of lighting, shading, and shadows. *ACM Transactions on Graphics*, 26(1):2, January 2007. ISSN 0730-0301. doi: 10.1145/1189762.1189764. URL http://doi.acm.org/10.1145/1189762.1189764. 72, 102, 117

John William Strutt Lord Rayleigh. On the scattering of light by small particles. *Philosophical Magazine*, 61:447–454, 1871. 64

Mark C. Reichert. A two-pass radiosity method driven by lights and viewer position. Master's thesis, Cornell University, January 1992. 22

Holly E. Rushmeier. *Realistic Image Synthesis for Scenes with Radiatively Participating Media*. Ph.d. thesis, Cornell University, 1988. 70

Holly E. Rushmeier and Kenneth E. Torrance. The zonal method for calculating light intensities in the presence of a participating medium. *Computer Graphics (Proceedings of SIGGRAPH 87)*, 21(4):293–302, July 1987. ISSN 0097-8930. doi: 10.1145/37402.37436. URL `http://doi.acm.org/10.1145/37402.37436`. 3, 69

Holly E. Rushmeier, Charles W. Patterson, and Aravindan Veerasamy. Geometric simplification for indirect illumination calculations. In *Proceedings of Graphics Interface '93*, 1994. 46

Bahaa E. A. Saleh and Malvin Carl Teich. *Fundamentals of Photonics*. Wiley-Interscience, 2 edition, March 2007. 8

David W. Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley Series in Probability and Statistics. Wiley-Interscience, September 1992. ISBN 0471547700. 178

Peter Shirley. *Physically Based Lighting Calculations For Computer Graphics*. PhD thesis, University of Illinois, Urbana–Champaign, November 1990. 21, 162

Peter Shirley. Discrepancy as a quality measure for sample distributions. In *Eurographics '91*, pages 183–94. Elsevier Science Publishers, Amsterdam, North-Holland, 1991. 21, 162

Peter Shirley, Bretton Wade, Philip M. Hubbard, David Zareski, Bruce Walter, and Donald P. Greenberg. Global illumination via density estimation. *Rendering Techniques 95 (Proceedings of Eurographics Workshop on Rendering 95)*, pages 219–230, 1995. 178

Robert Siegel and John R. Howell. *Thermal Radiation Heat Transfer*. Taylor & Francis, 4th edition, 2002. ISBN 1560328398. 19

François X. 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. 69

François X. Sillion and Claude Puech. A general two-pass method integrating specular and diffuse reflection. *Computer Graphics (Proceedings of SIGGRAPH 89)*, 23(3):335–344, July 1989. ISSN 0097-8930. doi: 10.1145/74334.74368. URL `http://doi.acm.org/10.1145/74334.74368`. 21

François X. Sillion, James R. Arvo, Stephen H. Westin, and Donald P. Greenberg. A global illumination solution for general reflectance distributions. *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):187–196, 1991. ISSN 0097-8930. doi: 10.1145/127719.122739. URL `http://dx.doi.org/10.1145/127719.122739`. 20, 167

Bernard. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, New York, NY, 1986. 124, 142, 178, 182, 183, 185

Peter-Pike Sloan. Stupid spherical harmonics (SH) tricks. Game Developers Conference, February 2008. URL `http://www.ppsloan.org/publications/`. 167

Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002)*, 21(3):527–536, 2002. ISSN 0730-0301. doi: 10.1145/566654.566612. URL `http://doi.acm.org/10.1145/566654.566612`. 173

Brian Smits, James Arvo, and Donald Greenberg. A clustering algorithm for radiosity in complex environments. In *Computer Graphics (Proceedings of SIGGRAPH 94)*, pages 435–442, New York, NY, USA, July 1994. ACM Press. ISBN 0-89791-667-0. doi: 10.1145/192161.192277. URL http://doi.acm.org/10.1145/192161.192277. 20

Brian E. Smits, James R. Arvo, and David H. Salesin. An importance-driven radiosity algorithm. *Computer Graphics (Proceedings of SIGGRAPH 92)*, 26(2):273–282, July 1992. ISSN 0097-8930. doi: 10.1145/142920.134080. URL http://doi.acm.org/10.1145/142920.134080. 20

Jos Stam. Multiple scattering as a diffusion process. In Patrick M. Hanrahan and Werner Purgathofer, editors, *Rendering Techniques '95*, Eurographics, pages 41–50. Springer-Verlag Wien New York, 1995. 3, 70

Marc Stamminger, Philipp Slusallek, and Hans-Peter Seidel. Three point clustering for radiance computations. In George Drettakis and Nelson Max, editors, *Rendering Techniques '98*, Eurographics. Springer-Verlag Wien New York, 1998. 20

Bo Sun, Ravi Ramamoorthi, Srinivasa G. Narasimhan, and Shree K. Nayar. A practical analytic single scattering model for real-time rendering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, 24(3):1040–1049, 2005. ISSN 0730-0301. doi: 10.1145/1073204.1073309. URL http://dx.doi.org/10.1145/1073204.1073309. 70

Frank Suykens and Yves D. Willems. Density control for photon maps. In *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)*, pages 11–22. Springer-Verlag, 2000. 122

Eric Tabellion and Arnauld Lamorlette. An approximate global illumination system for computer generated films. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004)*, 23(3):469–476, 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015748. URL http://dx.doi.org/10.1145/1015706.1015748. ix, 23, 45, 46

Jeremy Tanner. *The Invention of Art History in Ancient Greece: Religion, Society and Artistic Rationalisation*. Cambridge University Press, 2006. ISBN 0521846145. 1

Takehiro Tawara, Karol Myszkowski, and Hans-Peter Seidel. Localizing the final gathering for dynamic scenes using the photon map. In Günther Greiner, editor, *Proceedings of the Vision, Modeling, and Visualization Conference 2002*, pages 69–46. Aka GmbH, 2002. ISBN 3-89838-034-3. 51

Takehiro Tawara, Karol Myszkowski, Kirill Dmitriev, Vlastimil Havran, Cyrille Damez, and Hans-Peter Seidel. Exploiting temporal coherence in global illumination. In *SCCG '04: Proceedings of the 20th spring conference on Computer graphics*, pages 23–33, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-967-5. doi: 10.1145/1037210.1037214. URL http://dx.doi.org/10.1145/1037210.1037214. 50

Michael Tinkham. *Group Theory and Quantum Mechanics*. Dover Publications, 2003. 167, 174

Eric Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, December 1997. 129, 131, 133, 149, 159, 162

Eric Veach and Leonidas J. Guibas. Bidirectional estimators for light transport. In Georgios Sakas, Peter Shirley, and Stefan Müller, editors, *Rendering Techniques '94*, Eurographics, pages 145–167. Springer-Verlag Berlin Heidelberg New York, 1994. 21

Eric Veach and Leonidas J. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Computer Graphics (Proceedings of SIGGRAPH 95)*, pages 419–428, New York, NY, USA, 1995. ACM Press. ISBN 0-89791-701-4. doi: 10.1145/218380.218498. URL `http://doi.acm.org/10.1145/218380.218498`. 21, 159

Ingo Wald, William R. Mark, Johannes Günther, Solomon Boulos, Thiago Ize, Warren Hunt, Steven G. Parker, and Peter Shirley. State of the art in ray tracing animated scenes. In *STAR Proceedings of Eurographics 2007*, pages 0–0, Prague, Czech Republic, September 2007. Eurographics Association. 140

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. *Computer Graphics (Proceedings of SIGGRAPH 87)*, 21(4):311–320, 1987. ISSN 0097-8930. doi: 10.1145/37402.37438. URL `http://dx.doi.org/10.1145/37402.37438`. 20, 21

Bruce Walter, Philip M. Hubbard, Peter Shirley, and Donald P. Greenberg. Global illumination using local linear density estimation. *ACM Transactions on Graphics*, 16(3):217–259, 1997. ISSN 0730-0301. doi: 10.1145/256157.256158. 178

Bruce Jonathan Walter. *Density estimation techniques for global illumination*. PhD thesis, Cornell University, Ithaca, NY, USA, 1998. Chair-Donald P. Greenberg. 178

Changyaw Wang and Kelvin Sung. Multi-stage n-rooks sampling method. *Journal of Graphics Tools*, 4(1):39–47, 1999. 162

Greg Ward. Real pixels. *Graphics Gems II*, pages 80–83, 1991. 88

Gregory J. Ward. The RADIANCE lighting simulation and rendering system. In Andrew Glassner, editor, *Computer Graphics (Proceedings of SIGGRAPH 94)*, pages 459–472, New York, NY, USA, July 1994. ACM Press. ISBN 0-89791-667-0. doi: 10.1145/192161.192286. URL `http://doi.acm.org/10.1145/192161.192286`. 25

Gregory J. Ward and Paul S. Heckbert. Irradiance gradients. In Alan Chalmers, Derek Paddon, and François Sillion, editors, *Rendering Techniques '92*, Eurographics, pages 85–98, Bristol, UK, 1992. Consolidation Express Bristol. ix, 31, 34, 35, 36, 41, 43, 44, 72, 102, 104, 106, 107, 109, 111, 112, 113

Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. *Computer Graphics (Proceedings of SIGGRAPH 88)*, 22(4):85–92, 1988. ISSN 0097-8930. doi: 10.1145/378456.378490. URL `http://doi.acm.org/10.1145/378456.378490`. 21, 23, 24, 25, 26, 27, 29, 87, 101

Turner Whitted. An improved illumination model for shaded display. *Commun. ACM*, 23(6): 343–349, 1980. ISSN 0001-0782. doi: 10.1145/358876.358882. URL `http://dx.doi.org/10.1145/358876.358882`. 20, 21, 164

Christopher R. Wyman. *Fast Local Approximation to Global Illumination.* PhD thesis, University of Utah, 2004. 167

Yand Li Hector Yee. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. Master's thesis, Cornell University, August 2000. 52

Jr. John I. Yellott. Spectral consequences of photoreceptor sampling in the rhesus retina. *Science*, 221(4608):382–385, 1983. doi: 10.1126/science.6867716. URL `http://dx.doi.org/10.1126/science.6867716`. 162

David Zareski, Bretton Wade, Philip Hubbard, and Peter Shirley. Efficient parallel global illumination using density estimation. In *PRS '95: Proceedings of the IEEE symposium on Parallel rendering*, pages 47–54, New York, NY, USA, 1995. ACM. ISBN 0-89791-774-1. doi: 10.1145/218327.218336. 178

Index