# Analytical, Wavelet and Frequency based Mathematical Models for Real-Time Rendering

Bo Sun

Submitted in partial fulfillment of the
Requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2008

ABSTRACT

Analytical, Wavelet and Frequency based Mathematical Models
for Real-Time Rendering

Bo Sun

Real-time rendering techniques are critical and highly desirable in interactive graphics applications such as video games, flight simulations, and interactive design softwares. Offline rendering solutions such as raytracing, path tracing, photon mapping, and other Monte Carlo methods can generate very realistic images and capture fairly complicated effects such as caustics, soft shadows, atmospherical scattering or subsurface scattering, and environment lighting but take hours or even days to run. For interactive applications, a large number of real-time rendering techniques have been proposed. They, however, either capture effects that are too simplistic, missing critical complicated effects, or make very constraining assumptions that limit them to specific application scenarios. There is a wide gap and a large unexplored area between slow offline renderers and particular real-time solutions. In this thesis, we take analytical, frequency and wavelet based approaches to investigate efficient algorithms for more complicated natural phenomena. Practically, we aim to provide efficient solutions/tools that are general enough to be combined with existing real-time rendering techniques and expand the domain of tractable effects. On a theoretical level, we try to identify traits of graphics computation and uncover insights about the optimal mathematical representations, leveraging the advancing power of graphics hardware. While open topics are plenty, we choose to focus on key effects that are the most lacking in the current state of art of rendering techniques. In particular, we present solutions to four challenging problems. First, we consider real-time rendering of scenes in scattering media, capturing the effects of light scattering in fog, mist and haze. We present a physically based analytic model that captures these effects while maintaining real time performance and the ease of use of the OpenGL fog model. Our method is based on an explicit analytic integration of the single scattering light transport equations for an isotropic point light source in a homogeneous scattering medium. Second, we introduce a novel near-field relighting framework. At the core of this framework is an affine double and triple product integral theory - an important generalization of triple product wavelet integrals that enables one of the product functions to be scaled and translated. Our theoretical development overturns the long-held belief that operations such as affine transformation are difficult in wavelets and require converting to and from the pixel domain. We show through detailed analysis that while simple analytic formulae are not easily available, there is considerable sparsity that we can exploit computationally. The canonical coupling coefficients we derived and our way of exploiting unstructured sparsity are all new powerful insights. We demonstrate practical implementation of an intuitive lighting design system coupled with near-field relighting capabilities. We also illustrate initial examples of wavelet importance sampling with near-field area lights, and image processing directly in the wavelet domain. Finally, we present two frequency based approaches to normal map filtering and dynamic soft shadowing. Our main theoretical contributions in these applications are respectively formulating the normal map filtering as a convolution in the frequency domain and developing the spherical harmonic exponentiation and logarithm techniques. Our analysis has revealed important technical characteristics of light transport and reflectance. Our analytical, frequency and wavelet based approaches have opened up new perspectives for rendering novel effects that are conventionally viewed as difficult to achieve.

# Contents

# List of Figures

# List of Tables

# Acknowledgements

My gratitude first goes to my advisors Prof. Ravi Ramamoorthi and Prof. Shree K. Nayar. I thank them for their relentless guidance, arduous support, and unwavering trust, for the countless days and nights they spent teaching and working with me. They showed me structured ways of approaching problems and the manners of learned scholars. This thesis would not have been possible without them. I also thank Prof. Peter N. Belhumeur, Prof. Srinivasa Narasimhan, Dr. John Snyder, and Prof. Eitan Grinspun for their guidance, discussions and constructive feedbacks on my work. I also thank Prof. Ravi Ramamoorthi, Prof. Shree K. Nayar, Prof. Peter N. Belhumeur, Prof. Srinivasa Narasimhan and Dr. John Snyder for being my thesis committee members.

My special thanks to my colleagues at the Columbia Vision and Graphics Center, Aner Ben-Artzi, Craig Donner, Kevin Egan, Charles Han, Li Zhang, Ryan Overback, Jin-Wei Gu, Simon Premoze, Sujit Kuthirummal, Sebastian Enrique, Kshitiz Garg, Akash Garg, Anne Fleming, Kalyan Sunkavalli, Assaf Zomet, Gurunandan Krishnan, Moshe Ben-Ezra, Ronny Goldenthal, Bernhard Thomaszewski, Mikls Bergou, David Harmon, Etienne Vouga, Ko Nishino, Vlad Branzoi, Dhruv Mahajan, Neeraj Kumar, Vijay Nagarajan, Dimitri Bitouk, and Michael Grossberg. I thank them for their generous assistance and kind accommodation of my flaws and mistakes.

Finally, my deepest gratitude goes to my parents, for their endless support, care, and love. This thesis is dedicated to them.

# Chapter 1

# Introduction

## 1.1 Motivation

The mission of computer graphics is to visualize the world around us. For over 20 years, a perennial challenge for graphics rendering has been to create high quality realistic-looking images in real time. Real-time rendering introduces a whole new dimension in "interactivity and flexibility". This dimension is critical in applications where the rendering output depends on users' input and instant and continuous visual feedback is demanded. One popular application is for the user to navigate through the scene as in computer games, moving and panning the camera as needed. However, "interactivity and flexibility" exceeds far beyond camera navigation. For example, the lighting environment may change interactively, as the sun rises and falls and the weather condition varies. Variations in lighting dramatically change the appearance of the scene. In addition, objects inside the scene may move around or change in geometry by themselves. For example, a character walks around or a bird flaps its wings, and their cast shadow patterns should change accordingly. Further, material properties may evolve with time, e.g., paint will dry, solidify, and eventually peel off a surface, leading to the sharp change in its specularities and color contrast as we observe in real life. The current state of real-time rendering techniques is far from being able to accommodate all these effects and render all natural phenomena.

Because of the complexity of light transport and the high dimensionality of the representations of real lighting and materials, the cost of synthesizing a realistic image is prohibitive. To properly simulate light transport, all lighting coming from the upper hemisphere has to be integrated and light bouncing between different objects, scattering through participating media and occlusion by floating blockers have to be accounted for. To better model real-world effects, complicated material models such as BRDFs [78], BTFs [25], and BSSRDFs [39] and high dimensional lighting representations such as light field [64] or lumigraph [35] are often needed. Offline methods such as raytracing [124], path tracing [51], photon-mapping [47] and radiosity [34] can simulate light transport and add to the ambience of scenes with effects such as intricate shadows, complex inter-reflections and volumetric scattering. However, they are too slow to be real time and do not suit the need of interactive applications such as video games, simulations, previsualization and design systems. Though rapid advancements have been made in the real-time rendering domain, either the effects they capture are too simplistic or their assumption are too constraining. Many methods are still limited to simple, often static, effects because of the lack of suitable mathematical representations and efficient computational algorithms. The intimidating cost of rendering more complicated effects has impeded them from being further included in more real-time applications. The computational bottleneck has also constrained designers' capabilities to fully realize their design concepts and have set interactively generated images even further away from being realistic. Thus, we conduct a systematic study on some of the key rendering problems, exploring their optimal mathematical representation and exploiting the computational aspects of light transport and the theoretical characteristics of illumination, shadows and materials properties.

## 1.2 A Walkthrough with the Reflection Equation

In what follows, we provide a quick walkthrough of our research detailed in the rest of the thesis, starting from the reflection equation to introduce specific rendering problems that we focus on.

### 1.2.1 Reflection Equation

The reflection equation forms the basis of most modern real-time rendering algorithms. We first review the reflection equation:

$$L_r(x,\omega_o) = \int_{\Omega} L_i(x,\omega_i)V(x,\omega_i)\rho(x,\omega_i,\omega_o)(\omega_i \cdot n(x))d\omega_i, \tag{1.1}$$

where x is the location of a vertex, or a pixel in the image, $\omega_o$ is the viewing direction, $\omega_i$ is the incident direction, $L_i$ is the incident radiance from the light source at a vertex or a pixel, $V$ is the visibility function, $\rho$ is the BRDF at location x, and $(\omega_i \cdot n(x))$ is the cosine of the incident angle. It is sometimes convenient to fold the cosine term $(\omega_i \cdot n(x))$ into the BRDF $\rho(x,\omega_i,\omega_o)$ and write the reflection equation in a simpler form:

$$L_r(x,\omega_o) = \int_{x,\Omega} L_i(\omega_i)V(x,\omega_i)\rho(x,\omega_i,\omega_o)d\omega_i. \tag{1.2}$$

The BRDF, $\rho$, plays an important role in the above integration and is worth mentioning. It was first introduced by [78]. The BRDF describes how incoming light is distributed after reflection from the material's surface. To be physically accurate, the BRDF must satisfy the first law of thermodynamics (conservation of energy) and Helmholtz's reciprocity principle for general BRDFs or Rayleigh's reciprocity principle for arbitrary surfaces. For each point and a given $x$, the BRDF is a function of both the incoming and outgoing directions and hence is four-dimensional. For its high dimensionality, for real-time purposes, three main approaches are often used to approximate the BRDF: complex analytic models (Blinn-Phong [12], Torrance-Sparrow microfacet [115], Ward's anisotropic [122], Oren-Nayar diffuse [82], Ashikhmin-Shirley anisotropic [3] ), factorizations (SVD [55], Homomorphic [69]), and basis approximation (spherical harmonics [18], [123], sums of separable bicubic polynomials [33], wavelets [98], and Lafortune [62], ). BRDFs can be further divided into two classes: isotropic and anisotropic. Higher dimensional functions such as BTFs [25] and BSSRDFs [39] were also introduced to describe more complicated surface materials. However, BRDFs usually suffice for uniform surfaces.

The reflection equation described above focuses on direct occlusion or illumination and depends only on the local individual properties of the surfaces. To take the environment as a whole and consider interreflection between different points, one needs to extend the reflection equation to the full rendering equation [51], which is described in Sec. 2.1. Nevertheless, for introductory purposes, the reflection equation is a good starting point and has been widely used in many real-time rendering frameworks.

### 1.2.2 Problem Introduction and Formulation

Evaluating the reflection equation is non-trivial as one has to integrate all incoming rays over the entire upper hemisphere for each vertex or pixel. One modern approach is to use signal processing analysis [86, 91, 87] and project each integrand onto bases such as spherical harmonics or wavelets, converting expensive integrations to multiplications in the frequency or wavelet domain. E.g., assuming distant lighting and diffuse material, we can define a combined transport function:

$$T(x, \omega_i) = V(x, \omega_i)\rho(x, \omega_i). \tag{1.3}$$

Projecting the transport function in equation 1.3 and the lighting term onto some orthonormal basis, we rewrite the reflection integral as

$$\begin{aligned} L_r &= \int (\sum_j L_j \phi_j(\omega_i))(\sum_k T_k(x)\phi_k(\omega_i))d\omega_i & (1.4) \\ &= \sum_j \sum_k L_j T_k(x) \int \phi_j(\omega_i)\phi_k(\omega_i)d\omega_i & (1.5) \\ &= \vec{T}(x) \cdot \vec{L} & (1.6) \end{aligned}$$

which becomes a dot product of the coefficient vectors of $\vec{T}$ and $\vec{L}$. We notice that $\vec{T}$ captures the scene's response to lighting and forms a linear system for the input lighting. The linearity of the lighting system was employed by many relighting systems but recently popularized by precomputed radiance transfer (PRT) introduced by [102]. Equation 1.4 provides the foundation for precomputed radiance transfer, which will be described in more detail in Sec. 2.5. Preliminaries for frequency and wavelet analysis will also be covered in Sec. 2.2.

Despite advances in using frequency analysis, real-time rendering faces many challenges. Firstly, many phenomena such as light scattering of participating media cannot be modeled by the reflection equation. It is less clear how to apply recent frequency or wavelet analysis results to scattering media. In such cases, compact and descriptive analytic models are powerful and effective in capturing complicated effects [111]. Secondly, frequency and wavelet based rendering approaches are still under development and far from being complete. E.g. the lighting term in the reflection equation is often assumed to be distant for near-field lighting involves affine transformation in frequency and wavelet bases [110]. Rendering is also often limited to static scenes and rigid objects. Blocking functions of blockers are often precomputed and simply accumulated, as done in the shadow field technique [126]. Underlying these problems, one fundamental challenge is to generalize operations that are common in pixel space such as multiplication, dilation, translation, exponentiation and filtering to the frequency or wavelet space. This line of research carries significant theoretical influence not only on computer graphics but also on signal processing, numerical computation and applied mathematics. We follow these directions and focus on four specific rendering problems, each one of which can be related to individual terms in the reflection equation. We start from the lighting term.

**Area Lights:** The lighting term in the reflection equation is often treated as the input in relighting problems and critical for the scene's appearance. For real-time rendering, common light models are point lights, spot lights, directional lights, collimated lights and distant environment maps [15]. Light sources such as point and directional can remove the requirement for expensive integration. Distant environment maps were first introduced in [15] and later popularized by the environment and reflection mapping techniques [71, 26]. They significantly simplify the shading calculation by removing the spatial dependence ($x$) of the lighting term and therefore reduce its dimensionality to 2D ($\omega_i$). Near-field area lights produce convincing visual effects that are otherwise hard to achieve. However, they are rarely used in real-time rendering because they involve affine transforms of the source radiance at different points in space. In [110] (Chap. 4), we propose a wavelet based technique that efficiently performs light propagation in free space using their wavelet coefficients. Our key contribution is a novel affine double and triple product wavelet integral-that performs the affine transform operator in the wavelet basis. We study the computational complexity in a number of bases, with particular attention paid to the common Haar wavelets. We show that while simple analytic formulae are not easily available, there is considerable sparsity that can be exploited computationally. The achieved benefits are twofold: firstly, large area lights can be compressed efficiently using wavelets; secondly, light integration can be converted to simple dot product in the wavelet space.

**Scattering:** The lighting can be subject to atmospherical changes such as fog, mist, or haze. Before reaching the vertex, light may get reflected, refracted, attenuated, or scattered by particles in the scattering media. In turn, a vertex's incident radiance in scattering media may get softened or blurred. These effects are not modeled by the reflection equation and usually hard to capture in real time. A number of sophisticated approaches based on Monte Carlo and finite element simulation were adapted to render impressive effects including multiple scattering and non-homogeneous media [52, 68, 49]. However, those methods take hours to render a single image and do not work at interactive rates. The most common real-time methods are essentially simple variants of the OpenGL fog model. While easy to use and specify, the standard OpenGL fog model excludes many important qualitative effects like glows around light sources, the impact of volumetric scattering on the appearance of surfaces such as the diffusing of glossy highlights, and the appearance under complex lighting such as environment maps. In [111] (Chap. 3), we present an alternative physically based approach that captures these effects while maintaining real-time performance and the ease-of-use of the OpenGL fog model. Our method is based on an explicit analytic integration of the single scattering light transport equations for an isotropic point light source in a homogeneous participating medium. This model can be easily implemented in modern graphics hardware using a few small numerical lookup tables stored as texture maps, and hence can be widely used in real-time rendering.

**Soft Shadows:** The visibility term in the reflection equation can also be viewed as a global term as it changes with the movement of the surrounding objects. In a dynamic scene with multiple blockers, $V(x, \omega_i)$ is the product of multiple blocking functions $v_1, v_2, \ldots, v_m$. Previous methods for soft shadows numerically integrate over many light directions at each receiver point, testing blocker visibility in each direction. These numerical integration approaches are too slow to suit real-time applications. Dynamically accumulating blockers using triple product integrals [77] as done by the shadow field method [126] is also too expensive. In [92] (Chap. 5.2), we take a novel frequency based approach to render soft shadows in dynamic scenes by operating vectors representing visibility of blockers in spherical harmonics. At each receiver point, we compute the product of the visibility vectors. Instead of computing an expensive SH product, we perform inexpensive vector sums to accumulate the log of blocker visibility. SH exponentiation then yields the total product visibility vector over all blockers. Our key contribution is developing exponentiation operator in the spherical harmonic basis.

**Filtering:** Rendering can be performed in either vertex space or screen space. From the screen space's perspective, filtering is critical for representing detail, such as color textures or normal maps, across a variety of scales. While MIP-mapping texture maps is commonplace, accurate normal map filtering remains a challenging problem because of nonlinearities in shading - we cannot simply average nearby surface normals. In [36] (Chap. 5.1), we take a frequency based approach and show that normal map filtering can be formalized as a spherical convolution of the normal distribution function (NDF) and the BRDF and the generated effective BRDF can be used for efficient rendering.

## 1.3 Thesis Overview, Organization and Contributions

We apply a structured approach to the four specific rendering problems introduced in Sec. 1.2, i.e., single scattering, near-field relighting, dynamic soft shadowing, and normal map filtering. The general principle underlying all our research is based on exploiting compact analytical, frequency and wavelet based models and effectively combining novel mathematical representations with the advancing hardware support. At the theoretical level, the thesis will seek to discover the optimal mathematical representation and the most compact model for various complicated effects. The thesis also reports on the engineering challenges of our techniques, as studied in real experiments. The practical impact of the outcome of the thesis would not only be on real-time rendering but also on many related fields such as importance sampling, image processing, physics simulation and signal processing. Below is the organization of the rest of the thesis and our primary technical contributions.

**Background, Chapter 2:** In Chap. 2, we describe the background for our techniques and explain mathematical preliminaries required for understanding the technical derivations in this thesis. Readers who are more familiar with these basics and technologies may safely skip Chap. 2 to the detailed presentations of specific methods in Chap. 3, 4 and 5. Section 2.1 first introduces the rendering equation, the basis for the derivation of reflection equation. Section 2.2 provides an overview of the mathematical tools and key properties of the Fourier, spherical harmonic and wavelet bases. Section 2.3 introduces environment mapping methods [15, 71, 26]. Section 2.4 describes frequency analysis results of the lighting convolution [86, 91, 87]. Section 2.5 presents the precomputed radiance transport (PRT) framework [102] and reviews its recent developments [101, 103, 104, 76, 77, 120].

**An Analytic Single Scattering Model, Chapter 3:** We consider real-time rendering of scenes in participating media, capturing the effects of light scattering in fog, mist and haze. In Chap. 3, we present a physically based approach that captures these effects while maintaining real-time performance and the ease-of-use of the OpenGL fog model. This work is also presented and published in [111]. Our method is based on an explicit analytic integration of the single scattering light transport equations for an isotropic point light source in a homogeneous participating medium. We develop our analytic airlight and surface radiance models in Sec. 3.3. In Sec. 3.5, we present the complete shading model and show how to implement the model in modern programmable graphics hardware using a few small numerical lookup tables stored as texture maps. In Sec. 3.6, we adapt our model to generate the appearances of materials with arbitrary BRDFs, environment map lighting, and precomputed radiance transfer methods, in the presence of participating media. Our primary technical contributions are:

- Explicit Compact Formula for Single Scattering for Real-Time Rendering.
- Implementation on Programmable Graphics Hardware.
- Extensions of the Model to Complex Lighting and General BRDFs.

**Affine Double and Triple Product Wavelet Integrals, Chapter 4:** Many problems in computer graphics involve integrations of products of functions. Double and triple product integrals are commonly used in applications such as all-frequency relighting or importance sampling, but are limited to distant illumination. In contrast, near-field lighting from planar area lights involves an affine transform of the source radiance at different points in space. In Chap. 4, we present a novel affine double and triple product integral theory-this generalization enables one of the product functions to be scaled and translated. This work is also presented and published in [110]. In Secs. 4.3 and 4.4, we study the computational complexity in a number of bases, with particular attention to the common Haar wavelets. We show that while simple analytic formulae are not easily available, there is considerable sparsity that can be exploited computationally. In Sec. 4.5, we demonstrate a practical application to compute near-field lighting from planar area sources, which can be easily combined with most relighting algorithms. In Secs. 4.6, we also demonstrate initial results for wavelet importance sampling with near-field area lights, and image processing directly in the wavelet domain. Our primary technical contributions are:

- Computational Complexity Analysis in General, Pixel, Fourier and Haar wavelets Bases.
- An Affine Double and Triple Product Integral Theory.
- Practical Applications and Algorithms in Near-Field Relighting, Wavelet Importance Sampling, and Image Processing.

**Frequency Based Approaches, Chapter 5** We introduce two novel frequency based approaches to normal map filtering and soft shadowing in dynamic scenes.

**Frequency Domain Normal Map Filtering, Chapter 5.1:** Correct normal filtering is difficult due to its non-linearity. In Sec. 5.1.2, we show analytically that normal map filtering can be formalized as a spherical convolution of the normal distribution function (NDF) and the BRDF, for a large class of common BRDFs such as Lambertian, microfacet and factored measurements. This work is also presented and published in [36]. This is a joint work with my colleagues at Columbia University, Charles Han, Ravi Ramamoorthi, Eitan Grinspun. Our practical algorithms leverage a significant body of work that has studied lighting-BRDF convolution.

We also show how spherical harmonics can be used to filter the NDF for Lambertian and low-frequency specular BRDFs. Our specific technical contributions are

- Theory of Normal Map Filtering as Convolution.
- Novel Frequency Domain Representations.
- Extensions to Realistic BRDFs and Complex Lighting

**Soft Shadowing using Spherical Harmonic Exponential, Chapter 5.2:** Previous methods for soft shadows numerically integrate over many light directions at each receiver point, testing blocker visibility in each direction. In Chap. 5.2, we introduce a method that accumulate the product of visibility vectors in the log space using spherical harmonic exponentiation. This work is also presented and published in [92]. This research was conducted during my internship at Microsoft Research and is a joint work with researchers at Microsoft Research and Zhejiang University. In Secs. 5.2.2 and 5.2.3, we show mathematical derivations of SH exponentiation and logarithm and ways to optimize their computations. Section 5.2.4 gives an overview of our approximation using sphere sets and real-time implementation on GPU. Our primary technical contributions are

- First Real-Time Method for Soft Shadowing in Dynamic Scenes.
- Spherical Harmonic Exponentiation Technique.
- Sphere Set Approximation and Implementation on GPU.

**Conclusion, Chapter 6:** Chapter 6 concludes the thesis, offering a review of the common techniques exploited and discussing the primary benefits and limitations of our results. Finally, we suggest promising areas for future exploration.

# Chapter 2

# Background

## 2.1 Rendering Equation

The rendering equation was first simultaneously introduced by James T. Kajiya [51] and David Immel et al. [44] from radiative heat transfer and neutron transport literature. The rendering equation describes how light is reflected and scattered in a scene and can be used to characterize many known rendering algorithms. It is an integral equation in which the equilibrium radiance leaving a point is given as the sum of the emitted plus reflected radiance under a geometric optics approximation. Though it neglects for example wave effects, it has been proven useful to formalize the rendering process in computer graphics. The rendering equation can be written in the form

$$L_r(x, x') = L_e(x, x') + \int_S L_r(x'', x) V(x'', x) \rho(x'', x, x') dx'', \tag{2.1}$$

where $L_r$ is the reflected exitant radiance from $x$ to $x'$, $L_e$ is the outgoing emitted radiance from point $x$ to $x'$, $L_r$ is the incident radiance on point $x$ from $x''$, and $\rho$ is the BRDF at point from directions $x$ from $x''$ to $x$ and from $x$ to $x'$. The integration is performed for all surfaces in the scene. The difference between the reflection and the rendering equation is significant, i.e., $L_r$ inside the integral now includes incident radiance from not only light sources but also reflected radiance from nearby objects. Therefore, the rendering equation describes more global effects such as interreflection than the reflection equation. However, effects such as volumetric scattering and flurescence are not modeled. Two important features of the rendering equations are its linearity and spatial homogeneity, which means a wide range of factorings and rearrangements of the equation are possible.

Solving the rendering equation has been a major challenge for computer graphics and accurately evaluating it is expensive. Popular approaches include radiosity[34], ray tracing [124], path tracing [51], photon mapping [47], metropolis light transport [117], Monte Carlo simulations and etc. All these techniques are offline based rendering algorithms.

### 2.1.1 Reflection Equation

Omitting self-emitted radiance $L_e$ and assuming surfaces only reflect light sources (not reflected radiances from other surfaces), the rendering equation can be significantly simplified to the reflection equation, as we introduced in Sec. 1.1. Please note that in the reflection equation, it is customary to convert the integration over all surfaces to the integration over the more familiar angular domain and change variable $x''$ to $\omega_i$. We rewrite the reflection equation here:

$$L_r(x, \omega_o) = \int_{\Omega} L_i(x, \omega_i) V(x, \omega_i) \rho(x, \omega_i, \omega_o)(\omega_i \cdot n(x)) d\omega_i, \tag{2.2}$$

where x is the location of a vertex, or a pixel in the image, $\omega_o$ is the viewing direction, $\omega_i$ is the incident direction, $L_i$ is the incident radiance from the light source at a vertex or a pixel, $V$ is the visibility function, $\rho$ (different from its definition in Sec. 2.1)is the BRDF at location x for incoming angle $\omega_i$ and outgoing angle $\omega_o$, and

$(\omega_i \cdot n(x))$ is the cosine of the incident angle. Compared to the rendering equation, the reflection equation captures the first-order primary effects directly from the lighting but is much more tractable for real-time rendering.

## 2.2 Orthonormal Bases

We briefly review a few orthonormal bases such as the Fourier basis, spherical harmonics and Haar wavelets as they will be extensively used in later analysis in the thesis.

**Fourier Series:** The Fourier series is the most popular tool for frequency analysis. It is usually preferred for theoretical analysis, for example frequency analysis of light transport [31]. Fourier analysis, named after Joseph Fourier, is the decomposition of a function in terms of basis (sinusoidal) functions of different frequencies that can be recombined to obtain the original function. Originally introduced to solve the heat equation, it led to revolution in mathematics and reexamination of foundations for many modern theories. The complex form ($I = \sqrt{-1}$) of a 1D Fourier series on an azimuthal domain $[0, 2\pi]$ can be written as:

$$\phi_p(x) = \sqrt{\frac{1}{2\pi}} e^{Ipx}. \tag{2.3}$$

Fourier series also has a real form consisting of real sine and cosine functions. Periodic functions that are otherwise difficult to analyze can be decomposed using the Fourier basis to get their frequency coefficients. 2D and higher dimensional Fourier bases are products of multiple 1D Fourier basis functions. We use the Fourier analysis in Chaps. 4 and 5.1.

**Spherical Harmonics:** Spherical harmonics are the angular domain analog to Fourier series and are important in many theoretical and practical applications such as computation of atomic electron configurations, the representation of gravitational field, geoid, and magnetic field of planetary bodies. In computer graphics, spherical harmonics play an important role in a number of topics (ambient occlusion, global illumination, shape recognition, precomputed radiance transfer etc.) and are useful for representing low-frequency spherical functions such as radiance incident at a point and blocker visibility functions which modulate distant radiance. In mathematics, the spherical harmonics are the angular portion of an orthogonal set of solutions to Laplace's equation represented in a spherical coordinates. Spherical harmonics in complex form can be written as

$$Y_{lm}(\theta, \phi) = \alpha_{l|m|} P^l_m(\cos\theta) e^{Im\phi} \tag{2.4}$$

where $l \geq 0, -l \leq m \leq l$, $\alpha_{l|m|}$ is a normalization factor, and $P^l_m$ are associated Legendre polynomials. The azimuthal dependence is expanded in terms of Fourier basis functions. The $\theta$ dependence is expanded in terms of the associated Legendre functions $P^l_m$. The SH basis functions are orthogonal polynomials in $s = (x, y, z)$ restricted to the sphere $s \in S$. The indices obey $l \geq 0$ and $-l \leq m \leq l$. Thus, there are $2l + 1$ basis functions for a given order l. An order $n$ SH projection has $n^2$ vector coefficients. An important relation is that $Y_{l-m} = (-1)^m Y^*_{lm}$. Spherical harmonics of the first 3 order can be written as

$$Y_{00} = \sqrt{\frac{1}{4\pi}} \tag{2.5}$$

$$Y_{10} = \sqrt{\frac{3}{4\pi}} \cos\theta \tag{2.6}$$

$$Y_{11} = -\sqrt{\frac{3}{8\pi}} \sin\theta e^{I\phi} \tag{2.7}$$

$$Y_{20} = \sqrt{\frac{5}{16\pi}} (3\cos^2\theta - 1) \tag{2.8}$$

$$Y_{21} = -\sqrt{\frac{15}{8\pi}} \sin\theta\cos\theta e^{I\phi} \tag{2.9}$$

$$Y_{22} = \sqrt{\frac{15}{32\pi}} \sin^2\theta e^{2I\phi} \tag{2.10}$$

More importantly, spherical harmonics can be analytically rotated on the sphere using a rotation matrix $D^l_{mm'}$:

$$Y_{lm}(R_{\alpha,\beta,\gamma}(\theta,\phi)) = \sum_{m'=-l}^{l} D^l_{mm'}(\alpha,\beta,\gamma)Y_{lm'}(\theta,\phi). \tag{2.11}$$

Given a spherical function $f(\omega)$, we can *project* this function onto spherical harmonics to get a vector $\vec{f}$ representing its low-frequency behavior via

$$\vec{f} = \int_{\Omega} f(\omega)\, \vec{y}(\omega) d\omega \tag{2.12}$$

where $\vec{y}(\omega)$ is the vector of SH basis functions $Y_{lm}$. Conversely, Given an SH vector $\vec{f}$ we can *reconstruct* a continuous spherical function $f(\omega)$ via

$$f(\omega) = \sum_{i=0}^{n^2-1} \vec{f_i}\, y_i(\omega) = \vec{f} \cdot \vec{y}(\omega). \tag{2.13}$$

Spherical harmonics is used for efficient representation of irradiance maps [86], and frequency analysis of inverse and forward rendering [90, 87]. The original PRT

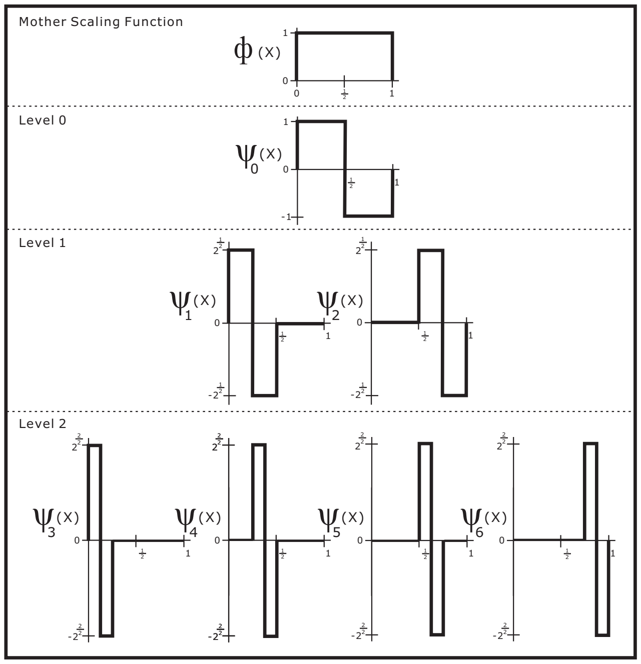


Figure 2.1: 1D Haar wavelet basis functions.

framework was also proposed in terms of spherical harmonics in [102]. We use spherical harmonics for frequency analysis in Chaps. 3, 5.1 and 5.2.

**Haar Wavelets:** The wavelets are scaled and translated copies of a finite length or fast-decaying oscillating waveform known as the ”mother wavelet”. Wavelet transforms have advantages over traditional Fourier transforms for representing functions that have discontinuities and sharp peaks, and for accurately deconstructing and reconstructing finite, non-periodic and stationary signals. The first known wavelets are Haar wavelets, proposed in 1909 by Alfred Haar. The Haar wavelet is also the simplest possible wavelet. The normalized 1D Haar basis can be written as

- The mother scaling function is

$$\phi(x) = \begin{cases} 1, & \text{for } 0 \le x < 1 \\ 0, & \text{otherwise} \end{cases} \tag{2.14}$$

- The mother wavelet function is

$$\psi(x) = \begin{cases} 1, & \text{for } 0 \le x < 1/2 \\ -1, & \text{for } 1/2 \le x < 1 \\ 0, & \text{otherwise} \end{cases} \tag{2.15}$$

- A normalized wavelet basis $\psi_j(x)$ at level $l_j$ and offset $t_j$ is

$$\psi_j(x) \quad = \quad 2^{l_j/2}\psi(2^{l_j}x - t_j), \tag{2.16}$$

  which is a scale and dilation of the mother wavelet $\psi(x)$.

Using either the standard or nonstandard decomposition, the 1D Haar wavelet transform can be easily extended to 2D or higher dimensions, as we show in Sec. 4.4.4. Below we recap a few key properties of Haar wavelets without proof. More interested readers can refer to [21].

- The wavelets at the same level are orthonormal: $\int \psi_i(x)\psi_j(x)dx = \delta(i-j)$.

- The wavelets have vanishing integrals: $\int \psi_i(x)dx = 0$.

- The product of the mother scaling function with any wavelet is simply still the same wavelet: $\phi(x)\psi_i(x) = \psi_i(x)$

- The square of a wavelet is equal to the scaling function restricted to $(l, t)$ and scaled by $4^l$.

- The product of two overlapping wavelets at different levels $l < l'$ is the finer wavelet scaled by $\pm 2^l$.

Wavelets have long been used in computer graphics to compress signals (images and videos) and approximate higher dimensional functions such as BRDFs. It has recently become popular in PRT based relighting algorithms for lighting and visibility representations. We use wavelets in our analysis in Chap. 4. It is also used in wavelet importance sampling for efficient offline renderings [22].

## 2.3 Environment Mapping

Environment mapping technique was first introduced by Blinn and Newell [15] to approximate mirror reflections and was later popularized by [71, 26]. Reflection mapping based methods first approximate the reflection equation by dropping the visibility term. Thus, it works best for simple BRDFs and objects and achieves real-time rates. For mirror reflection surfaces, the incident radiance gets perfectly reflected in a reflected direction. The reflected radiance can be written as

$$L_r(x, \omega_o) = L_i(x, \omega_r(n)), \tag{2.17}$$

where $\omega_r$ is the reflected direction of the outgoing angle $\omega_o$ with respect to the surface normal $n$. No integration is involved in the calculation. Reflection of vectors with respect to surface normal can be calculated using simple linear algebra. The reflected radiance can be obtained by indexing into environment map using the reflection vector $\omega_r$, greatly speeding up the performance. This is the reflection technique introduced in [15, 71] was popularized later in movies such as Interface, Flight of the Navigator, Abyss, Terminator 2 and etc.

Environment maps can be represented using a number of different parameterizations. For example, the map can be created by choosing a center of projection and project the environment onto it. Popular forms are spherical maps, latitude-longitude maps, cube maps, dual parabolic maps and etc. To achieve the highest level of fidelity, environment maps can also encode high dynamic range radiance values [27].

The original environment reflection mapping technique was also quickly extended to relighting with Lambertian or simple Phong BRDFs. In these cases, since the BRDF is given, the shading integration of the lighting with the BRDF can be performed beforehand to generate a prefiltered environment map. At run time, GPUs can be used for fast texture looks ups. In particular, Miller et al. [71] and Heidrich et al. [40] performed the following integrations before hand:

$$\begin{aligned} L_{diffuse}(x) &= k_d \int_\Omega L_i(x,\omega_i)(\omega_i \cdot n(x)) d\omega_i, & (2.18) \\ L_{phong}(x,\omega_o) &= \int_\Omega L_i(x,\omega_i)(\omega_r \cdot \omega_i)^n d\omega_i, & (2.19) \end{aligned}$$

where $k_d$ is the diffuse surface albedo that describes absorption, $\omega_r$ is the reflected outgoing direction, and $n$ is the Phong exponent. These calculations generate prefiltered environment maps indexed by the surface normal direction $n$ and the reflected outgoing direction $\omega_r$. Multiple environment maps may be needed for glossy materials such as the Phong BRDF with view point variations.

Environment mapping techniques uniformly assume the the surrounding environment is infinitely far away so that the spatial parallax can be safely ignored to speed up the computation. Often only one environment map or a few suffice. Because environment maps are preintegrated and tabulated, they cannot easily handle shadow and interreflection effects. As a result, they cannot be easily applied to planner and concave surfaces. Despite these limitations, environment mapping techniques are widely used for real-time applications such as games for their simplicity in hardware implementation.

## 2.4 Frequency Analysis of Irradiance Integration

Ramamoorthi and Hanrahan [86, 91, 87] pioneered important research on signal processing analysis of irradiance maps and shading convolution. Basri and Jacobs

[7] independently published similar frequency analysis results on face recognition. Their work has laid the foundation for much recent development in pre-computed radiance transfer (PRT) [102]. Ramamoorthi and Hanrahan [86] applied spherical harmonic analysis to irradiance environment maps. Spherical Harmonics, as introduced in Sec. 2.2, are the analogue to the Fourier basis on the sphere and approximate spherical functions well. Ramamoorthi and Hanrahan show that, when the filter kernel is broad, the prefiltered Lambertian BRDF can be well approximated using the nine lowest-order spherical harmonic coefficients. As convolution reduces to multiplication in the frequency domain, the lighting integration can be calculated very efficiently, dramatically increasing the performance and reducing the storage requirements. Specifically, they show that the irradiance convolution reduces to a multiplication:

$$E(\theta, \phi) = \sum_{lm} \hat{A}_l L_{lm} Y_{lm}(\theta, \phi), \tag{2.20}$$

where $E(\theta, \phi)$ is the irradiance, $L_{lm}$ is the spherical harmonic coefficients of the environment lighting, $Y_{lm}(\theta, \phi)$ is the spherical harmonic basis function, $\hat{A}_l$ is a simple analytic function whose explicit formula can be derived. They further show that that $\hat{A}_l$ vanishes for odd values of $l > 1$, and even terms fall off very rapidly as $l^{\frac{5}{2}}$. The analytic formulae are given by

$$l = 1 \qquad \hat{A}_1 = \frac{2\pi}{3} \tag{2.21}$$

$$l > 1, odd \qquad \hat{A}_1 = 0 \tag{2.22}$$

$$l = even \qquad \hat{A}_1 = 2\pi \frac{(-1)^{\frac{l}{2}-1}}{(l+2)(l-1)} \left[ \frac{l!}{2^l (\frac{l}{2}!)^2} \right] \tag{2.23}$$

Numerically, the first few terms are

$$\hat{A}_0 = 3.141593 \quad \hat{A}_1 = 2.094395 \quad \hat{A}_2 = 0.785398 \tag{2.24}$$

$$\hat{A}_3 = 0 \quad \hat{A}_4 = .0.130900 \quad \hat{A}_5 = 0 \quad \hat{A}_6 = 0.049087 \tag{2.25}$$

The key is that $\hat{A}_l$ decays very fast with $l$. Thus, the irradiance is well approximated by 9 spherical harmonic coefficients, instead evaluating a full hemispherical integral. For rendering, this simple algorithm can implemented using 4x4 matrices. Though efficient, this approach is limited to Lambertian diffuse reflectance and distant environment lighting. In [88], Ramamoorthi and Hanrahan extended the spherical harmonic approach to specular BRDFs under distant illumination using spherical harmonic reflection maps (SHRM). SHRM is a 4D spherical harmonic representation of the environment reflection at changing views. Sloan et al. [102] later applied spherical harmonic results to exploit the linearity of light transport and proposed precomputed radiance transfer (PRT).

One common limitation of spherical harmonic based approaches is that spherical harmonic basis is not well suited to approximate high frequency signals and display ringing artifact at higher orders. This prevents spherical harmonics from being applied on high frequency lighting environment or very glossy BRDFs. Ng et al. in [75] discussed the inability of a low-order SH basis to approximate high frequency environment maps. They propose non-linear wavelet approximation for representing the environment maps and light transport matrix to achieve all-frequency effects. A wavelet basis contains area lights that vary from the size of a cubemap pixel to essentially the size of the entire sky. They show that usually $0.5 \sim 1\%$ wavelet basis coefficients suffice to accurately approximate detailed photographed illumination, and that $0.1 \sim 0.2\%$ suffice for simpler environments or synthetic area lights. They precompute lighting and transport matrices in $6x64x64$

cubmaps and for rendering perform a giant sparse matrix multiplication. They later extended the use of wavelets to triple product wavelet integrals [76]. Other bases such as symmetric radial basis functions (SRBFs) [116, 36] are also experimented in the PRT relighting framework.

## 2.5 Precomputed Radiance Transfer (PRT)

Much of our work is built upon the precomputed radiance transfer (PRT) framework. Relighting techniques have been developed from the basic approach introduced by Nimeroff et al. [79] and Dorsey et al. [29] to much recent work on precomputed radiance transfer (PRT) [102], using the signal processing analysis results and leveraging the linearity of light transport. The original PRT algorithm proposed by Sloan et al. [102] is for static scenes in low frequency lighting environment. PRT solves the light integration problem using an tabulation of an object's response to the lighting in the frequency space, capturing the way an object shadows, scatters, and reflects light. Equation 1.4 forms the basis for precomputed radiance transfer. We rewrite it here as :

$$\begin{aligned} L_r &= \int (\sum_j L_j \phi_j(\omega_i))(\sum_k T_k(x)\phi_k(\omega_i))d\omega_i && (2.26) \\ &= \sum_j \sum_k L_j T_k(x) \int \phi_j(\omega_i)\phi_k(\omega_i)d\omega_i && (2.27) \\ &= \vec{T}(x) \cdot \vec{L}, && (2.28) \end{aligned}$$

where $L_r$ is the reflected radiance, $L$ is the incident lighting and $T$ is the transport function. PRT pre-computes light transport functions for each vertex and reduce complex integrations to a simple inner product of the light vector with the transport vectors in the frequency space. This work has rekindled a great interest not only in pre-computation based relighting but also in selection of better bases for lighting approximation.

Other methods [56, 101, 120, 121, 65] quickly improved the PRT technique to allow for better real-time performance and rendering of non-diffuse or translucent objects. As per-point transfer matrices form a high-dimensional surface signal, Sloan et al. [101] apply clustered principal component analysis (CPCA) to compress them to a low-dimensional set of per point weights on a per-cluster set of representative matrices. Ng et al. [75] used non-linear wavelet approximation for representing the environment map and light transport matrix to achieve all-frequency effects. Wang et al. [120] and Liu et al. [65] extended PRT to glossy objects using separable BRDF approximations. While [111] added atmospherical scattering effects to PRT, Wang et al. [118] further incorporated rendering of translucent objects in PRT. Tsai et al. [116] introduced for PRT a new data representation using spherical radial basis functions (SRBFs) and compressed the precomputed data using cluster tensor approximation. Ng et al. [76] further extend the general PRT framework to triple product wavelet integrals for handling higher dimensionality and sampling rate. Triple product wavelet integral is then further generalized by [112] to render dynamic glossy objects.

There has also been active research on extending PRT techniques to rendering more dynamic effects, such as moving and deformable objects, mid-range illumination, wrinkled surfaces and near-field lighting. Sloan et al. [104] capture local effects such as bumps, wrinkles, or other detailed features by applying zonal harmonics (ZH). Zhou et al. [126] present a shadow field technique for dynamic scenes, which precomputes for each scene entity a shadow field that describes the shadowing effects, and for each light sources a source radiance field that records radiance from an illumination. James et al. [46] take a data-driven approach to global illuminations of dynamic deformable scenes. Because they parameterize all possible scene deformations, they are able to precompute novel reduced coparameterizations of global scene illumination for low frequency lighting conditions. Annen et al. [2] use a first-order Taylor expansion of the spherical harmonic lighting coefficients and an interpolation scheme based on these gradients to render mid-range illumination. Spherical harmonic scaling [119] tries to approximate the effects of mid-range illumination by simple spherical function scaling. Spherical harmonic exponentiation [92] can render near-field soft shadowing effects in real time, but has to use sphere sets to approximate geometries. Kristensen et al. [60] extend work on PRT for distant lighting to local lighting by introducing unstructured light clouds. Sun and Ramamoorthi [110] proposed a novel affine double and triple product wavelet integral theory for near-field relighting.

This body of work has laid the foundation for much of our mathematical derivations for real-time rendering. For example, we have used the frequency based convolution framework to extend our analytic single scattering model, via the form of a point spread function (PSF), to arbitrary BRDFs, complex lighting and precomputed radiance transfer [111]. In Sec. 5.2, the spherical harmonic exponentiation technique presented for soft shadows is based on PRT, but extends it to multiple dynamic blockers. In chapter 4, our theory of affine double and triple wavelet product integrals for near-field relighting is an important generalization of the triple product wavelet integrals of Ng. et al [76]. From the next chapter, we start with our analytic single scattering model for real-time rendering.

# Chapter 3

# A Practical Analytic Model for Single Scattering

## 3.1 Introduction

Many real-time rendering applications like games or interactive simulations seek to incorporate atmospheric effects such as mist, fog and haze. These participating media lead to a number of qualitative effects not present in clear-day conditions (compare Fig. 3.1A with our result in Fig. 3.1C). In computer graphics, the approaches for capturing these effects represent two ends in the spectrum of speed and quality. For high-quality rendering, a number of Monte Carlo and finite element techniques have been proposed. These methods can model general volumetric phenomena and scattering effects. However, they are slow, usually taking hours to render a single image. At the other extreme, perhaps the most common approach for interactive rendering is to use the OpenGL fog model, which simply blends the fog color with the object color, based on the distance of the viewer (Fig. 3.1B). This model is also popular because of its simplicity. However, many qualitative effects are missing, such as the glows around light sources, the effect of

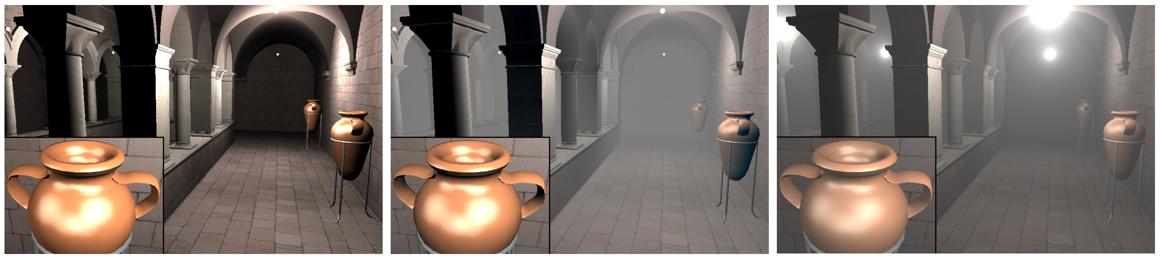


Figure 3.1: *Rendered images of a scene with 66,454 texture-mapped triangles and 4 point lights. (a) Standard OpenGL rendering (without fog), (b) OpenGL fog, and (c)Our real-time model, that includes the glows around light sources, and changes to surface shading such as dimming of diffuse radiance (floor and wall), brightening of dark regions (back side of pillars and vases) and dimming and diffusing of specular highlights (inset). All the visual effects are rendered by our method at about 20 frames per second.*

scattering on object shading, and the ability to incorporate complex lighting effects like environment maps.

Figure 3.2 illustrates three important visual effects due to light transport in scattering media. In this discussion, we assume single scattering (i.e. that light scatters at most once in the medium). Figure 3.2A corresponds to *direct transmission* of light from the source or surfaces to the viewer. This simple approach is essentially what interactive models like OpenGL fog implement. Figure 3.2B also includes the glows around light sources, commonly referred to as *airlight* [59]. Figure 3.2C further includes the effect of airlight on the outgoing *surface radiance*, leading to effects such as the spreading out of specular highlights and softening of shadows. Figure 3.2D illustrates the case where the surface radiance is single scattered in addition to being attenuated, before reaching the view point. Implementing the latter effect requires a depth-dependent convolution.

In this section, we present an alternative physically based approach that captures these effects while maintaining the real-time performance and ease of use of the OpenGL fog model. This work is published in [111]. Our method is based on an explicit analytic integration of the single scattering light transport equations for an isotropic point light source in homogeneous participating medium. Our model can be implemented as a simple vertex or pixel shader (pseudocode in Fig. 3.13). The method can also be applied with complex lighting allowing environment mapping and precomputed radiance transfer to be used interactively with participating media. Our specific technical contributions are:

**Explicit Compact Formula for Single Scattering:** One of the main contributions of our method is the derivation of an explicit compact formula for the single scattering from an isotropic point source in a homogeneous participating medium, by analytically integrating the single scattering equations. This *airlight model* (Sec. 3.3) allows us to simulate effects like the glows around light sources (Fig. 3.2B). We can also use the model to calculate the effects of scattering on the surface shading (Fig. 3.2C).

**Implementation on Programmable Graphics Hardware:** We reduce these difficult integrals to a combination of analytic functions that depend only on the physical parameters of the problem, and a few lookups of tabulated 2D functions, that are smooth and purely numerical. The numerical functions can be precomputed and stored as 2D texture maps, and the entire analytic computation and table lookups can be implemented in simple pixel or vertex shaders (Sec. 3.5).

**Extensions to Complex Lighting and BRDFs:** Mathematically, we derive a point-spread function (PSF) to represent the glow around a light source. We can convolve an environment map with this PSF to get the appearance of a foggy scene under natural lighting. This approach enables methods such as environment mapping and precomputed radiance transfer to be used with volumetric scattering effects.

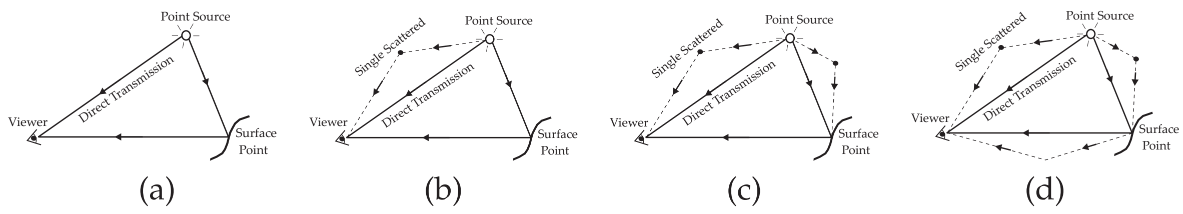


Figure 3.2: *Diagrams showing three cases of how light travels to the viewer through the participating medium. In (a) light travels in a straight line and directly reaches the surface and the viewer. This is essentially what previous interactive models such as OpenGL fog compute. In (b), in addition to what happens in (a), airlight scatters to the viewer and produces effects like glows around the light source. In (c), in addition to what happens in (b), airlight also scatters to the surface and gets reflected, leading to effects such as the diffusing out of specular highlights and brightening of darker regions. In image (d), reflected rays from the surface also scatter to the viewer.*

## 3.2 Previous Related Work

The literature on simulating volumetric effects is large, going back to [14], and we only discuss important representative papers. Most techniques are based on numerical or analytic approximations to the radiative transfer equation [20]. Monte Carlo ray tracing methods were adapted by computer graphics researchers to render impressive effects including multiple scattering and non-homogeneous media [53, 67, 48]. However, such methods can take hours to render a single image. To speed up rendering, numerical methods that only simulate single scattering have also been proposed [83, 72, 95, 94]. However, they still require significant running times, and are not suitable for interactive applications.

**Hardware-accelerated numerical methods:** A number of recent hardware accelerated techniques can significantly decrease the running times of numerical simulations, although they are still usually not fast enough for many interactive applications such as games. Dobashi et al. [28] describe a multi-pass rendering technique that numerically integrates the single scattering equations, using graphics hardware to accumulate the results at a number of planes in the scene, similar to volume rendering. Harris and Lastra [38] render clouds by including a forward scattering term in addition to single scattering. Note that their method is geared toward the case when the viewer is far from the clouds, and they apply a different and slower approach when the viewer and scene are immersed inside the medium, as is the scenario in our work.

These methods are intended to apply to specific phenomena like the sky or clouds [28, 93, 38]. This allows them to make use of complex tabular volume specifications, precomputed lighting solutions or multipass rendering techniques to produce effects including inhomogeneous media and simple heuristics for multiple scattering. They allow for viewpoint, and in a few cases interactive lighting variation, but usually fix the medium properties and scene specification.

In contrast, our technique, while focusing on homogeneous media and single scattering, can be encapsulated in a simple shader for general scenes, and allows for real time variation of the viewpoint, lighting, scattering properties of the medium, and even scene geometry and reflectance. Another major benefit of our method is that it addresses the effects of scattering on surface shading (Fig. 3.2C) and complex lighting like environment maps. These effects are not included in previous methods because they are difficult to numerically simulate efficiently, requiring an integration over all incident scattered lighting directions at each surface point.

**Analytically based methods:** The diffusion approximation for optically thick media was applied to subsurface scattering [106, 49]. An analytic form for the single scattering term was also derived by Hanrahan and Krueger [37]. However, the problem we are solving is very different from that of subsurface scattering, where the light sources and viewer are outside the medium. In our case, both the sources and viewer are immersed *inside* the medium. Also, unlike in the case of diffusion, we are interested in strongly directional effects like glows around sources.

| | |
|---|---|
| $s, v, p$ | Subscripts for Source, Viewer, surface Point |
| $\gamma$ | Angle between light source and viewing ray |
| $D_{sv}$ | Distance between source and viewer |
| $D_{vp}$ | Distance between viewer and closest surface point |
| $D_{sp}$ | Distance between source and surface point |
| $T_{sv}$ | Optical thickness between source, viewer ($\beta D_{sv}$) |
| $T_{vp}$ | Optical thickness between viewer, surface point ($\beta D_{vp}$) |
| $T_{sp}$ | Optical thickness between source, surface point ($\beta D_{sp}$) |
| $\beta$ | Scattering coefficient of the participating medium |
| $\alpha$ | Angle of scattering |
| $x$ | Distance along the ray from viewer (integration variable) |
| $d$ | Distance of single scattering from light source |
| $I_0$ | Radiant intensity of point light source |
| $f_r$ | BRDF of surface |

Figure 3.3: *Notations used in Chap. 3.*

Analytic expressions for airlight with directional light sources, based on the derivation by Koschmeider [59], are used frequently for rendering skies [84, 42, 73]. However, our focus is different. We wish to derive an analytic model with "near-field" point sources, which is a significantly more complex lighting situation as compared to distant lighting (collimated beams).

Analytic expressions for the glows around point light sources *inside* homogeneous media have also been derived [67, 11, 74]. Therefore, those methods could be used to render glows in real time. However, it is not clear how to extend them to a complete real-time rendering system that also considers the effects of airlight on surface shading, or handles complex environment map lighting. Furthermore, their derivations involve approximations that are not feasible in several common rendering scenarios. For instance, the model derived by Max [67] does not take into account attenuation. Biri et al. [11] use a polynomial approximation to single scattering which results in inaccurate glows along viewing directions near the source. The multiple scattering model in [74] is not strictly valid when objects are present in the medium, especially near the sources (as is generally true in most common scenes), or for optically thin media. Further, the integration required for surface

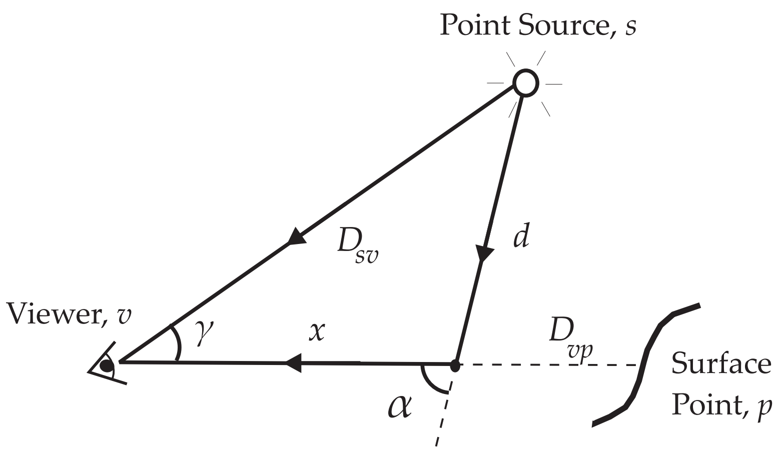


Figure 3.4: *Diagram showing how light is scattered once and travels from a point light source to the viewer.*

radiance cannot be computed analytically or simulated numerically in real time.

## 3.3 Airlight Model

In this section, we derive an explicit model for the single scattered radiance at a viewer, due to an isotropic point light source, assuming that both the viewer and the source are immersed in a homogeneous scattering medium. Consider the scenario illustrated in Fig. 3.4 (the notations used are indicated in Fig. 4.2). The point light source has a radiant intensity $I_0$ and is at a distance $D_{sv}$ from the view point, making an angle $\gamma$ with the viewing direction. The radiance, $L$, is composed of the direct transmission, $L_d$, and the single scattered radiance or airlight, $L_a$,

$$L = L_d + L_a. \tag{3.1}$$

The direct term $L_d$ simply attenuates the incident radiance from a point source $(I_0/D_{sv}^2)$ by an exponential corresponding to the distance between source and viewer, and the scattering coefficient[1] $\beta$,

$$L_d(\gamma, D_{sv}, \beta) = \frac{I_0}{D_{sv}^2} e^{-\beta D_{sv}} \cdot \delta(\gamma), \tag{3.2}$$

where the delta function indicates that for direct transmission, we receive radiance only from the direction of the source (no glows).

### 3.3.1 The Airlight Integral

We focus most of our attention on the airlight $L_a$. The standard expression [80] is given by an integral along the viewing direction,

$$L_a(\gamma, D_{sv}, D_{vp}, \beta) = \int_0^{D_{vp}} \beta k(\alpha) \cdot \frac{I_0 \cdot e^{-\beta d}}{d^2} \cdot e^{-\beta x} dx\,, \tag{3.3}$$

where $D_{vp}$ is the distance to the closest surface point along the viewing ray or infinity if there are no objects in that direction, and $k(\alpha)$ is the particle phase function. The exponential attenuation corresponds to the total path length traveled, $d + x$. The two parameters $d$ and angle $\alpha$ in the integrand depend on $x$. In particular, $d$ is given by the cosine rule as

$$d = \sqrt{D_{sv}^2 + x^2 - 2xD_{sv}\cos\gamma}. \tag{3.4}$$

Let us now substitute equation 3.4 into equation 3.3. For now, we also assume the phase function $k(\alpha)$ is isotropic and normalized to $1/4\pi$. In this case,

$$L_a(\gamma, D_{sv}, D_{vp}, \beta) = \frac{\beta I_0}{4\pi} \int_0^{D_{vp}} \frac{e^{-\beta\sqrt{D_{sv}^2 + x^2 - 2xD_{sv}\cos\gamma}}}{D_{sv}^2 + x^2 - 2xD_{sv}\cos\gamma} \cdot e^{-\beta x} dx\,. \tag{3.5}$$

We refer to this equation as the *airlight single scattering integral* and next focus on

[1]When there is light absorption in addition to scattering, $\beta$ is called the extinction coefficient and is given by the sum of the scattering and absorption coefficients. In this paper, we simply refer to $\beta$ as the scattering coefficient, and it is straightforward to include absorption in our models.

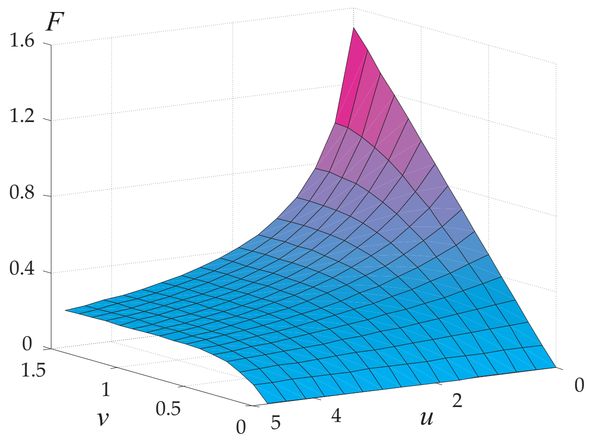


Figure 3.5: *3D plot of special function $F(u, v)$ in the range of $0 \le u \le 10$ and $0 \le v \le \frac{\pi}{2}$. The plot shows that the function is well-behaved and smooth and can therefore be precomputed as a 2D table. As expected from the definition in equation 3.15, the function decreases as u increases, and increases as v increases. The maximum value in the plot above therefore occurs at $(u = 0, v = \frac{\pi}{2})$. Also note from equation 3.15, that for $u = 0$, there is no attenuation so the function is linear in v.*

simplifying it further to derive an explicit form.

### 3.3.2 Solution to the Airlight Integral

We take a hybrid approach to solve equation 3.5. The key result is that this integral can be factorized into two expressions—(a) an **analytic expression** that depends on the physical parameters of the scene and (b) a two-dimensional numerically tabulated function that is **independent** of the physical parameters. Essentially, this factorization enables us to evaluate the integral in equation 3.5 analytically. A high-level sketch of the derivation is given below and detailed simplifications are included in appendix A.

**STEP 1. Reducing the dimensions of the integral:** Since the integral in equation 3.5 depends on 4 parameters, our first step is to apply a series of substitutions that reduce the dependency of the integrand to only one parameter. For this, we first write the expressions in terms of optical thicknesses $T_* = \beta D_*$ and $t = \beta x$. In most cases, this eliminates the separate dependence on both $\beta$ and the distance parameters, somewhat reducing the complexity, and giving us a simpler intuition regarding the expression's behavior. Then, we combine the dependence on $T_{sv}$ and $\gamma$ by making the substitution $z = t - T_{sv}\cos\gamma$, to obtain

$$L_a(\gamma, T_{sv}, T_{vp}, \beta) = \frac{\beta^2 I_0}{4\pi} e^{-T_{sv}\cos\gamma} \int_{-T_{sv}\cos\gamma}^{T_{vp}-T_{sv}\cos\gamma} \frac{e^{-z-\sqrt{z^2+T_{sv}^2\sin^2\gamma}}}{T_{sv}^2\sin^2\gamma + z^2}\, dz. \tag{3.6}$$

Now, the integrand really depends on only one physical parameter $T_{sv}\sin\gamma$, beginning to make the computation tractable.

It is possible to further simplify equation 3.6. To encapsulate the dependence on the physical parameters of the problem, we define the following two auxiliary expressions, corresponding respectively to the normalization term outside the integrand, and the single physical parameter in the integrand, $T_{sv}\sin\gamma$:

$$A_0(T_{sv}, \gamma, \beta) = \frac{\beta^2 I_0 e^{-T_{sv}\cos\gamma}}{2\pi T_{sv}\sin\gamma} \tag{3.7}$$

$$A_1(T_{sv}, \gamma) = T_{sv}\sin\gamma. \tag{3.8}$$

It is then possible to derive, that

$$L_a = A_0(T_{sv}, \gamma, \beta) \int_{\gamma/2}^{\frac{\pi}{4}+\frac{1}{2}\arctan\frac{T_{vp}-T_{sv}\cos\gamma}{T_{sv}\sin\gamma}} \exp[-A_1(T_{sv}, \gamma)\tan\xi]\, d\xi. \tag{3.9}$$

Now we briefly recap the mathematical derivation per the description above.

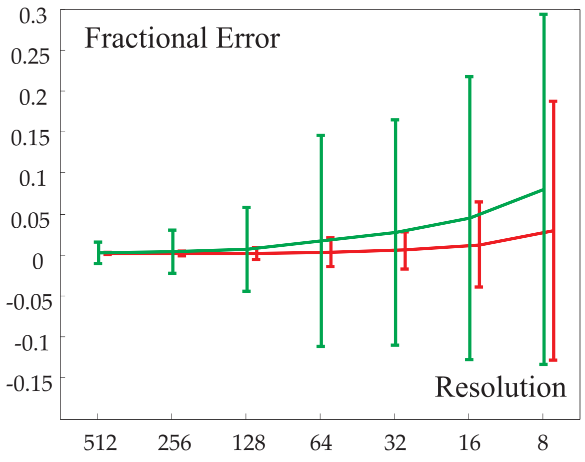


Figure 3.6: *Accuracy of the airlight model. The plots show the error (versus numerically integrating equation 3.5) as a function of the resolution for the 2D tables for $F(u, v)$. We report the fractional error, normalizing by the total airlight over the hemisphere. The error for each resolution is averaged over 40000 parameter values of $\beta$, $D_{sv}$, $D_{vp}$ and $\gamma$. Bilinear (red) and nearest neighbor (green) interpolation is used to interpolate $F(u, v)$ at non-grid locations of the indices $(u, v)$. The plots clearly indicate the high accuracy of our compact formula, and that a $64 \times 64$ table for $F(u, v)$ suffices for a maximum error of less than 2%.*

We start the derivation from equation 3.5.

$$L_a = \frac{\beta I_0}{4\pi}\int_0^{D_{vp}} \frac{e^{-\beta\sqrt{D_{sv}^2+x^2-2xD_{sv}\cos\gamma}}}{D_{sv}^2+x^2-2xD_{sv}\cos\gamma}\cdot e^{-\beta x}dx \tag{3.10}$$

$$\text{——>substitute } T_* = \beta D_* \text{ and } t = \beta x$$

$$= \frac{\beta^2 I_0}{4\pi}\int_0^{T_{vp}} \frac{e^{-\sqrt{T_{sv}^2+t^2-2tT_{sv}\cos\gamma}}}{T_{sv}^2+t^2-2tT_{sv}\cos\gamma}\cdot e^{-t}dt \tag{3.11}$$

$$\text{——>substitute } z = t - T_{sv}\cos\gamma$$

$$= \frac{\beta^2 I_0 e^{-T_{sv}\cos\gamma}}{4\pi}\int_{-T_{sv}\cos\gamma}^{T_{vp}-T_{sv}\cos\gamma} \frac{e^{-\sqrt{z^2+T_{sv}^2\sin^2\gamma}}}{z^2+T_{sv}^2\sin^2\gamma}\cdot e^{-z}dz \tag{3.12}$$

$$\text{——>substitute } z = T_{sv}\sin\gamma\tan\eta$$

$$= \frac{\beta^2 I_0 e^{-T_{sv}\cos\gamma}}{4\pi T_{sv}\sin\gamma}\int_{\gamma-\frac{\pi}{2}}^{\arctan\frac{T_{vp}-T_{sv}\cos\gamma}{T_{sv}\sin\gamma}} e^{-T_{sv}\sin\gamma\frac{1+\sin\eta}{\cos\eta}}d\eta \tag{3.13}$$

$$\text{——>substitute } \eta = 2\xi - \frac{\pi}{2}$$

$$= \frac{\beta^2 I_0 e^{-T_{sv}\cos\gamma}}{2\pi T_{sv}\sin\gamma}\int_{\gamma/2}^{\frac{\pi}{4}+\frac{1}{2}\arctan\frac{T_{vp}-T_{sv}\cos\gamma}{T_{sv}\sin\gamma}} \exp[-T_{sv}\sin\gamma\tan\xi]d\xi, \tag{3.14}$$

from which we obtain equation 3.9.

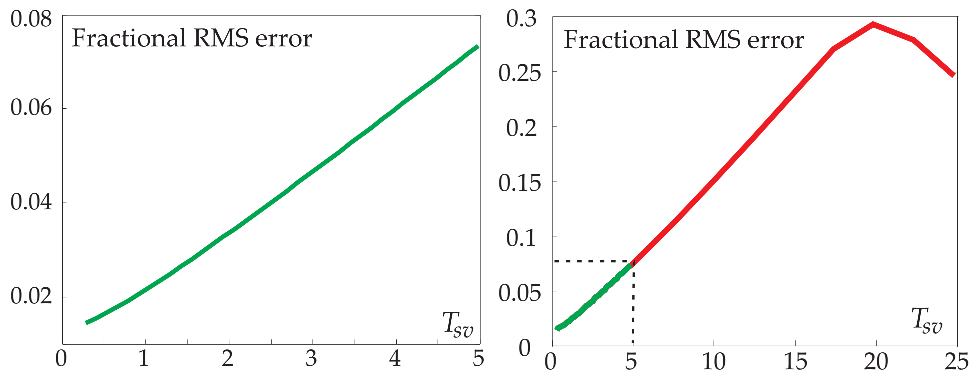


Figure 3.7: *Comparison of the airlight model with a standard Monte Carlo simulation that includes multiple scattering. The plots show the relative RMS error between the two methods for the case of isotropic phase function. [Left] The low RMS errors show that our model is physically accurate (less than 4% error) for optically thin media ($T_{sv} \leq 2$). [Right] From this plot, it is evident that multiple scattering becomes more important as optical thickness increases. However, the actual errors grow slowly and are still low for a wide range of optical thicknesses ($T_{sv} < 10$). It is also interesting to note that for very high optical thicknesses ($T_{sv} > 20$), attenuation dominates over scattering and once again the RMS errors decrease.*

Although equation 3.9 might seem complicated, it is really in a simplified form. We already have simple analytic expressions for $A_0$ and $A_1$. Further, the function $A_1$ is a numerical constant as far as the integration is concerned.

**STEP 2. Evaluating the integral using a special function:** To encapsulate the key concepts in the integrand of equation 3.9, we define the special function,

$$F(u, v) = \int_0^v \exp[-u \tan \xi] \, d\xi. \tag{3.15}$$

Unfortunately, there exists no simple analytic expression for $F(u, v)$. However, the function is a well behaved 2D function as shown in Fig. 3.5. Therefore, we can simply store it numerically as a 2D table. This is really no different from defining functions like sines and cosines in terms of lookup tables. In practice, we will use texture mapping in graphics hardware to access this 2D table. Note that $F(u, v)$ is purely numerical (independent of the physical parameters of the problem), and needs to be precomputed only once.

Finally, we can obtain for $L_a(\gamma, T_{sv}, T_{vp}, \beta)$,

$$L_a = A_0\Big[F(A_1, \frac{\pi}{4} + \frac{1}{2}\arctan\frac{T_{vp} - T_{sv}\cos\gamma}{T_{sv}\sin\gamma}) - F(A_1, \frac{\gamma}{2})\Big], \tag{3.16}$$

where we have omitted the parameters for $L_a$, $A_0$ and $A_1$ for brevity.

In the important special case of $T_{vp} = \infty$, corresponding to no objects along the viewing ray, we get $L_a(\gamma, T_{sv}, \infty, \beta)$ as

$$L_a = A_0(T_{sv}, \gamma, \beta)\Big[F(A_1(T_{sv}, \gamma), \frac{\pi}{2}) - F(A_1(T_{sv}, \gamma), \frac{\gamma}{2})\Big]. \tag{3.17}$$

**In summary, we have reduced the computation of a seemingly complex single scattering integral in equation 3.5 into a combination of an analytic function computation that depends on the physical parameters of the problem and a lookup of a pre-computed 2D smooth function that is independent of the physical parameters of the problem.** In the rest of the paper, we will demonstrate several extensions and applications of our model.

**Extension to General Phase Functions:** It is straight-forward to extend our model to more general phase functions. It is well known that phase functions $k(\alpha)$ of most media can be written as polynomials of $\cos\alpha$ [45, 20],

$$k(\alpha) = \sum_{k=0}^{n} W_k \cos^k \alpha\,, \tag{3.18}$$

where $W_k$'s are coefficients of the polynomial. Similar to the previous derivation for the isotropic case, we show in appendix B that the expression for general phase functions can be written using the special function $F^k(u, v)$ (analogous to $F(u, v)$),

$$F^k(u, v) = \int_0^v \cos^k 2\xi \exp[-u\tan\xi]\, d\xi. \tag{3.19}$$

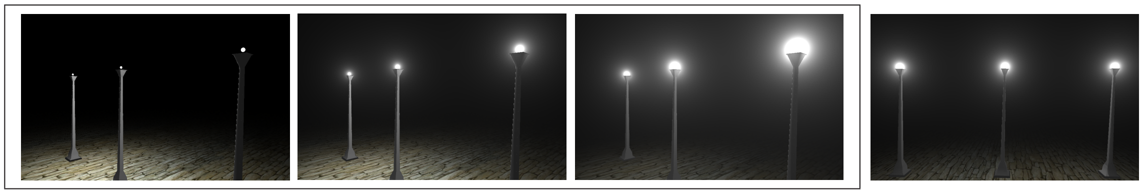


Figure 3.8: *The images show glows around three identical point light sources (street lamps) at different distances from the viewer. From left to right, we show three different values of the scattering coefficient $\beta$ ($\beta = 0, 0.01, 0.04$). Larger values of $\beta$ correspond to larger optical thicknesses $T_{sv}$. We clearly see the effect of greater glows for larger $\beta$. Also, the radiance from farther light sources is attenuated more in each individual image, resulting in smaller glows for the farther lights. In the fourth (rightmost) image, we show a different view with $\beta = 0.04$, where all the light sources are approximately equidistant, with the result that they have similar glows. (The shading on the surfaces is computed using the surface radiance model in Sec. 3.4.)*

Thus, if $n$ terms of the phase function are used, we need to store $n$ 2D tables. In practice, since a small value for $n$ (typically $n \leq 3$) typically suffices to represent most phase functions, only a few 2D tables need to be stored.

### 3.3.3 Accuracy of the Airlight Model

We first investigate the accuracy of our analytic model as compared to numerically integrating equation 3.5. Figure 3.6 shows plots of the mean error in $L_a$ as a function of the resolution of the 2D numerical table for the special function $F(u, v)$. We use interpolation to evaluate $F(u, v)$ at non-grid locations for the indices $(u, v)$ (bilinear and nearest neighbor interpolations are shown in Fig. 3.6). For each resolution, the error computed is averaged over 40000 sets of parameter values for $\beta$, $D_{sv}$, $D_{vp}$, $\gamma$. The error bars in the figure show the standard deviation of the errors. The plots indicate that even a low resolution 64×64 table suffices to compute $F(u, v)$ accurately, with a maximum error of less than 2%. As expected, bilinear interpolation performs better, but, for faster rendering, one can use nearest neighbor interpolation with only a small loss in accuracy.

We also validate the accuracy of the single scattering assumption in our airlight model. Figure 3.7 shows the relative RMS errors between glows around light sources computed using our model and a standard volumetric Monte Carlo approach that takes into account multiple scattering as well. The Monte Carlo simulation took approximately two hours to compute each glow, whereas our explicit model runs in real-time. The comparison was conducted for optical thicknesses over a wide range $T_{sv} \in (0.25, 25)$ and $T_{vp} \in (0.5, 50)$, which covers almost all real situations. As expected, for optically thin media ($T_{sv} \leq 2$), our model is very accurate (less than 4% relative RMS error). Interestingly, even for greater optical thicknesses ($T_{sv} > 2$), the error only increases slowly. Thus, our single scattering model may be used as a viable approximation for most common real-time rendering scenarios, such as games.

### 3.3.4 Visual Effects of the Airlight Model

The dependence of the model on the viewing direction $\gamma$ and the distance of the source from the observer $D_{sv}$, predicts visual effects like the glows around light sources and the fading of distant objects. As discussed above, these effects are physically accurate for thin fog (low $\beta$ and $T$), and qualitatively reasonable in other cases. In Fig. 3.8, we also see how these glows change as a function of the medium properties (the scattering coefficient $\beta$) and distance to the sources. As $\beta$ increases, we go from no glow ($\beta = T = 0$) to a significant glow due to scattering. The differences in the 3 light sources should also be observed. The farther lights are attenuated more, and we perceive this effect in the form of reduced glows around more distant sources. The final (rightmost) image in Fig. 3.8 shows a different viewpoint, where the sources are at approximately the same distance, and the glows therefore look the same.

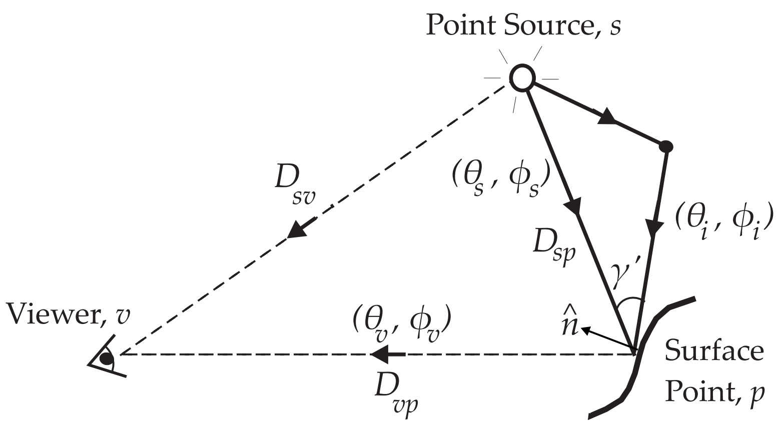


Figure 3.9: *Diagram showing how light travels from a point light source to a surface point and gets reflected towards the viewer by the surface point.*

## 3.4 The Surface Radiance Model

In this section, we discuss the effects of airlight on the outgoing surface radiance. Consider the illustration in Fig. 3.9, where an isotropic point light source $s$ illuminates a surface point $p$. We will calculate the reflected radiance at the surface. To get the actual appearance at the viewer, we need to attenuate by $\exp[-T_{vp}]$ as usual, where $T_{vp}$ is the optical thickness between viewer and surface point.

The reflected radiance $L_p$ is the sum of contributions, $L_{p,d}$ and $L_{p,a}$, due to direct transmission from the source, and single scattered airlight from the source respectively,

$$L_p = L_{p,d} + L_{p,a}. \tag{3.20}$$

The direct transmission corresponds to the standard surface reflectance equation, only with an attenuation of $\exp[-T_{sp}]$ added because of the medium, where $T_{sp}$ is the optical thickness between the source and the surface point:

$$L_{p,d} = \frac{I_0 e^{-T_{sp}}}{D_{sp}^2} f_r(\theta_s, \phi_s, \theta_v, \phi_v) \cos\theta_s, \tag{3.21}$$

where $f_r$ is the BRDF, $(\theta_s, \phi_s)$ is the direction to the source, and therefore also the incident direction, and $(\theta_v, \phi_v)$ is the viewing direction. All angles are measured with respect to the surface normal, in the local coordinate frame of the surface.

### 3.4.1 The Surface Radiance Integral

On the other hand, the single-scattered radiance $L_{p,a}$ is more complicated, involving an integral of the airlight ($L_a$ from equation 3.17) over all incident directions,

$$L_{p,a} = \int_{\Omega_{2\pi}} L_a(\gamma'(\theta_s, \omega_i), T_{sp}, \infty, \beta) f_r(\theta_i, \phi_i, \theta_v, \phi_v) \cos\theta_i \, d\omega_i \, . \tag{3.22}$$

Consider the parameters of $L_a$ in the integrand of the above equation. The angle $\gamma'$ in this case is the angle[2] between the incident direction $\omega_i$ and the source direction $(\theta_s, \phi_s)$. Note that for isotropic BRDFs, we can always rotate the coordinate system so $\phi_s = 0$, allowing us to write $\gamma'(\theta_s, \omega_i)$. Finally, $L_a$ also depends on the optical thickness between the source and the surface point $T_{sp}$ (instead of between source and viewer in equation 3.17).

We refer to equation 3.22 as the *surface radiance single scattering integral*, analogous to the airlight single scattering integral in equation 3.5, and next focus on deriving an explicit compact form.

### 3.4.2 Solution to the Surface Radiance Integral

First consider the Lambertian case, so the BRDF is a constant $k_d$. The integral will then depend only on the parameters of $L_a$, i.e. $\gamma'$, $T_{sp}$ and $\beta$. Of these, the dependency on $\beta$ is primarily a normalization factor and does not affect the integrand. The angle $\gamma'$ is a function of the source direction $\theta_s$, and the integration variable $\omega_i$. Hence, the integrand depends on only two physical variables, $T_{sp}$ and $\theta_s$. Thus, as in

[2] We use the prime on $\gamma'$ to make a technical distinction from the angle $\gamma$ between the source and viewer used in the previous section. An explicit trigonometric formula for $\gamma'$ is given in appendix B.

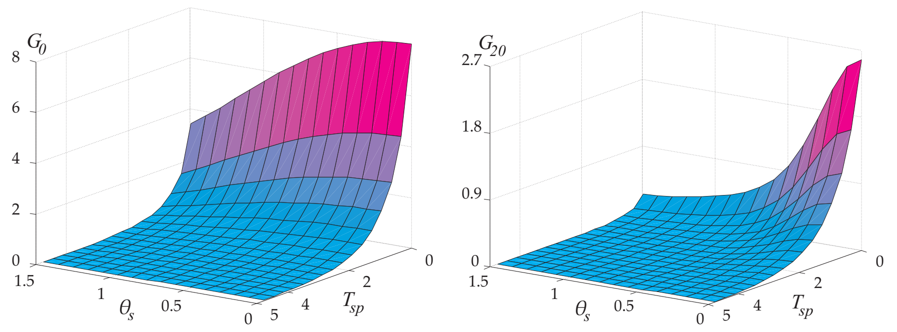


Figure 3.10: *3D plots of functions $G_0$ and $G_n$ for $n = 20$ in the range of $0 \leq T_{sp} \leq 10$ and $0 \leq \theta_s \leq \frac{\pi}{2}$. The plots show that both functions are well-defined and smooth and can therefore be precomputed as 2D tables. The functions reach their peak values for $\theta_s = T_{sp} = 0$, decaying with increases in both parameters. The decay is faster for the peakier $G_{20}$ on the right.*

the previous section, we can define a special two-dimensional numerical function $G_0(T_{sp}, \theta_s)$.

For the Phong BRDF, we employ the reparameterization method in [88], measuring angles from the reflection of the viewing direction about the surface normal, rather than the surface normal itself. To indicate this, we denote by $\theta_s'$ the angle the source makes with respect to this reflected direction. Upon making this transformation, it can be shown that the Phong BRDF is mathematically analogous to the Lambertian case. To allow for the Phong exponent $n$, we define the 2D function $G_n$ instead of $G_0$. These functions are well-defined and smooth as shown by the plots in Fig. 3.10. The details of these calculations are in appendix B, and the formula for $G_n$ is

$$G_n(T_{sp}, \theta_s') = \int_{\Omega_{2\pi}} \frac{e^{-T_{sp}\cos\gamma'}}{\sin\gamma'} \Big[F(A_1, \frac{\pi}{2}) - F(A_1, \frac{\gamma'}{2})\Big] \cos^n \theta_i \, d\omega_i, \tag{3.23}$$

where $\gamma'$ and $A_1(T_{sp}, \gamma')$ are functions of $\theta_s'$ and $\omega_i$, i.e. $\gamma'(\theta_s', \omega_i)$.

Now we briefly recap the mathematical derivation for the surface radiance model per the description above. We first consider the Lambertian BRDF, beginning with equation 3.22. Note that in the derivation below, $\gamma'$ is given from trigonometry by $\gamma'(\theta_s, \omega_i) = \cos\theta_i \cos\theta_s + \sin\theta_i \sin\theta_s \cos\phi_i$.

$$L_{p,a} = \int_{\Omega_{2\pi}} L_a(\gamma'(\theta_s,\omega_i), T_{sp}, \infty, \beta) f_r(\theta_i,\phi_i,\theta_v,\phi_v)\cos\theta_i \, d\omega_i \tag{3.24}$$

——>substitute equation 3.17 for $L_a$ and a constant $k_d$ for $f_r$

$$= \int_{\Omega_{2\pi}} A_0(T_{sp},\gamma,\beta)\Big[F(A_1(T_{sp},\gamma'),\frac{\pi}{2}) - F(A_1(T_{sp},\gamma'),\frac{\gamma'}{2})\Big] k_d \cos\theta_i \, d\omega_i \tag{3.25}$$

——>substitute equation 3.7 for $A_0$ and take constants out of integration

$$= \frac{\beta^2 I_0 k_d}{2\pi T_{sp}} \int_{\Omega_{2\pi}} \frac{e^{-T_{sp}\cos\gamma'}}{\sin\gamma'} \Big[F(A_1(T_{sp},\gamma'),\frac{\pi}{2}) - F(A_1(T_{sp},\gamma'),\frac{\gamma'}{2})\Big] \cos\theta_i \, d\omega_i$$

$$= \frac{\beta^2 I_0 k_d}{2\pi T_{sp}} G_0(T_{sp},\theta_s). \tag{3.26}$$

For the Phong BRDF after reparameterization, instead of $k_d \cos\theta_i$, we will obtain $k_s \cos^n\theta_i$, where $n$ is the Phong exponent. This can be handled exactly as above, simply replacing $G_0$ with $G_n$.

The final shading formula, considering both direct transmission and single scattering is then given by:

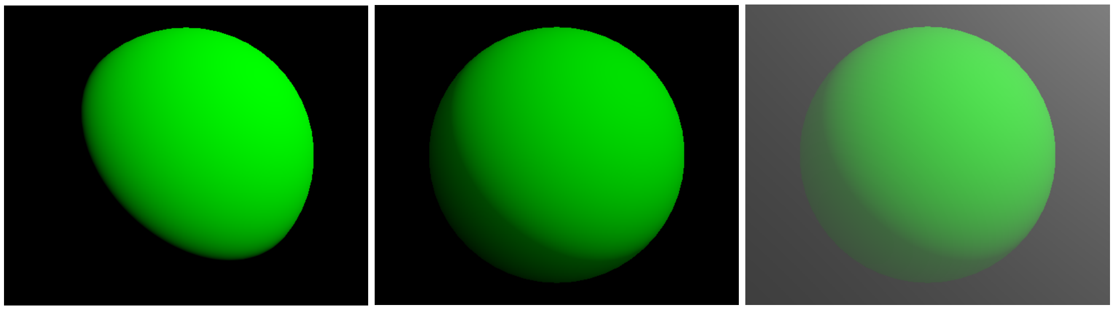
$$L_p = I_0 k_d \Big[\frac{e^{-T_{sp}}}{D_{sp}^2}\cos\theta_s + \beta^2 \frac{G_0(T_{sp},\theta_s)}{2\pi T_{sp}}\Big] + \tag{3.27}$$

$$I_0 k_s \Big[\frac{e^{-T_{sp}}}{D_{sp}^2}\cos^n\theta_s' + \beta^2 \frac{G_n(T_{sp},\theta_s')}{2\pi T_{sp}}\Big]. \tag{3.28}$$

As in the airlight model derivation, we have reduced the computation of surface radiance due to single scattering to a few analytic function evaluations and a few 2D table lookups.

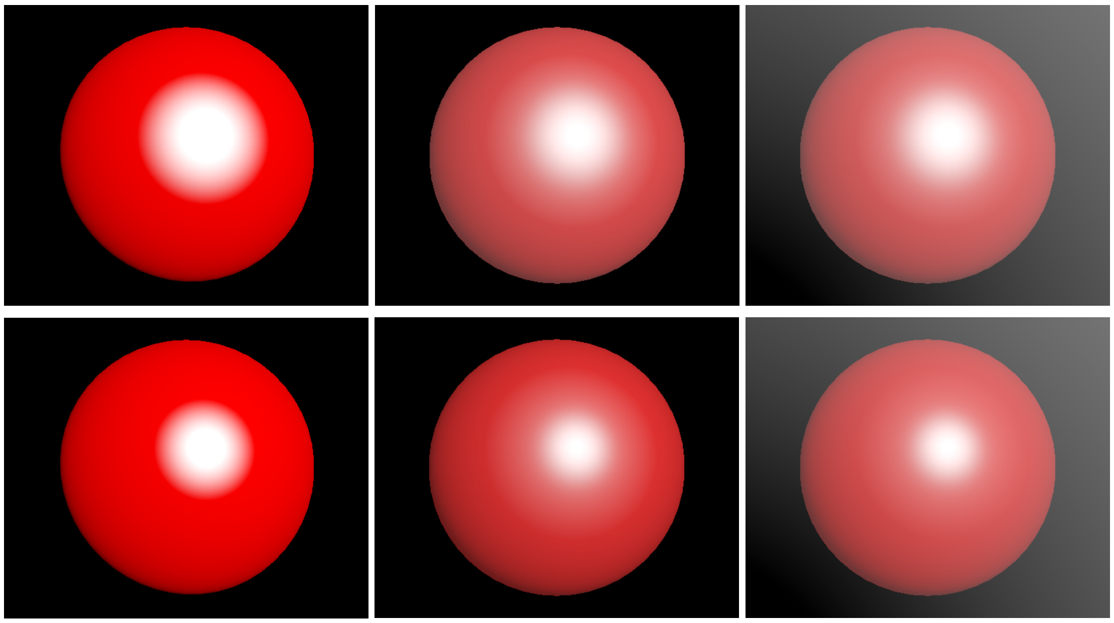
### 3.4.3 Visual Effects of the Surface Radiance Model

To illustrate the different qualitative effects we see on surfaces due to scattering, we rendered spheres with Lambertian (Fig. 3.11) and Phong BRDFs (Fig. 3.12) using



(a) (b) (c)

Figure 3.11: *Influence of scattering on Lambertian surface radiance. In the foggy image (b), created using our surface radiance model, we see a dimming due to attenuation and diffusing of shading (note the brightening of darker areas compared to the clear day image in (a)). These effects are perhaps more apparent in (c), where we also include airlight from the source.*



(a) (b) (c)

Figure 3.12: *Influence of scattering on specular surface radiance (top row has Phong exponent 10, bottom has Phong exponent 20). In the foggy images (b), we see a dimming and diffusing of the specular highlight compared to the clear-day image in (a). Note also the overall loss in color saturation and contrast, especially in (c). These are important visual effects, usually missing in previous interactive techniques.*

our model above. The columns are from left to right (a) no scattering, (b) the effects of scattering on surface shading, and (c) combining this with the airlight effects directly from the source. For the Lambertian sphere in Fig. 3.11, we see a dimming due to attenuation of light through the scattering medium, and the diffusing of

```
frag2app fmain(
   float4 objPos : TEXCOORD3,                                  // 2D texture coords
   ...
   uniform samplerRECT F,                                      // 2D special functions
   uniform samplerRECT $G_0$,
   uniform samplerRECT $G_n$)
{
   frag2app OUT;                                               // output radiance
   // Set up and calculate $T_{sv}$, $\gamma$, $D_{sv}$, $T_{vp}$, $\theta_s$ and $\theta'_s$

   /********** Compute $L_a$ from equation 3.16 ******/
   $A_0 = (\beta * I_0 * \exp[-T_{sv} * \cos\gamma])/(2\pi * D_{sv} * \sin\gamma);$      // equation 3.7
   $A_1 = T_{sv} * \sin\gamma;$                                // equation 3.8
   $v = \pi/4 + (1/2)\arctan\left[(T_{vp} - T_{sv} * \cos\gamma)/(T_{sv} * \sin\gamma)\right];$
                                                               // v is one of texture coords
   $f_1 = texRECT(F, float2(A_1, v));$                         // 2D texture lookup
   $f_2 = texRECT(F, float2(A_1, \gamma/2));$
   $airlight = A_0 * (f_1 - f_2);$                             // equation 3.16

   /********** Diffuse surface radiance from equation 3.27 ******/
   $d_1 = k_d * \exp[-T_{sp}] * \cos\theta_s * I_0/(D_{sp} * D_{sp});$
   $d_2 = (k_d * I_0 * \beta * \beta)/(2\pi * T_{sp}) * texRECT(G_0, float2(T_{sp}, \theta_s));$
   $diffuse = d_1 + d_2;$

   /********** Specular surface radiance from equation 3.28 ******/
   $s_1 = k_s * \exp[-T_{sp}] * \cos^n\theta'_s * I_0/(D_{sp} * D_{sp});$
   $s_2 = (k_s * I_0 * \beta * \beta)/(2\pi * T_{sp}) * texRECT(G_n, float2(T_{sp}, \theta'_s));$
   $specular = s_1 + s_2;$

   /********** Final Color (equation 3.29) ******/
   $OUT.color = airlight + (diffuse + specular) * \exp[-T_{vp}];$
   return   OUT;
}
```

Figure 3.13: *Pseudocode for the Cg fragment shader that implements our combined model for airlight and surface radiance.*

shading leading to a brightening of darker shadow regions. In the specular case, we see a dimming and diffusing out of the specular highlight due to scattering, combined with an overall reduction in color saturation and contrast. These are important qualitative shading effects that add to the realism of scene appearance in scattering media.

## 3.5 The Complete Model and Implementation

While the mathematical derivations in the previous two sections are somewhat involved, the actual implementation is straightforward. Our model provides an explicit form that can be easily implemented in modern programmable graphics hardware. This requires minimal changes to both the original rendering code and scene description, and can therefore be easily integrated into other existing real-time rendering methods. Indeed, the user need only specify the coefficient $\beta$ of the medium, as in standard OpenGL fog, and use the shader corresponding to our model (pseudocode is in Fig. 3.13).

To compute the final appearance, we sum up the attenuated reflected radiance from the surface and the airlight from the source,

$$L = e^{-T_{vp}} L_p + L_a. \tag{3.29}$$

$L_a$ is the airlight and is given by equation 3.16. $L_p$ is the exitant radiance at the surface and is given by equations 3.27 and 3.28. **We only need to compute a few simple analytic terms and do 4 texture lookups for each vertex or pixel, two for special function F, and one each for $G_0$ and $G_n$** (these texture lookups correspond to the texRECT function call in the pseudocode of Fig. 3.13). Clearly, these computations can be done by modern programable graphics cards interactively in a single rendering pass.

In practice, we implement the model using Cg in the fragment shader of an NVidia Geforce 6800 graphics card. The special functions $F$, $G_0$ and $G_n$ are precomputed and tabulated as $64 \times 64$ floating point textures. Since these textures are precomputed only once, we minimize frequent data transfer between the graphics card and main memory.

The rendering speed depends on a variety of variables, and rendering time is linear in the number of light sources. As shown in the video, we are able to achieve real-time rates even for fairly complex scenes with several light sources. As an example, we rendered the scene shown in Fig. 3.1, with 39,999 vertices and 66,454 triangles. We simulated the scattering effects from 4 light sources and achieved about 20 fps using the graphics hardware mentioned above. The model for the scene was obtained from http://hdri.cgtechniques.com.

## 3.6 Complex BRDFs and Lighting

So far, we have considered arbitrarily located point lights, and simple Lambertian and Phong BRDFs, showing how an explicit expression can be derived and implemented. Rendering time is linear in the number of lights. In this section, we show how these ideas can be extended to efficiently handle complex BRDFs and environment map lighting using convolution, if we are willing to make particular simplifying assumptions.

We first introduce the notion of a point-spread function (PSF) for the radiance or glow from a point source due to single scattering. This is similar in spirit to the PSFs derived by Narasimhan and Nayar [74] and Premoze et al. [85] in the context of multiple scattering for offline rendering. We will then discuss a number of applications including

- Rendering arbitrary BRDFs with point light sources, by convolving the BRDF with this PSF, as shown in Fig. 3.15. This approach can be used if we are willing to precompute a tabular BRDF representation, instead of using a simple explicit formula, as for Lambertian and Phong BRDFs.

- Convolving an environment map with the PSF to efficiently handle very complex lighting (with possibly thousands of lights, corresponding to the pixels of an environment map). This convolution is possible if we assume that all light sources are equally far away, as in a distant environment map. This enables us to obtain the characteristic glows and blurriness around light sources on foggy days, as shown in Fig. 3.16.

- Integrating volumetric scattering into precomputed radiance transfer methods that include complex lighting, realistic materials, cast shadows and interreflections (Fig. 3.17). The idea of convolving with the point-spread function can be applied to almost any technique that uses environment maps, enabling environment mapping and precomputed radiance transfer to be used in participating media for the first time.

Throughout the section, we apply the signal-processing results of Ramamoorthi and Hanrahan [87] and Basri and Jacobs [7] to efficiently compute the convolutions in the frequency domain using spherical harmonics.

### 3.6.1 Airlight Point Spread Function (PSF)

In Sec. 3.3, we determined the radiance $L_a(\gamma, D_{sv}, D_{vp}, \beta)$ from a point source reaching a viewer, due to single scattering. If we fix the distance to the source $D_{sv}$, the integrating distance $D_{vp}$, and the scattering coefficient $\beta$ of the medium, the radiance becomes a function only of the angle $\gamma$. We normalize this function by $I_0/D_{sv}^2$ to account for the intensity of the source, and define the PSF as

$$\mathrm{PSF}(\gamma)_{D_{sv},D_{vp},\beta} = \frac{D_{sv}^2 L(\gamma, D_{sv}, D_{vp}, \beta)}{I_0}. \tag{3.30}$$

Since the PSF is mostly applied for surface shading, we will generally set $D_{vp} = \infty$, as in Sec. 3.4.

### 3.6.2 Empirical PSF factorization for Speedup

The PSF defined above still depends on the parameters of the medium such as the coefficient $\beta$. So, changing these parameters changes the PSF and requires us to redo any convolutions. However, we have observed empirically that the PSF above can be factored into **a purely angular component that is independent of the**

**medium parameters and an amplitude component that depends on the medium parameters. This factorization enables us to change the medium parameters interactively without having to re-compute the PSF or redo any convolutions.** Specifically,

$$\mathrm{NPSF}(\gamma) = \frac{F(\sin\gamma, \frac{\pi}{2}) - F(\sin\gamma, \frac{\gamma}{2})}{2\pi\sin\gamma\cdot e^{(\cos\gamma-1)}} \tag{3.31}$$

$$\mathrm{PSF}(\gamma)_{D_{sv},\infty,\beta} = T_{sv}e^{-T_{sv}}\cdot\mathrm{NPSF}(\gamma), \tag{3.32}$$

where NPSF has only angular dependence, independent of other physical parameters.

The empirical PSF factorization is inspired by the observation that after being normalized by $T_{sv}e^{-Tsv}$, the PSF becomes essentially independent of the medium physical parameters (optical thickness) and largely depends on angle $\gamma$ as shown in Fig. 3.14 (left). This implies we can factor the PSF into a purely angular component and an amplitude component that depends on the medium parameters. We define the angular component NPSF($\gamma$) as the PSF$(\gamma)_{T_{sv}=1}$ normalized by $T_{sv}e^{-Tsv}$ and define the amplitude component as the normalization factor $T_{sv}e^{-T_{sv}}$. Then, the PSF can be expressed using these two terms as in equation 3.32. The absolute approximation error is plotted in Fig. 3.14 (right) for 11 different optical thickness ranging from 0.1 to 3.1.

### 3.6.3 Rendering with arbitrary BRDFs

We can use convolution with the PSF to render with arbitrary tabulated BRDFs, such as measured reflectance. For each outgoing direction, we tabulate the BRDF as a function over the sphere of incident directions. A new effective BRDF can be obtained for that outgoing direction by convolving this function with the PSF.

Mathematically, we first write the (isotropic) BRDF in terms of spherical har-

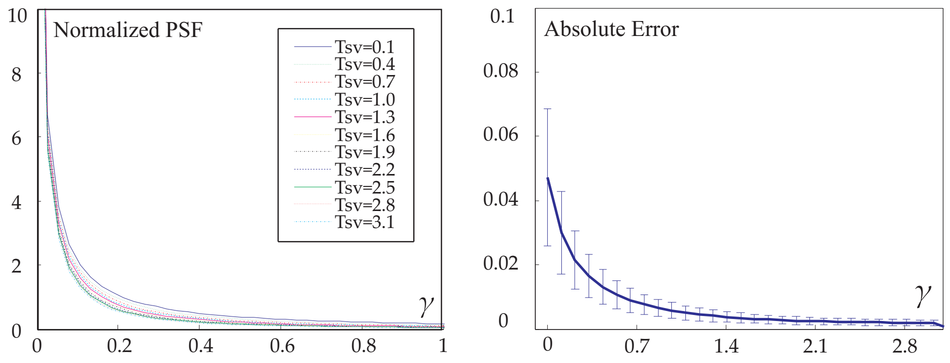


Figure 3.14: *[Left] Plot of PSF($\gamma$)$_{D_{sv},\infty,\beta}$ normalized by $T_{sv}e^{-T_{sv}}$ for different optical thicknesses $T_{sv}$ ranging from 0.1 to 3.1. After normalization, the PSF depends on$\gamma$ and is largely independent of $T_{sv}$. This implies that we can factor it into a purely angular component and an amplitude component depending on $T_{sv}$. [Right] The average and standard deviation of the absolute error of the empirical PSF. While the error increases for smaller angles $\gamma$, it remains well below 0.05.*

monic coefficients for each outgoing angle as

$$\rho^{\text{orig}}(\theta_i, \theta_o, \phi) = \sum_{l,m} \rho^{\text{orig}}_{lm}(\theta_o) Y_{lm}(\theta_i, \phi), \tag{3.33}$$

where $\rho^{\text{orig}}_{lm}$ are the coefficients, and $Y_{lm}$ is the spherical harmonic. To perform the convolution [87], we multiply the coefficients $\rho^{\text{orig}}_{lm}$ of the original BRDF by the corresponding coefficients of the point-spread function[3] PSF$_l$,

$$\rho^{\text{eff}}_{lm}(\theta_o) = \sqrt{\frac{4\pi}{2l+1}} \text{PSF}_l \rho^{\text{orig}}_{lm}(\theta_o). \tag{3.34}$$

Then, we can use the effective BRDF to compute the reflected radiance due to airlight, and the original BRDF for the reflected radiance due to direct transmission. Thus, standard rendering algorithms can be executed with only slight modification and at virtually no additional cost. Note however, that while our previous formulae for Lambertian and Phong models required no precomputations, the convolution approach requires precomputation of the spherical harmonic coefficients

[3]Since the PSF is radially symmetric, depending only on $\gamma$, only spherical harmonic coefficients with m = 0 are nonzero.

for a collection of outgoing angles.

Figure 3.15 shows images rendered with the Blue metallic BRDF measured by Matusik et al. [66]. In the left image, we simply render a tabular description of the BRDF without scattering. In the right image, we use the formula above to compute a new effective tabulated BRDF, including the effects of airlight. The brightening of darker regions owing to scattering is clearly visible on the right.

### 3.6.4 Rendering with Environment Maps

Our point spread function can be applied directly to environment maps, with the effects of scattering obtained by convolving the environment map with the PSF. To use a single PSF for all sources, we must assume that the lighting is made of small equidistant light sources (fixed $D_{sv}$). This is a good approximation when the size of the objects is small compared to the distance to the environment[4].

We first consider simply looking at the environment, where we would like to see the glows around the bright light sources, to create the effects of foggy or misty appearance. To achieve this effect, we simply need to convolve the environment map with the PSF,

$$L_{lm}^{convolved} = \sqrt{\frac{4\pi}{2l+1}} PSF_l L_{lm}^{original}. \tag{3.35}$$

Furthermore, similar to equation 3.1, we can simply use a combination of the original attenuated environment map $L^{attenuated}$ (for direct transmission, and corresponds to $L_d$ in equation 3.1) and the convolved version $L^{convolved}$ above (for airlight, and corresponds to $L_a$ in equation 3.1) to compute the surface shading,

$$L^{final} = L^{attenuated} + L^{convolved} \tag{3.36}$$

$$L^{attenuated} = L^{original} e^{-T_{sv}}. \tag{3.37}$$

[4]Note that while this assumption is similar to standard environment mapping, our PSF requires us to specify a finite (but possibly large) $D_{sv}$.

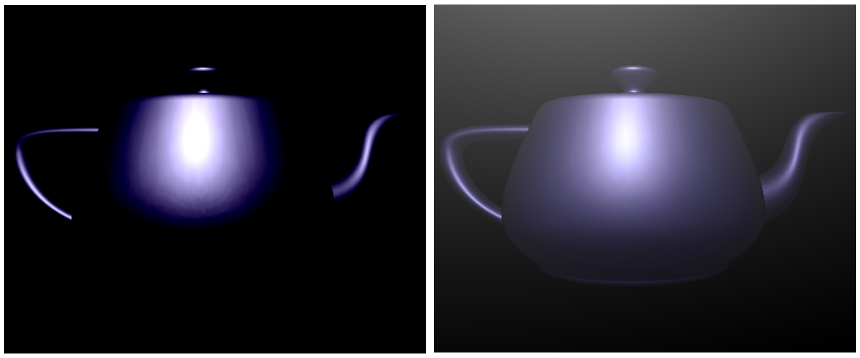


Figure 3.15: *[Left] A teapot rendered using the measured blue metallic BRDF. [Right] The teapot as it appears in a scattering medium. The brightening of darker regions, and softening of shading, is clearly visible.*

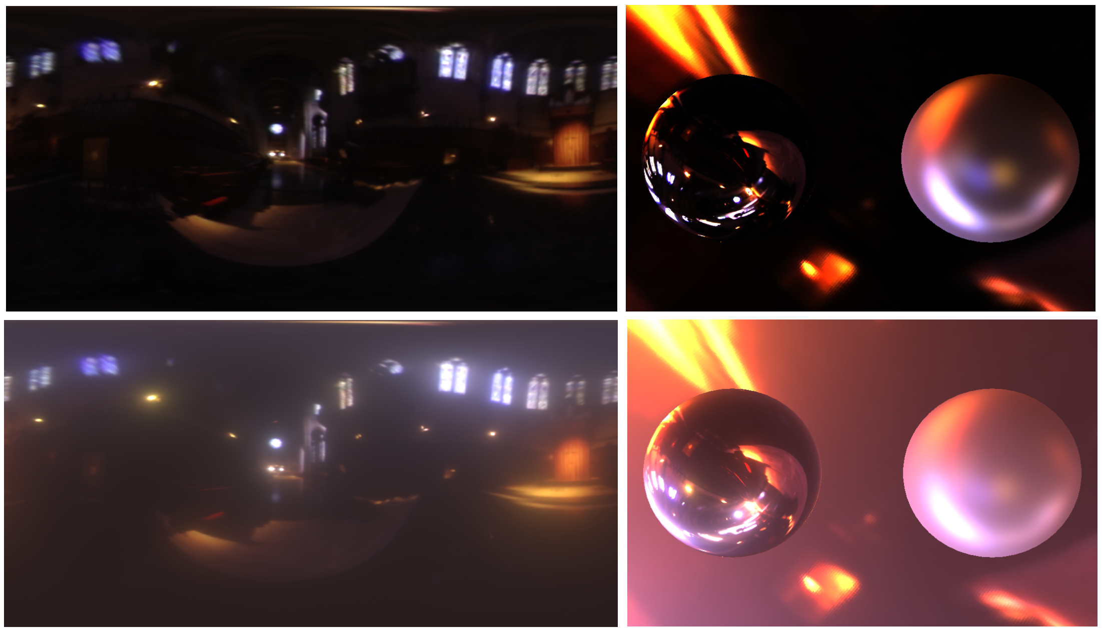


Figure 3.16: *[Top Left] Grace cathedral environment map with no scattering. [Bottom Left] The environment map is convolved with the airlight single scattering PSF to create a foggy/misty appearance. Notice the glows around the light sources, the blurring of the sources and the brightening of dark areas. [Right] A scene illuminated by the environment map without scattering (left) and with scattering (right). Notice the spreading of the highlights and brightening of the spheres due to scattering.*

Figure 3.16 shows results obtained by convolving the Grace Cathedral environment map [27] with the single scattering PSF. The blurring of light sources and the overall increase in brightness in dark regions can be immediately seen. Below that, we compare the appearances of spheres rendered illuminated by this environ-

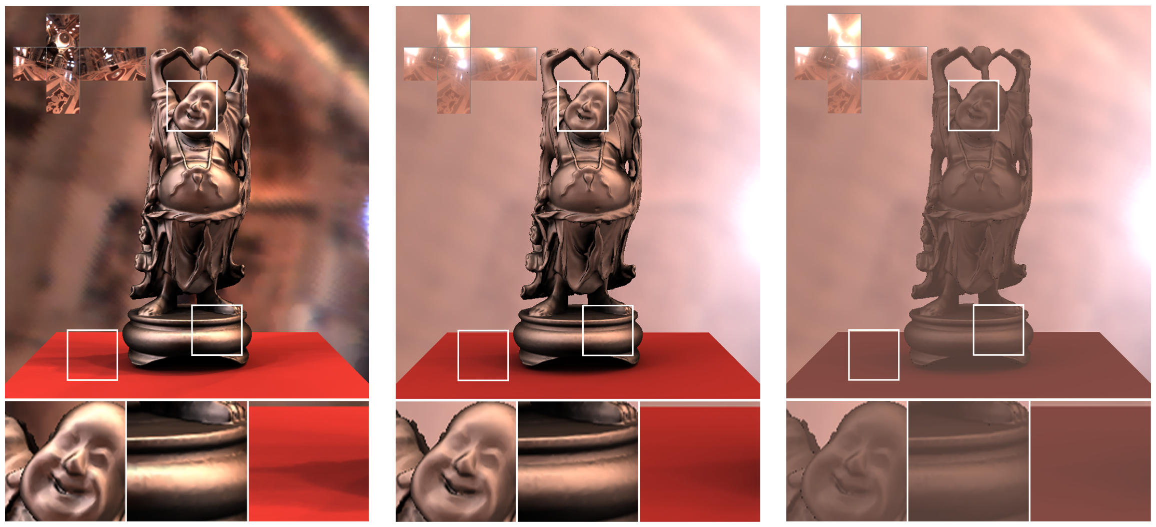


Figure 3.17: *[Left] A scene rendered using precomputed radiance transport, to capture the complex shadows and specular surface BRDFs. [Middle] The same scene rendered as though it was immersed in a scattering medium, with the observer close to the object. Notice the blurring and fogging of the environment in the background. In the insets, we see a number of shading changes, such as the brightening of dark regions in the face because of the scattering of light, the attenuation due to dimming and diffusing of specular highlights on the base, and the softening of shadows on the plane. [Right] The same scene including effects of airlight between viewer and object (as if the viewer were far away, seeing the Buddha through fog). Note the loss in contrast and saturation.*

ment map with and without scattering. Notice the spreading of highlights and the brightening of the objects.

### 3.6.5 Precomputed Radiance Transfer

The traditional environment map rendering techniques do not take shadows or interreflections into account. Precomputed radiance transport methods [102] compute the visibility in an off-line manner, followed by interactive rendering. To add participating media, we only need to convolve the lighting (environment map) with our PSF and use the result as input to the existing precomputed radiance transfer techniques. To demonstrate this, we used the technique of [120, 65], which handles non-diffuse objects under all-frequency environment illumination using a separable BRDF approximation.

We show the result of the Happy Buddha model rendered with the Ashikhmin-

Shirley BRDF [3] in Fig. 3.17. The left image is the standard result with no scattering. In the middle image, we show a view of the Buddha, where we include the effect of airlight from the environment on surface appearance, but there is no attenuation or scattering between the viewer and object itself (as if the observer were very close to the Buddha). We clearly see the foggy appearance of the background and the glows or airlight due to the light sources. On the face of the Buddha, we see a brightening of darker regions, along with a dimming and diffusing of specular highlights. A similar effect is seen on the base, where the dimming and diffusing of highlights reduces saturation and contrast. Finally, the shadows on the plane are blurred out, with a considerable softening and loss of detail. In the right image, there is also scattering or airlight between the object and the viewer (as if the observer were far away and seeing the Buddha through fog). This leads to a further loss of detail and contrast, so that the original glossy appearance of the object is essentially lost.

## 3.7 Discussion and Future Work

Our approach can be easily implemented in programmable graphics hardware and leads to a number of new effects in the real-time domain, such as interactive rendering with glows around light sources, the effects of scattering on surface shading, environment maps, and precomputed light transport. The key insight is a new analytic model for integrating the light transport equations assuming single scattering, which can also be extended to predict the impact of scattering or airlight on the inherent appearance of surfaces. More broadly, our approach indicates the power of using explicit formulae to simulate difficult effects like volumetric scattering, speeding up such a process by many orders of magnitude. We do sacrifice some generality, considering only isotropic point light sources, single scattering, homogeneous media, and excluding most cast and volumetric shadowing, but believe this is a worthwhile tradeoff to enable a simple technique that achieves real-time rates.

Future work can follow many avenues. For instance, we can attempt to extend our theoretical model to consider non-isotropic light sources (like spotlights) and inhomogeneous media. Our preliminary work in this area indicates that some of these generalizations, while relatively simple in standard surface calculations, are rather non-trivial for volumetric media. In general, we believe that analytic models of difficult to simulate volumetric phenomena are critical to achieving efficient renderings for real-time applications.

# Chapter 4

# Affine Double and Triple Product Wavelet Integrals for Rendering

## 4.1 Introduction

In last chapter, we presented an analytic single scattering model for real-time rendering. In this chapter, we focus on real-time near-field relighting and take a wavelet based approach to this problem. We start by reviewing some mathematical fundamentals for relighting. Integrations of products of functions are common in computer graphics and applied mathematics. For example, the reflection equation can be viewed as either a triple product integral [76] consisting of three factors: the lighting, BRDF, and visibility, or a double product integral where the BRDF and visibility are combined into the light transport function. A common assumption is that the illumination is distant, and each factor is represented in basis functions such as spherical harmonics or wavelets.

Local area lights have long been used as not only a practical modeling tool but also an indispensable artistic device to set up a mood for a scene, e.g., comfortable couches in a living room illuminated by a ceiling lamp (Fig. 4.1A), or stretch chairs

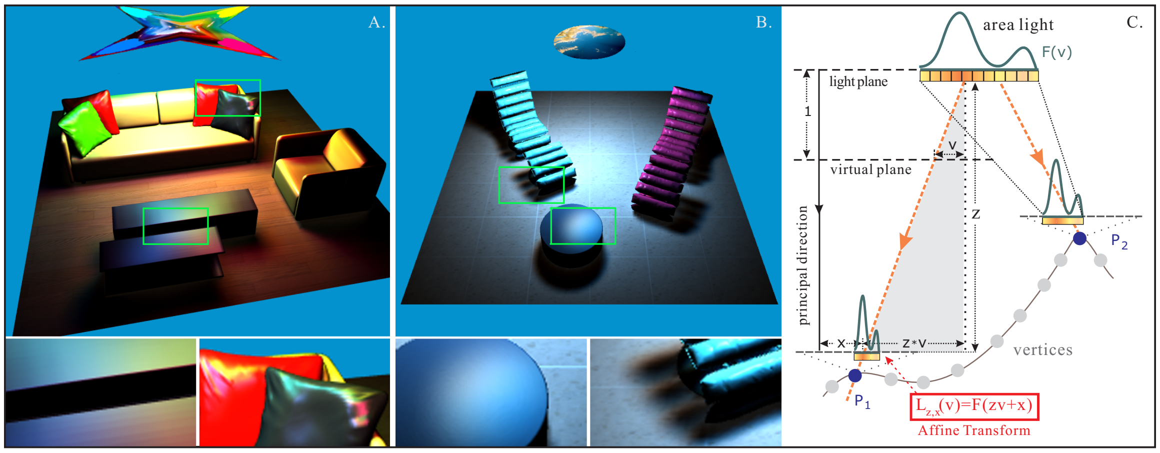


Figure 4.1: ***A and B:*** *Our method enables relighting of scenes lit with near-field illumination—a planar area source can be moved, retextured, reshaped and rotated (in plane) at real-time rates. Important effects like spatially varying shading on the floors in 1A and 1B, soft shadows under the chairs in the close-up of 1B, and colored specularities on the cushions and tables in the close-ups of 1A and 1B are clearly visible. These effects are difficult to capture using only distant lighting, as shown in Fig. 8C.* ***C:*** *Area lighting can be formulated as an affine transform. For simplicity, we parameterize the light field using a spatial coordinate x and an angular direction, given by the intercept v on a virtual plane, as shown in the diagram. The original area light is denoted as F(v), and the vertex's incident radiance as $L_{z,x}(v)$, where z and x respectively are the vertical and horizontal coordinates of the vertex $P_1$. The intensity at $P_1$, $L_{z,x}(v)$, is then given by an affine transform (F(zv+x)) from simple trigonometry. Our main contribution is a novel theory of affine double and triple product wavelet integrals that enables near-field area lighting to be used in almost any precomputation-based relighting framework.*

illuminated by the light from a circular vista window (Fig. 4.1B). Standard double and triple product integrals, however, are not suitable for near-field relighting. The basic problem is that the incident lighting varies across the scene, and needs to be evaluated at all points in space. With proper parameterization however, light propagation from planar area sources can be formulated as an affine transformation of the original source radiance, as shown in Fig. 4.1C. We call the transformed integrals affine double or triple product integrals (Sec. 4.3).

We present the first theoretical and computational analysis of affine double and triple product integrals in computer graphics. Our work is published in [110]. For actual computations, we focus our analysis on Haar wavelets, which have gained considerable attention in relighting. Haar wavelets are simple, and superior in compactly representing all-frequency effects such as natural lighting, specular BRDFs and intricate shadowing, often using only $1 \sim 2\%$ of coefficients [75]. Fundamentally, the problem we are trying to solve is to find an efficient representation for wavelets that are affinely transformed (scaled and translated). Wavelets, however, are known for their lack of even translation invariance [108]. For example, simply translating a Haar wavelet basis function one pixel to the left would change its coefficients dramatically, causing its power to spread across many different sub-bands.

Our main technical contribution is a novel affine double and triple product integral theory for Haar wavelets, which is presented in Sec. 4.4. The theory is developed primarily on 1D signals—since 2D and higher dimensional wavelets are simply products of 1D basis functions, a direct extension to higher dimensions is possible (Sec. 4.4.4). Note that we focus on 1D affine transforms, i.e. scaling and translations, and therefore do not consider rotations and shears in 2D or 3D. As seen in our practical applications, this is a general wavelet framework for many rendering problems.

The standard theory of double and triple product integrals is expressed in terms of standard coupling and tripling coefficients respectively. Our theory studies the affine analogs, that must now account not only for the different basis functions being coupled or integrated, but also the scale and translation in the affine transform. Affine coupling and tripling coefficients therefore gain two more degrees of freedom and are respectively 4D and 5D functions for 1D signals. In Sec. 4.4.2, we show that these coefficients can be boiled down to an intuitive 2D analytic core function, which we call the canonical coupling coefficient $M$. The canonical coupling coefficient exposes the inherent sparsity of the affine transform, which can be exploited to develop efficient computational methods. This is analogous to how standard tripling coefficients are theoretically complex, but actually sparse in

Haar wavelets [76].

Our theoretical development enables fast practical algorithms for affine transforms in Haar wavelets. This overturns a commonly held view that operations like shifts or scales are difficult in the wavelet domain. Our main practical focus is on relighting, and we take a significant step towards generalizing wavelet-based relighting methods to near-field settings with planar area light sources (Sec. 4.5; Figs. 4.1, 4.8 and 4.9). There are also applications to a variety of other problems that depend on wavelet representations. Section 4.6 describes initial solutions for wavelet importance sampling [22] with near-field area lights, and image processing (dilation and translation) directly in the wavelet domain. Readers more interested in implementation may want to first familiarize themselves with the basic concepts introduced in Sec. 4.3, and then focus on the applications in Secs. 4.5 and 4.6, skimming through the development of the theory in Sec. 4.4 as needed.

## 4.2 Previous Related Work

**Light Transport Analysis:** Recent papers [31, 89] have conducted a comprehensive analysis of light transport in Fourier and gradient representations. As noted in [31], one of the main aspects is the propagation of light from an area source, which can be written as an affine transform (in much the same form as Fig. 4.1C). Ramamoorthi et al. [89] characterized the basic mathematical operations of light transport, noting that linear or affine transforms are a key element, but there is no simple formula in wavelets. By developing a framework for affine double and triple product wavelet integrals, we take a significant step towards a full computational framework for rendering in the wavelet domain.

**Double and Triple Product Integrals:** Much of the work in relighting [102, 75] can be seen as double product integrals of the illumination and the light transport function. These integrals usually reduce to simple dot products in orthonormal bases like spherical harmonics and Haar wavelets. Subsequently, Ng et al. [76] developed the triple product integral framework to consider the integration of the lighting, BRDF and visibility, as needed for changing both view and illumination. The same mathematics can be applied to efficiently multiplying two wavelet signals [9]. Most recently, these results have been generalized to products of multiple functions [112]. Our work can be viewed as an important generalization of the *standard* double and triple product integral framework to *affine* double and triple product integrals.

**Affine Transforms of Basis Functions:** Affine transforms of Fourier basis functions are well known [16]. Spherical harmonics can be analytically rotated, as often used for environment maps [102]. However, the standard affine transform usually considered in the spatial domain has no simple analog in the spherical domain—therefore, we do not consider spherical harmonics in this paper. Researchers have approximated the affine transform using a combination of spherical rotations and a spherical scaling operation [118]. This approximation is limited to only mid-range illumination, since the distortion tends to be too severe in the near field.

Wavelets lack translation invariance, and have no simple formula for affine transforms. Beylkin [10] and others have studied the concurrent wavelet decomposition of all integer (not continuous) circulant shifts of a signal. In comparison, our goals are different and more ambitious in that we want to consider a general continuous affine transform (and not just all integer shifts). Nevertheless, we are inspired by the sparsity indicated by [10], and [119] who studied wavelet rotations. We have developed a fast algorithm for wavelet affine transforms, even when no simple analytic formula exists.

**Near Field Relighting and Image Processing:** Relighting techniques have been developed from the basic approach introduced by Nimeroff et al. [79] and Dorsey et al. [29] to much recent work on precomputed radiance transfer (PRT) [102]. In terms of our application, the most closely related works are methods extending PRT to near-field and dynamic settings. Spherical harmonic gradients [2] and scaling [118] try to approximate the effects of mid-range illumination. Spherical harmonic exponentiation [92] can render near-field soft shadowing effects in real time, but has to use sphere sets to approximate geometries so that affine transforms are avoided. As a precursor, Mei et al. [70] proposed decomposing the illumination into directional lights and searching through precomputed spherical radiance transport maps to render dynamic scenes. Zhou et al. [126] then developed precomputed shadow or source radiance fields to support all-frequency effects, but cannot support very high sampling rates nor general changes in the lighting (such as editing the pattern of an area source in lighting design). Kristensen et al. [60] also extend PRT to local lighting using unstructured light clouds. Overall, these methods have to assume pre-determined lights, allowing changes of only light positions or scale intensities. In contrast, our method assumes planar area sources, but allows near-field relighting with dynamic editable lights, and eliminates the need for lighting-dependent precomputations and storage.

Other applications shown at the end of the paper include wavelet importance sampling the product of the area lighting and general BRDFs for Monte Carlo offline rendering systems, which has previously only been applied to distant environment maps [22]. In addition, we also explore image processing. A number of image operations such as additions and multiplications can already be performed directly in the wavelet domain [30, 61]. Our algorithms extend these operations to dilations and translations, which can be cast as affine transforms.

| Notations | Descriptions | Acronyms |
|---|---|---|
| $v, u$ | Integration variables | |
| $s, o$ | Scaling and offsetting variables | |
| $l_i, t_i$ | Level and offset of a wavelet $\psi_i(v)$ | |
| $F(v), G(v)$ | Functions in pixel domain | |
| $H(v), W(v)$ | Functions in pixel domain | |
| $\Phi_i(v)$ | Orthonormal basis function | |
| $\psi_i(v)$ | Wavelet basis function | |
| $C_{ij}$ | Standard Coupling Coefficient | SCC |
| $C_{ijk}$ | Standard Tripling Coefficient | STC |
| $D_{ij}(s, o)$ | Affine Coupling Coefficient | ACC |
| $D_{ijk}(s, o)$ | Affine Tripling Coefficient | ATC |
| $M(r, c)$ | Canonical Coupling Coefficient | CCC |
| $r, c$ | Radius and center of the transformed mother wavelet in the CCC | |

Figure 4.2: *Notations used in Chap. 4.*

## 4.3 Affine Double and Triple Product Integrals

In this section, we introduce affine coupling and tripling coefficients. We use 1D wavelets for simplicity in our analysis (we will see that our results extend directly to higher dimensions in Sec. 4.4.4). Notation used in this paper is shown in Fig. 4.2.

### 4.3.1 Standard Coupling and Tripling Coefficients

Double and triple product integrals can be written respectively as:

$$H(v) = \int F(v)G(v)dv, \tag{4.1}$$

$$H(v) = \int F(v)G(v)W(v)dv, \tag{4.2}$$

where $H$, $F$, $G$ and $W$ are functions in the spatial or angular domain. For example, in relighting applications, $v$ could be the incident angle, $H$ could be the reflected radiance, and $F$, $G$ and $W$ could correspond to the lighting, visibility and BRDF respectively. For compact representation, it is common to expand them in basis functions. The double product integral becomes

$$\begin{aligned} H(v) &= \int \Big(\sum_i F_i \Phi_i(v)\Big)\Big(\sum_j G_j \Phi_j(v)\Big) dv \\ &= \sum_i \sum_j F_i G_j \int \Phi_i(v)\Phi_j(v) dv \\ &= \sum_i \sum_j F_i G_j C_{ij}, && (4.3) \\ C_{ij} &= \int \Phi_i(v)\Phi_j(v) dv, && (4.4) \end{aligned}$$

where $\Phi(v)$ are some set of orthonormal basis functions. We denote $C_{ij}$ as the **Standard Coupling Coefficient**, or SCC.

Similarly, the triple product integral becomes

$$\begin{aligned} H(v) &= \int \Big(\sum_i F_i \Phi_i(v)\Big)\Big(\sum_j G_j \Phi_j(v)\Big)\Big(\sum_k W_k \Phi_k(v)\Big) dv \\ &= \sum_i \sum_j \sum_k F_i G_j W_k \int \Phi_i(v)\Phi_j(v)\Phi_k(v) dv \\ &= \sum_i \sum_j \sum_k F_i G_j W_k C_{ijk}, && (4.5) \\ C_{ijk} &= \int \Phi_i(v)\Phi_j(v)\Phi_k(v) dv. && (4.6) \end{aligned}$$

We denote $C_{ijk}$ as the **Standard Tripling Coefficient**, or STC.

### 4.3.2 Affine Coupling and Tripling Coefficients

The SCC and STC only apply well to functions that are “fixed” and “static”. We now consider functions that are affinely transformed. Without loss of generality, we assume $F(v)$ is scaled and translated to $F(sv + o)$. With respect to the illustration in Fig. 4.1C, the offset $o$ corresponds to the horizontal position $x$, and the scale $s$ to the vertical distance $z$. This leads to an important variation of the standard double product integral, which we call the affine double product integral:

$$\begin{aligned} H(s,o;v) &= \int F(sv+o)G(v)dv \\ &= \int \Big(\sum_i F_i\Phi_i(sv+o)\Big)\Big(\sum_j G_j\Phi_j(v)\Big)dv \\ &= \sum_i\sum_j F_iG_jD_{ij}(s,o), \end{aligned} \tag{4.7}$$

$$\boxed{D_{ij}(s,o) = \int \Phi_i(sv+o)\Phi_j(v)dv.} \tag{4.8}$$

We denote $D_{ij}(s,o)$ as the **Affine Coupling Coefficient**, or ACC. Just as standard triple product integrals are used for multiplication, the same machinery as equation 4.8 is useful for affinely transforming a function. In fact, equation 4.7 is equivalent to an affine transform of $F$ followed by a standard double product integral with $G$.

Similarly, the affine triple product integral can be written as

$$\begin{aligned} H(s,o;v) &= \int F(sv+o)G(v)W(v)dv \\ &= \int \Big(\sum_i F_i\Phi_i(sv+o)\Big)\Big(\sum_j G_j\Phi_j(v)\Big)\Big(\sum_k W_k\Phi_k(v)\Big)dv \\ &= \sum_i\sum_j\sum_k F_iG_jW_kD_{ijk}(s,o), \end{aligned} \tag{4.9}$$

$$\boxed{D_{ijk}(s,o) = \int \Phi_i(sv+o)\Phi_j(v)\Phi_k(v)dv.} \tag{4.10}$$

We denote $D_{ijk}(s,o)$ as the **Affine Tripling Coefficient**, or ATC.

### 4.3.3 Discussion of Properties

**Properties of SCC and STC:** Because of the orthonormal relation between two different basis functions, the SCC reduces to a Kronecker delta function, $C_{ij} = \delta_{ij}$, and has exactly $N$ non-zero terms, where $N$ is the total number of basis functions. The STC is slightly more complicated. For general orthonormal bases, the complexity (i.e., number of nonzero coefficients) of the STC is $O(N^3)$. Sparsity exists for bases with special structures [76], e.g., the complexity is $O(N)$ for pixel bases, $O(N^2)$ for 2D Fourier series, and $O(N \log N)$ for Haar wavelets. The complexities of the SCC and STC are recapped in Sec. 4.4.3. In addition, note that both the SCC and STC are symmetric:

$$C_{ij} = C_{ji}, \qquad C_{ijk} = C_{perm(ijk)}, \tag{4.11}$$

where $perm(ijk)$ is any permutation of the triplet $(i, j, k)$.

**Properties of ACC and ATC:** By contrast, the ACC and ATC both gain two more degrees of freedom, since there are two new arguments: the scale $s$ and the offset $o$. In fact, the SCC and STC are special cases of the ACC and ATC, when the scale is 1 and the offset is 0,

$$C_{ij} = D_{ij}(1, 0), \qquad C_{ijk} = D_{ijk}(1, 0). \tag{4.12}$$

The ACC and ATC do not preserve the same symmetries of the SCC and STC, e.g., the ACC follows this more complex identity:

$$D_{ij}(s, o) = \frac{1}{s} D_{ji}(\frac{1}{s}, -\frac{o}{s}). \tag{4.13}$$

**Relation between ACC and ATC:**

**Lemma 1.** *For any orthonormal basis, the ATC can be represented using the the ACC and STC as*

$$D_{ijk}(s,o) = \sum_l D_{il}(s,o)C_{ljk}. \tag{4.14}$$

The Lemma can be proved by expanding $\Phi(sv+o)$ in terms of $\Phi(v)$ and then using associativity as follows. By definition, the ATC is

$$D_{ijk}(s,o) = \int \Phi_i(sv+o)\Phi_j(v)\Phi_k(v)dv. \tag{4.15}$$

The integrand function $\Phi_i(sv+o)$ can be viewed as a function of $s$ and $o$, and be expanded in basis $\Phi_l(v)$ as

$$\begin{aligned}\Phi_i(sv+o) &= \sum_l \Big( \int \Phi_i(su+o)\Phi_l(u)du \Big) \Phi_l(v) \\ &= \sum_l D_{il}(s,o)\Phi_l(v). \end{aligned} \tag{4.16}$$

Plugging equation 4.16 into 4.15, we obtain:

$$\begin{aligned} D_{ijk}(s,o) &= \int \Big( \sum_l D_{il}(s,o)\Phi_l(v) \Big) \Phi_j(v)\Phi_k(v)dv \\ &= \sum_l D_{il}(s,o) \int \Phi_l(v)\Phi_j(v)\Phi_k(v)dv \\ &= \sum_l D_{il}(s,o)C_{ljk}. \end{aligned} \tag{4.17}$$

Lemma 1 indicates that the computational complexity of the affine tripling coefficient $D_{ijk}(s,o)$ relies on those of the ACC and STC. Lemma 1 also suggests a way of evaluating the ATC using the ACC and STC.

## 4.4 Complexity of Affine Coupling and Tripling Coefficients

In this section, we study the computational complexity of the affine coupling and tripling coefficients (ACC & ATC), determining their numbers of non-zero terms in a number of bases. In wavelets, we focus on the Haar basis for its simplicity. We present the main results that are essential for understanding the key insights and implementing the theory, leaving many detailed mathematical derivations for the appendices.

### 4.4.1 General, Pixel and Fourier Bases

In general orthonormal bases, the complexities of the ACC and ATC are $O(N^2)$ and $O(N^3)$ respectively (can be lower for some values of $s$ and $o$). This can be contrasted to the linear complexity of the SCC, and highlights the fact that the original orthonormal relation between different basis functions no longer holds under an affine transform.

Pixel basis functions, in mathematical terms, are disjoint and discrete step functions. A unique aspect of the pixel basis is that both the integration and product of multiple *different* pixels is zero. This leads directly to the linear $O(N)$ complexity of the ACC and ATC. The pixel basis's complexity can formally analyzed as follows. Firstly, the ATC, according to Lemma 1, reduces to the ACC because in pixel basis $C_{ijk}$ equals 1 when $i = j = k$, and otherwise is 0. As a result, the complexities of the ATC and ACC are the same, and we only need to analyze the ACC $D_{ij}(s, o)$. Calculating the ACC in the pixel basis is essentially equivalent to determining if the two pixels $\Phi_i(sv + o)$ and $\Phi_j(v)$ overlap. Without loss of generality, assume that the original pixel basis is defined in the range $[0, 1]$ and there are $N$ pixel basis functions in total. Each pixel $\Phi_i(sv + o)$ covers an area of $1/(sN)$, and $\Phi_j(v)$ an area of $1/N$. Any given $\Phi_j(v)$, independent of the offset $o$, can overlap with at most $s+1$ basis functions $\Phi_i(sv+o)$ due to the area constraint. Taking all $\Phi_j(v)$ into account, we obtain a maximum of $(s+1)N$ overlapping pairs. Similarly, any given $\Phi_i(v)$ can overlap at most $s^{-1}+1$ basis functions $\Phi_j(v)$. Considering all $\Phi_i(v)$ gives maximally $(s^{-1}+1)N$ overlapping pairs. Combining these two maximums, we derive an upper bound $min\Big((s+1)N, (s^{-1}+1)N\Big)$ of the complexity, which can be proven tight easily. The tight upper bound $min\Big((s+1)N, (s^{-1}+1)N\Big)$ is a function of the scale $s$, and its maximum is $2N$ when $s=1$. So we have proved the complexity of the ACC and ATC in the pixel basis is $O(N)$. Despite its simplicity, the pixel basis is a poor basis for compression, which often outweighs its efficiency in calculation.

The Fourier series is mostly preferred in theoretical analysis and has a well-known 2D affine theorem [16]. As described in Sec. 2.2, the complex form ($I=\sqrt{-1}$) of a 1D Fourier series on an azimuthal domain $[0, 2\pi]$ can be written as:

$$\phi_p(v) \quad = \quad (2\pi)^{-1/2} e^{Ipv}.$$

Based on the 2D affine Fourier theorem [16], we recap the derivation for the 1D Fourier basis[1]:

$$\begin{aligned} D_{pq}(s,o) &= \int_0^{2\pi} \Phi_p(sv+o)\Phi_q^*(v)dv \\ &= e^{Ipo}\int_0^{2\pi} (2\pi)^{-1/2} e^{Ispv}(2\pi)^{-1/2}e^{-Iqv}dv \\ &= e^{Ipo}C_{sp,q} \quad = \quad e^{Ipo}\delta_{sp,q}. \end{aligned} \tag{4.18}$$

So we have shown that the ACC in the Fourier basis can be mapped to the SCC, and thus has the same linear $O(N)$ complexity. The specific mapping depends on the scale $s$ and offset $o$. Therefore same as the SCC, the complexity of the ACC in a 1D Fourier basis is $O(N)$.

[1]Since the Fourier basis is complex, complex conjugates of basis functions are used in ACCs and ATCs.

To relate the ATC to the ACC, we start with Lemma 1:

$$D_{npq}(s,o) = \sum_l D_{nl}(s,o)C_{lpq}.$$

Substituting $D_{nl}$ using equation 4.18 and $C_{lpq} = \sqrt{\frac{1}{2\pi}}\delta_{l+p,q}$ [76], we obtain

$$\begin{aligned} D_{npq}(s,o) &= \sum_l e^{Ino}\delta_{sn,l}(2\pi)^{-1/2}\delta_{l+p,q} \quad = \quad (2\pi)^{-1/2}e^{Ino}\sum_l \delta_{sn,l}\delta_{l+p,q} \\ &= (2\pi)^{-1/2}e^{Ino}\delta_{sn+p,q}, \end{aligned} \tag{4.19}$$

where $\delta_{sn+p,q}$ is non-zero when $sn+p-q=0$. Since one of the subscripts is uniquely determined by the other two, there are only two degrees of freedom as opposed to three. So the complexity of the ATC in the 1D Fourier basis is $O(N^2)$. All our analysis for the 1D Fourier basis can be easily extended to 2D and higher dimensions because high dimensional Fourier bases are just products of multiple 1D bases. So we have shown that the complexity of the ATC in the Fourier basis is $O(N^2)$.

Spherical harmonics are the extension of the Fourier basis to the sphere. They are not considered in this discussion since no standard operation in the spherical domain directly maps to the affine transform.

### 4.4.2 Haar Wavelets

We present our main theoretical contribution in this section, deriving the complexities of affine coupling and tripling coefficients in Haar wavelets. Readers interested in implementation may wish to skip to the summary of complexities in Sec. 4.4.3 on a first reading.

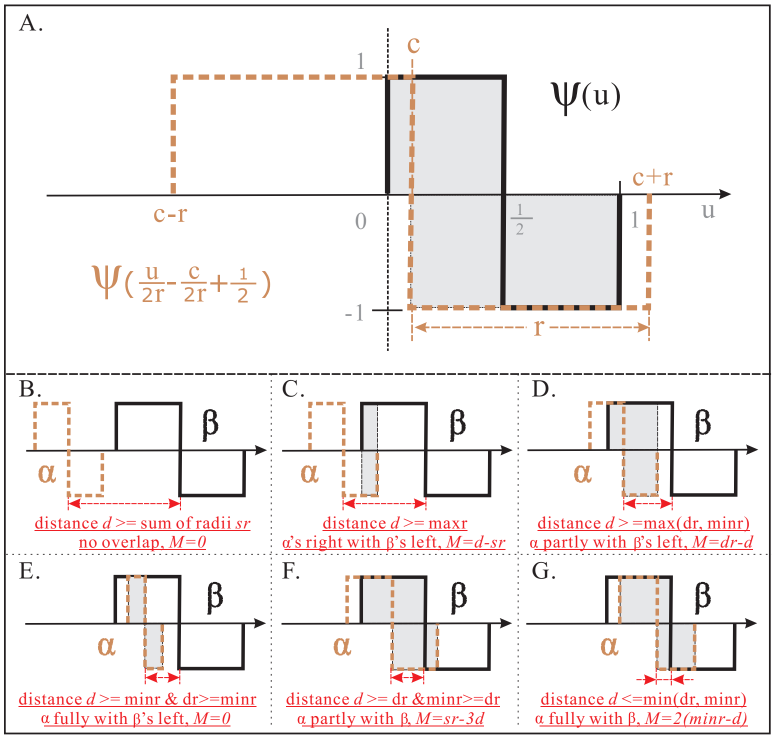

A.
1
c
ψ(u)
c-r
0
1/2
c+r
1
u
ψ(u/2r - c/2r + 1/2)
-1
r
B.
β
α
distance d >= sum of radii sr
no overlap, M=0
C.
β
α
distance d >= maxr
α's right with β's left, M=d-sr
D.
β
α
distance d > =max(dr, minr)
α partly with β's left, M=dr-d
E.
β
α
distance d >= minr & dr>=minr
α fully with β's left, M=0
F.
β
α
distance d >= dr &minr>=dr
α partly with β, M=sr-3d
G.
β
α
distance d <=min(dr, minr)
α fully with β, M=2(minr-d)

Figure 4.3: *The canonical coupling coefficient $M(r, c)$ is an integration of two wavelets denoted as $\alpha$ and $\beta$, with $\alpha$ being affinely transformed, as shown in A. Variables r and c correspond to the radius and center of the transformed wavelet $\alpha$. B-G show different overlapping relations of wavelets $\alpha$ and $\beta$. Variables maxr and minr are respectively the bigger and smaller of the radii of the two wavelets, and sr and dr their sum and difference.*

### Wavelet and Scaling Functions

The normalized 1D Haar basis [107] is defined as

- The mother scaling and wavelet functions are

$$\phi(v) = \begin{cases} 1, & \text{for } 0 \le v < 1 \\ 0, & \text{otherwise} \end{cases},$$

and

$$\psi(v) = \begin{cases} 1, & \text{for } 0 \le v < 1/2 \\ -1, & \text{for } 1/2 \le v < 1 \\ 0, & \text{otherwise} \end{cases}.$$

- A normalized wavelet basis $\psi_j(v)$ at level $l_j$ and offset $t_j$ is

$$\psi_j(v) = 2^{l_j/2}\psi(2^{l_j}v - t_j),$$

which is a scaled and dilated copy of the mother wavelet.

**Canonical Coupling Coefficient (CCC)**

The ACC is a 4D function of the two subscripts $i$ and $j$, the scale $s$, and the offset $o$. Similarly, the ATC is a 5D function. Since according to Lemma 1 the ATC can be reduced to the ACC, we first focus our analysis on the ACC. To reduce the dimensionality of the ACC, we invoke the standard property of Haar wavelets as a multi-resolution series and write wavelet basis functions in terms of the mother wavelet $\psi(v)$.

$$\begin{aligned} D_{ij}(s,o) &= \int \psi_i(sv+o)\psi_j(v)dv \\ &= 2^{\frac{l_i+l_j}{2}} \int \psi(2^{l_i}sv - t_i + 2^{l_i}o)\psi(2^{l_j}v - t_j)dv. \end{aligned} \tag{4.20}$$

To simplify equation 4.20, we make the substitution $u = 2^{l_j}v - t_j$ so that transformations of the two mother wavelets can be merged into a single combined transformation.

$$\begin{aligned} D_{ij}(s,o) &= 2^{\frac{l_i-l_j}{2}} \int \psi(2^{l_i-l_j}su - t_i + 2^{l_i}o + st_j2^{l_i-l_j})\psi(u)du \\ &= 2^{\frac{l_i-l_j}{2}} \int \psi(\frac{u}{2r} - \frac{c}{2r} + \frac{1}{2})\psi(u)du \\ \boxed{D_{ij}(s,o) = 2^{\frac{l_i-l_j}{2}}M(r,c)}\ , \end{aligned} \tag{4.21}$$

```
    //$c$ and $r$ are respectively the center and radius of the transformed wavelet.
    //$d$ is the distance between the centers of the two wavelets.
    //$sr$ and $dr$ are respectively the sum and difference between the two radii.
1.  $d = |c - 0.5|$;                  $sr = r + 0.5$;          $dr = |r - 0.5|$;
2.  $maxr = max(r, 0.5)$;             $minr = min(r, 0.5)$;

3.  if ($d >= sr$)                    $M = 0$;                 // Fig. 4.3B
4.  else if ($d >= maxr$)             $M = d - sr$;            // Fig. 4.3C
5.  else if ($d >= max(dr, minr)$)    $M = dr - d$;            // Fig. 4.3D
6.  else if ($d >= min(dr, minr)$)
7.      if ($dr >= minr$)             $M = 0$;                 // Fig. 4.3E
8.      else                          $M = sr - 3d$;           // Fig. 4.3F
9.  else if ($d <= min(dr, minr)$)    $M = 2(minr - d)$;       // Fig. 4.3G
```

Figure 4.4: *Analytic formula of $M(r, c)$ in pseudocode. Branches correspond to different overlappings between the two wavelets, as illustrated in Fig. 4.3.*

where

$$\boxed{M(r, c) = \int \psi\left(\frac{u}{2r} - \frac{c}{2r} + \frac{1}{2}\right)\psi(u)du}, \tag{4.22}$$

$$r = \frac{2^{l_j - l_i - 1}}{s}, \quad \text{and} \quad c = \frac{2^{l_j - l_i}t_i - 2^{l_j}o + 2^{l_j - l_i - 1}}{s} - t_j. \tag{4.23}$$

We call $M(r, c)$ the **Canonical Coupling Coefficient**, or CCC. The CCC encapsulates the core structure of the ACC. Variables $r$ and $c$ dictate the combined transformation in wavelet $\psi(\frac{u}{2r} - \frac{c}{2r} + \frac{1}{2})$. Intuitively, they are respectively the *radius* and *center* of the transformed wavelet $\psi(\frac{u}{2r} - \frac{c}{2r} + \frac{1}{2})$, as shown in Fig. 4.3A.

**Property 1.** *The evaluation of the 4D affine coupling coefficient reduces to a combination of a few simple analytic function computations and an estimate of the 2D analytic function $M(r, c)$ called the canonical coupling coefficient, as described by equation 4.21. Variables $r$ and $c$ are given by equation 4.23. $M(r, c)$ is a sparse, piece-wise linear, and symmetric function.*

In its analytic form, $M(r, c)$ is a branching function as shown in Fig. 4.4. The calculation of $M(r, c)$ is equivalent to determining the overlapping relation between the original mother wavelet $\psi(u)$ (denoted as $\beta$) and the transformed one $\psi(\frac{u}{2r} - \frac{c}{2r} + \frac{1}{2})$ (denoted as $\alpha$). Their overlapping relation depends on their *relative* positions

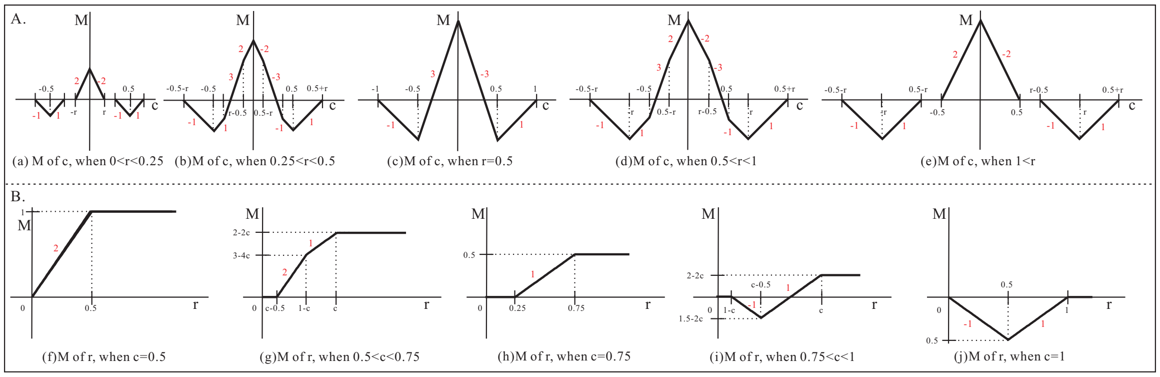


Figure 4.5: ***A:*** *Canonical Coupling Coefficient $M(r, c)$ in the c dimension (the horizontal axis is $c - 0.5$ to better demonstrate the symmetry) for all different ranges of the radius r. The red labels represent line slopes, while the black labels are measurement marks along the axes.* ***B:*** *$M(r, c)$ in the r dimension for $c \geq 0.5$. The horizontal axis is the radius r of the transformed wavelet. When $c \leq 0.5$, $M(r, c)$ can be computed using the symmetry $M(r, c - 0.5) = M(r, 0.5 - c)$. This graph illustrates a number of important properties of M, such as its sparsity, piece-wise linearity, symmetries and boundedness.*

(centers) and sizes (radii). We group their overlapping relations into six cases and show them in Fig. 4.3. For brevity in exposition, we assume that $\alpha$'s radius is smaller than that of $\beta$, and $\alpha$ is located on the left. Our analysis will still hold when $\alpha$'s radius is actually larger or it is located on the right, since $\beta$ can then be viewed as "the transformed wavelet" and exchange roles with $\alpha$. In all cases, $M(r, c)$ can be computed in no more than 9 lines of code (Fig. 4.4 also cross references the six cases in Fig. 4.3).

## Properties of CCC

$M(r, c)$ has special structures and important properties that can be exploited for computation. In Fig. 4.5, we plot $M(r, c)$ in both $r$ and $c$ dimensions to better expose many such properties. We examine a few important ones here:

**Sparsity:** $M(r, c)$ is sparse. As shown in Fig. 4.5, $M(r, c)$ is non-zero over only very limited ranges of the radius $r$ and the center $c$. In particular, $M(r, c)$ is zero when the right end-point of the transformed wavelet is less than 0 (meaning $c + r < 0$), or the left end-point is greater than 1 (meaning $c - r > 1$). In these cases, the two wavelets do not overlap and their integration is 0. These two boundary conditions compactly combine to $|c - 0.5| > r + 0.5$.

$$M(r, c) = 0, \quad \text{when } |c - 0.5| > r + 0.5. \tag{4.24}$$

Since the ACC reduces to the CCC, the ACC will be sparse if many combinations of $i, j, s$ and $o$ make $r$ and $c$ fall into the zero ranges. We will discuss the complexity of the ACC formally in Sec. 4.4.2.

**Piece-Wise Linearity:** $M(r, c)$ is a piece-wise linear function. As observed in Fig. 4.5, $M(r, c)$ has only a limited set of slopes $\{0, \pm 1, \pm 2 \pm 3\}$ in both $r$ and $c$ dimensions. This is because the partial derivatives of $M(r, c)$ with respect to $r$ and $c$ are just combinations of a few mother wavelets $\psi(u)$. These mother wavelets are valued respectively at one of the three break-points ($c - r$, $c + r$ and $c$, as in Fig. 4.3A) of the transformed wavelet $\psi(\frac{u}{2r} - \frac{c}{2r} + \frac{1}{2})$.

$$\frac{\partial M(r,c)}{\partial c} = 2\psi(c) - \psi(c - r) - \psi(c + r) = \{0, \pm 1, \pm 2, \pm 3\}, \tag{4.25}$$

$$\frac{\partial M(r,c)}{\partial r} = \psi(c - r) - \psi(c + r) = \{0, \pm 1, \pm 2\}. \tag{4.26}$$

The piece-wise linearity of M can be formally proved by computing its partial derivatives. We compute the partial derivatives of $M(r, c)$ with respect to $c$ and $r$ as follows[2]:

$$\begin{aligned}
\frac{\partial M(r,c)}{\partial c} &= \frac{\partial}{\partial c}\int \psi(\frac{u}{2r}-\frac{c}{2r}+\frac{1}{2})\psi(u)du \\
&= \frac{-1}{2r}\int \Big(\delta(\frac{u}{2r},\frac{c-r}{2r})+\delta(\frac{u}{2r},\frac{c+r}{2r})-2\delta(\frac{u}{2r},\frac{c}{2r})\Big)\psi(u)du \\
&= -\int \Big(\delta(u,c-r)+\delta(u,c+r)-2\delta(u,c)\Big)\psi(u)du \\
&= 2\psi(c)-\psi(c-r)-\psi(c+r) \\
\frac{\partial M(r,c)}{\partial r} &= \frac{\partial}{\partial r}\int \psi(\frac{u}{2r}-\frac{c}{2r}+\frac{1}{2})\psi(u)du \\
&= \frac{-1}{2r^2}\int \Big(\delta(\frac{u}{2r},\frac{c-r}{2r})+\delta(\frac{u}{2r},\frac{c+r}{2r})-2\delta(\frac{u}{2r},\frac{c}{2r})\Big)(u-c)\psi(u)du \\
&= \frac{-1}{r}\int \Big(\delta(u,c-r)+\delta(u,c+r)-2\delta(u,c)\Big)(u-c)\psi(u)du \\
&= \frac{-1}{r}\Big(-r\psi(c-r)+r\psi(c+r)\Big) \\
&= \psi(c-r)-\psi(c+r)
\end{aligned}$$

Consequently, the ACC is also piece-wise linear, and its gradient is easily computed from that of $M(r,c)$ using the chain rule.

**Symmetry:** $M(r,c)$ is symmetric. In the $c$ dimension, as shown in Fig. 4.5A, $M(r,c)$ is reflection-symmetric about 0.5. This reflection symmetry is also reflected in computation in Fig. 4.4 as the calculation depends not on $c$ directly, but on $d = |c-0.5|$. $d = |c-0.5|$ is reflection-symmetric about 0.5. In mathematical terms, we have

$$M(r, 0.5+c) = M(r, 0.5-c), \tag{4.27}$$

In the $r$ dimension, $M(r,c)$ also has a certain degree of symmetry. When $r$ is above 1, we can invert its value to below 1 by changing the integration variable in equation 4.21 to $w = \frac{u}{2r} - \frac{c}{2r} + \frac{1}{2}$.

[2]Kronecker delta is denoted as $\delta(x,y)$ here which equals 0 when $x = y$.

$$M(r,c) = 2rM(\frac{1}{4r}, \frac{1}{2} + \frac{1}{4r} - \frac{c}{2r}). \tag{4.28}$$

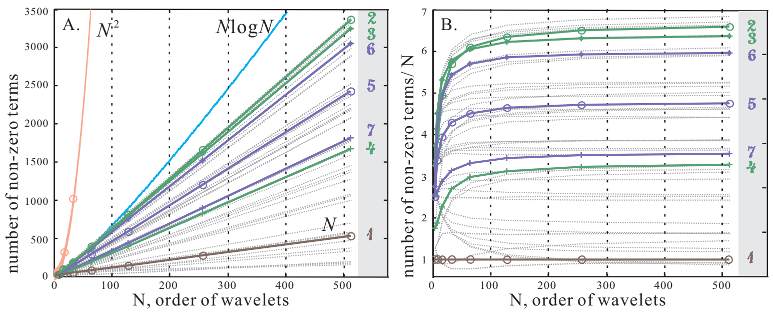
The symmetry properties of M can be formally proved in mathematics. We derive the symmetry properties of $M(r,c)$ in the $c$ and $r$ dimensions as follows:

$$\begin{aligned}
M(r,c+0.5) &= \int_{-\infty}^{\infty} \psi(\frac{u-c-0.5}{2r} + \frac{1}{2})\psi(u)du \\
&= -\int_{-\infty}^{\infty} \psi(\frac{u-c-0.5}{2r} + \frac{1}{2})\psi(1-u)du \qquad //\psi(u) = -\psi(1-u) \\
&= \int_{\infty}^{-\infty} \psi(\frac{-w-c+0.5}{2r} + \frac{1}{2})\psi(w)dw \qquad //\text{substitute } w = 1-u \\
&= \int_{-\infty}^{\infty} \psi(\frac{w+c-0.5}{2r} + \frac{1}{2})\psi(w)dw \quad //\psi(0.5+u) = -\psi(0.5-u) \\
&= M(r, 0.5-c) \\
M(r,c) &= \int \psi(\frac{u-c}{2r} + \frac{1}{2})\psi(u)du \\
&= 2r\int \psi(w)\psi\Big(2r(w-0.5)+c\Big)dw \qquad //\text{substitute } w = \frac{u-c}{2r} + \frac{1}{2} \\
&= 2rM(\frac{1}{2r}, \frac{1}{2} + \frac{1}{4r} - \frac{c}{2r})
\end{aligned}$$

Due to these symmetries, we only store a non-repeating quarter ($[0 < r \le 1, -r \le c \le 0.5]$) of the non-zero range of $M(r,c)$ and save three fourths of the storage space.

**Boundedness:** Finally, $M(r,c)$ is both upper and lower bounded, which makes it ideal for quantization and encoding in hardware textures.

$$\begin{aligned}
max\Big(M(r,c)\Big) &= 2min(r, 0.5) &\le 1, \\
min\Big(M(r,c)\Big) &= -min(r, 0.5) &\ge -0.5.
\end{aligned}$$



| C. | | | | | | | |
|---|---|---|---|---|---|---|---|
| $N$ | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
| **$N$log$N$** | **24** | **64** | **160** | **384** | **896** | **2048** | **4608** |
| $N^2$ | 64 | 256 | 1024 | 4096 | 16384 | 65536 | 262144 |
| s, o | | | | | | | |
| 1 1.0, 0.00 | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
| 2 0.9, 0.00 | 30 | 79 | 182 | 390 | 812 | 1665 | 3376 |
| 3 0.9, 0.23 | 36 | 85 | 185 | 387 | 795 | 1613 | 3254 |
| 4 0.9, 0.55 | 15 | 36 | 86 | 190 | 399 | 822 | 1676 |
| 5 0.6, 0.00 | 27 | 63 | 137 | 288 | 593 | 1205 | 2431 |
| 6 0.6, 0.23 | 33 | 79 | 174 | 364 | 748 | 1516 | 3053 |
| 7 0.6, 0.55 | 21 | 46 | 100 | 212 | 439 | 896 | 1814 |

Figure 4.6: *Complexity of affine Haar wavelet coupling coefficients for 50 randomly generated sets of scales and offsets. Figure 4.6A empirically shows the $O(N \log N)$ complexity of the ACC. 7 representative curves out of 50 are highlighted and their values listed in table C. Note that in Fig 4.6A for most scales and offsets, the actual numbers of non-zero ACC terms fall well below $N \log N$ (cyan line), and some even below $N$ (when significant offsets and scales transform a large portion of the wavelet tree out of $[0, 1]$). The formal proof of the complexity of the ACC is in Appendix D.*

### Complexities of Haar ACC and ATC

The number of non-zero ACC terms varies significantly with the scale and offset. We first show empirically in Fig. 4.6 the $O(N \log N)$ complexity of the ACC. In Fig. 4.6A, we plot the numbers of non-zero ACC terms versus the total numbers of the wavelets for 50 randomly generated sets of scales and offsets. 7 representative curves out of the 50 are highlighted, and their numerical values are tabulated in Fig. 4.6C. To better illustrate the logarithmic behavior part of the $O(N \log N)$ complexity, we divide all curves by $N$ and plot them again in Fig. 4.6B. Note that $O(N \log N)$ is just an upper bound. In practice, for most scales and offsets, the complexity of the ACC is well below $N \log N$ as shown in Fig. 4.6. The actual complexity of the ACC is comparable to that of standard tripling coefficients (STCs) developed by [76]. The $O(N \log N)$ complexity can also be mathematically proved by determining the upper-bound of the total overlapping pairs between the two wavelet trees, one of which is affinely transformed. In practice, indices of non-zero ACC terms can be picked either using equation 4.24 (the sparsity property of the CCC), or from a compact pre-tabulated table, as implemented in Sec. 4.5.5.

Now we give a formal proof of the complexity of Haar ACC. In the Haar wavelet basis, the ACC is non-zero only when wavelets $\Phi_i(sv + o)$ and $\Phi_j(v)$ overlap with each other. Counting the number of non-zero ACC terms reduces to determining the number of overlapping wavelet pairs. Since wavelets are best organized in a tree structure, we denote the two wavelet trees using **I** and **J**. The basic structure of a wavelet tree is that two wavelets in the same tree will overlap only if they are in the same tree branch. We start by fixing a level and picking a wavelet for each wavelet tree: level $l_i$ and wavelet $\Phi_i(sv + o)$ for tree **I**, and level $l_j$ and wavelet $\Phi_j(v)$ for tree **J**. Since respectively at levels $l_i$ and $l_j$ there are $2^{l_i}$ and $2^{l_j}$ wavelets, $\Phi_i(sv + o)$ and $\Phi_j(v)$ each must subtend $1/(s2^{l_i})$ and $1/(2^{l_j})$ in area. Any given wavelet $\Phi_j(v)$ at level $l_j$ in tree **J** can overlap with at most $s \cdot 2^{l_i - l_j} + 1$ wavelets $\Phi_i(sv + o)$ at level $l_i$ in tree **I** due to the area constraint. Summing over all levels in tree **I**, we get at most $\sum_{l_i}(s2^{l_i - l_j} + 1)$ overlapping pairs between the wavelet $\Phi_j(v)$ and wavelet tree **I**. Since the formula just derived is valid for any wavelet in tree **J**, we now sum over all wavelets $\Phi_j(v)$ in wavelet tree $J$ and get a maximum of $\sum_{l_j} 2^{l_i} \sum_{l_i}(s2^{l_i - l_j} + 1)$ overlapping pairs between trees **I** and **J**. Similarly, we can start from a given wavelet $\Phi_i(sv + o)$ in tree **I** and obtain a maximum of $\sum_{l_i} 2^{l_i} \sum_{l_j}(2^{l_j - l_i}/s + 1)$ overlapping pairs between trees **I** and **J**. Combining these two maximums, we obtain an upper bound of $min(\sum_{l_i,l_j}(s2^{l_i} + 1), \sum_{l_i,l_j}(2^{l_j}/s + 1))$, which is tight for most $s$ and $o$. The upper bound's maximum is $N \log N + \log^2 N$. So the complexity of the ACC in the Haar wavelet basis is $O(N \log N)$. Because of compression, only a small subset ($n$ out of $N$) of wavelet coefficients suffice to generate accurate results, and thus the complexity reduces to $O(n \log N)$.

To compute the complexity of Haar ATC, we invoke the standard property of

Haar wavelets that the product of two wavelets is either the finer wavelet up to a scale if they overlap, or zero otherwise. Therefore when wavelets overlap, the ATC reduces to the ACC, and subsequently to the CCC according to Property 1. When wavelets do not overlap, the ATC is simply zero. Based on this key observation, we derive Property 2 that relates the ATC to the CCC.

**Property 2.** *The Haar ATC, as defined in equation 4.10, is evaluated using the canonical coupling coefficient as follows:*

- *In equation 4.10, if basis functions $\Phi_j(v)$ and $\Phi_k(v)$ overlap,*

$$D_{ijk}(s,o) = 2^{\frac{l_i - l_m + l_n}{2}} M(r,c), \tag{4.29}$$

  *where $n = min(j,k)$, $m = max(j,k)$, and*

$$\begin{aligned} c &= \frac{2^{l_m - l_i} t_i - 2^{l_m} o + 2^{l_m - l_i - 1}}{s} - t_m, \\ r &= \frac{2^{l_m - l_i - 1}}{s}. \end{aligned}$$

- *Otherwise, $D_{ijk}(s,o) = 0$.*

As discussed, the ATC in the Haar wavelet basis reduces to an ACC when the non-zero segments of $\Phi_j(v)$ and $\Phi_k(v)$ overlap, or is 0 otherwise. Since there are at most $O(\log N)$ overlapping pairs of wavelets $\Phi_j(v)$ and $\Phi_k(v)$, the complexity of the ATC in the Haar basis is $O(n \log^2 N)$. For interested readers, Property 2 can also be verified against Lemma 1.

### 4.4.3 Summary of Complexities in All Bases

The following table summarizes the computational complexities of the SCC, STC, ACC and ATC in different bases. For general orthonormal bases, there is no sparsity due to the lack of special structures. The pixel basis, despite its simplicity, is a poor basis for compression and hence undesirable for practical applications. The Fourier basis is widely used in theoretical analysis, but is not good at capturing all-frequency lighting (or visibility/BRDF) as shown in [75]. Haar wavelets are preferred in all-frequency relighting and only need a handful of coefficients and basis functions to achieve good approximations. To distinguish from $N$, the total number of basis functions, we denote the number of terms retained after compression as $n$. $n$ is usually around $1 \sim 2\%$ of $N$ for wavelets. As shown in the following table, Haar wavelets have linear or close-to-linear complexities ranging from $n$ to $O(n \log^2 N)$ across all columns. In practice, we would need to compute far fewer terms after compression than in the other bases. This makes Haar wavelets ideal for many operations in practical applications.

| *Bases* | *SCC* | *ACC* | *STC* | *ATC* |
|---|---|---|---|---|
| General | N | $O(N^2)$ | $O(N^3)$ | $O(N^3)$ |
| Pixel | N | N | N | N |
| Fourier | N | N | $O(N^2)$ | $O(N^2)$ |
| Haar | n | O(nlogN) | O(nlogN) | $O(n\log^2 N)$ |

### 4.4.4 Generalization to Higher Dimensions

A high dimensional Haar basis can be viewed as a product of multiple 1D Haar wavelet basis functions for both standard and non-standard decompositions, as utilized by [22]. Similarly to a high dimensional Fourier basis, a high dimensional Haar basis can be written as

$$\Psi_{\mathbf{i}}(\mathbf{v}) = \prod_{q=1}^{Q} \psi_{i_q}(v_q), \tag{4.30}$$

where $\Psi_{\mathbf{i}}$ is a Q-dimensional basis function, variables $\mathbf{i}$ and $\mathbf{v}$ are Q-dimensional vectors, and $i_q$ and $v_q$ index into the $q$-th dimension of vectors $\mathbf{i}$ and $\mathbf{v}$.

The ACC becomes

$$\begin{aligned}
\mathbf{D_{ij}}(\mathbf{s},\mathbf{o}) &= \underbrace{\int\cdots\int}_{Q} \Psi_{\mathbf{i}}(\mathbf{sv}+\mathbf{o})\Psi_{\mathbf{j}}(\mathbf{v})d\mathbf{v} \\
&= \prod_{q=1}^{Q} \underbrace{\left(\int \psi_{i_q}(s_q v_q + o_q)\psi_{j_q}(v_q)dv_q\right)}_{\text{1D affine coupling coefficient}} \\
&= \prod_{q=1}^{Q} D_{i_q j_q}. \qquad (4.31)
\end{aligned}$$

Similarly, the ATC becomes

$$\begin{aligned}
\mathbf{D_{ijk}}(\mathbf{s},\mathbf{o}) &= \underbrace{\int\cdots\int}_{Q} \Psi_{\mathbf{i}}(\mathbf{sv}+\mathbf{o})\Psi_{\mathbf{j}}(\mathbf{v})\Psi_{\mathbf{k}}(\mathbf{v})d\mathbf{v} \\
&= \prod_{q=1}^{Q} \underbrace{\left(\int \psi_{i_q}(s_q v_q + o_q)\psi_{j_q}(v_q)\psi_{k_q}(v_q)dv_q\right)}_{\text{1D affine tripling coefficient}} \\
&= \prod_{q=1}^{Q} D_{i_q j_q k_q}, \qquad (4.32)
\end{aligned}$$

where $\Psi_{\mathbf{i}}$, $\Psi_{\mathbf{j}}$ and $\Psi_{\mathbf{k}}$ are Q-dimensional basis functions, and variables **i**, **j**, **k**, **s**, and **o** are vectors of Q elements. Equations 4.31 and 4.32 show that the high dimensional Haar ACC and ATC are products of multiple 1D ACCs and ATCs respectively. They enable scales and translations in wavelets, but not rotations or shears.

The complexities of the high dimensional ACC and ATC are respectively $O(n[\frac{\log N}{Q}]^Q)$ and $O(n[\frac{\log N}{Q}]^{2Q})$. If we denote the total number of basis functions in 1D as $\tilde{N}$, we obtain $N = \tilde{N}^Q$ for higher dimensions. Noting that $\log N = Q \log \tilde{N}$ and the complexities of 1D ACCs and ATCs are $O(\tilde{n} \log \tilde{N})$ and $O(\tilde{n} \log^2 \tilde{N})$, multiplying these complexities Q times respectively generates the complexities of the high dimensional ACC and ATC. For relighting, we will be working with 2D ACCs, whose complexity is $O(n \log^2 N)$ from the above analysis. However, we show in Sec. 4.5.5 that we can develop a more efficient algorithm with $O(n \log N)$ complexity.

## 4.5 Interactive Near-Field Relighting

We now develop our main practical application, showing how to integrate our theory with the PRT framework to render near-field lighting effects at interactive rates. Later Sec. 4.6 will present initial results for wavelet importance sampling for near-field planar area lights and image processing directly in the wavelet domain.

### 4.5.1 Basic Relighting Framework

In the reflection equation, the exitant radiance is

$$\begin{aligned} B(\mathbf{q}, \boldsymbol{\omega}_o) &= \int_\Omega L(\mathbf{q}, \boldsymbol{\omega}_i) V(\mathbf{q}, \boldsymbol{\omega}_i) \rho(\mathbf{q}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)(\boldsymbol{\omega}_i \cdot \mathbf{n}) d\boldsymbol{\omega}_i \\ &= \int_\Omega L(\mathbf{q}, \boldsymbol{\omega}_i) T(\mathbf{q}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) d\boldsymbol{\omega}_i, \end{aligned} \tag{4.33}$$

where $B$ is the reflected radiance as a function of the spatial location $\mathbf{q}$ and outgoing direction $\boldsymbol{\omega}_o$, $L$ is the incoming lighting, $\boldsymbol{\omega}_i$ is the incident direction, $V$ is the visibility, $\rho$ is the BRDF, and $\mathbf{n}$ is the surface normal. Symbols in bold represent 2D vectors. Often the visibility $V$, BRDF $\rho$ and cosine term $(\boldsymbol{\omega}_i \cdot \mathbf{n})$ are combined into the transport function $T$ as shown in Equation 4.33. Equation 4.33 is a double product integral of the lighting and the transport function, and is often expanded in the Haar wavelet basis in actual computations.

In most relighting algorithms, distant illumination is assumed so that $L(\mathbf{q}, \boldsymbol{\omega}_i) = L(\boldsymbol{\omega}_i)$ is the same for all vertices $\mathbf{q}$. However, in the near-field setting considered here, we need to propagate light from the planar area source to each spatial location.

We will show below that this corresponds to an affine transform of the original area source radiance. Thus, equation 4.33 becomes an affine double product integral, and can be efficiently computed on the fly for each vertex using the theory of Secs. 4.3 and 4.4.

### 4.5.2 Light Propagation

We consider light propagation from an area light source in 1D as shown in Fig. 4.1C. Extension to 2D planar sources in Sec. 4.5.5 is straightforward as explained in Sec. 4.4.4. The area light is $F(v)$, and the incident radiance at a vertex $L(z, x; v)$. Variables $z$ and $x$ are the vertical and horizontal coordinates of the vertex. $v$ is the intercept on a virtual plane a unit distance away. From simple trigonometry, the incident radiance can be written as

$$\boxed{L(z, x; v) = F(zv + x),} \tag{4.34}$$

which is an affine transformation of the original light $F(v)$. $z$ and $x$ respectively correspond to $s$ and $o$ previously used in the ACC.

It is worth making a note of the parameterization. In terms of the more familiar angular coordinates, $v = \tan\theta$, and we must include the correct angular/area measure for $dv/d\theta$ when changing the integration variable from $\theta$ to $v$. As is conventional, we incorporate this into the transport function $T(v)$. We emphasize that while our parameterization is similar to the linearization used in, for instance [31], equation 4.34 is exact, and not an approximation.

The coefficients of the incident radiance $L_k(z,x)$ can be computed as

$$\begin{aligned} L_k(z,x) &= \int F(zv+x)\psi_k(v)dv \\ &= \int \Big(\sum_i F_i\psi_i(zv+x)\Big)\psi_k(v)dv \\ &= \sum_i F_i D_{ik}(z,x), \end{aligned} \tag{4.35}$$

where $F_i$ are the wavelet coefficients of $F$. Equation 4.35 propagates light directly in the wavelet domain.

We emphasize that equations 4.34 and 4.35 simply express the incident radiance at a given spatial location. The PRT algorithm can be treated as a black box, that takes this incident lighting $L$ and applies the light transport function $T$. Therefore, our method can be incorporated into almost any PRT framework and representation for $T$, including those that are view-dependent.

By substituting the angle $\boldsymbol{\omega}_i$ with $v$ using appropriate normalizing weights (numerical cubature), we can write equation 4.33 as

$$B = \int L(z,x;v)T(v)dv = \sum_i \sum_j F_i T_j D_{ij}(z,x), \tag{4.36}$$

where $T_j$ is the wavelet coefficient of the transport function $T(v)$.

### 4.5.3 Relighting with Light Field

A general light field can be denoted as $F(v_1, v_2)$ and the local incident radiance $L(z,x;v)$ equals to $F(x+zv, v)$. Similarly, this relation can be written in wavelets using the ATC as

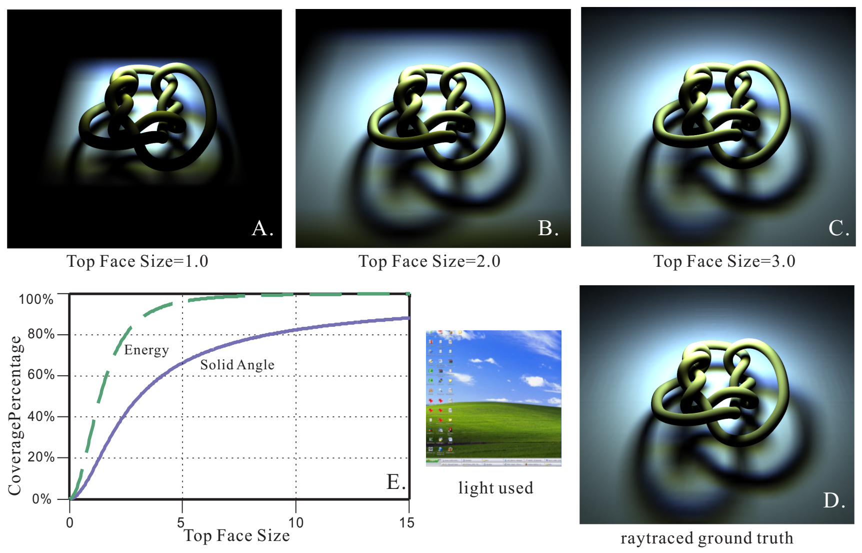


Figure 4.7: *A, B, and C are renderings using our method with different face sizes. D is the ground truth from a raytracer, using a full representation of the lighting environment at each vertex. E shows the coverage ratios of the top face over the upper hemisphere with different sizes. In the graph, the blue solid curve is the top face solid angle covered, and the green dotted curve the energy of a centrally aligned Lambertian lobe. For a standard cube map, its top face size is 2. Top face sizes 3 or 4 can generate visually accurate results.*

$$\begin{aligned}
L_k(z,x) &= \int L(z,x;v)\psi_k(v)dv \\
&= \int F(zv+x,v)\psi_k(v)dv \\
&= \int \Big(\sum_i \sum_j F_{ij}\psi_i(zv+x)\psi_j(v)\Big)\psi_k(v)dv \\
&= \sum_i \sum_j F_{ij}D_{ijk}(z,x). \qquad (4.37)
\end{aligned}$$

### 4.5.4 Parameterization

It is common to parameterize the sphere $S^2$ of directions using a cube map consisting of six faces. Propagations of light to the six faces would require six affine transformations. To speed up the computation, we make a trade off by using only the top face[3], but expand its size to cover a larger solid angle. The top face is aligned with the light plane. Our simplification is motivated by the observation

[3]Similar plane-angle and plane-plane parameterizations have been used in light field representations [64, 19].

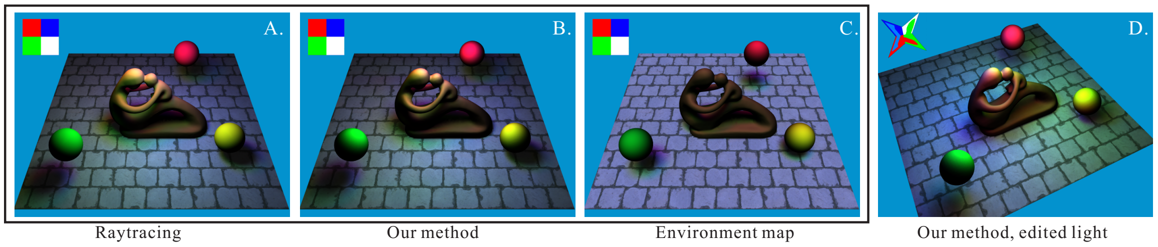


Figure 4.8: *A diffuse scene of a fertility sculpture and three spheres lit by a simple textured area light. The light textures are shown in the top left corners. The area sources are above the fertility sculpture, but cropped to save space.* ***A, B and C:*** *Images A, B and C compare raytracing, our method, and distant environment map lighting respectively. Compared to distant lighting in C, we see that our method in B correctly captures the spatially varying shading on the floor and sculpture and generates a result that is quite close to the ground truth. About* 1% *of source level and* 22% *of target level lighting coefficients are used in our method.* ***D:*** *Finally, we can edit the light texture and shape, and rotate it in real-time to obtain a distinct appearance in D. This important tool for lighting design would not be possible with previous techniques like precomputed shadow and source radiance fields.*

that in near-field settings for most vertices, at least one of the lighting, visibility or BRDF terms would tend to peak at the top face and die out towards peripheral regions. In addition, the cosine term (cosine of the incident angle) reduces contributions towards grazing angles.

We make the top face adjustable so that a large or smaller solid angle can be covered as needed. The top face in the standard cube map is assumed size 2. Figure 4.7 shows that mid-sized faces can cover a sufficient portion of the hemisphere. For example, a size 10 top face covers 82% solid angle of a hemisphere and captures 99% energy of a centrally aligned Lambertian lobe. The Lambertian lobe corresponds just to the cosine term. In our experiments, sizes 3 or 4 suffice to generate visually accurate results.

### Assumptions and Limitations

We have assumed planar area light sources, where the angular distribution of light is uniform. We have not implemented, but showed theoretically in Sec. 4.5.3 how to handle lights with angular variations such as light fields using ATCs. As noted above, we use an expanded top-face parameterization that may omit light incident at grazing angles. Our method is general enough to allow interactive scaling, translation and horizontal rotation of lights and general edits to the light textures. However, as in most wavelet-based relighting algorithms, we cannot support general out of plane rotations of the lighting.

### 4.5.5 Rendering Algorithm

We present key computation steps and major rendering results in this section. All renderings and time measurements are done on a commodity 3.0 GHz Dual-Quad-Core PC with 4GB memory.

#### Log-Linear-Time Light Propagation

To propagate light, we compute equation 4.35 in 2D,

$$\mathbf{L_k}(\mathbf{z}, \mathbf{x}) = \sum_{\mathbf{i}} \mathbf{F_i} \mathbf{D_{ik}}(\mathbf{z}, \mathbf{x}),$$

where $\mathbf{x}$ and $\mathbf{z}$ are 2D variables[4], and $\mathbf{i}$ and $\mathbf{k}$ are respectively 2D vectors of $(i_1, i_2)$ and $(k_1, k_2)$. Since we have to loop through all subscripts $\mathbf{i}$ and $\mathbf{k}$, we only evaluate the equation for non-zero 2D ACC $\mathbf{D_{ik}}(\mathbf{z}, \mathbf{x})$. Variables $\mathbf{z}$ and $\mathbf{x}$ respectively correspond to the scale $s$ and the translation $o$. Recall that we use $N$ to denote the total number of basis functions and $n$ for the number of terms retained after compression. The total cost appears to be the $O(n \log^2 N)$ complexity of the 2D ACCs, which is derived in Sec. 4.4.4.

In fact, a better $O(n \log N)$ algorithm exists if we separate 2D ACCs as products

[4]Since each vertex has only one depth from the light source, vector $\mathbf{z}$'s two elements are the same $z_1 = z_2$.

of 1D ACCs and perform the computations in each dimension in succession.

$$\begin{aligned}
\text{Step 1:} \quad \forall\ i_2, k_1 \quad & Z_{i_2 k_1}(z_1, x_1) = \sum_{i_1} F_{i_1 i_2} \cdot D_{i_1 k_1}(z_1, x_1), \\
\text{Step 2:} \quad \forall\ k_1, k_2 \quad & L_{k_1 k_2}(z_1, z_2, x_1, x_2) = \sum_{i_2} Z_{i_2 k_1} \cdot D_{i_2 k_2}(z_2, x_2).
\end{aligned}$$

$Z_{i_2 k_1}$ is an intermediate variable that carries the accumulation result from the first dimension. Step 1 involves looping over subscripts $i_1$, $i_2$ and $k_1$. For any given $i_2$, over all $i_1$ and $k_1$, there are approximately $O(\sqrt{n} \log N)$ non-zero $D_{i_1 k_1}(z_1, x_1)$ for the complexity of 1D ACCs is about the square root of that of 2D ACCs. Multiplying the number of $i_2$ which is about $\sqrt{n}$ gives the cost for step 1 as $O(n \log N)$. Similarly, step 2 also takes $O(n \log N)$ time, and thus the total complexity of both steps is $O(n \log N)$. This computation is performed independently for all three color channels.

### Precomputation and Rendering

Transport functions $T$ are precomputed similarly to [75], except using the expanded-top-face parameterization. Light propagation (equation 4.35) involves computing the ACCs. There are two practical approaches, one relying on memory looks ups and the other favoring faster computation.

**Memory Based Approach:** It is easy to store all non-zero 1D ACCs $D_{ik}(s, o)$ compactly in a 4D table and look up their values during the computation. Due to the sparsity of ACCs, the cost to compute and store the 4D table is minor. As shown in the following table, non-zero ACCs for wavelet order 32 and at spatial resolution 256 only takes 1.06 seconds to precompute and 18.66MB to store. Note that the space and time cost scales close to linearly with the order of the wavelets, confirming our analysis in Sec. 4.4.2. Precomputed tables at all different orders and resolutions will be downloadable from our web site.

| Pre-Computation and Storage Cost in Memory Based Approach | | | | | | |
|---|---|---|---|---|---|---|
| | space (MB) time (S) | "Spatial" Resolutions ( s, o) | | | | |
| Wavelet Orders ( i, j) | | 64 | 128 | 256 | 512 | 1024 |
| | 8 | 0.26MB 0.011S | 1.05MB 0.044S | 4.22MB 0.176S | 16.92MB 0.708S | 67.74MB 2.760S |
| | 16 | 0.56MB 0.029S | 2.25MB 0.114S | 9.03MB 0.457S | 36.24MB 1.801S | 145.09MB 7.093S |
| | 32 | 1.15MB 0.066S | 4.65MB 0.257S | 18.66MB 1.059S | 74.81MB 4.236S | 299.45MB 16.871S |
| | 64 | 2.33MB 0.147S | 9.44MB 0.589S | 37.83MB 2.368S | 151.60MB 9.498S | 606.78MB 37.768S |

Because of the independence of lighting coefficients across vertices, light propagation can be easily parallelized on multi-core machines or clusters. We implement both single-thread and multiple-thread rendering algorithms on a 3.0 GHz Dual-Quad-Core machine using the boost library. To ensure workload balance between threads, we choose a round-robin scheduling scheme among a pool of tasks, each carrying a small number (512) of vertices to compute. Compared to the single-thread, an eight-thread implementation generates the exact same rendering result, but improves the speed by about 6.5 times and easily obtains real-time rates. Most of our relighting results are generated using the multi-thread memory-based implementation.

**Computational Approach:** For machines with faster computation, we tabulate $M(r,c)$ (canonical coupling coefficient defined by equation 4.22) in a 2D texture and compute ACCs using equation 4.21. Since $M(r,c)$ is sparse and symmetric (Sec. 4.4.2), the entire M table can fit into the L2 cache. A 256x256 $M$ table in floating point only takes 0.25 MB space. In our experiment settings, the computational approach is about half as fast as the memory based approach.

### Non-Linear Lighting Approximation

Realistic illumination can be well approximated using a handful of wavelet coefficients. Our model allows compression at two levels, of both the original light $F$

(source level) and the local incident radiance $L$ (target level), drastically speeding up the performance.

**Source-Level Compression** At the source level, lighting coefficients $F_i$ are ordered, and the most significant ones are picked, as in standard PRT. Using only 1% of source level coefficients usually generates accurate renderings, as shown in Figs. 4.8 and 4.9.

**Target-Level Compression** Target lighting coefficients $L_k$ vary across all vertices and change as the light source is dynamically updated. It is difficult to predetermine which target coefficients are important, and generating and sorting them in real time is too expensive. Instead, we assume a heuristic light $\tilde{F}_i$ and precompute lighting predictions $\tilde{L}_k$ for a number of $z$ and $x$ values. The light predictions $\tilde{L}_k$ are used to pick target lighting coefficients $L_k$ at run time.

$$\tilde{\mathbf{L}}_{\mathbf{k}}(\mathbf{z}, \mathbf{x}) = \sum_{\mathbf{i}} \tilde{\mathbf{F}}_{\mathbf{i}} \mathbf{D}_{\mathbf{ik}}(\mathbf{z}, \mathbf{x}). \tag{4.38}$$

Rarely will the lighting predictions $\tilde{L}_k$ be exactly the same as the actual lighting $L_k$. They however, roughly track how power distributions of the wavelet coefficients under affine transformations change with respect to spatial locations. We use a simple constant light heuristic (a vector with only DC, $[1, 0, 0, ...]$) in our experiments, and need about $20 \sim 30\%$ of target level lighting coefficients for visually accurate results, as shown in Figs. 4.8 and 4.9.

### 4.5.6 Results

We demonstrate three scenes—fertility (39,391 vertices, Fig. 4.8), chairs (59,995 vertices, Figs. 4.1B and 4.9A-C), and couches (68,046 vertices, Figs. 4.1A and 4.9D-F). All scenes have static geometries and are rendered using PRT, with near-field

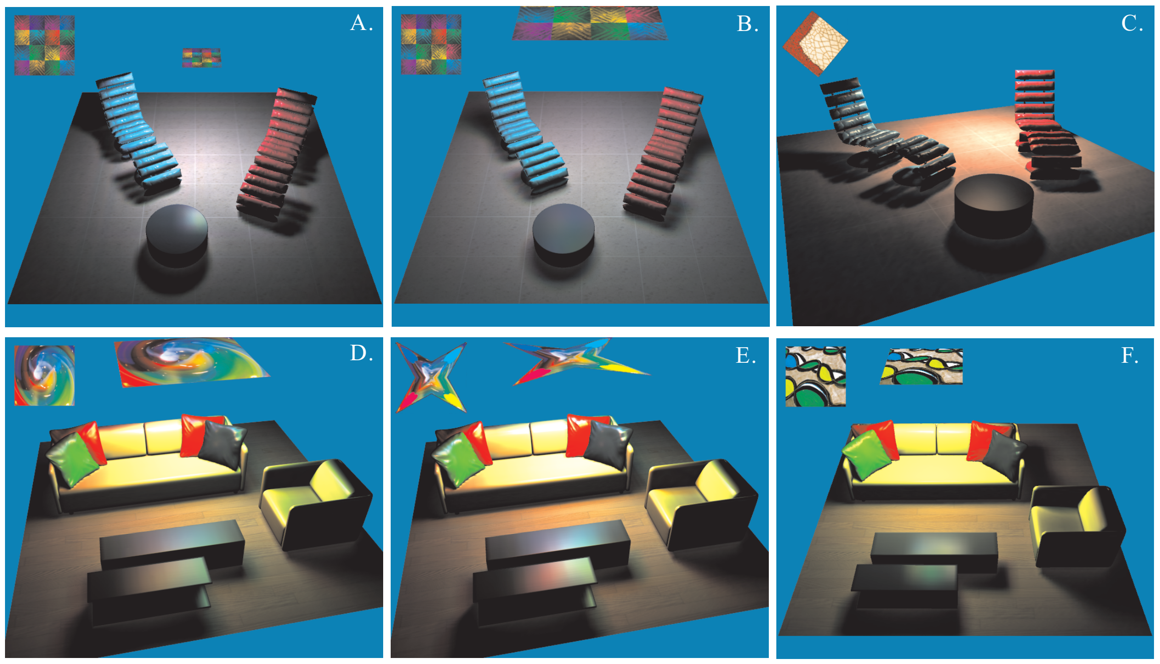


Figure 4.9: *Two specular scenes rendered with near-field sources using our method. Light textures used are shown in the upper left corners, and their sizes and positions vary. In the chair scene, note the sharper and larger shadows underneath the chairs when the light is small and close (A), and smaller and softer shadows when the light is larger and far away (B). Also note in the couch scene how light editing (change texture, reshape) from D to E influences specular reflections on the tables and cushions. Fig. 4.1A follows E but resizes and rotates the light. C and F render both specular scenes from another view point with a different light texture. Specular rendering from a different view point requires a separate precomputation. Close-ups can be found in Fig 4.1. Performance numbers are reported in Fig. 4.11.*

lighting effects generated using our method. The chairs, tables and couches have specular materials with precomputation done per vertex [75]. The fertility scene is diffuse and can be viewed from different angles. In all cases, we can interactively edit the light.

### Near-Field vs. Distant Lighting Effects

In Fig. 4.8B, we light a diffuse fertility scene with near-field lights. Spatially varying shadings and colored soft shadows on the floor are clearly visible. In contrast, Fig. 4.8C shows a rendering using the standard environment mapping technique,[5] which fails to capture the shading variations on the floor or sculpture that are critical to the mood of the scene.

In Fig. 4.9, we render two specular scenes with a number of lights at different positions. In particular, when the light is closer to the floor in Fig. 4.9A (as compared to Fig. 4.9B), shadows of both chairs expand sideways and the specularity on the table focuses. In Figs. 4.9D-E, we can clearly observe how editing the light texture, as well as its shape and size, changes the specular highlights on the tables and cushions. These effects are hard to capture with distant illumination. Close-ups are found in Figs. 4.1A and 4.1B.

### Light Editing

We develop a prototype light editing and design system, which allows artists to edit the lights in a more intuitive and interactive way. An artist can move, resize, or horizontally rotate the light. Lights can also be textured. Image processing methods such as blending, filtering, warping and painting can be easily applied to edit the light texture. Because the cost of compressing the edited light texture into wavelets in real-time is minor, changes can be immediately reflected in the realistically rendered images using our algorithm. For example, starting from Fig. 4.9D, we first paint and reshape the light to obtain Fig. 4.9E, and then resize and rotate the light to generate the image in Fig. 4.1A. Note the generality of our system to handle textured and editable/reshapeable light sources in Figs. 4.8 and 4.9, which cannot be addressed by previous near-field lighting techniques like precomputed source radiance fields [126].

[5]The incident lighting at the center of scene is used as the “environment map” and fed to all vertices for shading computations.

### Accuracy Analysis

**Validation:** We compare the ground truth image from a raytracer in Fig. 4.8A with our result in Fig. 4.8B. We also showed a ground truth comparison in our didactic example of a knot scene in Fig. 4.7. Since we only use a finite plane to parameterize the upper hemisphere, light incident at grazing angles is omitted, resulting in dimmer shading for some boundary vertices and the lack of grazing angle specularities. Also, lighting coefficients are compressed at both the source and target levels. Thus, energies at certain frequencies may be lost. Nevertheless, Figs. 4.7 and 4.8 clearly demonstrate that our results show little visual difference from the ground truth, but now can be rendered at real-time rates. Similar results hold for our other images.

**Lighting Approximation Error:** Figure 4.10 shows the approximation errors for different compressions at both source and target levels. A few representative light textures used in the renderings are included in the accuracy analysis. Note that the horizontal axes in Fig. 4.10 are on a log scale. The approximation accuracy improves quickly with increasing numbers of coefficients used. In addition, the target level compression is less efficient than the source level, requiring more coefficients for the same level of accuracy. This inefficiency is due to the fact that we rely on a constant heuristic light to predict significant lighting coefficients after the affine transformation. Our experiments show that 1% of the source level and 20 $\sim$ 30% of the target level lighting coefficients (using a constant light heuristic) usually suffice to generate visually accurate results.

### Performance

All images are rendered at 1200x900 resolution and a wavelet order of 32. The rendering speed depends on a number of factors such as the total number of

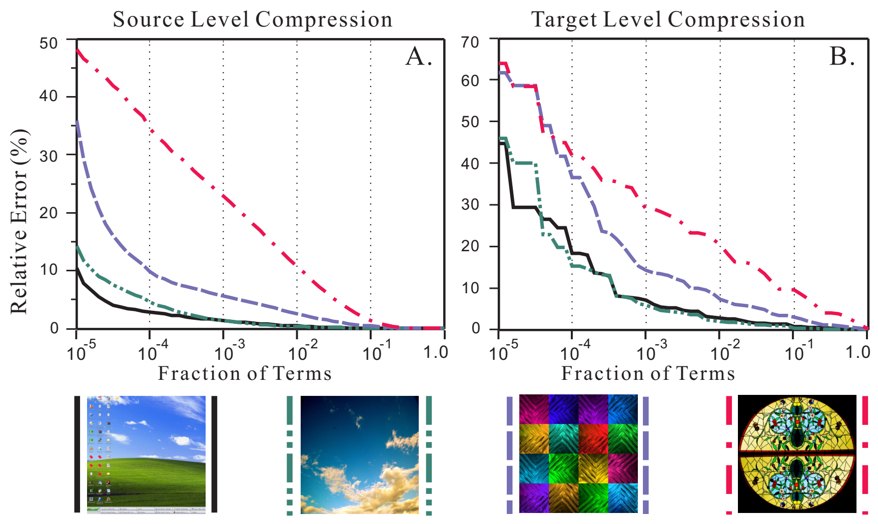


Figure 4.10: *Approximation errors of the source (graph A) and target (graph B) level compressions. Four representative light textures with different levels of high frequencies are used. The horizontal axes are on a log scale.*

vertices, numbers of the retained source and target level lighting coefficients, and the complexities of the scene materials. In table 4.11, we report the rendering performance for the fertility, chair and couch scenes. As shown in the second and third performance columns (which correspond to the common compression usage), our algorithm provides real-time performance, as also seen in the accompanying video.

| Scenes | #Verts | Threads | Source=1% Target=1% | Source=1% Target=20% | Source=5% Target=40% | Source=20% Target=70% | Source=100% Target=100% |
|---|---|---|---|---|---|---|---|
| diffuse fertility | 39,391 | 8 threads | 151.86fps | 38.71fps | 24.83fps | 13.78fps | 7.04fps |
| | | 1 thread | 32.79fps | 7.58fps | 4.17fps | 2.17fps | 1.08fps |
| specular chairs | 56,995 | 8 threads | 107.33fps | 25.87fps | 12.48fps | 7.87fps | 4.86fps |
| | | 1 thread | 20.43fps | 4.52fps | 2.03fps | 1.15fps | 0.75fps |
| specular couches | 68,046 | 8 threads | 101.64fps | 23.34fps | 11.98fps | 7.31fps | 4.80fps |
| | | 1 thread | 18.60fps | 3.91fps | 1.94fps | 1.10fps | 0.71fps |

Figure 4.11: *Rendering performance with different compressions. For each scene, the upper row contains the performance numbers using the multi-thread implementation, and the lower row for the single-thread implementation. All performance is measured on a 3.0 GHz Dual-Quad-Core PC with 4.0 GB memory. The second and third columns correspond to realistic compression levels for accurate rendering, and achieve real-time rates for both the single-thread and multi-thread algorithms.*

### 4.5.7 Comparison to Previous Near-Field Relighting Methods

In comparison with previous techniques, our method offers significant design flexibilities and achieves effects that are otherwise hard to capture. Annen et al. [2] and Wang et al. [118] pioneered rendering mid-range illuminations, using respectively spherical harmonic gradients and scaling. Lights are assumed some distance away from the scene so that the lighting can be smoothly interpolated and its propagation (affine transformation) approximated. Our work can be seen as extending these methods to near-field settings, as shown in Fig. 4.8. Moreover, our technique can also render specular scenes, such as the chair and couch scenes in Fig. 4.9. Zhou et al. [126] and Kristensen et al. [60] made important advances in near-field rendering of both diffuse and specular scenes. The light content however, is built into the precomputations and has to remain static during rendering. Designers can move the light or change its intensity, but not edit the light shape or texture. Our method can be seen as an important generalization of their techniques, and allows light editing to be fully integrated with any PRT framework. For example, interactively painting the lights and changing their shapes, as done in Fig. 4.8D and 4.9, or quickly flipping through several arbitrary light textures during relighting, as demonstrated in the video, are all feasible with our method. Finally, Annen et al. [2], Wang et al. [118] and Kristensen et al. [60] base their methods on the spherical harmonic basis and cannot capture all-frequency effects.

## 4.6 Other Applications

Besides our main problem of PRT near-field relighting, affinely transformed wavelets and wavelet integrals have many other potential applications in rendering, image and signal processing, and numerical analysis. As a first proof of concept, we demonstrate initial examples of wavelet importance sampling with near-field light-

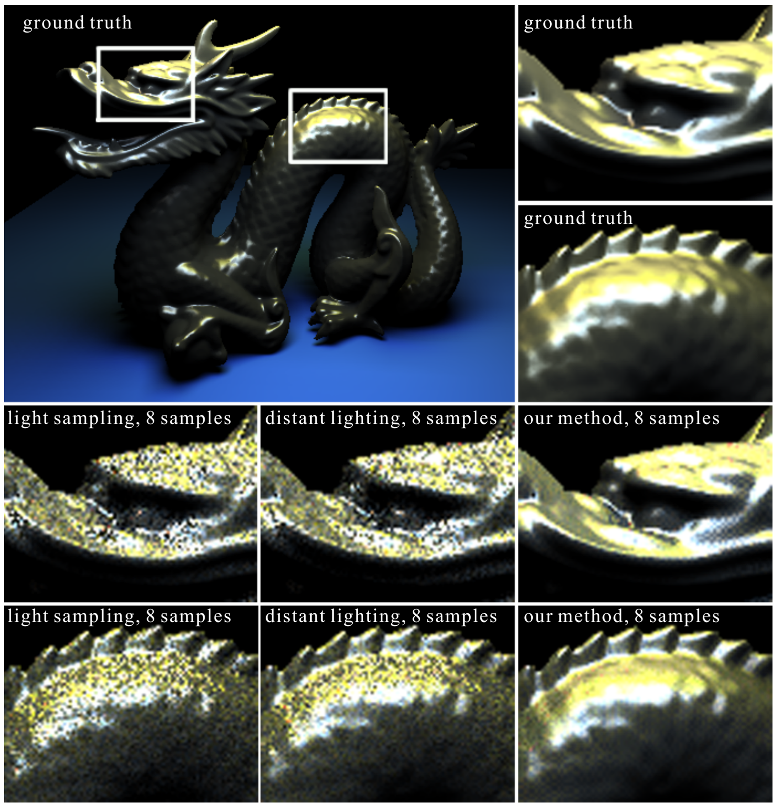

ground truth
ground truth
ground truth
light sampling, 8 samples
distant lighting, 8 samples
our method, 8 samples
light sampling, 8 samples
distant lighting, 8 samples
our method, 8 samples

Figure 4.12: *A gold dragon under a near-field area source rendered using wavelet importance sampling. The ground truth rendering is at the top. In the close-ups, for the same number samples (8), we compare the result of our method with those by standard wavelet importance sampling but assuming that the light is distant (distant lighting), and light importance sampling (light sampling). Order 64 wavelets are used for all three sampling methods. Our method converges to the ground truth an order of magnitude faster, exhibiting substantially less noise at this sample count.*

ing for offline Monte Carlo rendering, and image dilation and translation directly in the wavelet domain for image processing.

### 4.6.1 Wavelet Importance Sampling

Clarberg et al. [22] have shown that importance sampling the product of the BRDF and the distant lighting can greatly reduce the variance in Monte Carlo rendering. However, their method is limited to distant environment map lighting, since lighting-BRDF products are computed using standard triple product integrals. Our theory enables a direct extension to the near-field setting, with planar area sources. We pre-tabulate BRDFs as 4D functions as done in [22]. For each pixel, we affinely transform the original light source into the local incident radiance using equation 4.35 of Sec. 4.5, in a very similar fashion as for relighting. We then multiply the wavelet coefficients of the local radiance with those of the BRDF using the standard triple product wavelet integral [76]. Thereafter, we perform hierarchical wavelet warping to obtain importance samples, used for Monte Carlo estimation. Distinct from [22], we use the standard wavelet decomposition and an expanded-top-face parameterization, as we do in Sec. 4.5 for relighting.

We demonstrate a near-field rendering of a gold dragon under an area light source in Fig. 4.12. We observe visual effects such as highly contrasted and spatially varying shadings that are hard to obtain with distant lighting. In the close-ups, we compare the result of our method with those of standard wavelet importance sampling, but assuming the light is distant (distant lighting), and light importance sampling (light sampling). With only 8 samples per pixel, our method can significantly reduce the noise and generate results that are visually indistinguishable from the ground truth.

### 4.6.2 Image Dilation and Translation

Many image operations such as convolution, masking and zooming involve basic operators like dilations (scales) and translations. Using our theory, images compressed using wavelets can be scaled and translated directly in the wavelet domain, instead of first requiring decompression into the pixel domain. If needed, e.g., images are streamed from a remote server, image dilations and translations can also be performed in a multi-resolution fashion.

The basic equation for image scaling and translation remains the same as equation 4.35. $F(v)$, originally the area light, becomes the source image and $L(v)$, the incident radiance, becomes the target image. Variables z and x respectively de-

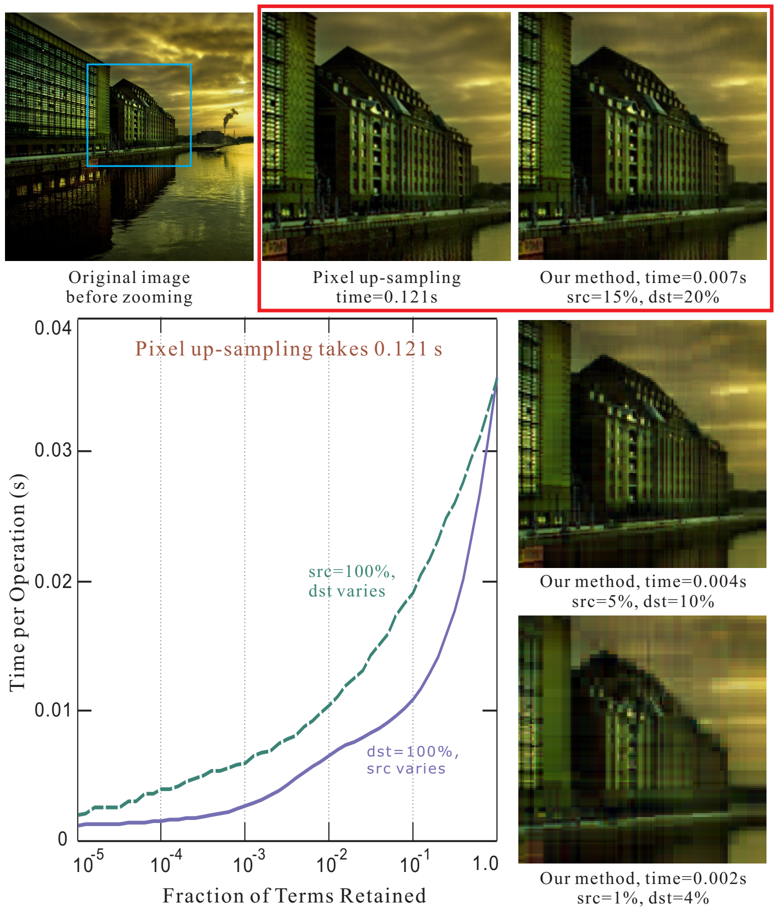


Figure 4.13: *Zooming in on a building in a picture of Berlin. The top left is the original image. The red box at the top contains zoomed images using up-sampling in the pixel domain and our method. Our method is visually as accurate as the pixel domain method, but more than 15 times faster. More results with different compressions at both the source and target levels (src and dst) are shown in the column beneath the red box. In the bottom left graph, we plot the performance curves of our method with different compressions. Each curve corresponds to either the source or the target level compressions (one level per curve). The horizontal axis is on a log scale. Time measurements of the pixel method include decompression and recompression of images encoded in wavelets.*

scribe the amount of dilation and translation. As before, compression of wavelet coefficients can be performed at both the source and target levels.

An example result is shown in Fig. 4.13, where we zoom in on a building in a photograph of Berlin, and plot the performance curves with different compressions for the source and target. As shown in the red box, our method is visually as accurate as the brute force up-sampling in the pixel domain, but more than 15 times faster. With more compression (two bottom right images), the speed-up can be up to 60 times at the cost of losing some details in the final images. Accuracy analysis can be referred to Fig. 4.10 in Sec. 4.5. Along with other wavelet domain image operations such as additions and multiplications, we believe our work is a significant step towards a complete wavelet domain toolbox for image processing.

## 4.7 Discussion and Future Work

We have presented a novel theory of affine double and triple product integrals. In particular, we have analyzed the sparsity of affine coupling and tripling coefficients in the Haar wavelet basis, showing that they have nearly linear complexity, which leads to efficient algorithms for computing affine transforms in Haar wavelets.

Besides being of substantial theoretical interest, our framework has significant practical implications. We develop some of the first methods for including near-field lighting effects in all-frequency PRT algorithms. A planar area source can be translated, dilated, rotated in its plane, and have its texture edited, all while the scene is rendered in real-time—local lighting is propagated to each vertex via an affine transform of the source radiance directly in wavelets. Our method can be integrated with almost any PRT algorithm.

Future work in relighting can follow many avenues. Better heuristics can be experimented with for more efficient lighting compression. Relighting with more general illumination sources like light fields and dynamic textures can be explored.

Our theory can also be applied to many other applications that depend on wavelet representations. As a proof of concept, we demonstrate implementation of wavelet importance sampling with near-field lights for offline Monte Carlo renderers. We also show the practical utility of our theory in dilating and translating images directly in the wavelet domain for image processing. However, we have only scratched the surface of applications, and we predict many future developments in graphics and applied mathematics.

Finally, our work reveals the fundamental sparsity of some basic wavelet operations (dilation and translation). This opens up a fresh perspective in approaching many other operations that have traditionally been viewed as difficult to compute directly with wavelets. Together with standard double and triple product integrals (multiplication), more complicated operators can be built, e.g. convolutions can be reduced to translations and multiplications. The basic mechanism and computational machinery we employed in developing our theory will shed key insights on constructing a complete suite of wavelet domain operations.

# Chapter 5

# Frequency Based Approaches to Real-Time Rendering

In last chapter, we presented a novel theory of affine double and triple product wavelet integrals for near-field relighting. In this chapter, we introduce two frequency based approaches respectively to correctly filtering normal maps [36] and generating soft shadows in dynamic scenes [92]. Mathematical tools in frequency analysis such as spherical harmonics are exploited. This chapter includes joint work with my colleagues at Columbia University and researchers at Microsoft Research and Zhejiang University. We focus on the key idea or my specific contributions, and detail on implementation and extensions may be skipped.

## 5.1 Frequency Domain Normal Map Filtering

### 5.1.1 Background and Preliminaries

Representing surface detail at a variety of scales requires good filtering algorithms. A common, linear approach to reduce aliasing is MIP-mapping [125]. Normal mapping (also known as bump mapping [13] or normal perturbation), a simple

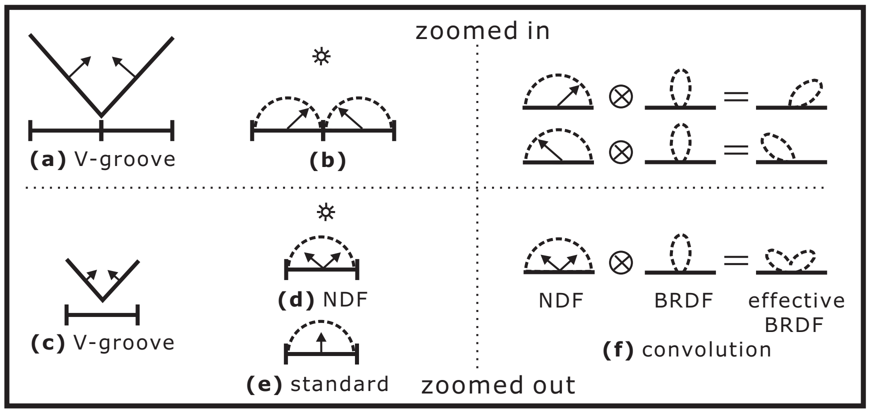


Figure 5.1: *Consider a simple V-groove. Initially in closeup (a), each face is a single pixel. As we zoom out, and average into a single pixel (c), standard MIP-mapping averages the normal to an effectively flat surface (e). However, our method uses the full normal distribution function or NDF (d), that preserves the original normals. This NDF can be linearly convolved with the BRDF (f) to obtain an effective BRDF, accurate for shading.*

and widely used analogue to color texture mapping, specifies the surface normal at each texel. Unfortunately, normal map filtering is very difficult because shading is not linear in the normal. For example, consider the simple V-groove surface geometry in Fig. 5.1a. In a closeup, this spans two pixels, each of which has distinct normals (b). As we zoom out (c), the average normal of the two sides (e) corresponds simply to a flat surface, where the shading is likely very different.

Our most important contribution in [36] is theoretical, formalizing these ideas and developing a comprehensive framework for normal map filtering. This is joint work with Charles Han, Ravi Ramamoorthi and Eitan Grinspun. In Sec. 5.1.2, we derive an analytic formula, showing that filtering can be written as a spherical convolution of the BRDF of the material with a function we define as the *normal distribution function* (NDF)[1] of the texel. This mathematical form holds for a large class of common BRDFs including Lambertian, Blinn-Phong, microfacet models like Torrance-Sparrow, and many measured BRDFs. However, the convolution does not apply exactly to BRDF models that depend on the "reflected direction"

[1]We define the NDF as a weighted mapping of surface normals onto the unit sphere; more formally, it is the extended Gaussian Image [43] of the geometry within a texel.

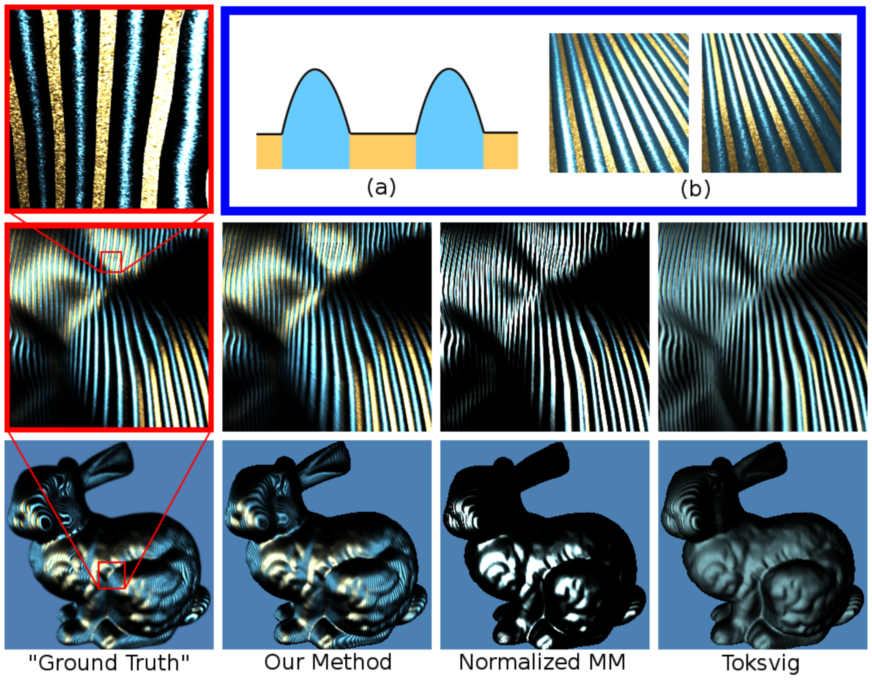


Figure 5.2: **Top:** *Closeup of the base normal map; all other methods are identical at this scale and are not shown. Schematic (a) and diffusely shaded (b) views are provided to aid in comparison/visualization.* **Middle:** *When we zoom out, differences emerge between our (6-lobe) spherical vMF method, the Toksvig approach (rightmost), and normalized MIP-mapping. (Unnormalized MIP-mapping of normals produces an essentially black image.)* **Bottom:** *Zooming out even further, our method is clearly more accurate than Toksvig's model, and compares favorably with ground truth. (The reader may zoom into the PDF to compare images.)*

between the light source and viewer, e.g., the standard Phong model. Our analytic result immediately connects geometrical normal map filtering with the older lighting-BRDF convolution result for appearance [6, 87]. This result also unifies many previous normal map filtering approaches, which can now be viewed as special cases.

Moreover, we can immediately apply a host of mathematical representations originally developed for lighting-BRDF convolution. In particular, we develop two novel algorithms. Our first method (Sec. 5.1.2) is a general framework that uses *spherical harmonics* (SH). Our second method, intended for high-frequency materials, uses an approximate function fitting technique known as spherical *expectation maximization* (EM) [5]. This method approximates or fits the NDF with *von*

| Acronym | Definition | Symbol |
|---|---|---|
| NDF | normal distribution function | $\gamma(n)$ |
| SH | spherical harmonics | $Y_{lm}$ |
| SRBF | spherical radial basis function | $\gamma(n \cdot \mu)$ |
| EM | expectation maximization | |
| vMF | von Mises-Fisher distribution | $\gamma(n \cdot \mu; \theta)$, $\theta = \{\kappa, \mu\}$ |
| movMF | mixture of vMF lobes | $\gamma(n; \Theta)$, $\Theta = \{\alpha_j, \theta_j\}_{j=1}^{J}$ |

Table 5.1: *Important abbreviations and acronyms used in Sec. 5.1.*

*Mises-Fisher* (vMF) distributions (refer [36] for details).

### Previous Related Work

**Normal Map Filtering:** Many previous methods approximate the normal distribution function using a single symmetric or asymmetric lobe. [97] described the lobe using covariance matrices, while [81] mapped normal distributions consisting of a single 3D Gaussian. A simple GPU method is described in [114]. In our framework, these methods can retrospectively be considered similar to using a single vMF lobe.

An early inspiration is [32], which uses up to seven Phong lobes per texel (and up to 56 at the coarsest scales). Note that [32] uses nonlinear least-squares optimization to fit lobes. In our experience, this is unstable and slow, especially considering the number of peaks and texels in a normal map. The most recent and closest previous work is [113], which uses EM to fit Gaussian lobes to a planar projection of the hemispherical NDF at each texel. It is easy to understand [113] as an important special case in our theoretical framework. [113] uses planar Gaussian fits, which has be shown to be considerably less accurate both in our work (please refer to [36] for details) and in other contexts [109]. Further, [113] treat the BRDF itself as a pre-baked distribution of normals at fine-scale texels.

**Hierarchy of Representations:** A hierarchy of scales, with geometry transitioning to bump or normal maps, transitioning to BRDFs, was first proposed by Kajiya [50]. This idea is explored in detail by [8], but they do not focus on normal map filtering as in our work. Similarly, appearance-preserving simplification methods replace fine-scale geometry with normal and texture maps [23]. It is likely that our approach could enable continuous level of detail and anti-aliasing in these methods. Separately, our formulation allows one to understand the tradeoff between a normal distribution and the BRDF, since the final image is given by a convolution of the NDF and BRDF.

Further, many of our mathematical representations and ideas derive from previous spherical convolution techniques [6, 87]. We also build on PRT methods that are introduced in Sec. 2.5.

### Mathematical Preliminaries

The reflected light $B$ at a spatial point $x$ in direction $\omega_o$ is

$$B(x, \omega_o) = \int_{S^2} L(x, \omega_i)\rho(\omega_i', \omega_o')\, d\omega_i, \tag{5.1}$$

where $L$ is the lighting at $x$ from incident direction $\omega_i$, and $\rho$ is the BRDF (including the cosine of the incident angle). This is the standard reflectance equation.

**Effective BRDF:** We define a new function, the *effective BRDF* or transfer function that depends on the surface normal (that we denote as $n(x)$ or simply $n$ for clarity) as,

$$\rho^{\text{eff}}(\omega_i, \omega_o; n) = \rho\left(R_n(\omega_i), R_n(\omega_o)\right),$$

allowing us to write equation 5.1 using the global directions,

$$B(x, \omega_o) = \int_{S^2} L(x, \omega_i) \rho^{\text{eff}}(\omega_i, \omega_o; n(x))\, d\omega_i. \tag{5.2}$$

**BRDF Parameterizations:** Many BRDFs such as Lambertian, Blinn-Phong or microfacet half angle (like Torrance-Sparrow), and many factored and measured BRDFs can be written as

$$\rho^{\text{eff}}(\omega_i, \omega_o; n) = f(n \cdot \omega(\omega_i, \omega_o)), \tag{5.3}$$

where the 1D function $f$ is radially symmetric about the shading normal $n$, and depends on the chosen parameterization $\omega(\omega_i, \omega_o)$ (henceforth $\omega$).

**Normal Map Filtering:** In screen space, the exitant radiance or pixel color $B(x, \omega_o)$ at a surface location $x$ should represent the average radiance at the $N$ corresponding finer-level texels $q$:

$$\begin{aligned} B(x, \omega_o) &= \frac{1}{N} \sum_{q \in x} \int_{S^2} L(x, \omega_i) \rho^{\text{eff}}(\omega_i, \omega_o; n(q))\, d\omega_i \\ &= \int_{S^2} L(x, \omega_i) \left( \frac{1}{N} \sum_{q \in x} \rho^{\text{eff}}(\omega_i, \omega_o; n(q)) \right) d\omega_i. \end{aligned}$$

This formulation allows us to define a new effective BRDF,

$$\rho^{\text{eff}}(\omega_i, \omega_o; x) = \frac{1}{N} \sum_{q \in x} \rho \left( R_{n(q)}(\omega_i), R_{n(q)}(\omega_o) \right). \tag{5.4}$$

Note that the effective BRDF now depends implicitly on all the normals $n(q)$ at $x$, rather than on a single normal. The next section shows how to explicitly represent $\rho^{\text{eff}}$ as a convolution of the original BRDF and a new function we call the NDF.

### 5.1.2 Theory of Normal Mapping as Convolution

In this section, we introduce our theoretical framework for normal map filtering as convolution and describe how spherical harmonics can be used as the mathematical representation for practical implementation.

#### Normal Distribution Function (NDF) and Effective BRDFs

Our first step is to convert equation 5.4 into continuous form, defining

$$\rho^{\mathrm{eff}}(\omega_i, \omega_o; \gamma(\cdot)) = \int_{S^2} \rho\left(R_n(\omega_i), R_n(\omega_o)\right) \gamma(n)\, dn, \tag{5.5}$$

where $\gamma(n)$ is a new function that we introduce and define as the *normal distribution function* (NDF), and the integral is over the sphere $S^2$ of surface orientations. Note that a unique NDF $\gamma(n)$ exists at each surface location $x$; for a discrete normal map, $\gamma(n)$ would simply be a sum of (spherical) delta distributions at $n(q)$, the fine-scale normals at $x$. Formally, $\gamma(n) = \frac{1}{N} \sum_{q \in x} \delta(n - n(q))$, as seen in Fig. 5.1d.

We further substitute the form of the BRDF from equation 5.3. Recall in this case that the BRDF only depends on the angle between $\omega$ and the surface normal $n$, and is given by $f(\omega \cdot n)$. The effective BRDF is now also only a function of $\omega$,

$$\rho^{\mathrm{eff}}(\omega; \gamma(\cdot)) = \int_{S^2} f(\omega \cdot n) \gamma(n)\, dn. \tag{5.6}$$

Note that the initial BRDF $\rho(\cdot) = f(\omega \cdot n)$ is symmetric about $n$, but the final result $\rho^{\mathrm{eff}}(\omega)$ is an arbitrary function on the sphere and is generally not symmetric.

#### Spherical Harmonics

We analyze equation 5.6 in the frequency domain using spherical harmonic (SH) basis functions $Y_{lm}(\cdot)$, which are the frequency domain analog to Fourier series on

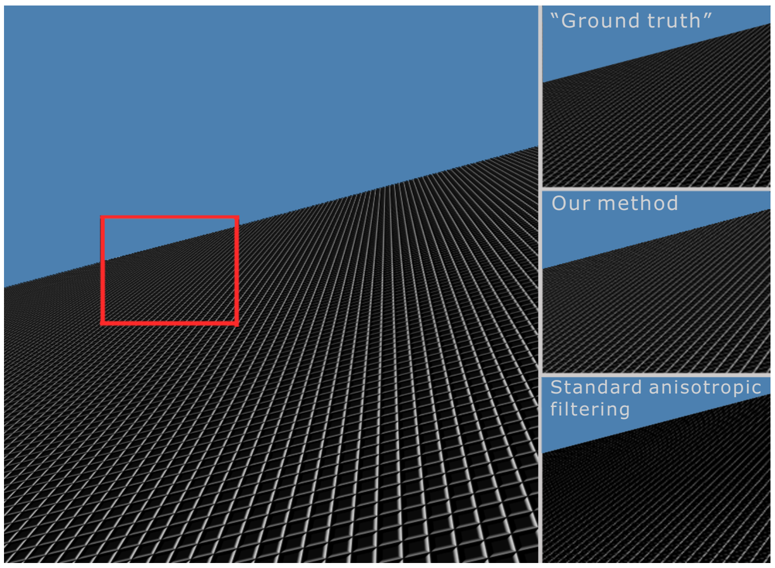


Figure 5.3: *Spherical harmonic anisotropic filtering for Lambertian reflection. Note the behavior for far regions of the plane. With standard normal filtering, these regions are averaged to a nearly flat surface. By contrast, our method is quite accurate in distant regions.*

the unit sphere. The $l$ index is the frequency with $l \geq 0$, and $-l \leq m \leq l$,

$$\gamma(n) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \gamma_{lm} Y_{lm}(n) \qquad f(\omega \cdot n) = \sum_{l=0}^{\infty} f_l Y_{l0}(\omega \cdot n)$$

$$\rho^{\text{eff}}(\omega) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \rho^{\text{eff}}_{lm} Y_{lm}(\omega).$$

The above is a standard function expansion, as in Fourier series. Note that the symmetric function $f(\omega \cdot n)$ is expanded only in terms of the zonal harmonics $Y_{l0}(\cdot)$ ($m = 0$), which are radially symmetric and thus depend only on the elevation angle.

Equation 5.6 has been extensively studied within the context of lighting-BRDF convolution for Lambertian or radially symmetric BRDFs [6, 87]. Since the theory is mathematically identical, we may directly use their results. Specifically, equation 5.6 expresses a spherical convolution of the NDF with the BRDF filter.

$$\rho^{\text{eff}}_{lm} = \sqrt{\frac{4\pi}{2l+1}} f_l \gamma_{lm}.$$

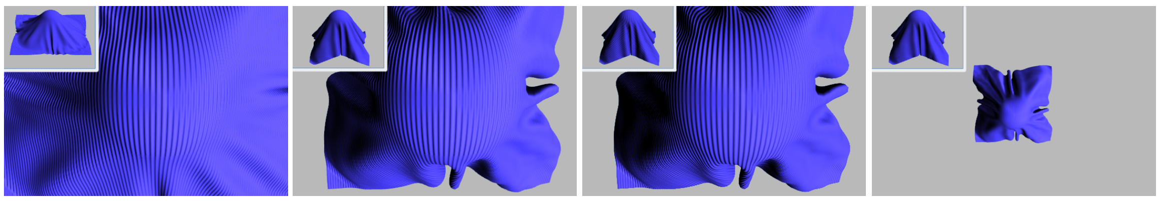

(a) Our method, frame 1 (b) Our method, frame 2 (c) “Ground truth”, frame 2 (d) Our method, zoomed out

Figure 5.4: *Stills from a sequence of cloth draping over a sphere, with closeups indicating correct normal filtering using our spherical harmonic algorithm (the full movie is shown in the video). Note the smooth transition from the center (almost no filtering) to the corners (fully filtered) in (b)—compare also with ground truth in (c). (d) is a zoomed out view that also filters correctly. We use a blue fabric material from the Matusik database as the BRDF.*

Explicitly making the NDF and effective BRDF functions of a texel $q$, we have

$$\boxed{\rho^{\text{eff}}_{lm}(q) = \hat{\rho}_l \gamma_{lm}(q)} \qquad \hat{\rho}_l = \sqrt{\frac{4\pi}{2l+1}} f_l \, , \tag{5.7}$$

where the NDF considers all normals covered by $q$. While $q$ usually corresponds to a given level and offset in a MIP-map, it can also consider more general “footprints”.

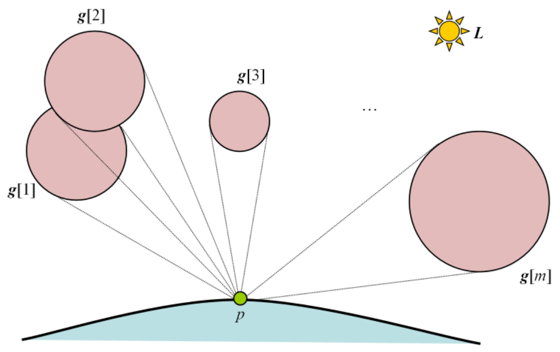
Practical implementation with spherical harmonics is straightforward, involving two basic steps: (1)computing a MIP-map of NDF coefficients for each level of texels q of the normal map. An important insight is that, unlike the original normals, these spherical harmonic NDF coefficients $\gamma_{lm}(q_0)$ can now correctly be *linearly* filtered or averaged for coarser levels $\gamma_{lm}(q)$. (2) rendering the final color by directly computing equation 5.7.

We implement the algorithm in a pixel shader using GLSL (see our website for example code). The spherical harmonics $Y_{lm}$ are stored in floating point textures, as are the MIP-mapped NDF coefficients $\gamma_{lm}(q)$. An example of Lambertian reflection using only nine spherical harmonic coefficients ($l \leq 2$) is shown in Fig. 5.3. Note that we preserve accuracy in far away regions of the plane, while naïve averaging of the normal produces a nearly flat surface that is much darker than the actual. Specular materials with BRDF f (wh n) also fit within our framework. Fig. 5.4 shows closeup views from an animation sequence of cloth draping over a sphere, using the blue fabric material from the Matusik database. Note the accuracy of our method (compare (b) with the supersampled "ground truth" in (c)).

## 5.2 Soft Shadowing using Spherical Harmonic Exponentials

### 5.2.1 Background and Preliminaries

Soft shadows are critical for realistic image synthesis. In [92], we propose a novel soft shadowing technique using spherical harmonic exponentials. This is joint work with researchers at Microsoft Research and Zhejiang University. Our goal is accurate generation of soft shadows in general, dynamic scenes where tabulation is impractical. Our method operates directly on vectors representing the low-frequency visibility function of a blocker as seen from a receiver point, expressed using the spherical harmonic (SH) basis. The fundamental difficulty is that SH rotations and products [77] are very expensive, precluding GPU implementation and restricting real-time CPU implementation to a few precomputed blockers.



Our solution approximates blocker geometry as a collection of spheres. Rather than computing blocker products directly, we represent blocker visibility in log space. The total log blocking vector can then be accumulated as a simple sum of log vectors over all blockers. For order-$n$ SH vectors, this reduces per-blocker computation from $O(n^5)$ to $O(n^2)$. Given the total log of blocker visibility at a receiver point, we perform SH exponentiation to yield the total blocker visibility. The result correctly accounts for blocker overlap and low-frequency directional dependence. Ours is the first real-time method for rendering soft shadows from low-frequency environmental lighting in dynamic scenes with many blockers. In particular, we introduce two technical contributions to do this. One is to accumulate low-frequency blocker visibility in log space. The Volterra series forms the mathematical foundation for SH exponentiation. The second is to make approximation of low-frequency visibility efficient using sets of bounding spheres. Our work is published in [92].

**Related Work**

Using triple products of lighting, reflectance, and visibility, [76] presents a method for fast relighting and view change in static scenes. Shadow fields [126] extend this method to account for dynamic visibility changes and form the basis for our approach of directly manipulating low-frequency visibility vectors. SH rotation and products used in these methods are more expensive with the "all-frequency" basis (wavelets over cube maps), which typically has an order of magnitude more basis components than low-order SH.

Ambient occlusion [17, 58] yields shadows, but maximally soft ones having no response to lighting directionality. [17] makes use of simple approximating elements for blockers organized as a hierarchy. The following work of [100] used a multipass algorithm that separates the ambient occlusion problem into high-frequency and low frequency ambient occlusions. [57] designed a method to work with deforming meshes based on a low-dimensional set of parameters, as in character animation. Several methods [54, 17, 126] including ours may be termed *blocker accumulation* methods because they process a list of blocker geometry at each receiver point. Other methods employ multiple shadow buffers [99, 1, 70] and so entail slow integration over lighting directions when rendering large lights. For example, [54] is a blocker accumulation method that also uses the directional lighting basis, but is limited only to very simple scenes. On the other hand, soft shadow volumes [4] can handle greater scene complexity but only small area light sources. Convolution can also be used to shadow [105], but it produces a scalar modulation rather than true hemispherical radiance and is difficult to apply in general scenes with non-planar receivers and large-depth blockers. Holomorphic factorization [69] uses log space to factor high-dimensional BRDFs into a sum of positive, lower-dimensional functions. Only scalar exponentiation is required at run-time, while we make use of the full exponentiated vector.

**Mathematical Preliminaries**

We use math italic for scalars and 3d points or vectors (e.g., $x$, $s$), boldface italic for SH vectors (e.g., $\vec{f}$, $\vec{g}$), and sans serif for matrices and higher-order tensors (e.g., $\mathsf{\Gamma}$, $\mathsf{M}$, $\mathsf{D}$). Spherical harmonic basis is introduced in Sec. 2.2.

**SH Products and the Triple Product Tensor** are useful for computing the combined shadowing effect of multiple blockers directly in the SH basis, without resorting to numerical integration over directions [54, 70] or performing complicated geometric clipping operations [4, 63]. The *SH product*, denoted $\vec{f} * \vec{g}$, represents the order-$n$ projected result of multiplying the reconstruction of two order-$n$ vectors, $\vec{f}$ times $\vec{g}$, or

$$\vec{f} * \vec{g} = \int_S f(s)\, g(s)\, \vec{y}(s)\, ds \;\Rightarrow\; (\vec{f} * \vec{g})_i = \sum_{jk} \mathsf{\Gamma}_{ijk}\, \vec{f}_j\, \vec{g}_k \tag{5.8}$$

where the *SH triple product tensor*, $\mathsf{\Gamma}_{ijk}$, is defined by

$$\mathsf{\Gamma}_{ijk} = \int_S \vec{y}_i(s)\, \vec{y}_j(s)\, \vec{y}_k(s) ds. \tag{5.9}$$

$\Gamma$ is a symmetric, sparse, order-3 tensor. This definition incurs truncation error because the product of two order-$n$ vectors is actually order $2n-1$. Order-$n$ SH products are $O(n^5)$ [76]. The following table records the number of nonzero entries in $\Gamma$ as function of $n$:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 1 | 10 | 83 | 369 | 1164 | 2961 | 6586 | 13018 |

Even at low orders, SH product is an expensive operation.

We can also define the *SH product matrix*, $\mathsf{M}_f$, given an SH vector $\vec{f}$. The product matrix is a symmetric matrix which encapsulates SH product with $\vec{f}$; in other words, $\vec{f} * \vec{g} = \mathsf{M}_f\,\vec{g}$ for an arbitrary vector $\vec{g}$. $\mathsf{M}_f$ is defined by

$$(\mathsf{M}_f)_{ij} = \sum_k \Gamma_{ijk}\,\vec{f}_k \tag{5.10}$$

**Shadowing using SH Products** [126] computes the product of a collection of $m$ blockers $\vec{g}[1], \vec{g}[2], \ldots, \vec{g}[m]$, via

$$\vec{g} = \vec{g}[1] * \vec{g}[2] * \cdots * \vec{g}[m] \tag{5.11}$$

where each $\vec{g}[i]$ is the SH projection of the corresponding blocker visibility function

$$g[i](s) = \begin{cases} 0, & \text{if object } i \text{ blocks in direction } s; \\ 1, & \text{otherwise.} \end{cases} \tag{5.12}$$

Although SH product is commutative, it is not associative, so the ordering in which the above products are performed matters.

**Shadowing in Log Space** instead accumulates the log of blocker visibilities, denoted by $\vec{f}[1], \vec{f}[2], \ldots, \vec{f}[m]$ where $\vec{f}[i] = \log(\vec{g}[i])$. Thus

$$\vec{g} = \exp(\vec{f}) = \exp\left(\vec{f}[1] + \vec{f}[2] + \cdots + \vec{f}[m]\right) \tag{5.13}$$

Accumulating the log now involves vector sums which are independent of the blocker ordering and much cheaper than SH products. Sec. 5.2.2 discusses how the SH exponential is computed while SH log is discussed in Sec. 5.2.3.

**Shading** then makes use of the total visibility vector $\vec{g}$. For diffuse surfaces, the computation is $(\vec{H}(N), \vec{L}, \vec{g})$ where $\vec{L}$ is the light vector, $\vec{g}$ is the total blocker visibility vector, and $\vec{H}(N)$ is the irradiance weighting function given the surface normal $N$ [86]:

$$\vec{H}(N) = \frac{1}{\pi} \int_s \max(s \cdot N, 0)\, \vec{y}(s)\, ds \tag{5.14}$$

$(\vec{a}, \vec{b}, \vec{c})$ for three SH vectors $\vec{a}$, $\vec{b}$, and $\vec{c}$ denotes the integral of the product of the three reconstructed functions and is given by

$$(\vec{a}, \vec{b}, \vec{c}) = (\vec{a} * \vec{b}) \cdot \vec{c} = (\vec{b} * \vec{c}) \cdot \vec{a} = (\vec{c} * \vec{a}) \cdot \vec{b} = \sum_{ijk} \Gamma_{ijk}\, \vec{a}_i\, \vec{b}_j\, \vec{c}_k$$

The total visibility vector can also be used to shade other types of BRDFs [56] or textural detail [101, 104]: the vector $\vec{g} * \vec{L}$ represents shadowed incident radiance to apply to the receiver.

## 5.2.2 Spherical Harmonic Exponential

Let $\vec{f}$ be an SH vector to be exponentiated, and $\vec{g}$ be the result of this exponentiation. The *Volterra series* [96] allows any analytic, univariate, scalar function $h(x)$ (e.g. $h(x) = \exp(x)$) to be applied to an SH vector, or indeed any discrete function basis.

We begin with the integral formulation

$$h(\vec{f}) = \int_S h\,(f(s))\;\vec{y}(s)\,ds = \int_S h\left(\sum_i \vec{f}_i\,y_i(s)\right)\vec{y}(s)ds \tag{5.15}$$

which applies $h$ to the reconstructed function $f(s)$ at each spherical point $s$ and then projects the result to the vector $h(\vec{f})$. Substituting the Taylor expansion of the function $h(x)$

$$h(x) = h_0 + h_1\,x + h_2\,x^2 + h_3\,x^3 + \cdots\,, \tag{5.16}$$

we obtain the *SH power series*

$$h(\vec{f}) = h_0\,\vec{1} + h_1\,\vec{f^1} + h_2\,\vec{f^2} + h_3\,\vec{f^3} + \cdots \tag{5.17}$$

where

$$\begin{aligned}\vec{1} &= (\sqrt{4\pi}, 0, 0, \ldots, 0)\\ \vec{f^p} &= \int f^p(s)\,\vec{y}(s)ds = \int\left(\sum_i \vec{f}_i\,\vec{y}_i(s)\right)^p\,\vec{y}(s)ds.\end{aligned}$$

The vector $\vec{1}$ corresponds to a constant value of 1 over the sphere, and satisfies $\vec{1} * \vec{f} = \vec{f}$ for any $\vec{f}$. Degree $p$ powers of $\vec{f}$ can be written in terms of order-($p$+1) tensors $\Gamma$, via

$$(\vec{f^p})_i = \sum_{i_1,i_2,\ldots,i_p} \Gamma_{i,i_1,i_2,\ldots,i_p}\,\vec{f}_{i_1}\,\vec{f}_{i_2}\ldots\vec{f}_{i_p} \tag{5.18}$$

where the tensor $\Gamma$ represents the *Volterra kernel* (when scaled by $h_i$) and generalizes the triple product tensor we encountered before:

$$\Gamma_{i,i_1,i_2,\ldots,i_p} = \int_S \vec{y}_i(s)\,\vec{y}_{i_1}(s)\,\vec{y}_{i_2}(s)\;\ldots\;\vec{y}_{i_p}(s)\,ds \tag{5.19}$$

Numerical integration (5.15) or high-order tensors (5.17) are too expensive to evaluate on-the-fly. The result can be approximated by substituting repeated SH products for true SH powers in the series. This incurs approximation error because it truncates after each binary product. For example, $\vec{f}^3 \approx (\vec{f} * \vec{f}) * \vec{f}$ because the result of the first square $\vec{f} * \vec{f}$ is truncated back to an order-$n$ SH vector before multiplying by $\vec{f}$ again. Nevertheless the approximation is typically accurate, especially for vectors representing bandlimited visibility functions. Using repeated SH products, we obtain the following approximation called the *SH product series*, more practical for real-time evaluation:

$$\begin{aligned} h_*(\vec{f}) &= h_0\,\vec{1} + h_1\,\vec{f} + h_2\,\vec{f} * \vec{f} + h_3\,\vec{f} * \vec{f} * \vec{f} + \cdots \\ &= h_0\,\vec{1} + h_1\,\vec{f} + h_2\,\vec{f}^{2*} + h_3\,\vec{f}^{3*} + \cdots \end{aligned} \tag{5.20}$$

We use the notation

$$\vec{f}^{p*} = \underbrace{\vec{f} * \vec{f} * \cdots * \vec{f}}_{\text{repeated } p \text{ times}}$$

and note that $\vec{f}^p \approx \vec{f}^{p*}$. For $p > 3$, product order matters; we assume the product is amassed from left to right.

Now applying the Volterra series using the Taylor expansion for $h(x) = \exp(x)$ in (5.20), we obtain the product series

$$\exp(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \cdots \tag{5.21}$$

$$\exp_*(\vec{f}) = \vec{1} + \vec{f} + \frac{\vec{f}^{2*}}{2} + \frac{\vec{f}^{3*}}{3!} + \cdots \tag{5.22}$$

**Product Series Approximation**

For a finite number of terms in (5.22), approximation error increases as $\|\vec{f}\|$ increases, just as it does in (5.21) as $|x|$ increases. For this reason, evaluation techniques try to reduce the magnitude of the input vector $\vec{f}$ and thereby increase accuracy for a fixed number of terms. Another technique factors the series to reduce the number of SH products. These techniques are analogous to ones used for the matrix exponential [41]. In fact, computing the exponential of an SH vector and a matrix are related since

$$\mathsf{M}_{f^{p*}} \approx (\mathsf{M}_f)^p \quad \Rightarrow \quad \mathsf{M}_{\exp_*(f)} \approx \exp\left(\mathsf{M}_f\right). \tag{5.23}$$

**DC Isolation** We express $\vec{f}$ as the sum of its DC component plus its remaining components, or $\vec{f} = \hat{\vec{f}} + \frac{\vec{f}_0}{\sqrt{4\pi}}\vec{1}$. $\hat{\vec{f}}$ simply zeroes out the DC component of $\vec{f}$, i.e.,

$$\hat{\vec{f}} = (0, \vec{f}_1, \vec{f}_2, \ldots, \vec{f}_{n^2-1}) = \vec{f} - (\vec{f} \cdot \vec{1})\vec{1}, \tag{5.24}$$

Then (5.22) becomes

$$\exp_*(\vec{f}) = \exp\left(\frac{\vec{f}_0}{\sqrt{4\pi}}\right) \exp_*(\hat{\vec{f}}) \tag{5.25}$$

which is easily derived since $\vec{f} * \vec{1} = \vec{f}$.

(5.25) *analytically* computes the exponential of the DC component, reducing the magnitude of the residual vector $\hat{\vec{f}}$. This vector is then exponentiated using the series method augmented by additional techniques described in the following.

**Scaling/Squaring** We also make use of a technique used in evaluating scalar (and matrix) exponentials, which observes that

$$\exp(x) = \left(\exp\left(\frac{x}{2^p}\right)\right)^{2^p} \quad \Rightarrow \quad \exp_*(\vec{f}) \approx \left(\exp_*\left(\frac{\vec{f}}{2^p}\right)\right)^{2^p *} \tag{5.26}$$

where $p$ is a positive integer. In other words, to compute $\exp(x)$ we first divide the input $x$ by a power of 2, compute the exponential of this scaled input, and finally repeatedly square the result $p$ times. The same idea can be applied to SH exponentiation using $p$ repeated squarings via the recurrence $\vec{f}^{2^p *} = \vec{f}^{2^{p-1} *} * \vec{f}^{2^{p-1} *}$.

(5.26) only approximates the product series in (5.22), but typically reduces error relative to the power series in (5.17). SH squares are also cheaper than general SH products, making this approximation more useful. We choose $p$ as a function of $\|\vec{f}\|$ using $p = \max(0, \lfloor \log_2 \|\vec{f}\| + 3 \rfloor)$. At most $p$=3 squarings are needed for low-order ($n \leq 6$) SH vectors in our examples.

**Factoring** (5.22) can be evaluated by accumulating successively higher powers of $\vec{f}$ via $\vec{f}^{p+1} = \vec{f}^{p} * \vec{f}$. This requires $p$-1 SH products for a degree $p$ expansion. The number of SH products can be reduced by segregating even and odd powers. (5.20) becomes

$$\begin{aligned} h_*(\vec{f}) \quad \approx \quad & \left(h_0\,\vec{1} + h_2\,\vec{f}^{2*} + h_4\,\vec{f}^{4*} + \cdots\right) + \\ & \vec{f} * \left(h_1\,\vec{1} + h_3\,\vec{f}^{2*} + h_5\,\vec{f}^{4*} + \cdots\right) \end{aligned} \tag{5.27}$$

improving to only ($p$+1)/2 products for degree $p$ expansion. Furthermore, fewer products implies smaller truncation error and thus a better approximation to (5.17). Powers of $\vec{f}$ should be computed so as to minimize the number of products in each term: $\vec{f}^{2*} = \vec{f} * \vec{f}$, $\vec{f}^{4*} = \vec{f}^{2*} * \vec{f}^{2*}$, $\vec{f}^{6*} = \vec{f}^{4*} * \vec{f}^{2*}$, and so on. Even better factorings can be obtained for series degree $p > 12$ [41].

We compare a number of algorithms for evaluating the SH exponential. PS-$p$ uses the simple product series evaluation of degree $p$ from (5.22). PS*-$p$ uses DC isolation (5.25) and scaling/squaring (5.26) applied to a factored degree-$p$ product series from (5.27). We also described in the original paper [119] an optimal linear approximation to the product series denoted as OL. We also extend the accuracy of this method via a hybrid method, called HYB, which applies scaling/squaring to the optimal linear method.

### 5.2.3 Spherical Harmonic Logarithm

A naive method for SH log applies (5.15):

$$\vec{f} = \log(\vec{g}) = \int_S \log\left(\max(g(s), \epsilon)\right)\ \vec{y}(s)\, ds \tag{5.28}$$

where we clip evaluations of $g(s)$ that are close to 0 or negative using a small threshold $\epsilon$. This method works poorly for two reasons. First, it neglects how truncation error from taking log affects the subsequent exponential. Substantial error $\|\exp_*(\log(\vec{g})) - \vec{g}\|$ results, manifested as attenuation of frequencies near the Nyquist band. Second, clipping to a constant introduces artificial high frequencies and so suboptimally picks a signal that's close to the original, $g(s)$, but avoids places where log is undefined.

We address both these problems by approximately inverting the exponential using an eigenanalysis of the product matrix $\mathsf{M}_g$. Inverting the $\exp_*$ operator takes truncation into account, reducing high-frequency attenuation after the final exponential. Clipping in the space of eigenvalues of $\mathsf{M}_g$ rather than sampled spherical values $g(s)$ eliminates artificially-introduced high frequencies.

#### Diagonalizing SH Exp

Rewriting (5.22) in terms of the product matrix $\mathsf{M}_f$, we obtain

$$\vec{g} = \exp_*(\vec{f}) = \vec{1} + \vec{f} + \frac{\mathsf{M}_f\, \vec{f}}{2!} + \frac{\mathsf{M}_f^2\, \vec{f}}{3!} + \cdots . \tag{5.29}$$

Then eigenanalysis on the symmetric product matrix yields

$$\mathsf{M}_f = \mathsf{R}_f^T\, \mathsf{D}_f\, \mathsf{R}_f \quad \Rightarrow \quad (\mathsf{M}_f)^p = \mathsf{R}_f^T\, \mathsf{D}_f^p\, \mathsf{R}_f \tag{5.30}$$

where $\mathsf{R}_f$ is a rotation matrix, $\mathsf{D}_f$ is a diagonal matrix, and powers of $\mathsf{D}$ are taken with respect to each of its diagonal components. Substituting these powers, (5.29) becomes

$$\vec{g} = \exp_*(\vec{f}) = \vec{1} + \mathsf{R}_f^T\, q(\mathsf{D}_f)\, \mathsf{R}_f\, \vec{f} \tag{5.31}$$

$$q(x) = 1 + \frac{x}{2!} + \frac{x^2}{3!} + \cdots = \frac{\exp(x) - 1}{x} \tag{5.32}$$

where $q$ is applied to each diagonal element of $\mathsf{D}_f$. Note that (5.31) represents an alternative method of evaluating SH exponential which avoids error from truncating to a finite series. However, it is not practical to compute the required eigenanalysis on-the-fly, and we use this formulation only to derive a method for SH log, which is computed as a preprocess.

**Inverting SH Exp**

We begin with an eigenanalysis of the product matrix of $\vec{g}$, $\mathsf{M}_g = \mathsf{R}_g^T\, \mathsf{D}_g\, \mathsf{R}_g$. (5.23) then implies that $\mathsf{M}_{\log(g)} \approx \mathsf{R}_g^T\, \log(\mathsf{D}_g)\, \mathsf{R}_g$ for positive definite product matrices $\mathsf{M}_g$. (5.31) can therefore be approximately inverted using

$$\log(\vec{g}) = \mathsf{R}_g^T\, q'(\mathsf{D}_g)\, \mathsf{R}_g\, (\vec{g} - \vec{1}) \tag{5.33}$$

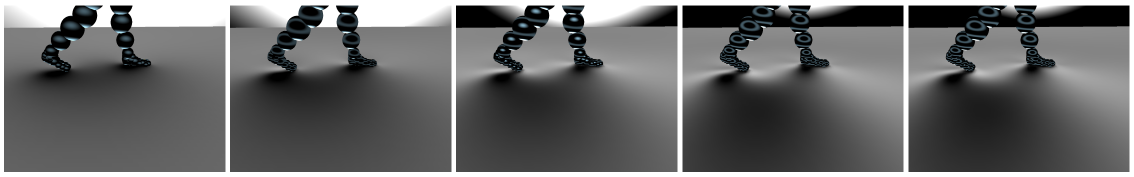
$$q'(x) = 1/q(\log(x)) = \log(x)/(x-1) \tag{5.34}$$

where the function $q'$ is applied to each diagonal component.

To avoid applying log to values that are negative or close to 0, we clip the eigenvalues of $\mathsf{M}_g$ via

$$\tilde{\mathsf{D}}_g = \max(\mathsf{D}_g, \epsilon), \quad \tilde{\mathsf{M}}_g = \mathsf{R}_g^T\, \tilde{\mathsf{D}}_g\, \mathsf{R}_g \tag{5.35}$$

and apply (5.33) to $\tilde{\mathsf{D}}_g$ rather than $\mathsf{D}_g$. Eigenvalue clipping yields smaller error



(a) $n$=3, 1.9x (b) $n$=4, 3.4x (c) $n$=5, 6.5x (d) $n$=6, 11.0x (e) $n$=6, product

Figure 5.5: *SH order comparison. The "walking man" model contains $n_S$=60 spheres.*

$||\mathsf{M}_g - \tilde{\mathsf{M}}_g||$ compared with clipping values of $g(s)$ over the sphere as in (5.28). In practice, we have found that setting the threshold $\epsilon$ to 0.02 times the largest eigenvalue works well for low-order SH vectors.

### 5.2.4 Implementation and Results

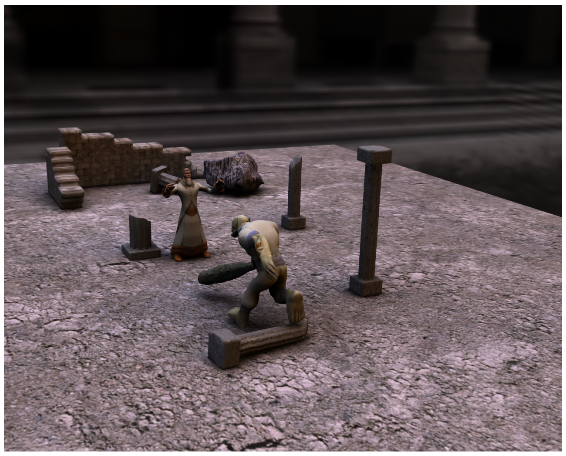
A single sphere's blocking function is easily represented as the SH visibility function for circles of angular radius $\theta$

$$g(s,\theta) = \begin{cases} 0, & \text{if } s \cdot (0,0,1) \geq \cos(\theta); \\ 1, & \text{otherwise.} \end{cases} \tag{5.36}$$

As a preprocess, we approximate the real geometry using sets of bounding spheres. Our approximation algorithm applies variational shape approximation [24] to the problem of bounding geometry within a set of spheres.

For diffuse surfaces in lighting environments, we tabulate $\vec{L}_H(N) = \vec{L} * \vec{H}(N)$ where $\vec{L}$ represents the lighting and $\vec{H}(N)$ is defined in (5.14). The result is then dotted with the exponentiated blocker vector $\vec{g}$ from (5.13) to produce the shadowed result, $\vec{L}_H(N_p) \cdot \vec{g}$. For static receiver points, local shadowing effects can be "baked in" by dotting with the precomputed vector $\vec{H}(N_p) * \vec{g}_p$ where $\vec{g}_p$ represents local visibility due to static occluders.

Timings were performed on a 3.2Ghz PC with 1Gb of memory and an NVidia 7800GTX graphics card. Figure 5.5 compares shadowing at various SH orders $n$.


(a) battle scene, 23.2Hz

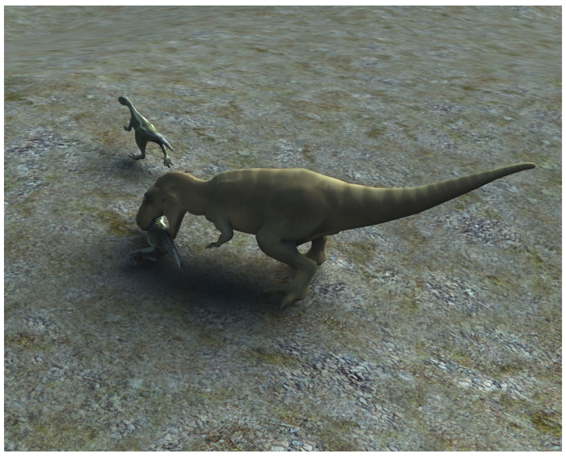

(b) dino scene, 22.4Hz

Figure 5.6: *Images from GPU rendering. Frame rate for the particular image is shown.*

The factor below each log space image represents total rendering speedup observed in a CPU implementation, when accumulating blockers in log space and applying PS*-2 rather than accumulating products. Approximation error from doing the computation in log space is difficult to see; Figure 5.6 presents images rendered on the GPU from two, more complex scenarios. SH order $n$=4 is used. We obtain good performance (10-30Hz) and high-quality soft shadows. The battle scene involves two characters (troll and wizard) and contains 65k vertices (41k static and 24k dynamic) and 244 leaf node sphere blockers. 120 receiver clusters were used: 100 for the ground plane, 8 each for the wizard and troll, 4 for the troll's club, and 1 each for each of the other objects (rock and columns). Frame rate was measured between 14.3 and 32.8Hz and averaged 22.4Hz. The dino scene contains a sequence of different clips. The most complicated, running at 10-12Hz, contains 8 moving dinosaurs, 120k vertices (75k static and 45k dynamic), 500 blocker leaf node spheres and 256 receiver clusters (192 for the ground surface and 8 for each dinosaur). Frame rate over all clips was measured between 10.1 and 26.3Hz and averaged 12.6Hz.

## 5.3 Discussion of Frequency Based Approaches

In Sec. 5.1, we have developed a comprehensive theoretical framework for normal map filtering with many common types of reflectance models. Our method is based on a new analytic formulation of normal map filtering as a convolution of the NDF and BRDF. The algorithms are simple enough to be implemented as GPU pixel shaders, enabling real-time rendering on graphics hardware. The convolution result unifies a geometric problem (normal mapping) with understanding of lighting and BRDF interaction in appearance. In [50], a hierarchy of level-of-details was spelled out including explicit 3D geometry, normal or bump maps, and BRDF or reflectance. Our method has addressed filtering of normal maps and to some extent, the transition to a BRDF at far distances. A critical direction for future work is filtering of geometry or displacement maps, where effects like local occlusions, shadowing, masking and interreflections are important.

In Sec. 5.2, we have presented a novel dynamic soft shadowing algorithm using spherical harmonic exponentials. Accumulating low-frequency blocker visibility in the spherical harmonic basis provides a direct method for rendering soft shadows without integrating over a huge number of lighting directions. We accelerate this approach by accumulating in log space rather than product space, and then computing the SH exponential required using new methods (HYB and PS*-$p$). Per-blocker computation is greatly reduced, allowing us to handle more blockers and to map the computation to the GPU in a single shading pass. Future work can extend to anisotropic blocker models, handling diffuse inter-reflection, and experimenting with alternative models for spatial shading variation.

# Chapter 6

# Conclusion and Future Work

This thesis (and related published works [111, 92, 36, 110]) has introduced important new analytical, wavelet and frequency based methods for four challenging rendering problems: single scattering, near-field relighting, normal map filtering and dynamic soft shadowing. In Chap. 3, we presented a practical analytic single scattering model for real-time rendering. Our analytical model easily achieves real-time performance while maintaining the ease of use of the standard OpenGL model. Our approach can be easily implemented in programmable graphics hardware and leads to a number of new effects in the real-time domain, such as interactive rendering with glows around light sources, the effects of scattering on surface shading, environment maps, and precomputed light transport. In Chap. 4, we introduced a novel theory of affine double and triple product wavelet integrals for near-field relighting. Our theory overturns the long-held beliefs that operations such as affine transformation are hard in wavelets, thus requiring converting to and from the pixel domain. Our work also significantly advanced the state of art in near-field relighting. We described two frequency based approaches in Chap. 5. We formulate the problem of normal map filtering as frequency domain convolution, leveraging a large body of mathematical developments on lighting-BRDF convolution. We also invented the spherical harmonic exponentiation technique to render soft shadows in dynamic scenes for skinned characters. All of our models are physically based, and can be easily implemented/reproduced from our publications.

My published work in this direction already made solid contributions to exploiting novel mathematical representations and faster numerical algorithms for interactive rendering. They have laid foundations for further investigations of better real-time rendering techniques and optimal models. Future work can follow two main directions. First, we should extend our presented ways of rendering scattering, lighting and shadowing effects to more general scenarios— scattering (inhomogeneous, time varying, subsurface), lighting (light field, indirect illumination), and shadowing (deformable objects, complex geometries). Second, we strive to build complete suites of frequency and wavelet domain operations, combining analytic derivations with computational methods. This line of investigation will hold the promise for a complete rendering framework in the frequency and wavelet domain. Further, our derived models, such as affine double and triple product integrals, should be tested for more applications beyond interactive rendering, such as video processing, tracking, and recognition.

## 6.1 General Interactive Effects

Sacrificing some generality is often a worthwhile tradeoff to enable simple techniques to achieve real-time rates. It is one of our main goals for future research to generalize these techniques to more complicated scenarios. As presented in this thesis, we developed a practical analytical single scattering model [111] but only considered isotropic point light sources, single scattering, homogeneous media, and excluding most cast and volumetric shadowing. For future research, it will be worth extending our theoretical model to non-isotropic light sources (like spotlights) because many non-isotropic sources can be described using analytic models. Secondary scattering of surface reflected radiance to the eye can be included using depth-dependent convolution. If the inhomogeneous media field can described by analytic models, it will be also worth conducing research on extending our models to inhomogeneous and possibly time-varying media. One can also experiment combining our analytical model with the popular dipole model for subsurface scattering.

In Chap. 4, we presented a novel affine double and triple product wavelet integral theory and applied it to near-field relighting. Immediate work can follow several avenues. We used a simple constant light heuristics for lighting compression. Better heuristics should be experimented. We also limited ourselves to simple planar area light sources and did not address general rotations. We, however, show theoretically in [110] that relighting with light fields can be formulated using our model. General rotations of light sources can be achieved using wavelet rotation [119]. For more near-field effects, a number of interesting areas are worth exploring: near-field scattering effects, relighting with non-planar light sources, and dynamic near-field shadows for deformable objects.

In Chap. 5, we presented two frequency based approaches. We first invented spherical harmonic exponentiation technique for dynamic shadowing. Direct future work can extend to anisotropic blocker models using ellipsoidal blocking functions, handling diffuse inter-reflection with precomputed global light transport. Finally, we proposed frequency domain normal map filtering. A future direction is filtering of geometry or displacement maps, where effects like local occlusions, shadowing, masking and interreflections are important.

Our preliminary work in these areas indicates that some of these generalizations, while relatively simple in concept and widely implemented in offline renderers, are rather non-trivial to incorporate in real-time rendering. For example, near-field lighting effects require efficient affine transform of signals in the wavelet basis, which was widely believed to be difficult. However, we are optimistic that with the development of more powerful mathematical representations and rapid advancement of hardware capabilities, problems will become more tractable with some challenges being conquered in the near future.

### 6.1.1 Development of Mathematical Representations

A compact and powerful mathematical representation is the key to real-time rendering, often speeding up the performance by many orders of magnitude. As in the development of the single scattering model, our key insight is a new analytic model for integrating the light transport equations assuming single scattering, which can also be extended to predict the impact of scattering or airlight on the inherent appearance of surfaces. Analytic models are also used for subsurface scattering effects [49]. In general, we believe that analytic models of difficult to simulate volumetric phenomena are critical to achieving efficient renderings for real-time applications.

Wavelet and frequency analysis can provide new insights and reveal the key structures of the problem. Much of the recent development in relighting and PRT methods have relied on insights derived from frequency and wavelet analysis: the lighting-BRDF convolution reduces to a dot product in the frequency or wavelet domain [86, 91, 76], high dimensional functions such as the lighting, BRDF and visibility can be more efficiently represented in spherical harmonic or wavelet bases [88, 77], the light transport can be analyzed in the frequency domain [31]. We have also leveraged recent results of frequency analysisand applied to normal map filtering. Our method is based on a new analytic formulation of normal map filtering as a convolution of the NDF and BRDF. The algorithms are simple enough to be implemented as GPU pixel shaders, enabling real-time rendering on graphics hardware. As a result, efficient and flexible operations in the wavelet and frequency domain become critical. For example, in wavelets affine transformation was conventionally considered as very difficulty and required converting to and from the pixel domain. We have presented a novel theory of affine double and triple product integrals that efficiently computes affine transforms in Haar wavelets. In spherical harmonics, we developed spherical harmonic exponentiation technique which converts prohibitive multiplications to inexpensive additions. However, operations in the wavelet and frequency domain are far from complete, e.g., geometric contraction or shears, trigonometric functions. Our work reveals the fundamental sparsity of some basic operations (dilation, translation and exponentation) and opens up a fresh perspective in approaching many other operations that have traditionally been viewed as difficult to compute directly with wavelets and spherical harmonics. Together with standard double and triple product integrals (multiplication), more complicated operators can be built, e.g. convolutions can be reduced to translations and multiplications. The basic mechanism and computational machinery we employed in these works will shed key insights on constructing a more complete suite of wavelet and frequency domain operations.

Our developed models also have broader applications. For example, our affine double and triple product integral theory can be widely applied to many other applications that depend on wavelet representations. As a proof of concept, we demonstrated initial results of wavelet importance sampling with near-field lights for offline Monte Carlo renderers. We also show the practical utility of our theory in dilating and translating images directly in the wavelet domain for image processing. However, we believe in this thesis we have only scratched the surface of many potential applications, and we predict many greater application and further developments of these mathematical models in computer graphics, vision, and applied mathematics.

# Appendix A

# Bibliography

[1] M. Agrawala, R. Ramamoorthi, A. Heirich, and L. Moll. Efficient image-based methods for rendering soft shadows. In *Proc. of SIGGRAPH '00*, pages 375–384, 2000.

[2] T. Annen, J. Kautz, F. Durand, and H. Seidel. Spherical harmonic gradients for mid-range illumination. In *Eurographics Symposium on Rendering*, 2004.

[3] M. Ashikhmin and P. Shirley. An anisotropic Phong BRDF model. *Journal of Graphics Tools: JGT*, 5(2):25–32, 2000.

[4] U. Assarsson and T. Akenine-Möller. A geometry-based soft shadow algorithm using graphics hardware. In *Proc. of SIGGRAPH '03*, pages 511–520, 2003.

[5] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.

[6] R. Basri and D. Jacobs. Lambertian reflectance and linear subspaces. In *International Conference on Computer Vision*, pages 383–390, 2001.

[7] R. Basri and D. Jacobs. Photometric stereo with general, unknown lighting. In *Computer Visiona and Pattern Recognition 01*, pages II–374–II–381, 2001.

[8] B. Becker and N. Max. Smooth transitions between bump rendering algorithms. In *Proc. of SIGGRAPH 93*, pages 183–190, 1993.

[9] G. Beylkin. On the fast algorithm for multiplication of functions in the wavelet bases. In *International Conference on Wavelets and Applications*, pages 53–61, 1992.

[10] G. Beylkin. On the representation of operators in bases of compactly supported wavelets. *SIAM Journal on Numerical Analysis*, 29(6):1716–1740, 1992.

[11] V. Biri, S. Michelin, and D. Arques. Real-time single scattering with shadows. In *In review http://igm.univ-mlv.fr/˜biri/indexCA_en.html*, 2004.

[12] J. Blinn. Models of light reflection for computer synthesized pictures. *SIGGRAPH Comput. Graph.*, 11(2):192–198, 1977.

[13] J. Blinn. Simulation of wrinkled surfaces. In *SIGGRAPH 78*, pages 286–292, 1978.

[14] J. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. In *Computer Graphics(Proceedings of ACM SIGGRAPH 82)*, pages 21–29. ACM, 1982.

[15] J. Blinn and M. E. Newell. Texture and reflection in computer generated images. *Commun. ACM*, 19(10):542–547, 1976.

[16] R. N. Bracewell, K.-Y. Chang, A. K. Jha, and Y.-H. Wang. Affine theorem for two-dimensional fourier transform. *Electronics Letters*, 29(3):304, 1993.

[17] M. Bunnell. Dynamic ambient occlusion and indirect lighting. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*, pages 223–233. Addison-Weseley Professional, 2004.

[18] B. Cabral, N. Max, and R. Springmeyer. Bidirectional reflection functions from surface bump maps. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 273–281, New York, NY, USA, 1987. ACM.

[19] J. Chai, X. Tong, and H. Shum. Plenoptic sampling. In *Proc. of SIGGRAPH 00*, pages 307–318, 2000.

[20] S. Chandrasekhar. *Radiative Transfer*. Oxford Univ. Press, 1960.

[21] C. K. Chui. *An Introduction to Wavelets*. Academic Press, 1992.

[22] P. Clarberg, W. Jarosz, T. Akenine-Möller, and H. W. Jensen. Wavelet importance sampling: Efficiently evaluating products of complex functions. *ACM Transactions on Graphics (SIGGRAPH 05 Proceedings)*, 24(3):1166–1175, 2005.

[23] J. Cohen, M. Olano, and D. Manocha. Appearance preserving simplification. In *SIGGRAPH 98*, pages 115–122, 1998.

[24] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. In *Proc. SIGGRAPH '04*, pages 905–914, 2004.

[25] K.J. Dana, B. van Ginneken, S.K. Nayar, and J.J. Koenderink. Reflectance and Texture of Real World Surfaces. Technical report, 1996.

[26] P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. *Computer Graphics*, 32(Annual Conference Series):189–198, 1998.

[27] P. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH 97*, pages 369–378, 1997.

[28] Y. Dobashi, T. Yamamoto, and T. Nishita. Interactive rendering of atmospheric scattering effects using graphics hardware. In *Graphics Hardware Workshop 02*, pages 99–109, 2002.

[29] J. Dorsey, J. Arvo, and D. Greenberg. Interactive design of complex time-dependent lighting. *IEEE Computer Graphics and Applications*, 15(2):26–36, 1995.

[30] I. Drori and D. Lischinski. Fast multiresolution image operations in the wavelet domain. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):395–411, 2003.

[31] F. Durand, N. Holzschuch, C. Soler, E. Chan, and F. X. Sillion. A frequency analysis of light transport. *ACM Transactions on Graphics (SIGGRAPH 05 Proceedings)*, 24(3):1115–1126, 2005.

[32] A. Fournier. Normal distribution functions and multiple surfaces. In *Graphics Interface Workshop on Local Illumination*, pages 45–52, 1992.

[33] A. Fournier. Separating reflection functions for linear radiosity. Technical report, Vancouver, BC, Canada, Canada, 1995.

[34] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile. Modeling the interaction of light between diffuse surfaces. *SIGGRAPH Comput. Graph.*, 18(3):213–222, 1984.

[35] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The lumigraph. In *SIGGRAPH 96*, pages 43–54, 1996.

[36] C. Han, B. Sun, R. Ramamoorthi, and E. Grinspun. Frequency domain normal map filtering. In *AMC SIGGRAPH*, 2007.

[37] P. Hanrahan and W. Krueger. Reflection from layered surfaces due to subsurface scattering. In *Proceedings of ACM SIGGRAPH 1993, ACM Press/ACM SIGGRAPH, New York. E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series*, pages 165–174. ACM, 1993.

[38] M. Harris and A. Lastra. Real-time cloud rendering. In *Eurographics 2001*, pages 76–84, 2001.

[39] P. S. Heckbert. *Simulating Global Illumination Using Adaptive Meshing*. PhD thesis, EECS Department, University of California, Berkeley, Jun 1991.

[40] Wolfgang Heidrich and Hans P. Seidel. Realistic, hardware-accelerated shading and lighting. In *SIGGRAPH 99*, pages 171–178, 1999.

[41] N. Higham. The scaling and squaring method for the matrix exponential revisited. In *SIAM Journal of Matrix Analysis Applications*, number 4, pages 1179–1193, 2005.

[42] N. Hoffman and A. J. Preetham. Real-time light-atmosphere interactions for outdoor scenes. *Graphics programming methods*, pages 337–352, 2003.

[43] B. K. P. Horn. Extended gaussian images. *Proceedings of the IEEE*, 72:1671–1686, 1984.

[44] D. S. Immel, M. F. Cohen, and D. P. Greenberg. A radiosity method for non-diffuse environments. *SIGGRAPH Comput. Graph.*, 20(4):133–142, 1986.

[45] A. Ishimaru. *Wave Propagation and Scattering in Random Media. Volume 1: Single Scattering and Transport Theory*. Academic Press, 1978.

[46] D. James and K. Fatahalian. Precomputing interactive dynamic deformable scenes. In *Proc. of SIGGRAPH '03*, pages 879–887, 2003.

[47] H. W. Jensen. Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30, New York, NY, 1996. Springer-Verlag/Wien.

[48] H. W. Jensen. *Realistic Image Synthesis Using Photon Mapping*. AK Peters, 2001.

[49] H. W. Jensen, S. Marschner, M. Levoy, and P. Hanrahan. A practical model for subsurface light transport. In *Proceedings of ACM SIGGRAPH 2001, ACM Press/ACM SIGGRAPH, New York, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series*, pages 511–518. ACM, 2001.

[50] J. Kajiya. Anisotropic reflection models. In *SIGGRAPH 85*, pages 15–21, 1985.

[51] J. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, 1986.

[52] J. Kajiya and B. Herzen. Ray tracing volume densities. In *Proceedings of ACM SIGGRAPH 1984, ACM Press/ACM SIGGRAPH, New York. E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series*, pages 165–174. ACM, 1984.

[53] J. Kajiya and T. Kay. Rendering fur with three dimensional textures. In *SIGGRAPH 89*, pages 271–280, 1989.

[54] J. Kautz, J. Lehtinen, and T. Aila. Hemispherical rasterization for self-shadowing of dynamic objects. *Proceedings Eurographics Symposium on Rendering*, pages 179–184, 2004.

[55] J. Kautz and M. D. McCool. Interactive rendering with arbitrary BRDFs using separable approximations. pages 253–253, 1999.

[56] J. Kautz, P. Sloan, and J. Snyder. Fast, arbitrary brdf shading for low-frequency lighting using spherical harmonics. *Eurographics Workshop on Rendering*, pages 291–296, 2002.

[57] A. G. Kirk and O. Arikan. Real-time ambient occlusion for dynamic character skins. In *I3D '07: Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 47–52, New York, NY, USA, 2007. ACM.

[58] J. Kontkanen and S. Laine. Ambient occlusion fields. In *Proc. of Interactive 3D Graphics and Games (I3D)*, 2005.

[59] H. Koschmeider. Theorie der horizontalen sichtweite. *Beitr. Phys. freien Atm.*,, 12, 1924.

[60] A. Kristensen, T.Akenine-Moller, and H. Jensen. Precomputed local radiance transfer for real-time lighting design. In *AMC SIGGRAPH*, 2005.

[61] R. Kutil. Approximating linear image operations in the wavelet domain. Technical Report 2005-08, Department of Scientific Computing, University of Salzburg, Austria, 2005.

[62] E. Lafortune, S. Foo, K. Torrance, and D. Greenberg. Nonlinear approximation of reflectance functions. In *Proc. of SIGGRAPH '97*, pages 117–126, 1997.

[63] S. Laine, T. Aila, U. Assarsson, J. Lehtinen, and T. Akenine-Möller. Soft shadow volumes for ray tracing. In *Proc. of SIGGRAPH '05*, pages 1156–1165, 2005.

[64] M. Levoy and P. Hanrahan. Light field rendering. In *Prof. of SIGGRAPH 96*, pages 31–42, 1996.

[65] X. Liu, P. J. Sloan, H. Shum, and J. Snyder. All-frequency precomputed radiance transfer for glossy objects. In *EuroGraphics Symposium on Rendering 04*, pages 337–344, 2004.

[66] W. Matusik, H. Pfister, M. Brand, and L. McMillan. A data-driven reflectance model. *ACM Transactions on Graphics*, 22(3):759–769, 2003.

[67] N. L. Max. Atmospheric illumination and shadows. In *Computer Graphics(Proceedings of ACM SIGGRAPH 86)*, pages 117–124. ACM, 1986.

[68] N. L. Max. Efficient light propagation for multiple anisotropic volume scattering. In *Eurographics Rendering Workshop 94*, pages 87–104, 1994.

[69] M. D. McCool, J. Ang, and A. Ahmad. Homomorphic factorization of brdfs for high-performance rendering. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 171–178, New York, NY, USA, 2001. ACM.

[70] C. Mei, J. Shi, and F. Wu. Rendering with spherical radiance transport maps. In *Eurographics '04*, pages 281–290, 2004.

[71] G. S. Miller and C. R. Hoffman. Illumination and reflection maps: Simulated objects in simulated and real environments, 1984.

[72] E. Nakamae, K. Kaneda, T. Okamoto, and T. Nishita. A lighting model aiming at drive simulators. In *Computer Graphics(Proceedings of ACM SIGGRAPH 90)*, pages 395–404, 1990.

[73] S. Narasimhan and S. Nayar. Vision and the atmosphere. *Interntiaon Journal on Computer Vision*, 48(3):233–254, August 2002.

[74] S. Narasimhan and S. Nayar. Shedding light on the weather. In *CVPR 03*, pages 665–672, 2003.

[75] R. Ng, R. Ramamoorthi, and P. Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. In *Proc. of SIGGRAPH '03*, pages 376–381, 2003.

[76] R. Ng, R. Ramamoorthi, and P. Hanrahan. Triple product wavelet integrals for all-frequency relighting. *ACM Transactions on Graphics (SIGGRAPH 2004)*, 23(3):475–485, 2004.

[77] Ren Ng. Fourier slice photography. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 735–744, New York, NY, USA, 2005. ACM Press.

[78] F. E. Nicodemus. Directional reflectance and emissivity of an opaque surface. *Appl. Opt.*, 4(7):767, 1965.

[79] J. S. Nimeroff, E. Simoncelli, and J. Dorsey. Efficient Re-rendering of Naturally Illuminated Environments. In *Eurographics Rendering Workshop 1994*, pages 359–373, 1994.

[80] T. Nishita and E. Nakamae. A shading model for atmospheric scattering considering luminous intensity distribution of light sources. In *Computer Graphics(Proceedings of ACM SIGGRAPH 1987)*, pages 303 – 310. ACM, 1987.

[81] M. Olano and M. North. Normal distribution mapping. Technical Report 97-041 http://www.cs.unc.edu/˜olano/papers/ndm/ndm.pdf, UNC, 1997.

[82] M. Oren and S.K. Nayar. Generalization of Lambert's Reflectance Model. In *ACM 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 239–246, Jul 1994.

[83] S. Pattanaik and S. Mudur. Computation of global illumination in a participating medium by monte carlo simulation. *Journal of Visualization and Computer Animation*, 4(3):133–152, 1993.

[84] A. J. Preetham, P. Shirley, and B. Smits. A practical analytic model for daylight. In *Proceedings of ACM SIGGRAPH 1999, ACM Press/ACM SIGGRAPH, New York. E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series*, pages 91–100. ACM, 1999.

[85] S. Premoze, M. Ashikhmin, J. Tesendorf, R. Ramamoorthi, and S. Nayar. Practical rendering of multiple scattering effects in participating media. In *EuroGraphics Symposium on Rendering 04*, pages 363–374, 2004.

[86] R. Ramamoorthi and P. Hanrahan. An efficient representation for irradiance environment maps. In *Proc. of SIGGRAPH '01*, pages 497–500, 2001.

[87] R. Ramamoorthi and P. Hanrahan. A signal-processing framework for inverse rendering. In *Proceedings of ACM SIGGRAPH 2001, ACM Press/ACM SIGGRAPH, New York. E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series*, pages 117–128. ACM, 2001.

[88] R. Ramamoorthi and P. Hanrahan. Frequency space environment map rendering. *ACM Transactions on Graphics (SIGGRAPH 02 proceedings)*, 21(3):517–526, 2002.

[89] R. Ramamoorthi, D. Mahajan, and P. Belhumeur. A first order analysis of lighting, shading, and shadows. *ACM Transactions on Graphics*, 26(1):2, 2007.

[90] Ravi Ramamoorthi and Pat Hanrahan. Analysis of planar light fields from homogeneous convex curved surfaces under distant illumination. In *SPIE Photonics West: Human Vision and Electronic Imaging VI*, pages 185–198, 2001.

[91] Ravi Ramamoorthi and Pat Hanrahan. On the relationship between radiance and irradiance: Determining the illumination from images of a convex lambertian object. *JOSA A*, 18(10):2448–2459, 2001.

[92] Z. Ren, R. Wang, J. Snyder, K. Zhou, X. Liu, B. Sun, P. J. Sloan, H. Bao, Q. Peng, and B. Guo. Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. *ACM Transactions on Graphics (SIGGRAPH 06 Proceedings)*, 25(3):977–986, 2006.

[93] K. Riley, D. Ebert, M. Kraus, J. Tessendorf, and C. Hansen. Efficient rendering of atmospheric phenomena. In *EuroGraphics Symposium on Rendering 2004*, pages 375–386, 2004.

[94] H. Rushmeier and K. Torrance. The zonal method for calculating light intensities in the presence of a participating medium. In *Proceedings of ACM SIGGRAPH 1987, ACM Press/ACM SIGGRAPH, New York. E. Fiume, Ed., Computer Graphics Proceedings*, pages 293–302. ACM, 1987.

[95] G. Sakas. Fast rendering of arbitrary distributed volume densities. In *Eurographics 90*, pages 519–530, 1990.

[96] M. Schetzen. *The Volterra and Wiener Theories of Nonlinear Systems*. John Wiley and Sons, 1980.

[97] A. Schilling. Toward real-time photorealistic rendering: Challenges and solutions. In *SIGGRAPH/Eurographics Workshop on Graphics Hardware*, pages 7–16, 1997.

[98] P. Schröder and W. Sweldens. Spherical wavelets: efficiently representing functions on the sphere. *Computer Graphics*, 29(Annual Conference Series):161–172, 1995.

[99] M. Segal, C. Korobkin, R. Van Widenfelt, J. Foran, and P. Haeberli. Fast shadows and lighting effects using texture mapping. In *Proc. of SIGGRAPH '92*, pages 249–252, 1992.

[100] P. Shanmugam and O. Arikan. Hardware accelerated ambient occlusion techniques on gpus. In *I3D '07: Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 73–80, New York, NY, USA, 2007. ACM.

[101] P. Sloan, J. Hall, J. Hart, and J. Snyder. Clustered principal components for precomputed radiance transfer. In *Proc. of SIGGRAPH '03*, pages 382–391, 2003.

[102] P. Sloan, J. Kautz, and J. Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics*, 21(3):527–536, 2002.

[103] P. Sloan, X. Liu, H. Shum, and J. Snyder. Bi-scale radiance transfer. In *Proc. of SIGGRAPH '03*, pages 370–375, 2003.

[104] P. Sloan, B. Luna, and J. Snyder. Local, deformable precomputed radiance transfer. In *Proc. of SIGGRAPH '05*, pages 1216–1224, 2005.

[105] C. Soler and F. Sillion. Fast calculation of soft shadow textures using convolution. In *Proc. of SIGGRAPH '98*, pages 321–332, 1998.

[106] J. Stam. Multiple scattering as a diffusion process. In *Eurographics Rendering Workshop 95*, pages 41–50, 1995.

[107] E. J. Stollnitz, T. D. DeRose, and D. H. Salesin. *Wavelets for Computer Graphics: Theory and Applications*. Morgan Kaufmann, 1996.

[108] G. Strang. Wavelets and dilation equations: a brief introduction. *SIAM Rev.*, 31(4):614–627, 1989.

[109] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Proc Natl Conf on Artificial Intelligence : Workshop of AI for Web Search (AAAI 2000)*, pages 58–64, 2000.

[110] B. Sun and R. Ramamoorthi. Affine double and triple product wavelet integrals for rendering.

[111] B. Sun, R. Ramamoorthi, S. Narasimhan, and S. Nayar. A practical analytic single scattering model for real time rendering. In *SIGGRAPH*, pages 1040–1049, 2005.

[112] W. Sun and A. Mukherjee. Generalized wavelet product integral for rendering dynamic glossy objects. *ACM Transactions on Graphics (SIGGRAPH 06 Proceedings)*, 25(3):955–966, 2006.

[113] P. Tan, S. Lin, L. Quan, B. Guo, and H. Shum. Multiresolution reflectance filtering. In *EuroGraphics Symposium on Rendering 2005*, pages 111–116, 2005.

[114] M. Toksvig. Mipmapping normal maps. *Journal of Graphics Tools*, 10(3):65–71, 2005.

[115] K. Torrance and E. Sparrow. Theory for off-specular reflection from roughened surfaces. *J. Optical Soc. America*, 57(9):1105–1114, 1967.

[116] Y. Tsai and Z. Shih. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Transactions on Graphics*, 25(3):967–976, 2006.

[117] E. Veach and L. J. Guibas. Metropolis light transport. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 65–76, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[118] J. Wang, K. Xu, K. Zhou, S. Lin, S. Hu, and B. Guo. Spherical harmonics scaling. *The Visual Computer*, 22(9-11):713–720, 2006.

[119] R. Wang, R. Ng, D. Luebke, and G. Humphreys. Efficient wavelet rotation for environment map rendering. In *Eurographics Symposium on Rendering 2006*, pages 173–182, 2006.

[120] R. Wang, J. Tran, and D. Luebke. All-frequency relighting of non-diffuse objects using separable brdf approximation. In *Proc. of 2004 Eurographics Symposium on Rendering*, 2004.

[121] R. Wang, J. Tran, and D. Luebke. All-frequency interactive relighting of translucent objects with single and multiple scattering. In *AMC SIGGRAPH*, 2005.

[122] G. J. Ward. Measuring and modeling anisotropic reflection. *SIGGRAPH Comput. Graph.*, 26(2):265–272, 1992.

[123] S. Westin, J. Arvo, and K. Torrance. Predicting reflectance functions from complex surfaces. In *SIGGRAPH 92*, pages 255–264, 1992.

[124] T. Whitted. An improved illumination model for shaded display. *Commun. ACM*, 23(6):343–349, 1980.

[125] L. Williams. Pyramidal parametrics. In *SIGGRAPH 83*, pages 1–11, 1983.

[126] K. Zhou, Y. Hu, S. Lin, B. Guo, and H. Shum. Precomputed shadow fields for dynamic scenes. In *Proc. of SIGGRAPH '05*, pages 1196–1201, 2005.