Representations and Uses of Light Distribution Functions

by

Paul Lalonde

B.Sc., Dalhousie University, 1990

M.Sc., Queen's University, 1992

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

we accept this thesis as conforming
to the required standard

The University of British Columbia

December 1997

Abstract

At their lowest level, all rendering algorithms depend on models of local illumination to define the interplay of light with the surfaces being rendered. These models depend both on the representations of light scattering at a surface due to reflection and to an equal extent on the representation of light sources and light fields.

Both emission and reflection have in common that they describe how light leaves a surface as a function of direction. Reflection also depends on an incident light direction. Emission can depend on the position on the light source. We call the functions representing emission and reflection light distribution functions (LDF's).

There are some difficulties to using measured light distribution functions. The data sets are very large - the size of the data grows with the fourth power of the sampling resolution. For example, a bidirectional reflectance distribution function (BRDF) sampled at five degrees angular resolution, which is arguably insufficient to capture highlights and other high frequency effects in the reflection, can easily require one and a half million samples. Once acquired this data requires some form of interpolation to use them. Any compression method used must be efficient, both in space and in the time required to evaluate the function at a point or over a range of points.

This dissertation examines a wavelet representation of light distribution functions that addresses these issues. A data structure is presented that allows efficient reconstruction of LDFs for a given set of parameters, making the wavelet representation feasible for rendering tasks. Texture mapping methods that take advantage of our LDF representations are examined, as well as techniques for filtering LDFs, and methods for using wavelet compressed bidirection reflectance distribution functions (BRDFs) and light sources with Monte Carlo path tracing algorithms.

The wavelet representation effectively compresses BRDF and emission data while inducing only a small error in the reconstructed signal. The representation can be used to evaluate efficiently some integrals that appear in shading computation which allows fast, accurate computation of local shading. The representation can be used to represent light fields and is used to reconstruct views of environments interactively from a precomputed set of views. The representation of the BRDF also allows the efficient generation of reflected directions for Monte Carlo ray tracing applications. The method can be integrated into many different global illumination algorithms, including ray tracers and wavelet radiosity systems.

Contents

List of Figures

Acknowledgements

There are a few people who need to be thanked for their help and kind words during the last five year. Pam and Neil, thanks. A little bit of stability goes a long way. Jason - thanks for putting up with me, during the proposal, and then the final oral. Alain Fournier, who supervised this work did all the right things to get me done in reasonable time and with a good thesis. And thanks to my parents Claude and Louise who have been supportive of my efforts all my years in university.

Paul Lalonde

The University of British Columbia

December 1997

Chapter 1

Introduction

One of the goals of image synthesis is to produce images of three dimensional scenes represented by a computer model that are indistinguishable from images of real scenes. The most successful methods used to render such images have been based on simulating the physics of light transport [CGIB86, Kaj86]. The process starts with a model of the scene to be rendered, including descriptions of light sources and surface reflection properties, and proceeds to simulate the transport of light from sources to surfaces. Ray tracing methods consider only direct illumination and perfect reflections and refraction [Whi80], although extensions exist to model other phenomena [Ama84, Kaj86, WRC88]. Radiosity methods consider the interreflections of light in scenes with only diffuse surfaces [CGIB86]. Again there are extensions for other lighting phenomena [ICG86, SAWG91]. Both methods use simple models of local reflection and of light sources. Many of the extensions presented to these methods can make use of more complex lighting and reflection models, but although some progress has been made using various analytical models of light reflection, little exploration has been done in the use of measured distributions. As the sophistication of the the rendering methods has increased the traditional reflectance models, such as those presented by Phong [Pho75], Blinn [Bli77], and Cook [CT82], and the light source models, such as those of Poulin [PA91], Warn [War83], and Verbeck [VG84], have become less adequate to the task at hand.

As the demand for more realistic computer generated images increases so does the need for more accurate models of light reflection and of light sources.

1.1 Light Distribution Functions

We use the term *light distribution function* (LDF) for functions that describe how light is distributed over a set of directions. The term is applied to scattering, as well as emissive and transmissive processes. Reflection is the process of redistributing incoming light according to a reflectance function that describes for each incident direction the distribution of reflected light in all reflected direction. Emission, as from a light source, requires characterization of the amount of light emitted in particular directions. Transmission considers how light is propagated through a space, requiring a field of directional samples. In particular, emission can be modeled by considering transmission at a boundary surrounding the source.

The main difficulty in using light distribution functions is in their representation. The light distribution functions considered share some key properties. The functions tend to be smooth over much of their domains, but have discontinuities and peaks that pose challenges in representing them compactly. If real surfaces or light sources are to be measured the functions need to be sampled at a fine interval in order to capture the reflection and illumination effects that make that LDF different visually from other LDFs. This in turn leads to very large representations - reflection, as is refraction, is a function of four angular variables (an incoming direction and an outgoing direction); emission over a surface, or transmission through a planar region, is a function of a positional variable and a direction.

This dissertation addresses a range of issues surrounding representations and uses of light distribution functions including, but not limited to:

1. compact representations;

2. efficient evaluation;

3. ease of application to shading computations;

4. construction by simulation;

5. using LDFs to filter texture maps and shading computations; and,

6. using LDFs in a Monte-Carlo path tracing application

1.2 Outline

The remainder of this dissertation is divided into 7 parts. Chapter 2 presents background and prior work. Chapter 3 presents a simple mechanism for using tabulated reflectance data, and in so doing motivates the work in later chapters. Chapter 4 introduces our wavelet representation of the BRDF and the point-reconstruction algorithm. Chapter 5 shows how the structure can be applied similarly to represent light emitted from a surface. Chapter 6 uses the material from Chapters 4 and 5 to examine the local shading problem and shows how our representation allows efficient filtering of light reflected from the entire incident hemisphere. Chapter 7 shows how to use our BRDF representation to pre-filter texture maps, including bump maps and micro geometry textures. Chapter 8 shows how Monte Carlo integration can be used to evaluate many of the integrals that arise when solving lighting equations.

In particular, it shows how we can use our representation to generate samples that are distributed according to the BRDF and emissive textures.

Chapter 2

Background

The study of optics is divided into three subareas: physical optics, concerned with the wave nature of light and interactions of light with objects of comparable size to its wavelength; quantum optics, concerned with the interaction of light with atoms and molecules [BW75]; and geometrical optics, concerned with macroscopic effects.

In computer graphics we are most concerned with geometrical optics, where we can represent light as directional distributions representing how much light is traveling in a given direction. We use *radiance* as the measure for the amount of light. Radiance is the power per unit of projected area perpendicular to the incoming light per solid angle in the direction of the incoming light, which in SI units is $watts \times m^{-2} \times sr^{-1}$. In computer graphics applications radiance is usually described as the amount of light power leaving a surface through a solid angle. The projected area accounts for the foreshortening of the area of the surface caused by viewing it at an angle.

A related quantity is *irradiance*, E, which is the total energy per unit area falling on a surface, regardless of direction. To refer to the irradiance due to light arriving in a specific direction we write $E_i(\vec{\omega})$. A more complete introduction to these quantities is presented in Appendix A.

2.1 Reflectance Functions

The bidirectional reflectance distribution function (BRDF) characterizes the reflection of light at a surface [Ins86]. Following the basic definition of a surface from *differential geometry* [dC76], we will assume that there is a mapping $\mathbf{x} = \mathbf{x}(u, v)$ of an open set U in the uv plane into S, a set of points in E^3. This mapping is a *regular parameterization* of class $C^{[m]}$ if the function is $C^{[m]}$ in U and given a basis $\mathbf{i}, \mathbf{j}, \mathbf{k}$ in E^3 and writing:

$$\mathbf{x}(u, v) = x_1(u, v)\mathbf{i} + x_2(u, v)\mathbf{j} + x_3(u, v)\mathbf{k}$$

then we have:

$$rank \begin{pmatrix} \frac{\partial x_1}{\partial u} & \frac{\partial x_1}{\partial v} \\ \frac{\partial x_2}{\partial u} & \frac{\partial x_2}{\partial v} \\ \frac{\partial x_3}{\partial u} & \frac{\partial x_3}{\partial v} \end{pmatrix} = 2$$

This of course means that there is at least one of the 2x2 minors of this matrix which is non-zero. We will often refer to $\mathbf{x}(u, v)$ as a point of S, when in fact it is the *image* of the mapping $\mathbf{x} = \mathbf{x}(u, v)$ which is a point of S. We will use similar shorthand for the *parametric curves* such as $\mathbf{x} = \mathbf{x}(u_0, v)$ and the vectors such as the tangent vector with respect to u:$\mathbf{x}_u = (\frac{\partial x_1}{\partial u}, \frac{\partial x_2}{\partial u}, \frac{\partial x_3}{\partial u})$ of the tangent vector with respect to v:$\mathbf{x}_v = (\frac{\partial x_1}{\partial v}, \frac{\partial x_2}{\partial v}, \frac{\partial x_3}{\partial v})$. Note that the definition of a regular parameterization means that the vector $\mathbf{N} = \mathbf{x}_u \times \mathbf{x}_v$ cannot be of length 0, and therefore can always be normalized. The normalized vector:

$$\mathrm{N} = \frac{\mathbf{x}_u \times \mathbf{x}_v}{|\mathbf{x}_u \times \mathbf{x}_v|}$$

is the normal to the surface at point $\mathbf{x}(u_0, v_0)$ if the partial derivatives are computed for these values of the parameters.

We will later restrict the parameterization a little more, but this is enough for now to consider the geometry of the reflection as illustrated in Figure 2.1.

An orthonormal basis for E^3 is given by $\mathbf{i}, \mathbf{j}, \mathbf{k}$ which define the axes X, Y and Z respectively. Without loss of generality we will compute the partial derivatives and $\mathbf{N}$ at $u_0 = 0$ and $v_0 = 0$. The vector $\mathbf{k}$ is chosen so that $\mathbf{k} = \mathbf{N}$. Without loss of generality we have assumed that the unit vector $\mathbf{i}$ defining the X axis is the image of $\mathbf{x}_u$ normalized, and that the unit vector $\mathbf{j}$ defining the Y axis is $\mathbf{j} = \mathbf{N} \times \mathbf{i}$. We know from the definition of the regular parameterization that this vector cannot be null.

We will consider two points in S, $\mathbf{P} = \mathbf{x}(u_i, v_i)$ and $\mathbf{Q} = \mathbf{x}(u_r, v_r)$. Directions will be noted in polar angles $\vec{\omega} = (\theta, \phi)$ in the frame XYZ[1]. For instance the direction of the X axis is $\vec{\omega}_X = (\frac{\pi}{2}, 0)$, the direction of the Y axis is $\vec{\omega}_Y = (\frac{\pi}{2}, \frac{\pi}{2})$, and the direction of the Z axis is $\vec{\omega}_Z = (0, \phi)$ where ϕ can take any value.

2.1.1 BRDF Definitions

The definition of the *bidirectional reflection distribution function* is based on the fact that if a quantity of light falls at a point $\mathbf{P} = \mathbf{x}(u_i, v_i)$ from the direction $\vec{\omega}_i = (\theta_i, \phi_i)$ over an infinitesimal solid angle $d\omega_i$ and causes an amount of reflected light at point $\mathbf{Q} = \mathbf{x}(u_r, v_r)$ in direction $\vec{\omega}_r = (\theta_r, \phi_r)$, another amount from the direction will cause a proportional amount of reflected light at the same point in the same direction.

Note that if we consider photons, that means that if a photon takes a path

[1] Differential solid angles have associated with them a unique radial direction. When we write $\vec{\omega}$ we are referring to the radial direction associated with the differential solid angle $d\omega$.

from **P** from direction $\vec{\omega}_i$ to re-emerge at point **Q** in direction $\vec{\omega}_r$, any other photon incident into the same point in the same direction will re-emerge at the same point in the same direction. In quantum optics this applies only to probabilities, but we will work only with amounts of light for which the variance can be ignored. This will be true if the medium did not change appreciably in the time between the two paths being taken (such as in a very rapid chemical reaction).

This simple assumption of proportionality fails to hold in many cases, for instance when *fluorescence* or *phosphorescence* (when the delay between absorption and emission is longer) intervene. The incident quantity of light is best specified as a flux Φ, of dimension of power. Since this flux will be distributed over an area on the receiving surface and over a solid angle involving a a range of direction, one can specify it either as a second order differential of flux, or as a first order differential of irradiance over a differential of area, or as a radiance over a differential area and a differential element of solid angle:

$$d^2\Phi(u_i, v_i, \vec{\omega}_i) = dE(u_i, v_i, \vec{\omega}_i)dA_i = L_i(u_i, v_i, \vec{\omega}_i)\cos\theta_i d\omega_i dA_i \tag{2.1}$$

The $\cos\theta_i$ term is necessary in the relation 2.1 above because with radiance the power is expressed per solid angle and per area projected in the direction of the ray considered. In this case the ray carrying the radiance makes an angle θ_i with the surface element dA_i. To specify the reflected light we will use the radiance at that point in that direction $L_r(u_r, v_r, \vec{\omega}_r)$. If the power of the reflected light is distributed both over an area and over a solid angle, then the reflected radiance has to be a second order differential element $d^2L_r(u_r, v_r, \vec{\omega}_r)$. So we will define the *bidirectional scattering surface reflectance distribution function* (BSSRDF) as the ratio:

$$S_{PQ}(u_i, v_i, \vec{\omega}_i, u_r, v_r, \vec{\omega}_r) = \frac{d^2L_r(u_r, v_r, \vec{\omega}_r)}{d^2\Phi_i(u_i, v_i, \vec{\omega}_i)} \tag{2.2}$$

The dimensions of this quantity are $L^{-2}sr^{-1}$.

It is important to note that the same concepts and notations can be used when the light does not emerge on the same surface, but on another surface (virtual or real). This is the basis of the *Lucifer* approach to light transport and global illumination as described in [LF96]. In this case the reflection process is substituted by an explicit light transport mechanism.

The *Helmholtz reciprocity principle* [SH92] states that the BRDF remains unchanged if incident and reflected directions are switched. It is clear that in the general case of the BSSRDF the incident and reflected positions have to be switched as well for the reciprocity principle to apply.

Equations similar to 2.2 have been of course been developed before, especially by Nicodemus and his collaborators [NRH+77] also reprinted in [SHW92]. The best exposition of this material for computer graphics is to be found in Glassner's book [Gla95].

If we replace the differential incident flux by its expression as a function of the incident radiance and the differential elements of area and solid angles (Equation 2.1) we get:

$$S_{PQ}(u_i, v_i, \vec{\omega}_i, u_r, v_r, \vec{\omega}_r) = \frac{d^2L_r(u_r, v_r, \vec{\omega}_r)}{L_i(u_i, v_i, \vec{\omega}_i)\cos\theta_i d\omega_i dA_i} \tag{2.3}$$

We can easily find the relationship between the BSSRDF and the more commonly used *bidirectional reflectance distribution function* (BRDF) by integrating the reflected radiance over an area A_r:

$$dL_r(\vec{\omega}r) = \int_{A_r} d^2L_r(u_r, v_r, \vec{\omega}_r)$$

Replacing $d^2L_r()$ by its expression in terms of $d^2\Phi()$ and S_{PQ} (Equation 2.3):

$$dL_r(\vec{\omega}r) = \int_{A_r} S_{PQ}(u_i, v_i, \vec{\omega}_i, u_r, v_r, \vec{\omega}_r) L_i(u_i, v_i, \vec{\omega}_i)\cos\theta_i d\omega_i dA_r.$$

Since $L_i()$ is not a function of (u_r, v_r) we can remove it from the integral. We can also remove $\cos\theta_i$ and $d\omega_i$ from the integral. We then obtain:

$$dL_r(\vec{\omega}r) = L_i(u_i, v_i, \vec{\omega}_i)\cos\theta_i d\omega_i \int_{A_r} S_{PQ}(u_i, v_i, \vec{\omega}_i, u_r, v_r, \vec{\omega}_r)dA_r.$$

If we note:

$$F(u_i, v_i, \vec{\omega}_i, \vec{\omega}_r) = \int_{A_r} S_{PQ}(u_i, v_i, \vec{\omega}_i, u_r, v_r, \vec{\omega}_r)dA_r$$

then we have in $F()$ an expression of the BRDF:

$$F(u_i, v_i, \vec{\omega}_i, \vec{\omega}_r) = \frac{dL_r(\vec{\omega}_r)}{L_i(u_i, v_i, \vec{\omega}_i)\cos\theta_i d\omega_i} \tag{2.4}$$

In this expression, we have removed from the list of arguments the position of the point at which we define the BSSRDF by integrating over an area. If this area is such that all the reflected light flux is included, then the function $F()$ is expressing some intrinsic property of the surface. One can also note that the denominator in Equation 2.4 is $L_i(u_i, v_i, \vec{\omega}_i)\cos\theta_i d\omega_i = dE(u_i, v_i, \vec{\omega}_i)$. The dimension of $F()$ is clearly sr^{-1} that is the reciprocal of the solid angle.

The expression in equation 2.4 is still a function of the coordinates of $\mathbf{P}$, the incident point. There are two simplifications that will lead us to the "classic" formula for the BRDF. The first step is if we assume that there is no reflection except at the point of incidence. That means that the BSSRDF includes a Dirac $\delta()$ function of the form:

$$S_{PQ}(u_i, v_i, \vec{\omega}_i, u_r, v_r, \vec{\omega}_r) = \delta(u_r - u_i, v_r - v_i)\mathcal{F}(u_i, v_i, \vec{\omega}_i, \vec{\omega}_r)$$

and that the integral giving $F()$ will give us the right answer if the area A_r includes the incidence point $\mathbf{P}$:

$$F(u_i, v_i, \vec{\omega}_i, \vec{\omega}_r) = \int_{A_r} \delta(u_r - u_i, v_r - v_i)\mathcal{F}(u_r, v_r, \vec{\omega}_i, \vec{\omega}_r)dA_r = \mathcal{F}(u_i, v_i, \vec{\omega}_i, \vec{\omega}_r).$$

The second step is to assume that the material of the surface is *homogeneous*, that is the function $f()$ is independent of the position of the incident point $\mathbf{P}$. In this case we merely drop the arguments (u_i, v_i) from the expression of $f()$ and obtain:

$$\mathcal{F}(\vec{\omega_i}, \vec{\omega_r}) = \frac{dL_r(\vec{\omega_r})}{L_i(\vec{\omega_i}) \cos\theta_i d\omega_i} \tag{2.5}$$

Several points should be noted here about Equation 2.5.

1. We have also removed (u_i, v_i) from the expression of the incident radiance, even though the radiance will be in general a function of the position on the surface. The subscript i should remind us of that.

2. The BRDF here is not assumed to be *isotropic*, and therefore depends on the local frame of reference used at the point of incidence. Even if it is isotropic it will clearly depend on the direction of the normal vector at the point of incidence.

We have avoided difficulties by assuming that all the reflected light comes out at the point of incidence, and therefore it is obvious which frame of reference to use. In the general case of the light emerging at different points, or even in the case where we want to average the "classic" BRDF over a curved surface a second coordinate frame is necessary to define the exiting light.

2.1.2 Illumination

Consider again the geometry in Figure 2.1. It shows a differential area, dA, being illuminated by a light L_i coming from an infinitesimal solid angle $d\omega_i$. The light is then reflected in a given direction $\vec{\omega_r}$. Let $E(\lambda)$ be the irradiance at wavelength λ arriving at the element dA from the entire hemisphere centered on the normal $\vec{N}$.

Then

$$dE(\vec{\omega_i}) = L_i(\vec{\omega_i}) \cos\theta_i d\omega_i \tag{2.6}$$

where $dE(\vec{\omega_i})$ is the change in exitance caused by the illumination from $d\omega_i$ at wavelength λ. Given the BRDF, we can relate the change in irradiance to the change in exitance, $dL_r(\vec{\omega_r})$, emitted in direction $\vec{\omega_r}$:

$$dL_r(\vec{\omega_r}) = f_r(\vec{\omega_i} \rightarrow \vec{\omega_r}) dE(\vec{\omega_i}) \tag{2.7}$$

The only contribution to the radiance L_r is from the hemisphere surrounding dA, written Ω_N. We can integrate $\vec{\omega_i}$ over the hemisphere Ω_N to get

$$L_r(\vec{\omega_r}) = \int_{\Omega_N} L_i(\vec{\omega_i}) f_r(\vec{\omega_i} \rightarrow \vec{\omega_r}) \cos\theta_i d\omega_i \tag{2.8}$$

This is commonly referred to as the fundamental equation of physical shading. Most work done in illumination and shading has concentrated on simplifying one or both of the $L_i(\vec{\omega_i})$ and $f_r(\vec{\omega_i} \rightarrow \vec{\omega_r})$ terms. A common simplification is to assume that the BRDF does not change with the rotation of the surface dA about the normal $\vec{N}$. Such a BRDF is called isotropic and can then be expressed as

$$f_r(\phi_i, \theta_i, \phi_r, \theta_r, \lambda) = f_r^{iso}(\theta_i, \phi_r - \phi_i, \theta_r, \lambda) \tag{2.9}$$

which can reduce storage and computational costs considerably. DeYoung *et al.* examine how to determine if a measured BRDF is isotropic or not, and show how to force a dataset to have this property, at the possible cost of accuracy [DLF96].

There are several other simplifications made. Lambertian reflectors, used in radiosity solutions, assume the BRDF is a constant. Blinn-Phong shaders assume the BRDF is given by a diffuse Lambertian term and a specular reflection term [Pho75, Bli77]. Other shaders approximate real BRDFs with varying degrees of complexity and success [War92a, CT82, HTSG91].

2.1.3 Commonly Used Local Illumination Models

Although real materials' BRDFs can be measured[2] they are rarely used in practice so far. Instead, BRDFs are approximated by simple mathematical models of varying levels of accuracy, both with regards to appearance of the rendered surface and to their physical properties, such as energy conservation and reciprocity [Lew93].

Early local illumination work in computer graphics dates to the mid to late 1970's with the work of Phong, Blinn, and Cook, [Pho75, Bli77, CT81], and to earlier work in optics [TS66, TS67]. The early computational models were designed to produce reasonable images at very low computational cost—the computers of the day were not up to the task of evaluating substantially more complex systems.

Phong-Blinn Shading

One of the most widely used computer graphic local illumination models was developed by Phong Bui-Tuong [Pho75]. The strength of the Phong model lies in its simplicity. The model allows non-ideal specular reflection with a specular peak of adjustable width. The width of the peak is controlled by a $\cos^n \alpha$ term, originally written as $(\vec{R} \cdot \vec{V})^n$, where $\vec{R}$ is the direction of perfect reflection (as in a perfect mirror) and $\vec{V}$ is the reflected direction. The total illumination at a point on a surface at a given wavelength λ is given by

$$I(\lambda) = I_s(\lambda)(k_{diff}(\vec{N} \cdot \vec{L}) + k_{spec}(\vec{R} \cdot \vec{V})^n)$$

where $I_s(\lambda)$ is the illumination arriving at the surface from the source, k_{diff} and k_{spec} are the fractions of diffuse and specular transport respectively, and $\vec{L}$ is the

[2] Some argue this point on the basis that BRDFs may be comprised of delta functions. However, since measuring equipment does not sample an infinitesimal area or solid angle but an integral over some finite solid angle, this concern is unjustified. Moreover, delta functions can't occur in real materials.

direction to the light source. This formulation has a serious error. The highlight is controlled only by the difference between the viewing angle and the reflected angle, without taking into account surface geometry directly, leading to cases where the highlight can spill below the visible hemisphere centered on the normal to the surface.

A frequently used reformulation of Phong shading, due to Blinn [Bli77], replaces the $\vec{R} \cdot \vec{V}$ term with the term $\vec{N} \cdot \vec{H}$, where $\vec{N}$ is the normal to the surface, and $\vec{H}$ is the *halfway* vector, pointing half way between the light source direction and the viewer direction. Blinn's formulation generates more sensible results at glancing angles and similar results to Phong's elsewhere.

Neither the Phong model, nor Blinn's reformulation of it, is based on any theoretical model of reflection.

A Physics Based Model

The Torrance-Sparrow model [TS66, TS67] is a physics based model of the interaction of light with a reflecting surface. Early implementations for computer graphics are due to Blinn [Bli77] and to Cook and Torrance [CT81].

The model represents a surface as a uniform distribution of microscopic mirrors, called micro-facets. The distribution and geometry of these reflectors controls how light reflects from a surface. The specular term of the reflection is given by

$$\rho_s = \frac{F_\lambda}{\pi} \frac{DG}{(\vec{N} \cdot \vec{V})(\vec{N} \cdot \vec{L})},$$

where D is the distribution function of the micro-facets; G is the geometrical attenuation factor, and accounts for self-shadowing and self-reflection of micro-facets with one another and is a simple geometric term; and F_λ is the Fresnel term that relates incident light to reflected light at each micro-facet. The $\vec{N} \cdot \vec{V}$ term scales the result by the visible (foreshortened) area, and the $\vec{N} \cdot \vec{L}$ scales for the area visible to the light source.

The complexity of the Cook-Torrance model is hidden in the D, G, and F functions. The micro-facet distribution function D, determines what proportion of the facets lie in the direction of the halfway vector. Choices used for this function include a Gaussian bump distribution, as used by Torrance and Sparrow [TS66, TS67]; the Trowbridge and Reitz distribution [TR67], used by Blinn; and the Beckmann distribution [BS87]. The Beckmann distribution has the advantage of a good theoretical basis and no arbitrary constants, unlike the other two, but does not produce particularly nice images. Cabral *et al.* propose an extension that allows algorithmic simulation of the micro-facet distribution allowing a wider range of micro-facet distributions to be used [CMS87].

The Fresnel term F relates the amount of light reflected from a surface at a certain wavelength to the incoming wavelengths, geometry of the surface and light, to the angle of incidence, and the polarization of the incoming light. The Fresnel term can be derived from Maxwell's equations at the surface boundary, ensuring that energy and continuity constraints are conserved after reflection. The derivation can be found in many modern optics texts, such as Born and Wolf [BW75]. For our purposes it is sufficient to note that to use these equations it is necessary to know the index of refraction and the coefficient of extinction of the surface. The expressions are also computationally expensive, requiring the evaluation of a number of trigonometric functions, square roots, and computation on complex numbers.

The G term is geometrically defined to attenuate light because of self-blocking and intereflection by the micro-facets. As the incident and reflected directions approach the horizontal more of the light will be masked by the micro-facet geometry.

2.1.4 Anisotropic Shaders

The shaders examined to date have been independent of surface orientation. However, many surfaces exhibit anisotropy, having different appearances from different directions. For example, fabrics frequently have a nap to them, appearing considerably different depending on the orientation at which they are viewed.

The earliest work on anisotropic shaders (where the BRDF varies with the orientation of the surface) in computer graphics is due to Kajiya, who developed a shader derived from the equations of electromagnetism [Kaj85]. His model uses an extension to Beckmann's scattering formula [BS87] extended to include anisotropic effects. Kajiya also extends bump mapping to frame mapping, where the entire local coordinate frame at a point can modified, allowing local deformation of anisotropy effects.

The Poulin-Fournier Model

Poulin and Fournier developed a simple model of anisotropic reflection in which small cylinders are distributed in various orientations over a surface [PF90]. The intensity of reflected light is calculated by determining the visible and illuminated portions of the cylinder, taking self-blocking into account. Their images are satisfactory although the model seems limited to simulating brushed metals and Christmas ornaments.

He-Torrance Shading

He *et al.* present a reflectance model based on physical optics that computes reflective effects based on wavelength, incident angle, two surface roughness parameters, and the surface refractive index [HTSG91]. Their model also treats polarization and directional Fresnel effects, allowing the simulation of effects such as depolarization and cross-polarization. Their model can be seen as an extension to the Cook-Torrance model, replacing the micro-facet distribution made of Gaussian peaks, with a roughness described by the size of the Gaussian bumps and an autocorrelation length that is a measure of the distance between the Gaussian peaks, and carrying the polarization terms through their derivation. Their formulation approximates the diffuse behaviour of the surface with a uniform (Lambertian) term, and breaks the specular term into an ideal specular term, whose magnitude is controlled by the surface roughness, and a directional diffuse term which depends on the statistical distribution of the Gaussian bumps on the surface.

The model yields visually attractive results, but the accuracy of the result depends crucially on selecting the correct surface statistics for the material being simulated. The authors only give a method for approximating the uniform diffuse term, leaving the reader at a loss for how to compute the other parameters required to shade a surface.

2.1.5 Sampled BRDF data

In addition to analytical models there are a number of techniques which depend on sampling either physical BRDF data, or else building up a sampled description of a BRDF from stochastic or other types of processes.

Cabral *et al.* present a method of generating BRDFs filtering micro-geometry distributions and approximating the result using spherical harmonics [CMS87]. Westin *et al.* use a similar method but sample the geometry by ray-tracing. Their results are visually attractive, but their method requires storing tens of thousands of coefficients; they do not state the angular sampling density used to generate their images. One disadvantage of using spherical harmonics is that the basis functions themselves are relatively expensive to compute. When used to represent BRDFs they are dependent on four variables, which makes them impossible to tabulate efficiently.

Another approach to modeling the reflective properties of micro-geometries was taken by Fournier [Fou92] who represents a micro-geometry as a set of representative normal directions and magnitudes, each with an associated Phong shader. To shade such a surface the contributions of each normal are summed.

Ward presents a method for measuring and storing BRDF data for real objects. He describes his imaging gonioreflectometer, based on capturing the entire hemisphere of reflected light for an incident direction at once by using a reflecting hemisphere and a camera with a fish-eye lens. He then fits an anisotropic surface micro-facet slope distribution to his acquired data [War92a].

2.1.6 Other work

Apart from the general and measured illumination models a number of more specialized reflection models have been developed.

Hanrahan and Krueger present a model of reflection caused by subsurface scattering [HK93]. They decompose surfaces into multiple layers with different reflection and absorption properties, solve the interactions using linear transport theory, and propose a ray-based Monte Carlo solution of the resulting integrals. Their model is particularly applicable to materials such as skin and leaves, where the effects of layering have a profound effect on the appearance of the surface.

Nakamae *et al.* present a lighting model aimed at simulating driving conditions, including wet roads, puddles, streaks of light, and the interaction with the viewer's eye [NKON90].

2.2 Luminaires

In illumination engineering a luminaire refers to a light source and its associated shades and reflectors. We will use the word luminaire in this way instead of the more common 'light source' used in compute graphics since we are more interested in the complete lamp. This section examines current computer graphics models of light sources and luminaires.

The light source plays as important a part in shading calculations as does the surface. Most light source models used currently are directional, point, linear, and polygonal area sources. Directional and point sources are attractive because of their simplicity. Finding the direction to the light is simple and efficient. Linear and area sources are more difficult, requiring an integration over the surface of the light in order to compute the lighting. Some attempts have been made, particularly in the field of near-field photometry, to measure the shape and light distribution of actual luminaires [Ash92]. A related luminaire representation used when shading is the hemi-cube representation from radiosity techniques [CG85]. One view of the hemi-cube is that it represents a luminaire comprised of the rest of the environment. This interpretation becomes relevant when using LDF representations for global illumination [Lew96].

2.2.1 Directional and Point Light Sources

Directional sources simplify the fundamental shading equation (Eqn. 2.8) by treating $L_i(\vec{\omega_i}, \lambda_i)$ as a product with a delta function parameterized by direction.

$$L_i(\vec{\omega_i}, \lambda_i) = E(\lambda_i)\delta(\cos\theta_i - \cos\theta_s)\delta(\phi_i - \phi_s)$$

Point light sources are considered similarly, by including a distance term, r, to the source to correct for the solid angle of the source subtended by the surface element. The irradiance of a point source can vary by direction. The irradiance on an element dA due to the point source is given by

$$E(\lambda_i) = I(\vec{\omega}, \lambda_i)\frac{\cos\theta}{r^2} \tag{2.10}$$

If the direction to the source is $\vec{\omega_s}$ then the radiance from the point source is:

$$L_i(\vec{\omega_i}, \lambda_i) = \frac{I(-\vec{\omega}_s, \lambda_i)}{r^2}\delta(\cos\theta - \cos\theta_s)\delta(\phi - \phi_s) \tag{2.11}$$

Common extensions to the point light model include adding baffles and cones [War83], in effect specifying the directional distribution of $I(\omega, \lambda)$.

2.2.2 Linear and Area sources

Directional and point sources are rare in the physical world. Real light sources have an extent, and this shows itself in the form of soft shadows, where a surface is illuminated by only part of a source. Point sources can be used to simulate area sources, but only by using a large number of them [VG84]; even then aliasing artifacts remain [Pou91].

Poulin and Amanatides give an analytical model for linear light sources based on integrating over the length of the light. They also present data structures to handle partial occlusion of the source [Pou91].

Nishita and Nakamae present a method based on contour integration to determine the illuminance at a point due to a polygonal light source in the presence of polygonal blockers [NON85]. Vedel approximates the contours due to blockers using a quad-tree approach, allowing non-polygonal objects to be used [Ved93].

Arvo developed analytic expressions for solving illumination due to Lambertian emitters [Arv94], and due to emitters with Phong distributions [Arv95]. His techniques are based on *irradiance tensors* which use weighted integrals of the radiation field to simulate various non-diffuse phenomena. He gives analytic forms for piecewise polynomial distributions over polygonal light sources.

Houle extends the area light source representation, allowing the brightness and directional emissivity distribution of an area source to vary over the source's extent [Hou91].

2.2.3 Radiance Representations

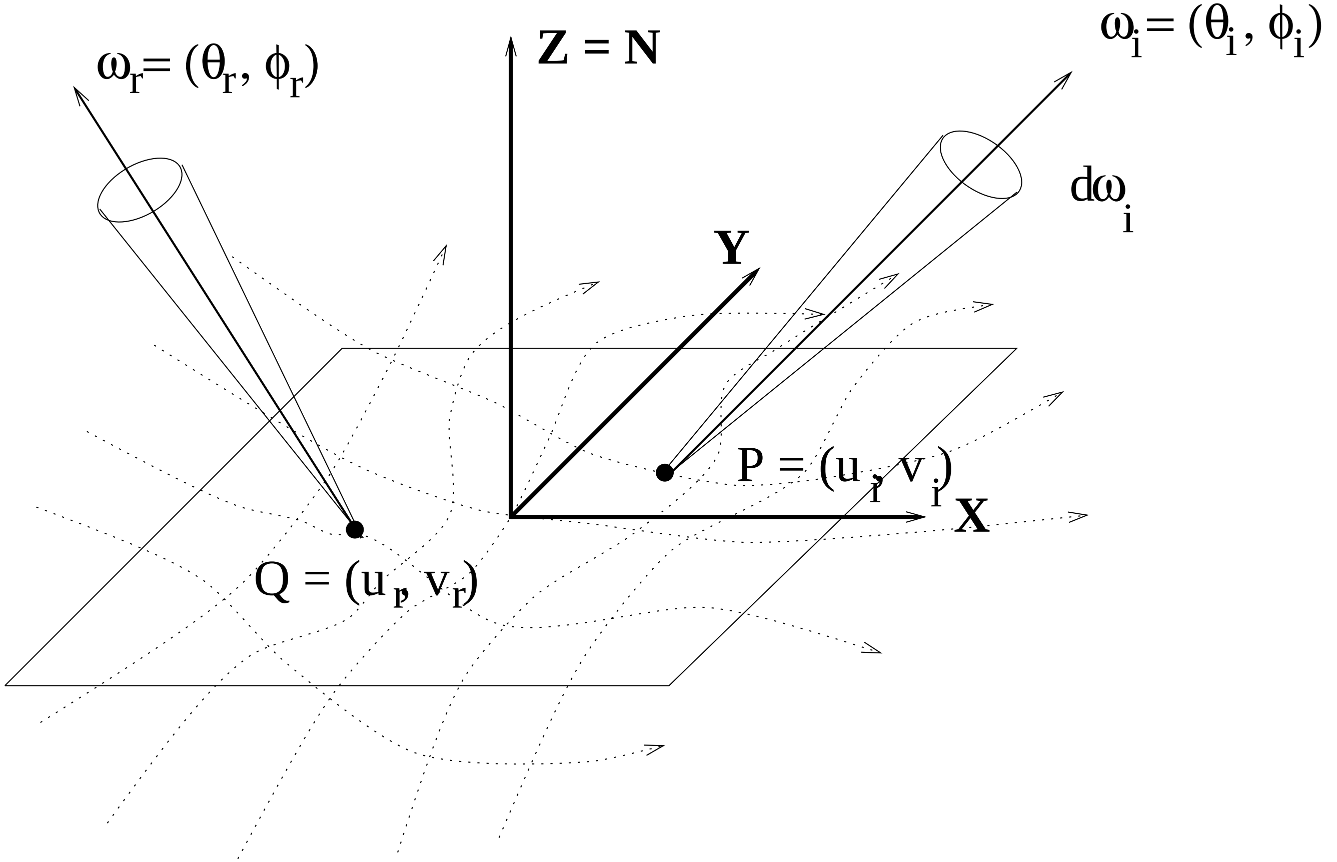
At the next level of refinement, light passing through a a finite area can be characterized. By recording the radiance in each direction at each point on the area a compete representation of the distribution is obtained.

Levoy and Hanrahan [LH96] and Gortler *et al.* [GGSC96] parameterize the space of positions and directions by the intersection of a line with two parallel places, one of which represents a viewing window while the other specifies a direction. A set of views of the volume are stored, and a new view is generated by interpolating from these views. Levoy and Hanrahan use a lossy compression system that reduces the accuracy of the simulation. Gortler *et al.* do not compress their data, keeping a large number of views in memory and using hardware texture mapping to perform the interpolations.

Christensen *et al.* develop a wavelet representation of radiance distributions using a non-standard wavelet decomposition over a unit patch and a radially stretched gnomonic projection of the hemisphere of directions. They then solve two point light transport equations by numerical sampling of the BRDF and of the transport integrals [CSSD96].

Lewis *et al.* present a similar representation of radiance distributions based on a Nusselt embedding of the angular components [Lew96]. This will be examined in more detail in Section 4.1.

Ashdown [Ash92] presents a method for measuring such a data set for arbitrary luminaires, based on using a CCD-based video camera to record the emitted radiance from a number of positions around the luminaire. He also presents results claiming that a 5 degree sampling over the sphere surrounding the luminaire is sufficient for near-field illumination calculations. He uses a resolution of about 480 samples per position. A quick calculation indicates that this representation still requires on the order of one and a half million samples, which is unwieldy for most graphics computations, particularly when multiple light sources are present in the scene. The storage costs are simply too high.



$\omega_r = (\theta_r, \phi_r)$
Z = N
$\omega_i = (\theta_i, \phi_i)$
$d\omega_i$
Y
$P = (u_i, v_i)$
X
$Q = (u_r, v_r)$

Figure 2.1: The geometry of local illumination

Chapter 3

Representing BRDFs

The BRDF is a function of four variables that needs to be sampled at a fairly high resolution to capture the reflection behaviour of a surface. Although many analytic BRDF models of varying complexity and accuracy are in use few of these rely on actual measurements of surfaces to fit parameters. Part of the reason for this is that actual measurements are expensive and hard to come by, and part is because real BRDFs are considerably more complex than those generated by computer models. In addition, the computational cost of those models that start to approach accurate representations are prohibitive.

3.1 Measured Reflectance Functions

Measuring a BRDF from an actual surface is problematic for several reasons. A gonioreflectometer measures a BRDF by physically moving a light source and a radiometer over the hemispheres of incident and reflected directions, taking measurements every few degrees. Gonioreflectometers are relatively expensive and slow, and the data returned from them is often quite noisy, both in position and in value.

Ward attempted to address the problems of cost and speed [War92a] but the samples obtained from his gonioreflectometer are both nonuniform and noisy, complicating their use in rendering. Ward overcomes these limitations by fitting the parameters of a simple anisotropic shader to his data and using this shader to render surfaces, but his images are limited by his shading model.

Greenberg *et al.* describe their experimental setup in which a gonioreflectometer is used to gather BRDF data. They acknowledge that the amount of data collected is prohibitive to use, although they do use the data to try to validate analytic light reflection models[GTS+97].

Pai *et al.* have designed a measurement workbench, ACME, that should allow measurement of BRDFs, as well as geometry. It is expected to be operational by December 1997 [PAI97].

3.1.1 The Virtual Gonioreflectometer

It is also possible to simulate a gonioreflectometer in software. Using a ray-casting strategy, a micro-geometry representing the small scale surface properties of a material is sampled [CMS87, WAT92, DLF96]. A directional light source is moved over the hemisphere surrounding the surface normal, and at each light position the eye is fixed and the lighting evaluated over the micro-geometry being sampled. Multiple samples scattered over the area of the sample provide greater accuracy. The method used to evaluate the lighting of the sample patch can have a considerable impact on the resulting BRDF. In particular some strategy that accounts for inter-reflection over the surface will produce different results than a simple ray-casting strategy. For our experiments we used a path tracing algorithm similar to Kajiya's [Kaj86].

The resulting samples are then tabulated and can be used in the same way as a physically measured BRDF can. For research purposes these synthetic BRDFs have a number of advantages. The noise and errors can be simulated, and samples can be taken at specific points without having to deal with the inaccuracies involved with mechanical systems. The sampling is both fast and repeatable.

Once measured and tabulated, BRDF data sets are large enough that their size is an impediment to their use. Without a priori knowledge of a material being sampled with a gonioreflectometer, or of the properties of a micro-geometry being sampled by a virtual gonioreflectometer, it is not possible to establish what sampling density is required to represent the BRDF within a given error tolerance. Either a relatively low frequency sampling is performed, possibly missing important features such as sharp reflection peaks, or a high frequency sampling is used, resulting in a data set which is too large to be useful. Consider for example a Phong model with an exponent of 50. Its specular component falls by half in 9.6 degrees. Sampling every 10 degrees gives $(36 * 9)^2 = 104976$ samples. Each of these may involve multiple spectral samples. It is possible to establish if a given BRDF has certain properties, such as anisotropy for example, and modifying the BRDF to have a certain property. DeYoung and Fournier show methods for quantifying anisotropy, reciprocity, separability, and energy conservation [DF97].

3.2 Requirements of the Representation

To determine if a particular representation is successful we must establish what is required of our representation. There are a number of operations that the representation must support, and some constraints on efficiency and compactness.

3.2.1 Point Reconstruction

At a minimum the representation must allow reconstruction and interpolation of the BRDF at a given set of incident and reflected angles. This is the basic operation involved in using a BRDF for almost any computation.

3.2.2 Accurate Shading

One common operation that uses BRDFs is local shading. We would like to perform shading computations in a more sophisticated way than just point samples of light directions. In particular, the representation should allow the reflected radiance to be filtered over all directions of incident irradiance. This corresponds to evaluating the shading equation (Eqn. 2.8):

$$L_r(\vec{\omega_r}, \lambda) = \int_{\Omega_N} L_i(\vec{\omega_i}, \lambda) f_r(\vec{\omega_i} \rightarrow \vec{\omega_r}, \lambda) \cos\theta_i d\omega_i. \tag{3.1}$$

This can be done by summing a number of point-wise samples, but more efficient methods would be useful.

3.2.3 Texture Maps

Surfaces used in computer graphics frequently exhibit shading detail at a level between that of pure geometry and that of shading differential areas. Texture maps modify shading parameters in that range of scales, adding visual richness without extending the geometry. In this range the level of detail used depends critically on the projected area of a pixel onto the object. Methods that allow our BRDF representation to be used in conjunction with texture mapping would be useful.

3.2.4 Compactness

It is not uncommon to have several hundred different surfaces with different surface properties appear in single frames of large scale commercial animations. These surface properties, of which the BRDFs play an important role, must be reasonably compact in their representation, or else the scenes could not be computed in a timely fashion because of memory constraints. As will be shown, raw representations of BRDF data are generally too large to fit this requirement.

3.2.5 Level of Detail

Ideally the reconstructed BRDF would be identical to the original sampled signal at every sample point. However, such a reconstruction may be prohibitively expensive. It would be good if our method allowed fidelity to be traded off for faster rendering, letting the user rapidly pre-view an image, while being able to compute a full fidelity image when required. Likewise, the values returned by our presentation between samples points should not add important features to the BRDF that do not exist in the original. This rules out most interpolating polynomials, and many other representations, because of ringing effects.

3.3 Shading with Measured Reflectances

3.3.1 Selecting a Parameterization

The topology of the BRDF function space is an hemisphere crossed with an hemisphere; if we consider refraction as well, a sphere crossed with a sphere ($S^2 \times S^2$). In selecting a parameterization to use we must consider a few factors:

- The topology of the basis functions used to represent the functions;

- The error induced by the conversions from the measurements; and
- The distortion in sample spacings caused by different embeddings.

Fortunately the embedding of $S^2 \times S^2$ is straightforward, save for the matter of redundancy at the poles. For our initial experiments we chose to use a polar mapping, as in [CMS87]. This mapping was chosen because of its simplicity and because it is consistent with the inputs to the shading code in the ray-tracer used. It provided a very quick way to test the usefulness of measured reflectance function as well as allowing us to easily visualize the functions directly using interactive techniques[DLF96]. Another obvious choice is to map by direction cosines or using a Nusselt embedding [Lew96]. This parameterization will be examined in more detail in Section 4.1.

3.3.2 Using Measured BRDFs

The simplest way to use measured reflectance functions while rendering in a ray tracing environment is to use interpolated table lookup. Fortunately BRDFs have the property of being smooth over much of their range, but they also contain discontinuities and regions of rapid change that are sufficiently important that they should not be ignored.

Another fact to consider is that the original BRDF may have discontinuities, but the reconstruction method used should not introduce new discontinuities where none exist. Fitting our simple basis functions over the data will misrepresent the discontinuities already present, but this cannot be helped, particularly since we are using discrete samples. The Nyquist theorem tells us that we can only accurately reconstruct a signal up to a frequency dictated by our sampling rate, but our signal may contain arbitrarily high frequencies. In fact, in the presence of discontinuities our signal contains arbitrarily high frequencies that cannot be reconstructed from our samples.

We are then left in a position where more samples are required to arrive at a reconstruction that is accurate to some degree, but where the samples have a high cost in terms of storage. Choosing an appropriate set of basis functions with which to represent the measured BRDFs should lead to considerable space savings and increased accuracy in reconstruction.

For initial experiments data were gathered with the virtual gonioreflectometer and the resulting BRDFs used to shade surfaces with a ray tracer. We used 2 micro-geometries in our tests. The sawtooth micro-geometry, shown in figure 3.2 was chosen because of sharp discontinuities and ease of visualization. The velvet, shown in figure 3.3 was chosen because of its interesting shading characteristics. Much of the reflection on velvet occurs in regions of low incident angle, with a large uniform scattering, and a small contribution from the surface under the individual cut threads.

The most naive approach to using a tabulated BRDF is to use a piecewise constant reconstruction strategy. This technique, though easy to implement and computationally efficient, is usually insufficient for even the simplest BRDFs. The lack of continuity in the representation causes highly visible artifacts in the generated images (Figure 3.1(a)).

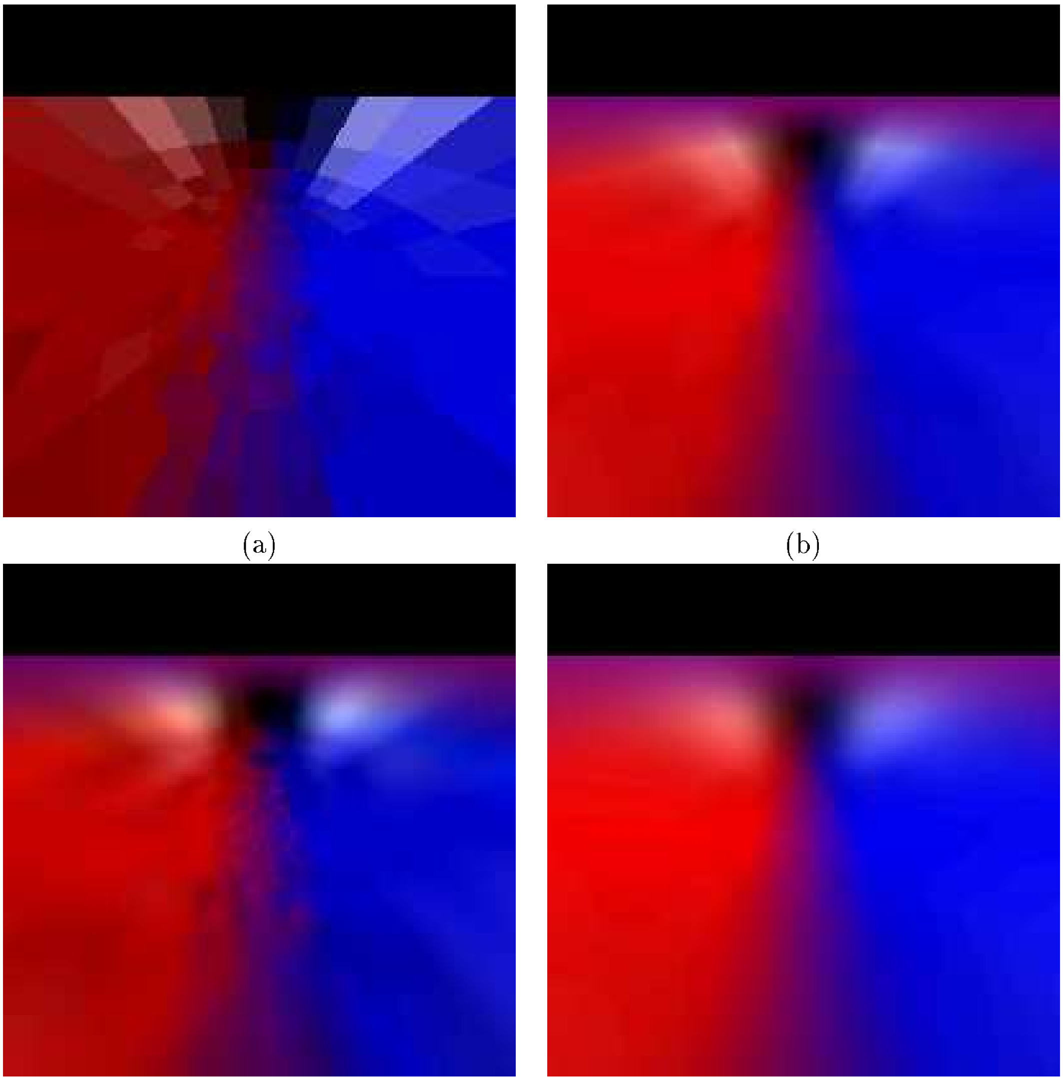
Quadrilinear interpolation on the parameters $\phi_i, \theta_i, \phi_r, \theta_r$, provides geometric $(C^{(0)})$ continuity, but the lack of $(C^{(1)})$ tangential continuity leads to visible artifacts (Figure 3.1(b)).

Higher order interpolation functions can provide us with $C^{(1)}$ tangential continuity. In particular, we have used a few different quadricubic interpolation methods. The simplest was to fit a tangentially continuous interpolating four dimensional tensor product cubic polynomial (Hermite) through the 256 samples surrounding the requested sample point (Figure 3.1(c)). It has the unfortunate drawback of oscillating considerably, as do all interpolating splines. We also used a quadricubic B-Spline scheme where we took the samples points to be the B-spline control mesh (Figure 3.1(d)). Although this last method is non-interpolating it produces good results because of the dense sampling over the hemisphere relative to the curvature of the BRDF. The resulting surface closely approximates the BRDF. This indicates that non-uniform sampling schemes should be examined. Figure 3.3 shows a more complex surface and the result of using B-spline bases to approximate the BRDF.

3.3.3 Sampling Density

An important issue to consider when measuring BRDFs is how many samples are required to obtain data that accurately represents a BRDF within some error tolerance. Such a figure is difficult to establish for real measured BRDFs since the gonioreflectometer can induce a considerable amount of error, both systematic and random. The effects of these errors can be reduced by making sufficient measurements, but this increases the expense of gathering BRDF data. However, for synthetic BRDFs we can generate samples from any direction and compare them against data obtained by re-sampling the corresponding BRDF, without worries about the errors inherent in a physical apparatus.

Figure 3.4 shows the root mean squared error rates between sub-sampled versions of the measured BRDFs and the BRDFs generated at the same sample points for different re-sampling methods. The RMS error is not an ideal error measure for these data, but because the measures are being used for comparison



(a)

(b)

(c)

(d)

Figure 3.1: Different interpolation methods for measured BRDFs. (a) piecewise constant; (b) quadrilinear interpolation; (c) tangentially continuous cubic tensor product polynomial; (d) tensor product cubic B-Spline; The micro-geometry, shown in Figure 3.2, is sampled at a resolution of 30 samples in ϕ_i and ϕ_r and 15 samples in θ_i and θ_r, in each of the red, green, and blue colour channels.

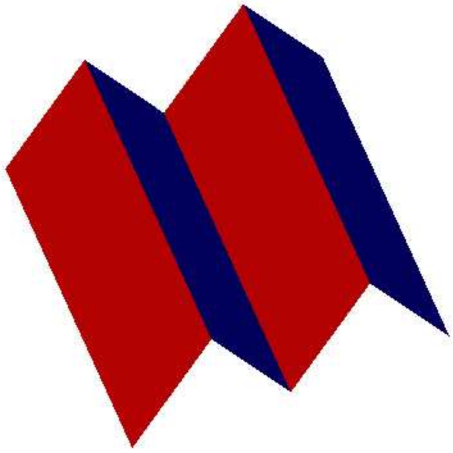


Figure 3.2: The sawtooth micro-geometry.

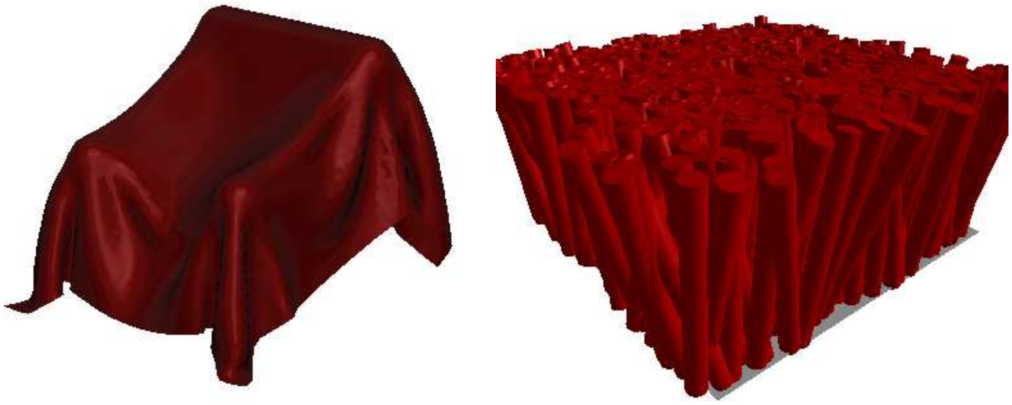


Figure 3.3: A velvet-like BRDF applied over a chair. The micro-geometry, shown to the right, is Phong shaded.

between representations of the same function this is not as much of an issue as in the case of image comparisons.

The linear interpolation performed consistently well although the the visual artifacts caused by the lack of tangential continuity were problematic. The quadrilinear case was also the fastest of the methods described (excluding the piecewise constant approach rejected because of its highly objectionable artifacts). To contrast the linear interpolation with the tensor product cubics, observe that the linear interpolation requires only 16 samples to be accessed compared to the 256 required for the tensor product cubics. Further augmenting the computational work required at each sample to evaluate a cubic polynomial causes the cubic bases to lose their attractiveness.

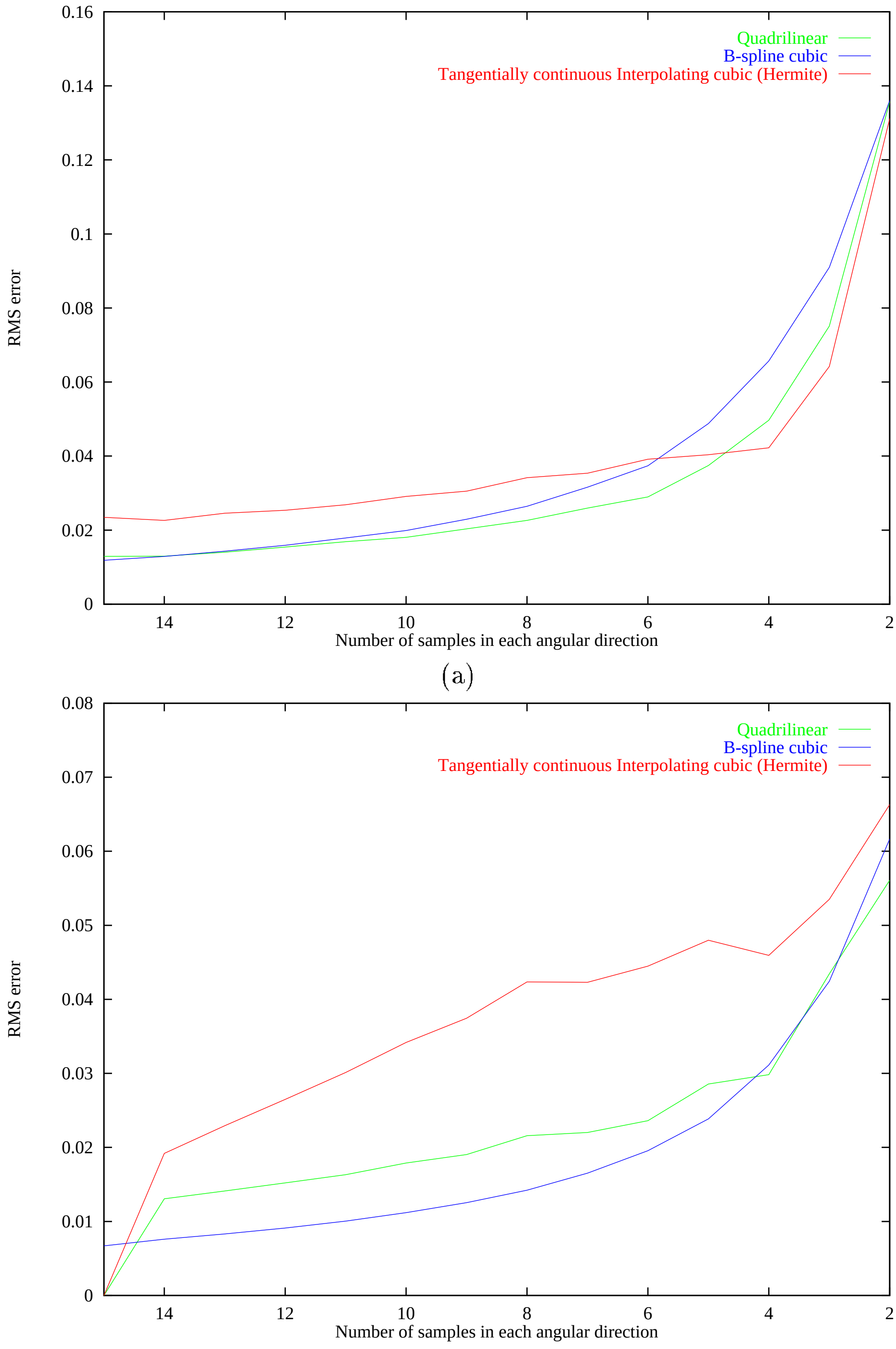
The tangentially continuous interpolating spline shows itself to be a poor choice. Interpolating splines have a tendency to oscillate considerably in their spans, and this is demonstrated in our results.

The best images overall were produced using the B-Spline bases, providing low error rates and tangential continuity. Its major drawback was the cost of computation compared to the quadrilinear basis. The cubic bases require 16 times as many operations as the linear bases, and each operation is more expensive (evaluating a cubic instead of a linear function).

3.3.4 Summary

It remains to examine whether the tabulated representations of the BRDF successfully addressed the representational issues described in Section 3.2. Examining the points in turn:

Point Reconstruction: The method allows easy evaluation of the BRDF making



(a)

(b)

Figure 3.4: Root mean squared error for two measured BRDFs: (a) BRDF generated from a sawtooth micro-geometry (Figure 3.2) originally sampled at a resolution of 30 samples in ϕ_i and ϕ_r and 15 samples in θ_i and θ_r; (b) BRDF generated from a velvet micro-geometry (Figure 3.3) originally sampled at a resolution of 15 samples in each angular direction.

it easy to integrate into point-shading computations typical to ray tracers.

Filtered Shading: The method provides no special facilities for filtering the reflection computation over incident light directions.

Compactness: The method requires large amounts of storage, making the technique infeasible when a large number of shaders are required.

Texture Maps: The method allows various forms of texture mapping to be used, but, as will be seen in chapter 7 most filtering schemes would require vast amounts of storage since the tabulations do not reduce storage requirements.

Level of Detail: The method is computationally efficient, although there is no way to trade off fidelity for additional speed without seriously compromising the fidelity. The main run-time inefficiency is caused by virtual memory thrashing when the BRDF data sets are too large or numerous.

3.4 Summary

We have presented some simple methods for using tabulated representations of BRDFs. The methods performs well in terms of point-shading evaluation and in fidelity to the original data. The compactness issue unfortunately impinges on the efficiency of the method when many BRDFs are used, causing excessive memory swaps to disk. The lack of explicit filtered shading techniques is also a drawback.

Chapter 4

A Wavelet Representation

The raw tabulated representations used in Chapter 3 fail in fulfilling our requirements for a representation. We have examined using the raw tabulations and spline representations thereof. Other researchers have examined spherical harmonics as a solution. None of these methods are sufficiently compact, nor do they simplify the process of filtering reflection computations.

We propose a wavelet based solution for a number of reasons:

1. The representation allows a good control of the accuracy/space trade-off;

2. The point-wise reconstruction is logarithmic in the number of original BRDF samples;

3. A wavelet reconstruction can smoothly interpolate the original data;

4. Error metrics are easy to compute;

5. Incremental levels of accuracy can be used; and

6. BRDFs are largely smooth with important discontinuities, which can be interpreted as a signal which is mostly low-frequency, but with localized high

frequencies. Wavelets are ideally suited to represent this kind of data as they can isolate features at various resolutions, and provide considerable compression in regions where the signal is largely constant. The goal of the remainder of this dissertation is to study the appropriateness of wavelet bases for representing the BRDF and to examine the algorithms related to shading that follow from the representation.

4.1 Parameterization

Recall that the topology of the BRDF function space is an hemisphere crossed with an hemisphere, or if we consider the transmission as well, a sphere crossed with a sphere ($S^2 \times S^2$). The most common wavelet formulations are applied on Cartesian spaces. Fortunately there are straightforward embeddings of $S^2 \times S^2$ into R^4 if one is willing to compromise on the matter of required discontinuity in the mapping. In our initial experiments, described in chapter 3, we used a polar embedding. For our wavelet based representation we will use the *Nusselt* embedding, which is closely related to the direction cosines [Lew96].

To express a direction given by a vector $\vec{v}$ in a local coordinate frame given by the normal $\vec{N}$ and surface tangent $\vec{T}$, we let $\mu_x = \vec{v}.\vec{T}$ and $\mu_y = \vec{v}.(\vec{N} \times \vec{T})$. If the angles are defined in polar and azimuthal angles (θ, ϕ) we obtain:

$$\mu_x = \sin\theta\cos\phi$$

$$\mu_y = \sin\theta\sin\phi$$

We normalize these to the range (0..1), and call them κ and λ:

$$\kappa = (\mu_x + 1.0)/2.0$$

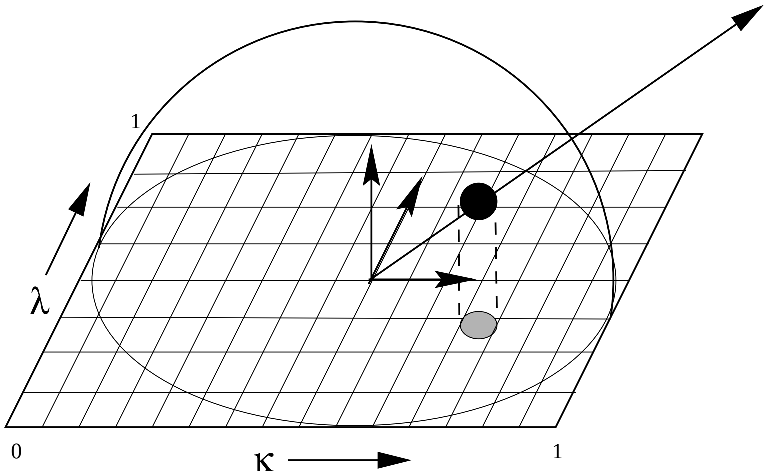


Figure 4.1: The Nusselt embedding. The given direction is mapped to a particular pair (κ, λ) by projecting the intersection with the hemisphere onto the unit square.

$$\lambda = (\mu_y + 1.0)/2.0.$$

κ and λ then index in the unit square. Directions then cover only the circle at the center of the square. We define the value of any function of directions for which $\mu_x^2 + \mu_y^2 > 1$ as zero. Figure 4.1 shows graphically how a direction maps to a position in Nusselt coordinates.

To perform the change of variable in integrals over ϕ and θ we multiply by the determinant of the Jacobian, which is:

$$\left|\frac{\partial(\theta, \phi)}{\partial(\kappa, \lambda)}\right| = \frac{4}{\cos\theta \sin\theta}$$

If we take for example Equation 2.8 expressed in terms of (θ, ϕ):

$$L_r(x, y, \theta_r, \phi_r) = \int\limits_{\theta}\int\limits_{\phi} L_i(x, y, \theta, \phi) f_r(\theta, \phi, \theta_r, \phi_r) \cos\theta d\omega \tag{4.1}$$

which becomes in Nusselt coordinates (recalling that $d\omega = \sin\theta d\theta d\phi$):

$$L_r(x, y, \kappa_r, \lambda_r) = 4 \int\limits_0^1\int\limits_0^1 L_i(x, y, \kappa, \lambda) f_r(\kappa, \lambda, \kappa_r, \lambda_r) d\kappa d\lambda \tag{4.2}$$

The Nusselt parameterization has a number of advantages. The change of variables in the local shading expressions eliminates the cosine foreshortening term, making the integrals much simpler to evaluate. The projection assigns more samples in the region of the poles, while neglecting glancing angles, where the areas are generally foreshortened and of less importance to global light levels. The Nusselt parameters are easily converted to vectors in the surface coordinate frame, requiring only two multiplications and a square root, as opposed to the trigonometric operations required when using polar embeddings.

4.2 Wavelets and BRDFs

Schröder and Sweldens showed a method for encoding the reflections from one incident direction using a spherical wavelet representation [SS95]. This restriction to one incident direction is a serious shortcoming. This method does not easily extend to encoding a large number of incident directions since the dimensionality of the function space of a BRDF does not map to the topology of their spherical space. The other difficulty is that even extending the method to higher dimensions, the basis functions of their transforms are built by subdivision of the input space. The basis functions are no longer self-similar, and cannot be composed of sets of uni-dimensional functions, which as we shall see in sections 4.2.2 and 6.3 is critical to efficient evaluation of shading computations.

We chose instead to use a multi-dimensional wavelet transform, mapping the parameter space of the BRDF to a Cartesian grid. The multi-dimensional wavelet decomposition allows a choice in how the uni-dimensional decomposition is extended to higher dimensions. In the standard case a product of one-dimensional decompositions is used. In the non-standard case a product of the basis functions is used, leading to a multi-dimensional multi-resolution analysis that is analogous to the uni-dimensional case [Dau92]. We will examine each in turn, and the algorithms that follow.

4.2.1 Standard Decomposition

The standard four-dimensional wavelet decomposition yields:

$$\mathcal{F}_r(\kappa_i, \lambda_i, \kappa_r, \lambda_r) = \sum_g \sum_h \sum_j \sum_k c_{g,h,j,k} B_g(\kappa_i) B_h(\lambda_i) B_j(\kappa_r) B_k(\lambda_r) \tag{4.3}$$

where B_n is defined as

$$B_n(x) = \begin{cases} \phi(x) & \text{if } n = 0 \\ 2^{-l/2}\psi(2^{-l}x - m) & \text{if } n = 2^l + m \text{ for some } l \text{ and } m \geq 0 \end{cases} \tag{4.4}$$

and ψ is our mother wavelet and ϕ is our smoothing function.

Consider the coefficients and basis functions that must be examined to reconstruct the BRDF at point $\mathbf{q}$ in the standard decomposition. Each coefficient $c_{g,h,j,k}$ for which any of $B_g(\kappa_i), B_h(\kappa_i), B_j(\kappa_i), B_k(\kappa_i)$ is non-zero needs to be examined. Recall that $B_n(x) = 2^{-l/2}\psi(2^{-l}x - m)$ if $n = 2^l + m$ for some l and $m \geq 0$ when $n > 0$. We can then, for each l and m, establish which functions $\psi_{l,m}(x)$ are non-zero. If the wavelet has a width of support w then there will be w basis functions to examine at each level l. Observe then that each term in the summations of equation 4.3 can reference basis functions $\psi_{l,m}()$ at different levels. This interaction means that a reconstruction costs $O(w^4 l^4)$; if there are n^4 original samples in the signal the cost is $O(w^4 \log_2^4 n)$. This cost is too high to use the standard decomposition to evaluate the value of the BRDF at a point. The non-standard transform gives more encouraging results.

4.2.2 Non-Standard Decomposition

We will express a BRDF as its projection onto a multi-resolution wavelet space. For a univariate wavelet basis, the basis functions are all derived from two functions by scaling by a factor l (we will us $l = 0$ as the coarsest level of the pyramid, that is a larger l in the discrete case corresponds to a larger number of elements) and translations by m, which can range from 0 to 2^l. One group comes from the *scaling function* $\phi()$:

$$\phi_{lm}(p) = 2^{l/2}\phi(2^l p - m)$$

and the other from the *mother wavelet function* $\psi()$:

$$\psi_{lm}(p) = 2^{l/2}\psi(2^l p - m)$$

To manage the notation for multidimensional wavelet bases, we will use a four-variable vector $\mathbf{q}$:

$$\mathbf{q} = (\kappa_i, \lambda_i, \kappa_r, \lambda_r)$$

and a multi-resolution index $\mathbf{j}$:

$$\mathbf{j} = (\nu, l, m_{\kappa_i}, m_{\lambda_i}, m_{\kappa_r}, m_{\lambda_r})$$

Here ν is the *selector index* for the wavelet functions, l is the level in the wavelet subspaces, 0 being the coarsest, and m_p the offset at that level for parameter p. The level is the same for all variables because we use non-standard multidimensional wavelet bases [Dau92, SDS95]. Since they all have an hyper-cube support (same extent for all variables at a given level) the data structure is simpler, as are many geometric operations. The selector ν determines the product of wavelet and smooth basis functions that together define $B_j(\mathbf{q})$. We introduce a special notation to indicate the selection, where $\nu[h]$ is interpreted as the hth bit of ν:

$$\Gamma^h_{l,m,\nu}(p) = \left\{ \begin{array}{ll} \phi_{l,m}(p) & \text{if } \nu[h] = 0 \\ \psi_{l,m}(p) & \text{if } \nu[h] = 1 \end{array} \right\}$$

Then we then write the basis function $F_j(\mathbf{q})$ as:

$$F_j(\mathbf{q}) = \Gamma^0_{l,m_{\kappa_i},\nu}(\kappa_i) \times \Gamma^1_{l,m_{\lambda_i},\nu}(\lambda_i) \times \Gamma^2_{l,m_{\kappa_r},\nu}(\kappa_r) \times \Gamma^3_{l,m_{\lambda_r},\nu}(\lambda_r)$$

Then the projection of the BRDF $\mathcal{F}_r$ onto the multi-resolution space is given by:

$$\mathcal{F}_r(\kappa_i, \lambda_i, \kappa_r, \lambda_r) = \sum_j f_j F_j(\kappa_i, \lambda_i, \kappa_r, \lambda_r) \tag{4.5}$$

where $F_j(.)$ are the appropriate basis functions, and $\tilde{F}_j(.)$ are their duals. The f_j are given by

$$\begin{aligned} f_j &= < \mathcal{F}_r(\kappa_i, \lambda_i, \kappa_r, \lambda_r) | \tilde{F}_j(\kappa_i, \lambda_i, \kappa_r, \lambda_r) > \\ &= \int_0^1\int_0^1\int_0^1\int_0^1 \mathcal{F}_r(\kappa_i, \lambda_i, \kappa_r, \lambda_r) \tilde{F}_j(\kappa_i, \lambda_i, \kappa_r, \lambda_r) d\kappa_i d\lambda_i d\kappa_r d\lambda_r \end{aligned} \tag{4.6}$$

Since our functions are defined as zero when $\mu_x^2 + \mu_y^2 >= 1$ we can safely narrow the bounds of integration from $(-\infty..\infty)$ to $(0..1)$.

If we now examine the time-complexity of the reconstruction at a point we find that since the basis functions $F_j()$ used in the analysis are multi-dimensional, each of the terms of the summation in equation 4.5 has a unique level l for all combinations of ψ and ϕ required to assemble $F_j()$. There are then 15 basis functions with w^4 translations at each level that support the point $\mathbf{q}$ at which we are reconstructing, where w is the width of the basis functions. This means we have only $O(w^4 \log_2 n)$ terms to examine during our traversal to evaluate at a point.

In view of these results, we choose to use only the non-standard decomposition in our investigations.

4.2.3 Choice of Basis

Our choice of wavelet basis functions is determined by two properties: the width of support of the basis, and the amount of compression a basis can provide. Since one of our goals in using the wavelet representation is to reduce the memory requirements we see immediately why greater compression is a goal. It is not so obvious why the width of support of the basis is important, or how it relates to compression rates.

Firstly, the narrower the width of support of a wavelet, the less computational work is required in the reconstruction, since the scaled and translated wavelets will cover fewer terms. When extending to higher dimensions the amount of work required by wider bases increases with the power of the dimension. The number of terms examined then becomes a dominant factor in the cost of the reconstruction. In the case of bi-orthogonal wavelets all we care about is the width of the reconstruction wavelet. The analysis wavelet can be as wide as necessary, since the wavelet transform is performed off-line using the fast wavelet transform.

Secondly, wider bases tend to compress better. The number of vanishing moments of a wavelet tells us the highest degree polynomial that the wavelet basis can represent exactly with only as many terms as the width of the basis. This ability to represent high-order polynomial correlates strongly with the amount of compression that a basis provides. In general, wavelets with more vanishing moments have a wider width of support. Thus, wider bases provide better compression, but at the cost of much more expensive computation.

It follows then that the best bases for our application will compress the data well, while being as narrow as possible. It is only possible to determine experimentally which bases are best for our application.

4.3 Representation

The remaining difficulty is in representing the coefficients. The major advantage of using a wavelet based representation is that the representation is sparse, with many of the wavelet coefficients being zero or small. To increase our compression rate at the cost of accuracy we threshold small coefficients to zero. A property of the orthogonal and bi-orthogonal wavelet transform is that the sum of these thresholded coefficients gives us the root mean square error induced in the reconstruction by ignoring the thresholded terms. If the original BRDF is f, the coefficients of its wavelet transformed representation are w_i, and f_A represents the BRDF reconstructed by using only the coefficients in the set A then the error is given by for some constants c_0 and c_1:

$$c_0(\sum_{(i) \notin A} |w_i|^2)^{1/2} <= ||f - f_A|| <= c_1(\sum_{(i) \notin A} |w_i|^2)^{1/2}. \tag{4.7}$$

The difficulty is that there are as many wavelet coefficients as original data points, even though a large number of them may be zero. Thus we want to find a representation that stores only the non-zero coefficients, allowing us considerable economies in storage.

4.3.1 Hash Table

One solution to the problem of storing zeros is to use a hash table representation of the array containing only the non-zero coefficients. We use an open-addressing scheme keyed on the index of the coefficient. This scheme is compact, requiring only storing the non-zero entries and the keys[Knu73].

In an open addressing scheme the keys are stored in a table at an address given by some hash function $h(k, i)$ where k is the key, and i is a count of the number of probes required to find an empty slot. If the table is only lightly loaded most keys will hash to an empty slot on the first try. If a collision occurs, i is incremented and the process repeats. To do a lookup in the table the same sequence of hashes is examined until either the key is found at the position $h(k,i)$ in the table or an empty slot is found, at which time it is known that the key is not in the table, a negative probe.

In our application we expect most of our searches to have a negative result, indicating that the coefficient is zero. It is then important that negative searches be very quick. If the table is too full then negative probes will have to check a large number of entries and performance will be seriously affected.

Instead we allocate a table that is about twice as large as the number of non-zero coefficients. Since the expected number of probes required for a negative result is given by $1/(1-\alpha)$ where α is the load factor (ratio of full slots to table length), we find that with $\alpha = 0.5$ that a negative probe is expected to take 2 lookups. We feel that this is an acceptable trade-off.

We choose for our table size a prime number m about twice the size of our number of non-zero coefficients in our thresholded wavelet representation. Then our hash function is:

$$h(k,i) = h_1(k) + i * h_2(k)$$

$$h_1(k) = k \bmod m$$

$$h_2(k) = k \bmod (m-1)$$

Since m and $h_2(k)$ are relatively prime $h(k,i), i = 1..m$ will probe the entire table before repeating itself. See Knuth for more details [Knu73].

Unfortunately, if our data set is sparse most of our lookups are expected to return negative probes. This means that our most expensive lookups are most common. A simple extension to address this would be to structure our evaluation in a tree structure and to augment out hash table with tree-structure information. At this point though, it becomes more efficient to store the tree explicitly.

4.3.2 Wavelet Coefficient Tree

In a one dimensional transform it is possible to use a binary tree structure that at each node records the value of the coefficient and mimics the structure of the wavelet analysis. Then it is possible to prune any subtrees that contain only zero coefficients. There may remain nodes in the tree that record zero values (if any of its descendants are non-zero) but these add no more than a small linear factor to the size of the stored tree. For reconstruction it is sufficient to traverse a path down the tree, following a child pointer if the child's basis function supports the point being evaluated. We let this structure inspire us in higher dimensions.

An efficient option for the representation is as a *Wavelet Coefficient Tree.* This tree is a sparse hexa-decary tree (16 ordered children), where the depth of the nodes indicates the depth of the coefficients in the representation, and where each node stores the non-zero coefficients with the same offset m, indexed by ν, the wavelet basis selector.

We use a structure called a *wavelet node* that stores all non-zero wavelet coefficients with the same indices (l, m_0, m_1, m_2, m_3), and different basis selector ν. The wavelet node also maintains a list of non-empty subtrees rooted at a wavelet node with indices $l{+}1, 2m_0{+}b\{0\}, 2m_1{+}b\{1\}, 2m_2{+}b\{2\}, 2m_3{+}b\{3\}$ where $b = 0..15$. The addition of a 15-bit mask indicating which coefficients are present (the value mask) and and a 16 bit mask indicating which children (the child mask) are present allow very compact storage of the tree.

```
function Eval(BRDFTree brdf, vector d)
{
    result = 0;
    for b = 1 .. 15
        if (brdf->valuemask & (0x01>>b))
            j = (b,brdf->l,brdf->m[0..4]);
            result += brdf->value[b] * B_j(d);
    for b = 0 .. 15
        if (brdf->childMask & (0x01>>b))
            j = (b, brdf->level+1, brdf->m[0] * 2 + ((b>>0)&0x01),
                                   brdf->m[1] * 2 + ((b>>1)&0x01),
                                   brdf->m[2] * 2 + ((b>>2)&0x01),
                                   brdf->m[3] * 2 + ((b>>3)&0x01));
            // Check the support of B_j()
            if (InSupportOf_B_j(d))
                result += Eval(brdf->child[b], d);
    return result;
}
```

Figure 4.2: Reconstruction of a 4-variable Wavelet compressed function at point **q**

This method is both simple to implement and efficient. The tree traversal for an evaluation is comprised of a number of bit-mask operations and tests for coverage by the relevant child's basis function. This coverage operation can frequently be implemented much more efficiently than a basis function evaluation. For efficiency the non-Haar wavelet and scaling functions $\Gamma(x)$ are tabulated off line, and basis function evaluation becomes a table lookup after scaling and translating the parameter x. Haar bases are coded algorithmically because of their simplicity. Fig. 4.2 shows the traversal pseudo-code for a point-evaluation.

4.4 Compressing the BRDF

Now, with a data structure that can store sparse wavelet coefficients, and an algorithm for point reconstruction, we must briefly examine the thresholding operations that we hope will allow us to compress BRDF efficiently using wavelets.

A key result for approximation is that the wavelet coefficients of a sufficiently smooth function decay by levels at least as a power of 2^N, provided ψ has N vanishing moments:

$$\mathrm{max}_j |w_{ij}| \leq C\ 2^{-iN}.$$

The rate of decay is governed by the number of vanishing moments of the wavelet used. As an example, Haar wavelets have only one vanishing moment, which means that they do not approximate functions very rapidly. Similarly, the Haar coefficients do not tend to zero fast at finer levels, so Haar wavelets will not produce as marked a contrast in coefficient size between smooth and non-smooth sections of data as wavelets with more vanishing moments.

A consequence of this is a bound on the error if we approximate a function with a limited set of coefficients. If f is the function to be approximated, A is the set of coefficients (i) chosen to be in the approximating function f_A, the L^2 norm of the error is bounded by

$$c_0(\sum_{(i)\notin A} |w_i|^2\ )^{1/2} <= ||f - f_A|| <= c_1(\sum_{(i)\notin A} |w_i|^2\ )^{1/2}.$$

This means that the L^2-error will be smallest when the n largest wavelet coefficients are chosen for the approximation. This corresponds to simple threshold compression of the wavelet coefficient information. From the above results it is clear that for smooth data, compression rates improve as the number of vanishing moments of the wavelet increases. It remains to show that these results apply to BRDF data.

4.5 Wavelet Compression Results

Our results fall into three categories: BRDF size reduction vs. error in the BRDF, BRDF size reduction vs. error in the resulting image, and timings. We will examine these in turn, after having described our test data.

4.5.1 Test Data

The BRDFs we used to test our method were generated using a virtual gonioreflectometer [WAT92, DLF96]. This process gives us considerable control over the data generated. We can easily capture the same BRDF at different sampling resolutions and be assured that measurement errors are not an issue.

The BRDFs we examined are the sawtooth and the velvet from chapter 3, and a Phong shader with an exponent of 30. These were sampled at a resolution of 32 samples for each of $\kappa_i, \lambda_i, \lambda_r$ and κ_r.

4.5.2 BRDF Size Reduction

The measure of success of our BRDF representation is of compression vs. error, both in the representation of the BRDF, and in images shaded with the BRDF. Figure 4.4 shows compression vs. error rates for the sawtooth BRDF and various wavelet bases, where the error is the root-mean-square (RMS) of the difference between the complete BRDF and the thresholded BRDF. The important points to note from these plots is that the simple bases with narrow support perform quite competitively with the broader bases with more vanishing moments. In particular, the Daubechies basis with 4 vanishing moments is not as efficient as the simple linear spline[1] . The added cost of evaluating the Daubechies basis however (about

[1] This is the spline 2,2 from page 277 of [Dau92], shown in Figure B.2

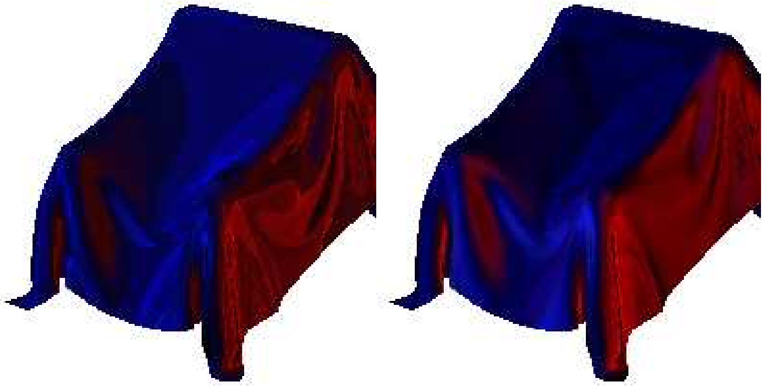


Figure 4.3: Haar versus linear spline. Both BRDFs are thresholded to 0.1% of original size

16 times the work — the Daubechies basis has about twice the coverage of the linear spline) makes this a gain of only marginal utility. Overall the linear spline bases seem to be adequate to compressing BRDF data to usable sizes with few visible artifacts. Figure 4.3 contrasts the Haar and linear spline bases, each with the wavelet coefficients thresholded to 0.1% of the original number of coefficients. The smooth interpolation of the spline helps mask the artifacts visible from the Haar's piecewise constant basis functions.

Table 4.1 shows a comparison of running times and accuracy in the rendered image with thresholding. The image RMS error is computed from pixel values, scaled to the range (0..1], of the difference images, shown in Fig. 4.5.

4.5.3 Images

The suite of images in figure 4.5 shows a chair[2] shaded with each of the BRDFs and at increasing levels of accuracy. The light source is directional. The corresponding

[2] Chair data is courtesy of David Breen at the Caltech Computer Graphics Lab.

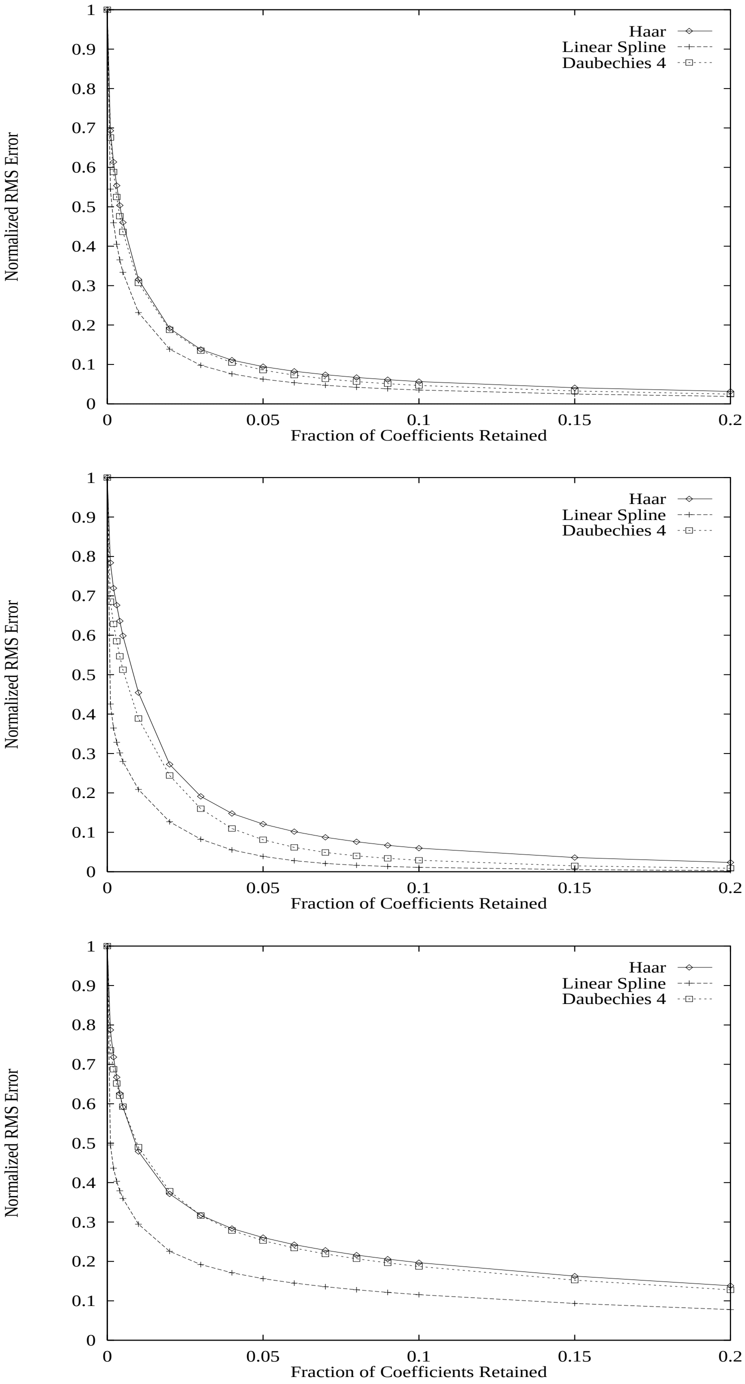


Figure 4.4: Compression vs. Percent total error. From top to bottom: the sawtooth BRDF, a Phong illumination model with exponent 30, and a velvet BRDF.

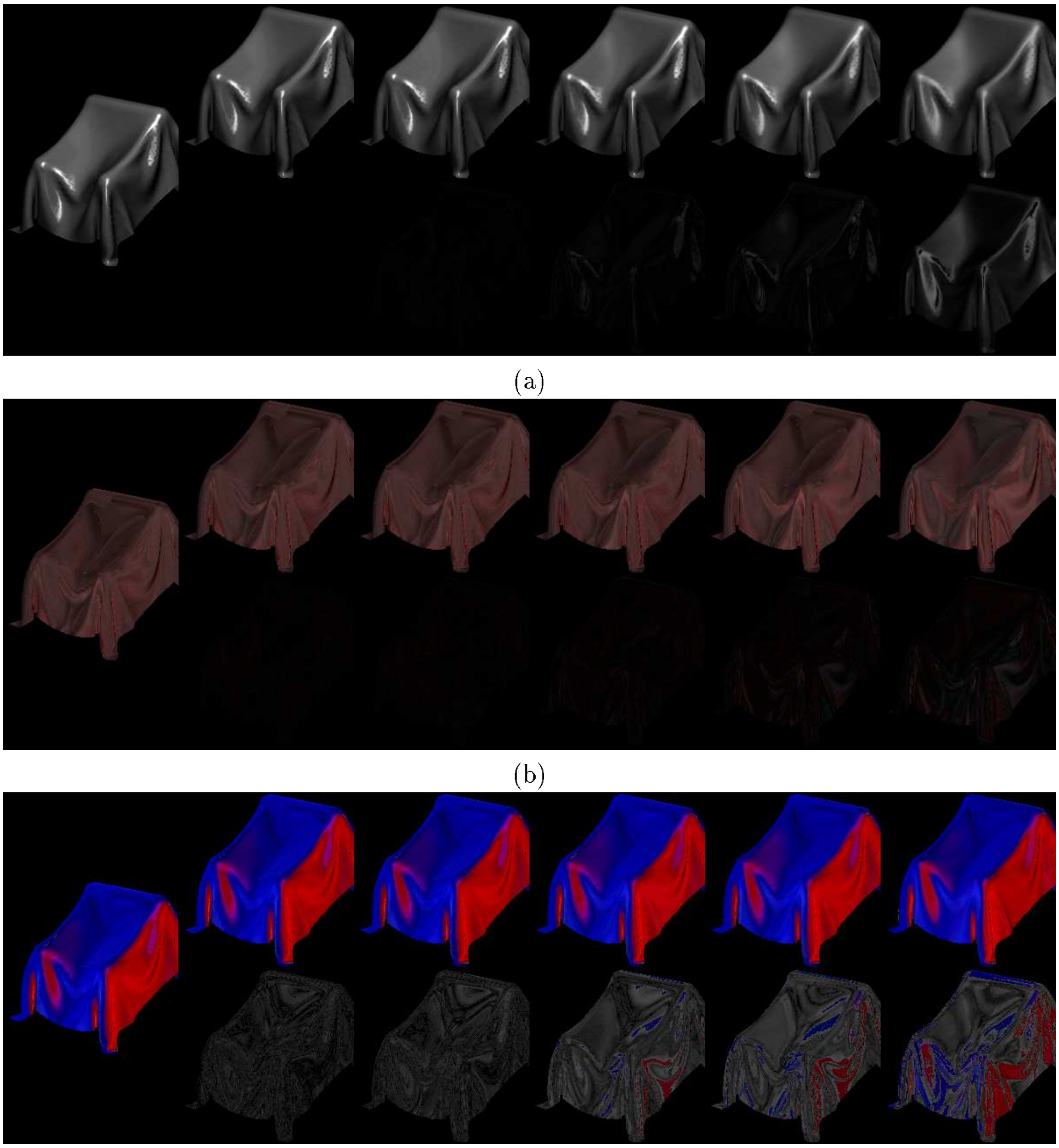
BRDF	Size	# Coef.	Time	Image RMS	BRDF RMS
Phong (n=50)	1.0	150933	50.9	0.0000	0.0000
	0.2	30183	36.4	0.0020	0.0244
	0.1	15051	30.1	0.0031	0.0549
	0.05	7542	22.7	0.0054	0.1049
	0.01	1503	12.5	0.0159	0.2334
	0.005	747	8.8	0.0236	0.2814
	0.001	147	3.5	0.0825	0.4331
	0.0001	12	1.5	0.1877	0.7486
Velvet	1.0	93911	30.0	0.0000	0.0000
	0.20	18779	16.2	0.0010	0.0437
	0.10	9388	13.2	0.0028	0.0944
	0.05	4692	10.9	0.0029	0.1612
	0.01	936	6.4	0.0152	0.2984
	0.005	467	5.5	0.0216	0.3535
	0.001	91	2.8	0.0362	0.5160
	0.0001	6	1.3	0.0776	0.8279
Sawtooth	1.0	127334	40.4	0.0000	0.0000
	0.2	25463	22.8	0.0075	0.0386
	0.1	12728	17.7	0.0149	0.0719
	0.05	6363	13.0	0.0202	0.1207
	0.01	1268	6.7	0.0352	0.3479
	0.005	632	5.2	0.0459	0.4639
	0.001	123	2.8	0.0762	0.6809
	0.0001	8	1.5	0.1486	0.9231

Table 4.1: Comparison of running times vs thresholding and accuracy. Size is given as the fraction of original non-zero coefficients; time is in milliseconds per pixel on a 200 mhz MIPS R4400 processor; RMS errors are calculated after normalization of data to the range (0..1]. The BRDF RMS is for the red channel.

difference images below them show the absolute difference from the chair rendered from the unthresholded BRDF. The most important point to note is that the images are consistently good when the BRDFs are compressed to 1% of their original sizes. Even at 0.1% the surfaces are still recognizable. The relationship between the size of the BRDF and the speed of evaluation makes these highly compressed BRDFs even more attractive.

4.6 Summary

In this chapter we have presented a wavelet based representation of the BRDF. The original BRDF data are transformed into the wavelet domain using a multi-dimensional non-standard wavelet transform. The coefficients are then stored in a sparse hexa-decary tree that does not store subtrees whose nodes are all zero-valued. The representation allows rapid evaluation of the BRDF at a point and provides interpolation between originally sampled points. The reconstruction at a point costs $O(\log n)$ time, where there are n^4 original samples. The method allows a simple trade-off between space used and error induced by the lossy compression.



(a)

(b)

(c)

Figure 4.5: A chair shaded with our wavelet compressed BRDFs, and the difference images. From left to right the images are generated with BRDFs thresholded to 10%, 5%, 1%, 0.5%, and 0.1% or their original sizes. The (a) is a Phong shader with an exponent of 50,(b) a velvet, and (c) is a sawtooth, red on one side, blue on the other.

Chapter 5

Light Field Representations

Recent developments have made image-based rendering techniques useful as a supplement to traditional rendering methods. The main idea is that an object can be represented as a series of views of itself and that its appearance in a rendered scene is generated by interpolating a new view from these views and compositing this view into the image[GGSC96, LH96]. One of the main difficulties of this approach is that to obtain good reconstructions from many different directions a dense sampling of the space of views is required. This quickly leads to an explosion in the amount of data to be stored and in the reconstruction times. Our interest is to use techniques similar to those used for storing and reconstructing BRDFs to store and reconstruct light field objects.

We are interested not only in generating new views of the objects, but also in using such objects as light sources in a scene. To date light field objects have only been used to add views of objects into a scene. Recognizing that a light field object (LFO) is a representation of all light emitted from the volume surrounding the object allows us to treat the object as a light source. In addition, this kind of representation is appropriate for maintaining a representation of light arriving at a surface, which can then be used to shade the surface, as will be shown in chapter 6.

5.1 Representing Light Field Objects

The problems of representing light fields and of representing the light emitted from a lamp are closely related to BRDF representations. Both require representing a function of 4 variables, some or all of which are directional. In the case of light fields we exchange two of the directional parameters of the BRDF with positional parameters on some surface. Evaluating the direct illumination at a point $\mathbf{x}$ due to an object L_e is done by integrating the product of the BRDF $\mathcal{F}$ and the light arriving from the surface area of the object. This is a restating of Equation 2.8 integrating over points in space rather than directions:

$$L_r(\mathbf{x}, \vec{\omega_i}) = \int_{\mathbf{x}' \in L_e} \mathcal{F}((\mathbf{x}' - \mathbf{x}), \vec{\omega_r}) L_e(\mathbf{x}', (\mathbf{x} - \mathbf{x}')) cos\theta_i d\mathbf{x}'. \tag{5.1}$$

We will revisit this equation in chapter 6. This chapter examines the problem of efficiently representing the distribution of light emitted by a luminaire, corresponding to the $L_e(.)$ term of equation 5.1.

One particularly suitable application of LFOs is in modeling the luminaires used to illuminate a scene. Luminaires are available in a variety of shapes, sizes, with vastly different lighting characteristics. Although the emissive surface of the lamp (the filament or excited gas) plays an important role in determining the characteristics of the emitted light a very significant role is played by the rest of the lighting fixture—the lamp shade and reflectors. The fixtures focus or diffuse light, allowing many variations on how the light emitted from an incandescent bulb or fluorescent tube will illuminate a scene. Although it is possible to model the fixture as part of the scene and to render it like any other surface in the scene this approach has major drawbacks. Because of the intensity of the light near its source and the proximity to the surfaces of the fixture, any variations in the fixture surfaces tend to be magnified and be made visible in the outgoing distribution of light—the distribution of light from a faceted lamp shade will be visibly different than that from a continuously curved one. Consider for example a lamp with an imperfect bulb that is rippled. The light projected would have brighter and dimmer areas, depending on how the light was focused or diffused by the imperfections in the glass. The rippling is nearly invisible geometrically, and appears only at a very fine scale, but clearly apparent in the light falling on the wall. Likewise, any errors in the computations involving the source and the other parts of the luminaire will be magnified throughout the scene. In the case where a fixture has to be simulated, the fine scale geometry of the real fixture will affect how the lamp emits light, but the computational and storage costs caused by the geometry involved may be prohibitive to accurate simulation.

Many rendering techniques are also poorly suited to the accurate simulation of scenes that include complete luminaires. Traditional ray tracers do not handle reflective lamp shades at all—the lighting calculation traces a ray to the light source (typically a point) and cannot determine if a surface is reflecting a sufficient amount of light to be worth of considering as a light source. Radiosity engines are particularly ill equipped because of the $O(n^2)$ (or $O(n \log n)$ for hierarchical radiosity) dependence on the number of elements in the discretization required to simulate high frequency lighting changes [HSA91]. The refinement required on a luminaire tends to be prohibitive for radiosity methods.

In order to represent efficiently the light emitted by a luminaire it is necessary to express the radiance emitted by the luminaire in all directions. Were all light sources points this would be an easy measurement requiring only the sampling of the illumination in the sphere surrounding the point. Instead, since most luminaires have a finite size, we have to measure the emissivity in every direction from every point on the luminaire's surface. This quickly becomes unwieldy, more so when the luminaire is not convex, as lighting calculations using the luminaire are complicated by the need to integrate over the surface of the luminaire.

Fortunately, it is possible to represent a luminaire by its bounding volume. The radiant flux at each point of its boundary is measured. The interaction of light in any volume can be completely replaced by a description of the radiant flux at every point on the boundary of the volume. This is quite similar to the approaches taken by Levoy and Hanrahan [LH96] and Gortler *et al.* [GGSC96], that they use to represent objects that are sampled from multiple viewpoints.

5.2 Representation Requirements

The representation chosen for the LFO is closely related to the intended application. Bearing in mind that we are interested using our light field objects primarily as light sources in ray-tracing and radiosity type applications, we propose the following criteria.

Units

We are particularly interested in the steady state solutions of light transfer. As such, we want a unit of light that is independent of time. Also, we will frequently be point sampling our representations, and do not want to have to account for area each time. As power is measured in Watts, the appropriate units for our representation is in Watts per meter squared per solid angle.

Accuracy

The method must provide an accurate representation of the luminaire. In particular, the user must be able to specify the tolerances to which the representation must adhere. The method should not introduce artifacts into the shaded scene. Since light sources are critical to the illumination in a scene any artifacts must be quantifiable.

Compactness

The representation must be sufficiently compact that representing many LFOs in one scene is reasonable. A raw fine-grained sampling of the object in space and direction is therefore inadequate.

Efficiency

Computing the shading at a point due to a luminaire must be fast—this is the most frequent operation in most rendering systems. Our representation should also allow other shading computations to be done efficiently.

Geometric approximation

The representation should allow some form of a geometric approximation for the luminaire, in order to calculate inter-reflection effects of the scene upon the luminaire. This becomes more relevant when the luminaires are large compared to the scene. Usually the light reflected at a luminaire from elsewhere in a scene is considerably less important than the light emitted from the luminaire itself, mitigating the inaccuracies due to coarse geometry.

5.3 LFO Approximations

A representation of a light field object is a dense sampling in both positions and directions on the surface of the lamp's boundary surface. The approaches examined in this thesis explicitly parameterize directions, using the Nusselt parameterization described in section 4.1. We choose to use this to parameterize directions on our luminaires because this reduces the number of coordinate changes required when using our BRDF representation. Various parameterizations of position on the bounding surface can be easily substituted without radically changing the representation of the directional distributions.

5.3.1 Spherical Bounds

One obvious boundary is a bounding sphere enclosing the luminaire. There are two simple ways to parameterize the sphere: by polar coordinates, or by two Nusselt-parameterized hemispheres. The disadvantage of this sampling is that a polar embedding of the sphere of directions is non-uniformly sampled with regards to its spatial embedding, allocating too much work at the poles. Since a luminaire will frequently not have preferred viewing positions this is problematic. Likewise, a Nusselt parameterized sphere is also non-uniform in its sampling, and artifacts are introduced at the horizon. The Nusselt embedding also requires two hemispheres to be recorded and computed with separately in order to represent a sphere. For these reasons, and the fact that domains without these problems exist, we shall not examine spherical spaces any further.

5.3.2 Rectilinear domains

An important class of light field objects is formed by the “light through a window” parameterization. Here the light field object is measured as if seen through a window. The most common geometry used is a rectilinear region, although other parameterizable domains can be used. This domain is particularly useful to represent light passing through a region of space which forms a particularly good basis for accumulating light arriving at a surface. This will be used extensively in chapter 6. A ray tracer can easily be adapted to compute such light fields. The parameterization is also easily mapped onto objects, making the integration of light field object luminaires into a scene quite simple.

5.3.3 Polyhedral domains

Another representation to use for the boundary surface is a polyhedral bounding volume. In particular a bounding box is good. Each face of the box is easily discretized, and the directional distribution gathered over it. For any point in the scene at most three faces are visible, reducing the shading from a lamp to the shading from three area sources with spatially-varying emittance distributions. More complex polyhedra could be used, such as a convex hull of the luminaire, but as the number of faces increases, the number of area source-like surfaces that must be considered increases rapidly. Using only one wall of the bounding box trivially allows surface mounted luminaires to be used.

5.3.4 Arbitrary domains

A related representation allows the mapping of a luminaire onto an arbitrary object. By taking a luminaire representation parameterized by u and v and identifying these parameters with the parameters of a geometric object, the object can be made to have a spatially and directionally varying emissivity distribution. There are a few difficulties with this approach however. Sampling issues (addressed in Chapter 8) are complicated by the geometry. The mapping from the object's parametric space to world space is usually not the same as the mapping from the luminaire's original space to its u, v parameterization, inducing distortions in the brightness of the luminaire over the surface. A re-parameterization may help this, but is not easy to obtain in general.

5.4 Sampling Luminaires

Ashdown has shown a system based on CCD video cameras and a moving gantry for measuring the emissivity of a luminaire over an encircling sphere [Ash92]. His method records video images from a number of directions, and these then need to be transformed into positional and directional samples. The camera registration and lens distortion effects can cause serious artifacts. These problems may also appear in the Gortler *et al.*'s Lumigraph [GGSC96] as their data are collected with a hand-held video camera.

Instead of building a physical instrument we have chosen to instrument our test-bed ray tracer to sample detailed geometries of luminaires. Similar measurements can be made with Lucifer, a global illumination software package that uses a wavelet representation of light, with the added benefit of obtaining our data already in the wavelet domain [Lew96]. Both these methods make measurements completely repeatable and lets us ignore calibration issues allowing us to concentrate on representational issues. Using our system we sampled 3 luminaires at a resolution of 32 samples in each of u, v, κ, and λ:

- A simple lamp shade made of spheres;
- A ceiling mounted fluorescent lamp with sharp baffles; and
- A simple room environment with a red wall, a blue wall, a green floor, and a sphere with a sawtooth BRDF.

A view of each is show in Figure 5.1.

5.5 Luminaire Representation

Directional radiance distributions have many of the same properties as BRDFs. Over much of their angular range luminaires are smoothly varying, but any discontinuities are important. Thus, we choose to adapt our wavelet representation of BRDFs to the problem of representing the exiting flux through a surface parameterized by positional parameters (u, v) ranging from 0 to 1, and in directions (κ, λ), again ranging from 0 to 1.

The four-variable vector $\mathbf{q}$ corresponds to $\mathbf{d}$ of section 4.2.2:

$$\mathbf{q} = (u, v, \kappa_i, \lambda_i)$$

and the multi-resolution index $\mathbf{k}$ corresponds to $\mathbf{j}$ of of section 4.2.2:

$$\mathbf{k} = (\nu, l, m_u, m_v, m_{\kappa_i}, m_{\lambda_i}).$$

Then the radiance $L(\mathbf{q})$ can be expressed as its projection onto a multi-resolution wavelet space:

$$L(\mathbf{q}) = \sum_k b_k \tilde{B}_k(\mathbf{q}),$$

entirely analog to the case of the BRDF. The b_k'are are given by:

$$b_k \quad = \quad < L_r(u, v, \kappa_i, \lambda_i) | \tilde{B}_k(u, v, \kappa_i, \lambda_i) >$$

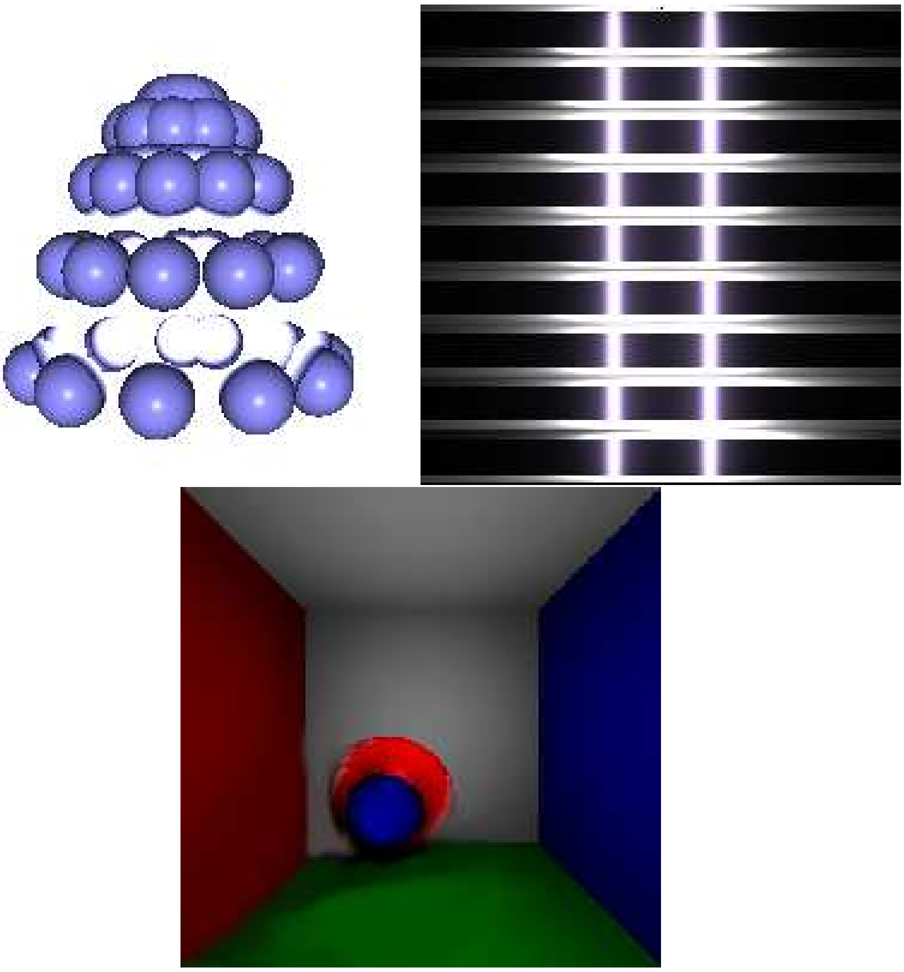


Figure 5.1: The three objects used in our experiments.

$$= \int_0^1\int_0^1\int_0^1\int_0^1 L_r(u,v,\kappa_i,\lambda_i)\tilde{B_k}(u,v,\kappa_i,\lambda_i)dudvd\kappa_i d\lambda_i \qquad (5.2)$$

Likewise, the basis functions B_k are given by:

$$B_k(\mathbf{q}) = \Gamma^0_{l,m_u,\nu}(u) \times \Gamma^1_{l,m_n,\nu}(v) \times \Gamma^2_{l,m_{\kappa_i},\nu}(\kappa_i) \times \Gamma^3_{l,m_{\lambda_i},\nu}(\lambda_i).$$

This is precisely what Lewis *et al.* do [Lew96], but they only use these radiance representations on flat surfaces. The same data structures then apply to radiance distributions as to BRDFs. Lewis *et al.* and Christensen *et al.* also restrict their wavelet representation to Haar wavelets. We will not do so, except in specific circumstance when general wavelet bases are simply too computationally expensive to use. For instance if the radiance distributions are compressed using non-Haar wavelets, the direct evaluation of the illumination equation becomes dramatically more expensive, as will be seen in Chapter 6.

The reconstruction algorithm is entirely analogous to the BRDF reconstruction.

5.6 Results

One of the goals of the luminaire representation is to be able to take a complex luminaire and sample it in a reasonable fashion that will allow integration of complicated luminaire geometry without an inordinate amount of work for the renderer. Figure 5.2 shows our three test luminaires at different compression levels. The top row is at 100%, the center at 10% of original size, and the last at 1% of original size. Figure 5.4 shows compression rates of luminaires plotted against the RMS error induced.

A viewer for our data sets achieves interactive rates at reconstructing views from our luminaire information. Figure 5.3 shows screen captures of the application at work. The mouse position determines the viewing direction, and the image is redrawn in incremental steps. Instead of using the mouse to determine viewing positions it should be possible to use a head-tracking interface to complete the illusion of looking at light through a window.

Further results will be presented in Chapter 6 where local shading computations are examined.

5.7 Summary

This chapter has shown how to use a wavelet representation similar to that presented in Chapter 4 to store and reconstruct the light field through a surface. The representation works particularly well for rectangular luminaires and representations of light through a window. As in Chapter 4 the representation allows a tradeoff of space versus accuracy.

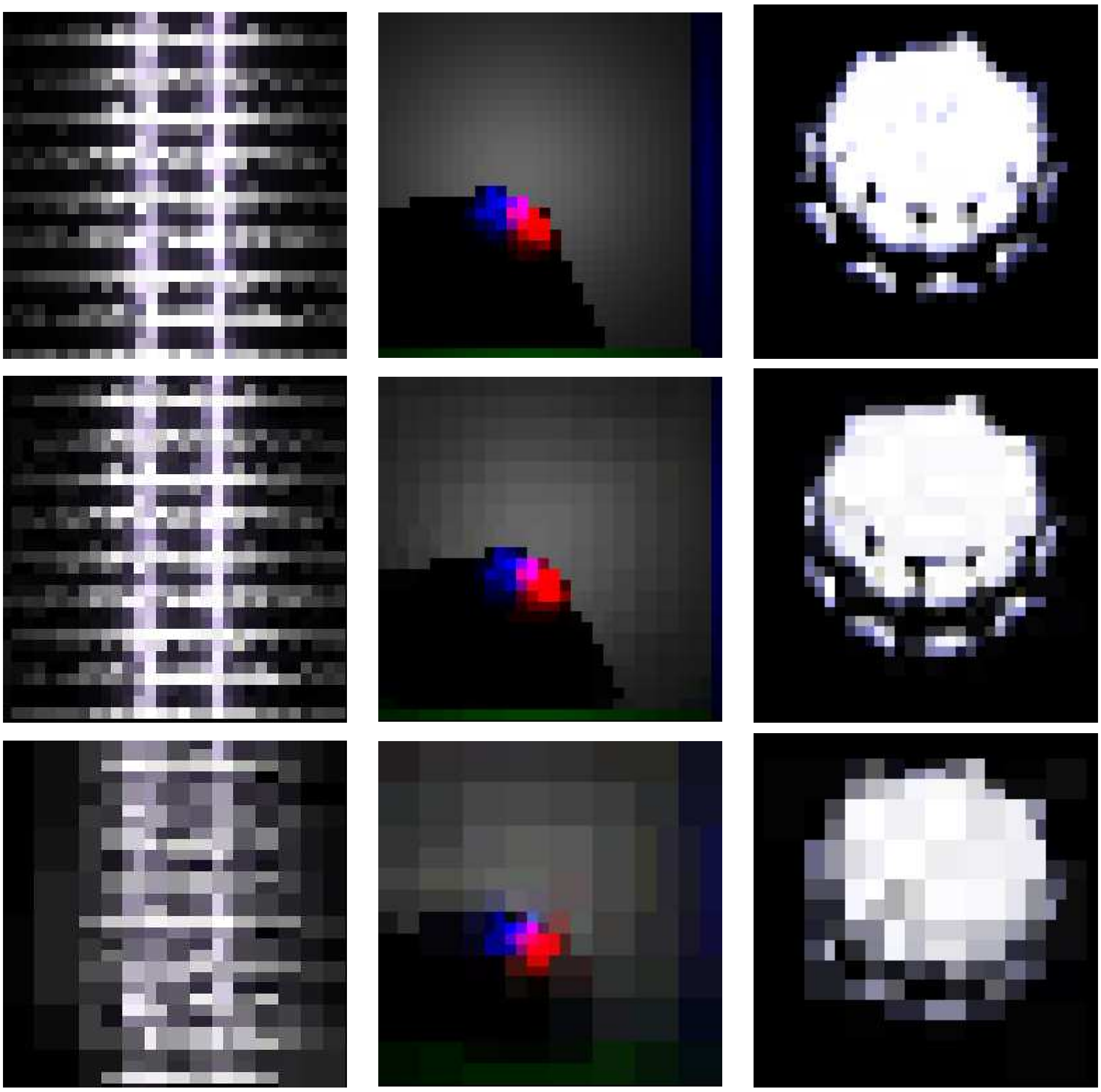


Figure 5.2: Three Luminaires at (from top to bottom) 100%, 10% and 1% of original sizes. The lamp is viewed from nearly below.

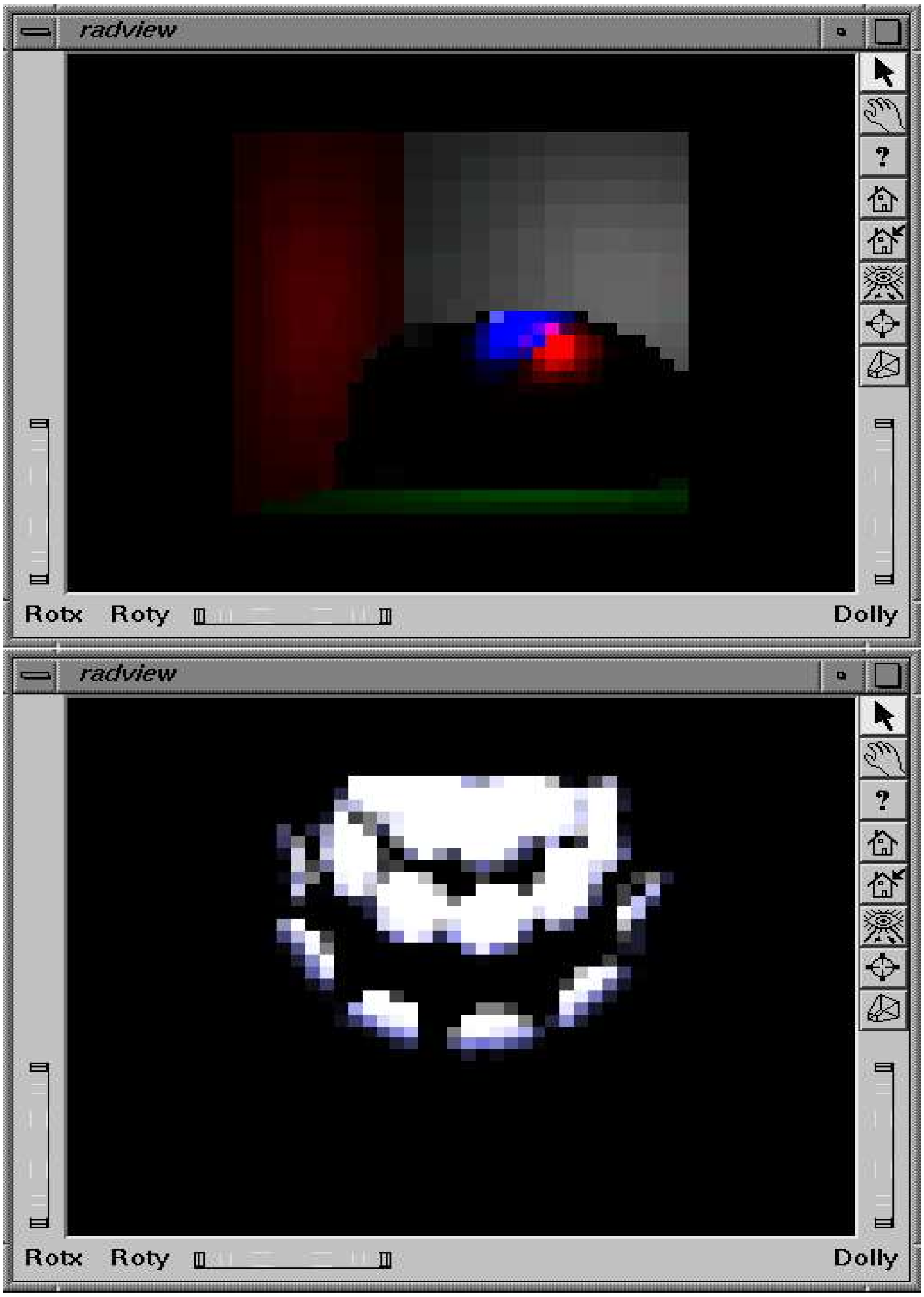


Figure 5.3: 2 Screen shots of RadView, an interactive program for viewing light field objects and luminaires. A view of the room scene is shown above and a bottom view of the sphere luminaire below.

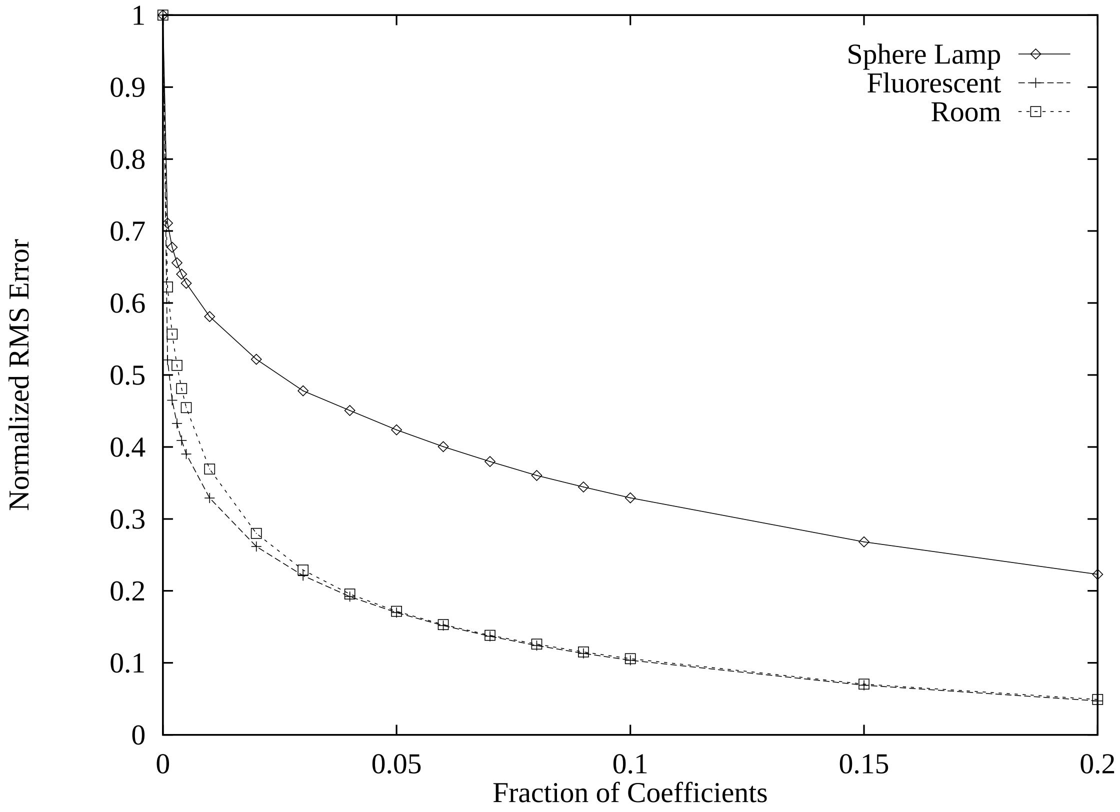


Figure 5.4: Compression rates vs. error for our three objects.

Chapter 6

Local Illumination and Filtering

With BRDFs and light sources now defined, it is possible to examine the process of surface shading. Consider again equation 2.8, without the wavelength dependence:

$$L_r(\vec{\omega_r}) = \int_{\Omega_N} L_i(\vec{\omega_i}) f_r(\vec{\omega_i} \rightarrow \vec{\omega_r}) \cos\theta_i d\omega_i \tag{6.1}$$

The challenge of local shading is to evaluate this integral efficiently at every point on a surface. Different assumptions about the incident light function $L_i()$ lead to various different solutions. Early ray tracers assumed either point or directional lights, while more sophisticated techniques such as Monte Carlo ray tracing [Kaj86], extended radiosity methods [ICG86, CRMT91], and photon tracing [Jen95], for example, allow more general formulations.

6.1 Point Lights

Computing the illumination at a point due to a point light source is straightforward. Recall from section 2.2.1 that radiance $L_i(\vec{\omega_i})$ at a point when the direction to the source is $\vec{\omega_s}$ is given by:

$$L_i(\vec{\omega_i}) = \frac{I(-\vec{\omega}_s)}{r^2}\delta(\cos\theta - \cos\theta_s)\delta(\phi - \phi_s) \tag{6.2}$$

Substituting into equation 2.8:

$$L_r(\vec{\omega_r}) = \int_{\Omega_N} \frac{I(\vec{\omega_s})}{r^2}\delta(\cos\theta - \cos\theta_s)\delta(\phi - \phi_s) f_r(\vec{\omega_i} \rightarrow \vec{\omega_r}) \cos\theta_i d\omega_i \tag{6.3}$$

Rewriting with the Nusselt parameterization and integrating we find:

$$L_r(\kappa_r, \lambda_r) = 4\frac{I(\kappa_s, \lambda_s)}{r^2} f_r(\kappa_s, \lambda_s, \kappa_r, \lambda_r) \tag{6.4}$$

6.2 Light Field Objects

The most frequent use of a luminaire is to compute the direct illumination at a point due to that luminaire. This is an evaluation of the shading equation for a limited part of the hemisphere. This section is concerned with how to use our luminaire representation to evaluate the direct lighting at a point.

Consider again the shading equation but with the addition of a parameter $\mathbf{x}$ denoting the point being shaded, in world coordinates:

$$L_r(\mathbf{x}, \vec{\omega_r}) = \int_{\Omega_N} L_i(\mathbf{x}, \vec{\omega_i}) f_r(\vec{\omega_i} \rightarrow \vec{\omega_r}) \cos\theta_i d\omega_i \tag{6.5}$$

If we are interested only in direct illumination due to a particular luminaire $\mathcal{L}_e$ we can rewrite this as:

$$L_r(\mathbf{x}, \vec{\omega_r}) = \int_{\mathbf{x}' \in L_e} \mathcal{F}_r(\overrightarrow{(\mathbf{x}' - \mathbf{x})} \rightarrow \vec{\omega_r}) L_e(\mathbf{x}', \overrightarrow{(\mathbf{x} - \mathbf{x}')}) \frac{g(\mathbf{x}, \mathbf{x}') \cos\theta_i \cos\theta_e}{||x' - x||^2} d\mathbf{x}' \tag{6.6}$$

where $g(\mathbf{x}, \mathbf{x}')$ is a geometry term accounting for occlusion (both self-occlusion and blockers) that returns 1 if the path from $\mathbf{x}$ to $\mathbf{x}'$ is unobstructed and zero otherwise.

Converting to Nusselt parameters and making explicit that the integration is over parameters u, v of the light source:

$$L_r(x, y, \kappa_r, \lambda_r) = 4 \int_0^1 \int_0^1 \mathcal{F}_r(\kappa(u,v), \lambda(u,v), \kappa_r, \lambda_r) L_e(u, v, \kappa(u,v), \lambda(u,v)) \times \frac{g(\mathbf{x}, \mathbf{x}') \cos\theta_i \cos\theta_e}{||x' - x||^2} du dv \tag{6.7}$$

where $\mathbf{x}$ and $\mathbf{x}'$ are derived from the parameters on their respective surfaces, and the functions $\kappa()$ and $\lambda()$ establish the Nusselt parameters corresponding to the direction to the point being shaded. In all but the simplest geometries, these coordinate transformations make the integral analytically intractable. Chapter 8 examines the application of Monte Carlo integration to solving this integral.

6.3 Radiance Field

Local shading can be computed efficiently if the incident radiance field is known. Currently, both Lucifer [LF96] and wavelet radiosity methods [GSCH93] can be adapted to generate wavelet-represented radiance fields. We will examine the particular case where the incident field is described using a planar wavelet projected radiance field presented in section 5.3.2. Recall that the incident radiance is represented as a function $L_i(u, v, \kappa, \lambda)$, and the BRDF is given by $\mathcal{F}_r(\kappa, \lambda, \kappa_r, \lambda_r)$ where all the parameters range from 0 to 1. To shade in the direction of the eye where $\vec{e} = (u, v, \kappa_r, \lambda_r)$ where (u, v) are the spatial coordinates of the point being shaded, and (κ_r, λ_r) is the direction to the eye from that point, then the required radiance is:

$$L_r(\mathbf{e}) = L_r(u, v, \kappa_r, \lambda_r) = 4 \int_0^1 \int_0^1 \mathcal{F}_r(\kappa_i, \lambda_i, \kappa_r, \lambda_r) L_i(u, v, \kappa, \lambda) d\kappa d\lambda \tag{2.8}$$

Recall that the wavelet projection of L_i is:

$$L_i(\mathbf{q}) = \sum_{\mathbf{k}} b_k B_k(\mathbf{q})$$

where $\mathbf{q}$ is the vector:

$$\mathbf{q} = (u, v, \kappa, \lambda)$$

and k is the index vector:

$$\mathbf{k} = (v^k, l^k, m_u^k, m_v^k, m_\kappa^k, m_\lambda^k)$$

Likewise the projection of the BRDF is given by:

$$\mathcal{F}_r(\mathbf{d}) = \sum_{\mathbf{j}} f_j F_j(\mathbf{d})$$

where $\mathbf{d}$ is the vector:

$$\mathbf{d} = (\kappa, \lambda, \kappa_r, \lambda_r)$$

and $\mathbf{j}$ is the index vector:

$$\mathbf{j} = (v^j, l^j, m_\kappa^j, m_\lambda^j, m_{\kappa_r}^j, m_{\lambda_r}^j)$$

Then equation 2.8 becomes:

$$L_r(u, v, \kappa_r, \lambda_r) = 4 \sum_{\mathbf{j}} \sum_{\mathbf{k}} f_{\mathbf{k}} b_{\mathbf{j}} \int_0^1 \int_0^1 F_{\mathbf{k}}(\mathbf{d}) B_{\mathbf{j}}(\mathbf{q}) d\kappa d\lambda$$

Using the $\Gamma()$ notation to expand the basis functions $F_{\mathbf{j}}$ and $B_{\mathbf{k}}$ we arrive at:

$$\begin{aligned} L_r(u, v, \kappa_r, \lambda_r) = & \\ 4 \textstyle\sum_{\mathbf{j}} \sum_{\mathbf{k}} f_{\mathbf{k}} b_{\mathbf{j}} \iint_{00}^{11} \quad & \Gamma^0_{l^j m^j_{\kappa_i} \nu^j}(\kappa) \Gamma^1_{l^j m^j_{\lambda_i} \nu^j}(\lambda) \Gamma^2_{l^j m^j_{\kappa_r} \nu^j}(\kappa_r) \Gamma^3_{l^j m^j_{\lambda_r} \nu^j}(\lambda_r) \times \\ & \Gamma^0_{l^k m^k_u \nu^k}(u) \Gamma^1_{l^k m^k_v \nu^k}(v) \Gamma^2_{l^k m^k_\kappa \nu^k}(\kappa) \Gamma^3_{l^k m^k_\lambda \nu^k}(\lambda) d\kappa d\lambda \qquad (6.8) \end{aligned}$$

Leaving only functions of κ and λ in the integrals:

$$
\begin{aligned}
L_r(u, v, \kappa_r, \lambda_r) \quad = \quad & 4 \sum_{\mathbf{j}} \sum_{\mathbf{k}} f_{\mathbf{k}} b_{\mathbf{j}} \Gamma^2_{l^j m^j_{\kappa_r} \nu^j}(\kappa_r) \Gamma^3_{l^j m^j_{\lambda_r} nu^j}(\lambda_r) \Gamma^0_{l^k m^k_u \nu^k}(u) \Gamma^1_{l^k m^k_v \nu^k}(v) \\
& \int\limits_0^1 \Gamma^0_{l^j m^j_\kappa \nu^j}(\kappa) \Gamma^2_{l^k m^k_\kappa \nu^k}(\kappa) d\kappa \int\limits_0^1 \Gamma^1_{l^j m^j_\lambda \nu^j}(\lambda) \Gamma^3_{l^k m^k_\lambda \nu^k}(\lambda) d\lambda \qquad (6.9)
\end{aligned}
$$

Recall that the bases for the radiance (the $\vec{k}$ indices) were constrained to be Haar bases while the BRDF bases may be arbitrary. The $\Gamma()$ terms outside the integrals are evaluations of the basis. In most cases one of these will be 0 for bases of finite support. The integrals can be computed efficiently by tabulating the integral of the non-Haar bases. To evaluate, the range of integration is first clipped against either the whole range of the Haar smooth function and integrated, or against the two halves of the Haar wavelet separately and summed. If the radiance wavelet were not restricted to Haar this integration would become more expensive, as the tabulation of product of all the basis functions would be required. The additional width of the radiance wavelet would also increase the number of wavelets at every level that share support with the BRDF wavelets, further increasing the computational expense. The cost of the method would be prohibitive because of the exponential growth of number of coefficients covered at finer levels of the wavelet transform. Thresholding helps this, but only by a linear amount.

The implementation now becomes a set of nested traversals of the radiance tree and of the BRDF tree, comparing the supports of the children's bases before recursing. At each pair of nodes the summand of equation 6.3 is computed. Fig. 6.1 shows pseudo-code for the traversal.

```
float RadTraverse(RadianceTree wct, BRDFTree brdf, u, v, kappa, lambda)
    // u and v are parametric positions on the surface being shaded,
    // in wct coordinates
    // kappa and lambda indicate the direction to the eye
{
    result = BRDFTraverse(brdf, wct, u, v, kappa, lambda);

    // Here the recursion checks for support against the positional
    // components, u and v, at indices U and V in the m vector,
    // which is the m terms from the index vector k in the text.
    // The zero and one bits of b index the U and V indices of m.
    for b = 0..15
        if (wct->childmask & (0x01>>b))
            // test our given u,v against the
            // support of the child basis given by b
            m[U] = wct->m[U] * 2 + ((b>>0)&0x01);
            m[V] = wct->m[V] * 2 + ((b>>1)&0x01);
            if (InHaarSupport(wct->level,m[U],u)
             && InHaarSupport(wct->level,m[V],v))
                result += RadTraverse(wct->child[b],brdf,
                                      u,v,kappa,lambda);
    return result;
}

float BRDFTraverse(BRDFTree brdf, RadianceTree wct, u, v, kappa, lambda)
{
    result = Integrate(brdf, wct, u, v, kappa, lambda);

    // Here the recursion checks for support against the reflected
    // directions, kappa and lambda, at indices Kr and Lr in the m
    // vector, which is the m terms from the index vector j in the text.
    // Bits 2 and 3 of b index the Kr and Lr indices of m.
    for b = 0..15
        if (brdf->childmask & (0x01>>b))
            m[Kr] = brdf->m[Kr] * 2 + ((b>>2)&0x01);
            m[Lr] = brdf->m[Lr] * 2 + ((b>>2)&0x01);
            if (InGammaSupport(brdf->level,m[Kr],kappa)
             && InGammaSupport(brdf->level,m[Lr],lambda))
                result += BRDfTraverse(brdf->child[b],wct,
                                       u,v,kappa,lambda);
    return result;
}
```

Figure 6.1: Tree-traversal evaluation of nested summations of equation 6.3. The function Integrate() calculates the summand of equation 6.3.

6.4 Results

Fig. 6.2 show the results of applying this computation. The image shows the back wall of a cubic room with a green floor, a red left wall and a blue right wall, similar to the classic Cornell box [GTGB84]. The BRDF is Lambertian, which is to say that the BRDF is a constant. The radiance field was sampled with a ray tracer at a resolution of 32 by 32, both positionally and angularly. The image on the left is computed without thresholding, on the right with. The unthresholded image took 90 milliseconds per pixel, the thresholded one 17 milliseconds per pixel.

Fig. 6.3 shows the back wall of a similar room for different Phong exponents. The wall was shaded from an incident light field. The important part to note is the glossy reflections. As the Phong exponent increases the images become sharper.

Fig. 6.4 shows the result of a scene with a wall illuminated by an area light source in the presence of a blocker. Again, the radiance field was sampled at 32 by 32 positionally and angularly. The BRDF was sampled at 16 samples per angular component. Note the effect of the filtering. The dimmer regions correspond to regions where the light was partially occluded. The upper left image, with wavelet coefficients thresholded to 10% of original size, took 113 milliseconds per pixel, the upper right 54 milliseconds per pixel at 1% of original size. The bottom left shows the same scene with the BRDF represented with a linear spline basis, thresholded to 0.5% of original size.

An examination of the running times shows that thresholding does vastly improve running times with very little degradation in image quality. Using non-Haar bases for the BRDF is too expensive to be considered reasonable, particularly as the sampling rates of the radiance field and of the BRDF increase.

In this chapter we have shown how to use the BRDF representation of Chap-

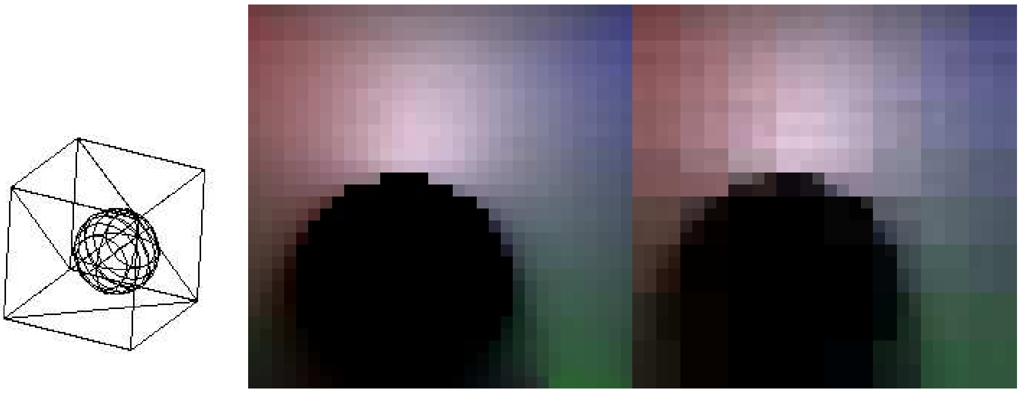


Figure 6.2: Orthographic view of a Lambertian wall in a cubic room. A sphere close to the wall occludes much of the room. Leftmost, a wire-frame view of the geometry. Center, the shaded view of the back wall, computed without thresholding. Right, the back wall shaded with the incident radiance thresholded to 1% of the original size.

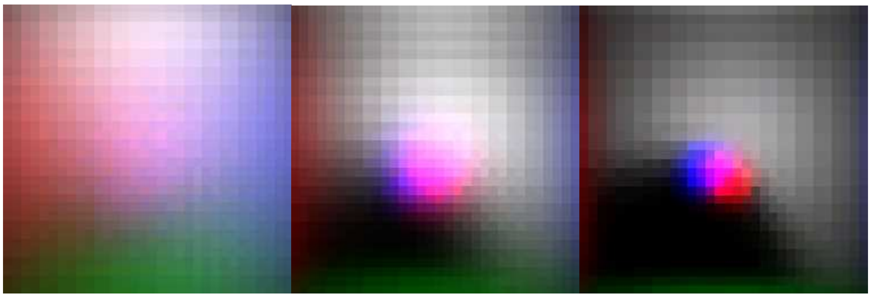


Figure 6.3: The back wall of a room, with Phong BRDFs. From left to right the exponents are 5, 50, and 200.

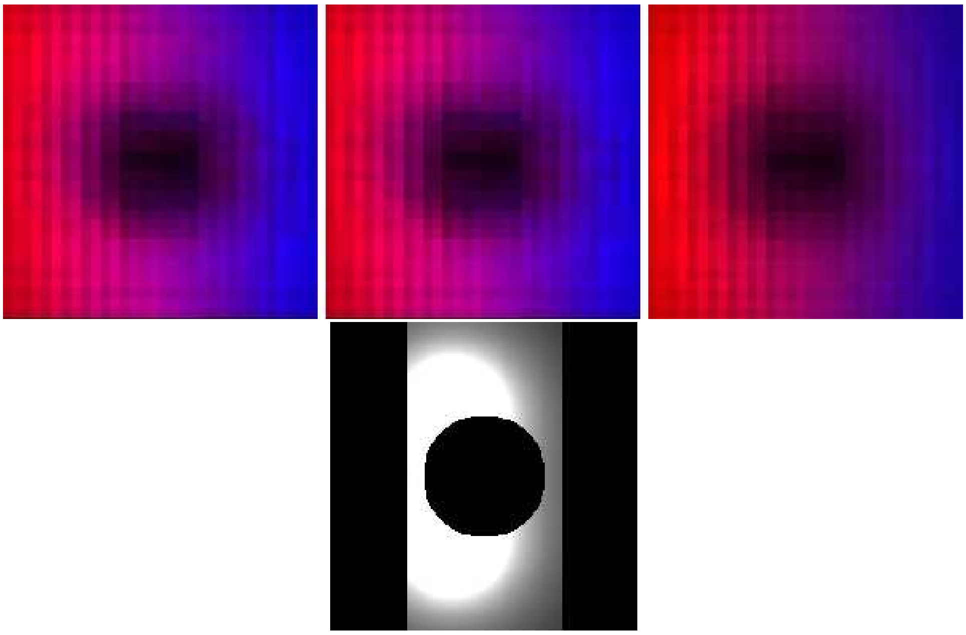


Figure 6.4: A wall illuminated by an area light source. A sphere partially occludes the light source. The BRDF is generated from a sawtooth micro-geometry. The top-left image is computed without thresholding. In the top-center the incident radiance and the BRDF thresholded to 1% of the original size. The top right image is the result of thresholding the incident radiance to 1% of the original size, and using a linear spline basis (spline 2,2 from Daubechies, p. 277) for the BRDF, thresholded to 5% of its original size. The bottom image shows a view of the geometry for reference.

ter 4 and the luminaire representation of Chapter 5 to efficiently evaluate the shading equation at a point. The luminaire representation is used to describe the light arriving at a surface from all directions. The BRDF is then used to mediate the interaction. The sparse nature of the wavelet transforms, as well as some observations about how the wavelet basis functions can be decomposed into separable parts within the shading integral, allow us to evaluate the integral directly, without sampling the light field and BRDF at points, as previous general solutions to this integral have had to. The solution is sufficiently fast to replace point sampling methods, and can be integrated into such rendering algorithms as Lucifer [LF96] and wavelet radiosity [GSCH93] that generate wavelet representations of incident light.

Chapter 7

Filtering Textures

The use of texture maps has a long history in computer graphics [BN76]. First used to provide more interesting surfaces by varying the colour of a surface according to its local parameterization, texture maps have since been applied as methods of modifying many parameters in analytic shading models [Hec86]. In general texture maps add detail to the surface appearance without increasing the geometric complexity. Whenever texture maps are used some sort of filtering is required to overcome sampling problems. Some of these techniques can be applied to surfaces whose properties are defined by measured BRDFs.

There are 5 ways in which BRDFs and filtering can interact:

1. Treat a colour map as a colour modifier over the exiting (or incident) light at a surface element, as in Figure 7.1 where the mandrill was applied over a BRDF-specified Phong shaded plane; existing texture map filtering technologies apply [Wil83, Fou92, GH86];

2. A texture map can vary the parameters of an analytical shader. If the parameters are linearly separable from the shading equation then methods from (1)

apply, otherwise considerable knowledge of the shader is required and it is not obvious how to filter them. We will not examine this option further;

3. The BRDF can vary over the surface according to a texture map: entries in the texture map are indices into a table of BRDFs. Figure 7.2 shows a checkerboard texture made from a Blinn-Phong sampled BRDF and the sawtooth BRDF from Figure 3.1(e);

4. The normal at the surface can be perturbed, requiring some filtering of the BRDF to adequately capture these perturbations, particularly in those cases where highlights are induced;

5. A complete micro-geometry can be specified over the surface, and this needs to be filtered. Measured BRDF based techniques provide useful tools for this.

The cases addressed in this dissertation are items three, four and five. Methods three and four will be shown to be subsets of method five.

Two approaches present themselves for filtering BRDFs: pre-filtering and filtering at the time of rendering. We will examine pre-filtering in particular, since existing methods for filtering at rendering time are easily extended to include BRDF surfaces. The disadvantage of rendering time methods is that they can be very expensive, as their cost is usually related to the number of texture pixels covered by a screen pixel. When pre-filtering, however, the cost per pixel is essentially constant [Hec86]. The drawback is that when pre-filtering we must assume that the light source direction and the viewer direction do not change appreciably over the surface area subtended the the pixel being shaded. This assumption may be wrong, particularly in cases where point light sources are very near the surfaces and when the viewer is very far away from the object, or viewing it at a glancing angle.

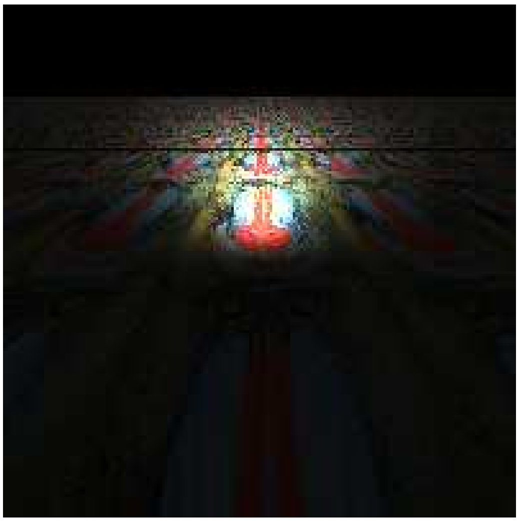


Figure 7.1: The plane is illuminated by a directional source. The BRDF is a tabulated monochromatic Phong shader with a low ambient term. The colour of the mandrill texture map modulates the monochromatic BRDF. No anti-aliasing is performed.

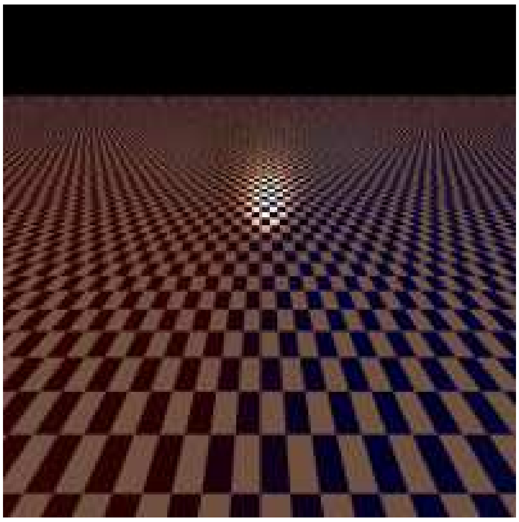


Figure 7.2: A checkerboard texture made of a Blinn-Phong shader and a sawtooth texture, filtered by super-sampling at 16 samples per pixel.

This assumption is implicit in all pre-filtering methods, without which no linear separation of light and viewer directions and shader parameters can be made.

7.1 Linearity and BRDFs

Most view and light independent filtering techniques need the parameter being filtered to cause only a linear effect on the shading of a surface. Consider for example a surface shaded using the Phong illumination model. If the property being changed is the exponent in the cosine term then the texture cannot be pre-filtered using traditional linear methods. If on the other hand the textured property is the specular scaling term it is easy to see that the linear methods apply:

$$\begin{aligned} \rho &= \frac{k_{s1}(N \cdot H)^n + k_{s2}(N \cdot H)^n}{2} && (7.1) \\ &= \frac{k_{s1} + k_{s2}}{2}(N \cdot H)^n && (7.2) \end{aligned}$$

If we want to pre-filter BRDFs independently of light and view directions we must show that the tabulated values can be factored from the point-shading equation. Rewriting equation 2.7 with explicit directional dependences we get:

$$dL_r(\vec{\omega_r}) = \mathcal{F}_r(\vec{\omega_i}, \vec{\omega_r}) dE(\vec{\omega_i})$$

Substituting for $dE(\lambda)$ (Equation 2.1):

$$dL_r(\vec{\omega_r}) = \mathcal{F}_r(\vec{\omega_i}, \vec{\omega_r}) L_i(\vec{\omega_i}) \cos\theta_i d\omega_i \qquad (7.3)$$

Consider a surface composed of n different materials each with BRDF $\mathcal{F}_j, j = 1..n$, each of which subtends a fraction a_j of the surface under the pixel being rendered. Then the light reflected from a direction $\vec{\omega_i}$ in direction $\vec{\omega_r}$ at that pixel is given by:

$$dL_r(\vec{\omega_r}) = \sum_{j=1}^{n} a_j \mathcal{F}_j(\vec{\omega_i} \rightarrow \vec{\omega_r}) L_i(\vec{\omega_i}, \lambda) \cos\theta_i d\omega_i \tag{7.4}$$

$$= L_i(\vec{\omega_i}) \cos\theta_i d\omega_i \sum_{i=1}^{n} a_j \mathcal{F}_j(\vec{\omega_i} \rightarrow \vec{\omega_r}). \tag{7.5}$$

We see that the BRDFs can be extracted linearly, and summed separately to yield an equivalent BRDF for the surface when viewed from far enough away that the individual parts of the inhomogeneous surface are not visible.

The last operation required to filter BRDFs independently of viewer or light source positions is to compute the weighted sum of BRDFs. The sum required is the point-wise sum of the BRDF functions. In a case where the BRDFs are represented by point samples on the same grids a point-wise addition is sufficient. Things are more complicated when the samples fall on different grids, or when the BRDFs are approximated by more complex bases.

The case of particular interest to us is how to perform efficiently the summation when the BRDFs are represented using wavelet coefficients. If the BRDFs are sampled on the same grid and transformed with the same wavelet function, then the summation is simple. Wavelets are sums of linear terms, which makes addition in the wavelet domain the same as addition in the untransformed domain:

$$\mathcal{F}_{r1}(\mathbf{d}) + \mathcal{F}_{r2}(\mathbf{d}) = \sum_j f_{1j} F_j(\mathbf{d}) + \sum_j f_{2j} F_j(\mathbf{d}) \tag{7.6}$$

$$= \sum_j (f_{1j} + f_{2j}) F_j(\mathbf{d}) \tag{7.7}$$

Thus we can sum wavelet coefficients to find our representation of the summed BRDFs, saving the considerable computational expense of reconstructing the original BRDFs, summing them, and transforming back to the wavelet domain. Fortunately, in practice our BRDFs are sampled on the same grids. It is difficult to sample

BRDFs at more than 64 samples in each of $\kappa_i, \lambda_i, \kappa_r$ and λ_r. The next reasonable increment to perform the wavelet transform without requiring extensive padding of the data is at 128 samples each, requiring 2^{28} samples, which is prohibitive. Our results show that a resolution of 32 samples per parameter is frequently sufficient, and that 64 works quite well. This leads us to use BRDFs that are sampled at the same rates, obviating the need to sum BRDFs sampled on different grids. Scaling a BRDF data set up to the next sampled resolution is easy in the wavelet domain, as this can be done trivially by setting the new coefficients to zero. In our representation none of these subtrees will be stored, and so the representation of a 32x32x32x32 BRDF is identical to its scaled up 64x64x64x64 representation.

7.2 Pre-filtering BRDF Textures

Williams presented a texture mapping system for colour maps based on pyramidal data structures where each larger level of the pyramid represents extra detail [Wil83]. By choosing the texture map at the appropriate level of detail to use at each pixel the colour texture can be adequately filtered. Williams' MIP (*multum in parvo*) maps store a pixel-based texture at the lowest level of the pyramid and filtered versions of the texture at higher levels. This approach bears a resemblance to a wavelet scheme. We could extend our BRDF representation to include two more variables - the texture parameters u and v on the surface. Then by integrating over the range of u and v visible from a pixel a correctly shaded pixel could be obtained. However, the advantages of the wavelet transform quickly disappear as the number of dimensions increases. Recall from section 4.2.2 that the cost of a non-standard wavelet transform's reconstruction at a point includes a w^D term. Even for Haar, this constant becomes large. Integrating over u and v would increase the cost beyond practicality.

Fortunately, a MIP-map like approach can be applied to BRDF textures. Our texture maps stores indices into a table of BRDFs. We then use a MIP-map scheme to filter the texture. At each level of the pyramid we store the BRDFs indexed by the texture. The top level of the pyramid contains one measured BRDF, that of the surface when viewed from sufficiently far away that texel differences are invisible. Each level below it stores a more detailed map of BRDFs and the BRDFs necessary to shade the texels in that map. Since the BRDF has been shown to be linearly separable from the shading equation (Section 7.1) and that the wavelet-represented BRDFs can easily be added, we can apply this pyramidal scheme to our measured BRDFs. Each BRDF in the coarser level $l+1$ of the pyramid can be built from the finer level l's BRDFs (u and v are texel indices in the texture maps):

$$\mathcal{F}_{r}{}^{l+1}_{(u,v)}(\mathbf{d}) = \frac{1}{4}\sum_{i,j=0,1} \mathcal{F}_{r}{}^{l}_{(2u+i,2v+j)}(\mathbf{d}) \tag{7.8}$$

In essence, we have pre-computed a number of integrals over u and v without using the wavelet transform to store the BRDF as a function of u and v.

To use the pyramid the area subtended by a pixel being rendered is estimated, as per Williams [Wil83]. A non-integral index into the depth of the pyramid is derived so that a texel at level μ would correspond in size to the pixel being rendered. Consider filtering a texture on a surface as seen from a particular pixel. We project a circular filter at the pixel against the plane tangential to the point on the surface corresponding to the center of the pixel. This yields an ellipse[1]. A measure of the size of the ellipse (in this case the maximum of the major and minor axis lengths) is used to determine the level μ of the pyramid at which one screen pixel would subtend

[1] This is not strictly true — this yields a conic section, which may by a hyperbola or a parabola. We can nevertheless get a measure of the area subtended in screen space.

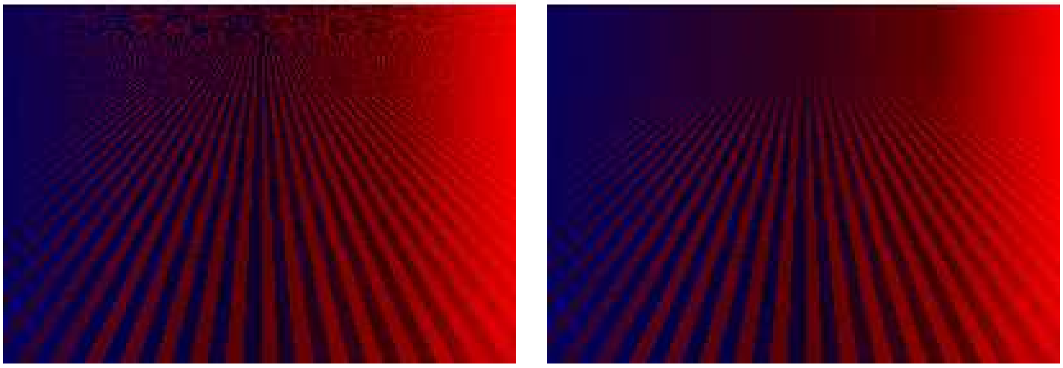


Figure 7.3: A plane shaded with a sawtooth BRDF, with and without the pyramidal filtering scheme. The BRDF was generated from the micro-geometry. The base texture is 4x4.

one texture sample. Since this μ does not necessarily correspond to a computed level of the pyramid, possibly being non-integral, ${\mathcal{F}_r}^{\mu}_{(u,v)}$ is approximated by a weighted sum of the two nearest levels:

$${\mathcal{F}_r}^{\mu}_{(u,v)} = (1.0 - (\mu - \lfloor\mu\rfloor)){\mathcal{F}_r}^{\lfloor\mu\rfloor}_{(u,v)} + (\mu - \lfloor\mu\rfloor){\mathcal{F}_r}^{\lceil\mu\rceil}_{(u,v)} \tag{7.9}$$

The entire BRDF ${\mathcal{F}_r}^{\mu}_{(u,v)}$ is never calculated. Instead ${\mathcal{F}_r}^{\mu}_{(u,v)}$ is only evaluated for the required incident and reflected directions:

$${\mathcal{F}_r}^{\mu}_{(u,v)}(\mathbf{d}) = (1.0 - (\mu - \lfloor\mu\rfloor)){\mathcal{F}_r}^{\lfloor\mu\rfloor}_{(u,v)}(\mathbf{d}) + (\mu - \lfloor\mu\rfloor){\mathcal{F}_r}^{\lceil\mu\rceil}_{(u,v)}(\mathbf{d}). \tag{7.10}$$

7.3 Filtering Micro-Geometry Textures

We now will consider using our measured BRDF representation to filter micro-geometries. One model for reflective surfaces is as a small scale micro-geometry where each facet is shaded using a simpler reflection model. Previous work has

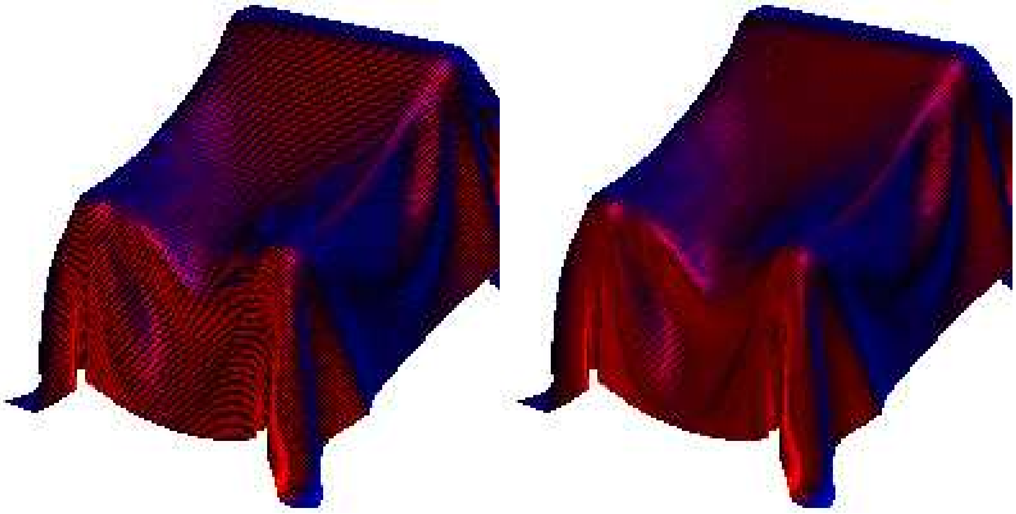


Figure 7.4: A chair shaded with the sawtooth BRDF, with and without the pyramidal filtering scheme.

shown how to sample such a micro-geometry to generate a BRDF to use in further shading [CMS87, Kaj85, WAT92, GMN94]. However, relatively little work has been done on rendering within the transitional area where the surface goes from being best represented by its detailed geometry to being best represented by some aggregate BRDF. The BRDF map presented above is a simplified case of this, where the micro-geometry is flat.

Consider the sawtooth pattern in Figure 3.2. Recall that a BRDF can be measured for a micro-geometry with a virtual gonioreflectometer. This allows us to shade a surface that is viewed from far enough away that the micro-geometry pattern itself is invisible. Only the overall shading effect is visible in this case. If however the surface is close enough that the details of the micro-geometry should be visible, then we need to build a texture to represent it. This can be done by sampling subregions of the micro-geometry to acquire a BRDF representative of those subregions, and then using this texture to shade the object. The difficulty arises when the viewer is in the zone of transition when parts of the micro-geometry may be visible while others might not. In this case some sort of filtering is necessary. Failure to do so yields artifacts like those seen in Figure 7.5.

A second issue that arises in this kind of filtering is masking and self-shadowing. The micro-geometry may occlude itself and light sources in various ways. Care must be taken when generating filtered versions of the texture that these effects are adequately captured. Simply averaging BRDFs will not provide this. Consider a surface with a sharply ridged strip running down its center. On either side the BRDF may not include shadowing effects, and if the BRDFs are averaged light might still fall on the flat area that should be blocked by the ridge.

We will use the pyramidal scheme presented in Section 7.2 to store the textures, although the BRDFs at each level will be computed from the micro-geometries rather than from the level below. To build the pyramid we use a virtual gonio-reflectometer, sampling appropriate sub-regions of the micro-geometry to generate the various BRDFs in the pyramid. This approach guarantees that self-shadowing and inter-reflection are properly accounted for.

Let the BRDF of a particular texel (u, v) of a unit square texture within level l be ${\mathcal{F}_r}^l_{(u,v)}$, for integral l. The micro-geometry represented by ${\mathcal{F}_r}^l_{(u,v)}$ is the rectangle r from $(\frac{u}{2^l}, \frac{v}{2^l})$ to $(\frac{u+1}{2^l}, \frac{v+1}{2^l})$, where the complete geometry of the texture map is scaled to the unit square.

Figure 7.3 shows a plane textured without and with our sawtooth texture using the pyramidal scheme. Figure 7.4 shows the same texture applied to a chair.

7.4 Bump Maps

Bump maps perturb the normal of each point on a surface before shading calculations are performed. Each texel of the bump map stores a perturbation vector which is added to the computed normal for the surface before computing the shading. Bump maps are more challenging to filter than colour maps, and the same techniques do not generally apply because perturbations of the normal are not linearly separable from shading calculations [Fou92]. Becker and Max show some methods for forming the transition between bump maps and BRDFs, which implicitly includes filtering bump maps, but they pass directly from bump map to a single BRDF in one step which is not adequate for complex bump maps. Nor do they discuss their BRDF representation in detail [BM93].

Consider shading a pixel that subtends a region of a surface which includes two polygons with normals, N_0 and N_1. The colour resulting from shading the average normal will be different than the average of the shading computed with the two normals separately. Consider the case where the shader is a Phong shader. The average of the two normals N_0 and N_1 could point in such a direction as to cause a specular highlight to appear, when neither N_0 nor N_1 cause such behaviour. Bump maps cannot then be pre-filtered using the same techniques as colour maps. Instead the pre-computed information must factor in the effect of the shader characteristics on the final colour.

One benefit of filtering measured BRDFs explicitly is that bump maps are easily filtered. The observation to make is that applying a bump map to a surface shaded with a BRDF $\mathcal{F}_r$ is equivalent to rotating $\mathcal{F}_r$ about the normal, producing a new BRDF $\mathcal{F}_r'$. If the normal is perturbed by some rotation κ_n, λ_n then the perturbed BRDF ${\mathcal{F}_r}'$ is given by

$$ {\mathcal{F}_r}'(\kappa_i, \lambda_i, \kappa_r, \lambda_r) = \mathcal{F}_r(\kappa_i + \kappa_n, \lambda_i + \lambda_n, \kappa_r + \kappa_n, \lambda_r + \lambda_n) \tag{7.11} $$

which is simply a rotation of the BRDF about the normal. We set the value of $\mathcal{F}_r$ to zero outside of the visible hemisphere. To filter at run-time we need only filter the perturbed BRDFs over the projected area of the screen pixel, which only requires four more additions than filtering a non-bump mapped texture.

Filtering is still computationally expensive, however. As an alternative it is possible to pre-filter the texture using the pyramidal scheme described above. In this case it is necessary to build the texture up from the low level BRDFs. Williams' averaging scheme to generate the next level is adequate:

$$ {\mathcal{F}_r}^{l+1}_{(u,v)}(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{1}{4} \sum_{i,j=0,1} {\mathcal{F}_r}^{l}_{(2u+i,2v+j)}(\theta_i, \phi_i, \theta_r, \phi_r) \tag{7.12} $$

More complex filters can, of course, be used. Figure 7.5 contrasts a bump texture filtered in different ways. The first image shows the bump map with no filtering. The second shows it filtered using our pyramidal bump map filtering scheme. The third shows the same bump map, but generated directly from the micro-geometry. The interesting point to note is that generating the bump map pyramid was fast compared to the micro-geometry pyramid because the bump map geometry is simple while the micro-geometry was comprised of 1600 polygons.

7.5 Summary

This chapter has described a set of methods for filtering BRDF-based texture maps. The texture maps can be flat surfaces with varying BRDFs, bump maps, or general micro-geometry. The method is based on pyramidal representations of BRDFs,

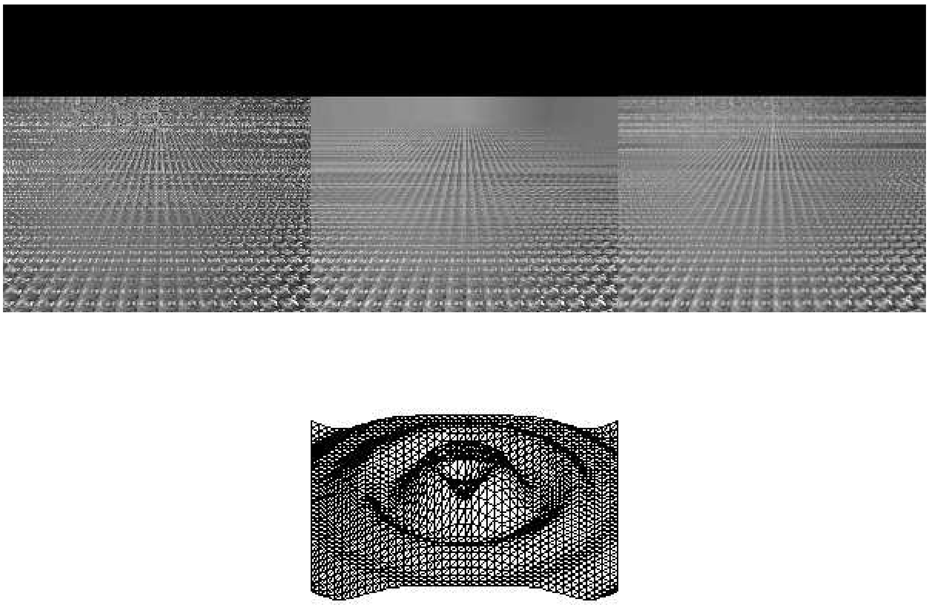


Figure 7.5: A plane with three versions of the same texture. From left to right: the unfiltered bump map, the bump map filtered with our pyramidal scheme, and the bump map rendered by super-sampling each pixel 4 times. The micro-geometry is shown below for reference.

where each BRDF in the pyramid can be generated using a gonioreflectometer to sample the BRDFs and geometry of the underlying surface. Our compact BRDF representation makes this approach feasible, although storage requirements are still high.

Chapter 8

BRDFs and Probability Distributions

Image synthesis algorithms based on Monte-Carlo path tracing [Kaj86, CRMT91, Jen96, SWZ96] have been used to generate impressively realistic images of artificial environments. As these environments become more complex and as the techniques are applied more and more to modeling real environments, the use of reflectance functions measured from real materials becomes more important. To represent adequately these functions a large number of incident and reflected directions must be measured [War92a, War92b].

8.1 Monte-Carlo Path Tracing

Monte Carlo solutions are a set of techniques applied to solving integrals by point sampling the integrand. The material in this section is can be found in many texts [HH64], and this particular derivation is taken largely from Shirley *et al.*'s work on Monte Carlo methods for direct lighting [SWZ96].

Consider a function $h : \mathcal{G} \to \mathcal{R}$ where $\mathcal{G}$ can be a multidimensional space, and we want to estimate the expected value of $h(\psi)$, where ψ is a random variable with probability density function $p : \mathcal{G} \to \mathcal{R}^+$. If μ is a measure defined on $\mathcal{G}$ then $\int_{\mathcal{G}} p(\psi) d\mu(\psi) = 1$. The expected value of $h(\psi)$, $E[h(\psi)]$ can then be approximated by a sum:

$$E[h(\phi)] = \int_{\mathcal{G}} h(\psi')p(\psi')d\mu(\psi') = \frac{1}{N}\sum_{i=1}^{N} h(\psi_i) \tag{8.1}$$

where the samples ψ_i are a set of N instance of the the random variable ψ. Substituting $f = hp$ for the integrand we find:

$$\int_{\mathcal{G}} f(\psi')d\mu(\psi') = \frac{1}{N}\sum_{i=1}^{N} \frac{f(\psi_i)}{p(\psi_i)}. \tag{8.2}$$

The variance of this expression is

$$V\left[\frac{1}{N}\sum_{i=1}^{N} \frac{f(\psi_i)}{p(\psi_i)}\right] = \frac{1}{N}V\left[\frac{f}{p}\right]. \tag{8.3}$$

To decrease the variance we want to use a large N and also want f/p to have low variance. Choices of p so that p is large when f is large lead to having more samples in more important regions, which helps reduce variance.

If we choose to approximate the fundamental equation of physical shading, in Nusselt parameters (Equation 4.2), using only one sample we arrive at:

$$L_r(x, y, \kappa_r, \lambda_r) \approx \frac{L_i(x, y, \kappa, \lambda)f_r(\kappa, \lambda, \kappa_r, \lambda_r)}{q(\kappa, \lambda)} \tag{8.4}$$

where the pair (κ, λ) has distribution q. Selecting the proper distribution q can greatly reduce the variance of the computation. Shirley *et al.* show how to choose q to sample the $L_i()$ term efficiently when the light is incident from a luminaire. The BRDF is assumed to vary only slightly over the angular range of the luminaire, and is not considered in the generated probability distributions. If instead we are interested in the incident light from all directions, not just from luminaires, then the BRDF must be considered. Since little can be known about the illumination in the scene apart from the light sources (without considerable extra computation), the distribution of the BRDF may be the most important term we should consider when selecting the distribution q. In practice we want a distribution that includes the BRDF term when sampling the non-source illuminators of the surface (since we don't know their brightnesses), and the light source geometry, as per Shirley *et al.* when the light sources are known. Veach and Guibas give a simple technique for combining two such distributions, based on randomly selecting which distribution to use [VG95]. We will examine how to build and generate deviates from a distribution based on the BRDF, and then show briefly a technique to sample light sources in our framework.

8.2 The Distribution Function

Monte-Carlo path tracing techniques can benefit dramatically from accurate generation of reflected directions. Generating reflected directions according to the distribution of the BRDF can dramatically reduce the variance of the process. Lafortune and Willems show how using a probability distribution proportional to the BRDF and a cosine factor can reduce variance, but do not give a method for efficiently generating these directions [LW95]. Shirley *et al.* [SWZ96] examine the problem of sampling light sources, but do not include the BRDF's mediating effect on the computation.

Thus, we want to generate reflected rays according to the distribution of the BRDF. The first observation to make is that the BRDF is everywhere positive. In addition, the BRDF exactly determines the magnitude of the light reflected, which means that if the incident light is held constant, the BRDF alone determines the magnitude of the reflection. This means that the BRDF, suitably normalized, can be used as the probability distribution q. The important part to note is that the distribution can be different for each κ_r, λ_r. The normalization factor then is

$$\rho(\kappa_r, \lambda_r) = \int_0^1 \int_0^1 f_r(\kappa_r, \lambda_r, \kappa, \lambda) d\kappa d\lambda. \tag{8.5}$$

The distribution of q for a given reflected direction κ_r, λ_r is then given by

$$q(\kappa, \lambda) = \frac{f_r(\kappa_r, \lambda_r, \kappa, \lambda)}{\rho(\kappa_r, \lambda_r)}. \tag{8.6}$$

The difficulty is in generating values of κ and λ distributed according to q. Usual methods involve storing the inverse of the cumulative probability density functions, but this is prohibitive in our case, since q is parameterized by the reflected direction. There is however some advantage to be gained from using our wavelet representation. A Haar transformed version of the data lends itself to efficient computation of reflected data. Only a coarse representation of q is required to effectively reduce the variance of the integration. The wavelet representation can be used to integrate the function in about the same amount of time as an evaluation of the BRDF. Since our representation is adaptive in trading accuracy with time we should arrive at good results. It remains to show how to extract from our wavelet representation κ's and λs's distributed according to q.

We will begin by examining the univariate case, and from there extend the result to higher dimensions and show how it works with our BRDF data in a Monte-Carlo path tracing framework.

8.3 Generating Random Deviates

The problem is to generate random deviates according to a given probability distribution. Given a uniform probability distribution function so that the probability of generating a number between x and $x + dx$, denoted $p(x)dx$ is uniform on the range $0 < x < 1$, we can take a given function of it, say $y(x)$ and find that its probability distribution $y(x)dy$ is given by

$$p(y) = p(x)|\frac{dx}{dy}|$$

To generate some distribution of y then, say one with $p(y) = f(x)$ we need to solve

$$\frac{dx}{dy} = f(y).$$

If the indefinite integral of $f(x)$ is $F(x)$ then

$$y(x) = F^{-1}(x).$$

The geometric interpretation is that one should choose a uniform random x and find the value y at which the area under the curve $f(x)$ to its left has fraction x of the total area [PTVF92].

8.3.1 Univariate

In most cases we do not have an analytic representation of $F^{-1}(x)$. In our particular case though, we can easily compute some values of $\int_a^b f(x)dx$ which we can use to our advantage to let us evaluate $F^{-1}(x)$ by searching.

The key observation is that the smooth terms of a Haar wavelet-compressed signal gives the average value of the signal in the region covered by the smooth terms. By performing a binary search down the hierarchy of wavelet coefficients we can then find the point at which the area to the left of that point is equal to our uniform random deviate x. The detail that must be attended to is a scaling term for the total area under the curve. Fortunately the top-level smooth term gives us the area under the entire signal. The cost of this algorithm is then $O(\log_2 n)$ where n is the number of samples in the original signal. The cost can be lowered at the cost of accuracy by thresholding the wavelet coefficients.

Consider the Haar transform of a discrete signal $f(x)$ defined over the domain $0..1$, whose wavelet coefficients are $w_{l,m}$, $l = 0..d$, $m = 0..2^l - 1$, and w_s is the smooth term. Then

$$f(x) = w_s\phi(x) + \sum_{l=0}^{d}\sum_{m=0}^{2^l-1} w_{l,m}\psi_{l,m}(x) \tag{8.7}$$

Integrating we find:

$$\int_a^b f(x)dx = w_s \int_a^b \phi(x)dx + \sum_{l=0}^{d}\sum_{m=0}^{2^l-1} w_{l,m} \int_a^b \psi_{l,m}(x)dx \tag{8.8}$$

It must also be recalled that for any wavelet $\int_{-\infty}^{\infty}\psi(x)dx = 0$. In the particular case of the Haar basis a wavelet $\psi_{l,m}(x)$ is supported entirely in the range $(m/2^l, (m+1)/2^l)$, and zero everywhere else. Consider what happens when we reconstruct our function $\int_a^b f(x)dx$ only considering the wavelet coefficients $w_{l,m}$, $l = 0..k$, effectively truncating the evaluation at a certain level. Let the function $f^k(n)$ represent the reconstruction of $f(x)$ obtained by truncating the evaluation at level k:

$$f^k(x) = \phi + \sum_{l=0}^{k}\sum_{m=0}^{2^l-1} w_{l,m}\psi_{l,m}(x) \tag{8.9}$$

The error induced is then

$$\begin{aligned} e^k(x) &= f(x) - f^k(x) && (8.10)\\ &= \sum_{l=k+1}^{d}\sum_{m=0}^{2^l-1} w_{l,m}\psi_{l,m}(x) && (8.11) \end{aligned}$$

If we now consider the error induced by integrating $f^k(x)$ rather than $f(x)$ we find:

$$\int_a^b f(x)dx - \int_a^b f^k(x)dx = \int_a^b e^k(x)dx \tag{8.12}$$

```
function Search( $y, l, r, total, \phi, d, m$ )
if $total = y$ return $r$
   if $total + \frac{r-l}{2}(\phi + w_{d,m}) < y$
      return Search( $y, (l+r)/2, r,$
         $total + \frac{r-l}{2}(\phi + w_{d,m}), \phi - w_{d,m}, d+1, 2m+1$)
   else
      return Search( $y, l, (l+r)/2, total,$
         $\phi + w_{d,m}, d+1, 2m$ )
```

Figure 8.1: Generating a random deviate from a Haar-transformed signal.

and

$$\int_a^b e^k(x)dx = \sum_{l=k+1}^{d} \sum_{m=0}^{2^l-1} w_{l,m} \int_a^b \psi_{l,m}(x)dx \tag{8.13}$$

However, $\int_a^b \psi_{l,m}(x)dx = 0$ unless the interval (a, b) partially intersects the interval $(\frac{m}{2^l}, \frac{m+1}{2^l})$, since that is the entire width of support of the wavelet $\psi_{l,m}(x)$. It is then the case that

$$\int_{\frac{m}{2^k}}^{\frac{m+1}{2^k}} f(x)dx = \int_{\frac{m}{2^k}}^{\frac{m+1}{2^k}} f^k(x)dx \tag{8.14}$$

In our particular application we are interested in performing a search to find some value b such that given y:

$$y = \int_0^b f(x)dx \tag{8.15}$$

We can do this using a binary search technique. Observing that the binary search will follow the dyadic divisions of the wavelet transform, we find that we can compute $\int_l^{(l+r)/2} f(x)dx$ incrementally at each iteration, by keeping track of the smooth term during the traversal (Figure 8.1). Other wavelet bases are not as accommodating as Haar, and require that the evaluation be performed to the bottom level of the wavelet coefficient tree. The cost then becomes $O(\log^2 n)$ where n is the number of terms, since the search requires an $O(\log n)$ reconstruction at each of the $O(\log n)$ levels.

8.4 Multivariate

Extending the result to higher dimensions is straightforward. The largest complication is that if a similar search is to be used we want to choose a point that has a certain fraction of the volume under the curve to its left, in some sense. Consider the two dimensional case. There are a large number of rectangular regions with one corner at the origin where the volume under the curve is the same. Our task is to choose one of these. If we are consistent in how we choose our area then we will achieve an unbiased result. One way to be consistent in our enumeration of the space is to use a quad-tree like search, where the area in each quadrant is examined in a consistent order. In 4 dimensions we use a hexa-decary search instead of a quadtree search, and find that the search space is again divided along the dyadic subdivisions of the input. It is easy to show that because this subdivision is dyadic that the finer-scale terms need not be examined to find the integral over the subdivision regions. The key observation is that if any of the $\Gamma(.)$ functions that comprise the basis functions $B_{l,m}(.)$ at a depth l of the search is the wavelet then the value of the integral of $B_{l,m}(.)$ over the range of the search is zero. Likewise the integral of $B_{l',m'}(.)$ for $l' > l$ will also be zero. The only time that the $B()$'s don't include a wavelet component is when the $B(.)$ is a pure smoothing term, but this only happens at the root of the wavelet tree and can be dealt with as a special case.

8.5 Multivariate with Fixed incident Directions

The problem becomes more interesting when two variables are held fixed, which is the case when we are performing Monte-Carlo path tracing, and requesting a representative distribution of reflected rays. The integral to search for a reflected

```
function Integrate(WaveletNode wn, $\kappa_i$, $\lambda_i$, $\kappa_r^{min}$, $\kappa_r^{max}$, $\lambda_r^{min}$, $\lambda_r^{max}$ )
$r = 0$
for $b = 0..15$
    if bit $b$ of wn.ValueMask is set
        $r$ += $\Gamma^0_{wn.l,wn.m_{\kappa_i},b}(\kappa_i) \times \Gamma^1_{wn.l,wn.m_{\lambda_i},b}(\lambda_i)$
            $\times \int_{\kappa_r^{min}}^{\kappa_r^{max}} \Gamma^2_{wn.l,wn.m_{\kappa_r},b}(\kappa_r) d\kappa_r \times \int_{\lambda_r^{min}}^{\lambda_r^{max}} \Gamma^3_{wn.l,wn.m_{\lambda_r},b}(\lambda_r) d\lambda_r$
for $b = 0..15$
    if bit $b$ of wn.ChildMask is set
        if $\kappa_i$ is in the support of $\Gamma^0_{wn.l+1,2wn.m[0]+b:0,b}(.)$
            and $\lambda_i$ is in the support of $\Gamma^1_{wn.l+1,2wn.m[1]+b:1,b}(.)$
            $r$ += Integrate(wn.Child[b], $\kappa_i$, $\lambda_i$, $\kappa_r^{min}$, $\kappa_r^{max}$, $\lambda_r^{min}$, $\lambda_r^{max}$)
return $r$
```

Figure 8.2: Integration over exiting directions of the BRDF from a 4-variable Wavelet compressed BRDF

ray is:

$$g(\kappa_i, \lambda_i) = \frac{\iint \mathcal{F}_r(\kappa_i, \lambda_i, \kappa_r, \lambda_r) d\kappa_r d\lambda_r}{\rho(\kappa_i, \lambda_i)} \tag{8.16}$$

Writing this in wavelet terms and leaving within the integral only functions of κ_r and λ_r we find:

$$\begin{aligned} g(\kappa_i, \lambda_i)\rho(\kappa_i, \lambda_i) = \quad & \sum_l \sum_m \sum_{\nu=1}^{15} f_{l,m,\nu} \Gamma^0_{l,m_{\kappa_i},\nu}(\kappa_i) \Gamma^1_{l,m_{\lambda_i},\nu}(\lambda_i) \\ & \times \int \Gamma^2_{l,m_{\kappa_r},\nu}(\kappa_r) d\kappa_r \int \Gamma^3_{l,m_{\lambda_r},\nu}(\lambda_r) d\lambda_r \end{aligned} \tag{8.17}$$

The terms within the integrals are easy to compute. The integrals of the basis functions can be tabulated for fast lookup, or evaluated analytically in the case of simpler bases, such as Haar. Pseudo-code for the integration is given in Figure 8.2. The search itself is a straightforward quad-tree search over the range of the integration variables κ_r and λ_r.

8.6 Sampling Light Sources

In section 6.2 we saw how the shading equation can be transformed from an integral over the incident hemisphere to an integral over the area of the light source. We also saw that because of the coordinate transformations involved, the analytic solution of this integral is is not available in general. Monte Carlo techniques give us some tools for evaluating this integral.

Before examining the solution to this integral we must examine which geometric domains we are interested in. Simple modifications to a ray tracer allow a (u, v) parameterized emissive texture to be mapped onto an object in the same way as other textures are mapped. To sample from the point being shaded we need only to be able to convert a point of intersection in world coordinates to a (u, v) pair. To evaluate the integral from the point of view of the lamp requires that a (u, v) pair be converted into world coordinates. Although this operation is supported for most primitives it is not usually supported for collections of primitives, such as polygonal meshes. For purposes of our discussion we will only examine primitives for which this mapping is well defined.

Shirley *et al.* make the assumption that the emissivity of the object does not vary much over the range of directions represented while illuminating an object. However, it is easy to see that given a spherical emitter with a directionally varying emission, that the emissivity can vary considerably between the horizon and the center of the visible disk, since the surface normal changes by 90 degrees. Shirley's methods are all based on sampling the geometry of the luminaire from the point of view of the surface being shaded. This sampling is straightforward if the geometry is simple. In particular Shirley addresses spherical, planar, and cylindrical luminaires.

There are two ways to proceed if we want to generalize to arbitrary geometries. We can either use a test and reject sampling method that generates points on a bounding volume of the luminaire and then casts a ray from the view point to the luminaire. If the ray misses the luminaire's geometry a new sample is computed. Another approach is to generate samples in the luminaire's parametric space. This approach has the advantage that the random deviates are easy to compute, and may allow the brightness of the luminaire to be taken into account. However, a uniform sampling in parametric space will almost never generate a uniform sampling in world space. Likewise, this distortion will considerably affect the distribution of the emission from the luminaire, likely making any importance sampling generated from the emissivity map less useful. The additional complication is that unlike the BRDF directional sampling we present, any importance measure generated from the luminaire will be closely tied to the geometry of the luminaire.

For arbitrary geometries we choose to generate sample rays toward the luminaire's bounding box because the mapping from (u, v) parameters to world space is not straightforward for luminaires made of groups of polygons or bicubic patches. Shirley's techniques are then largely sufficient for this.

For planar luminaires we choose to generate samples in the luminaire's parametric space. We can use a technique similar to that used to generate random deviates from BRDF data to generate points on the luminaire distributed according to the intensity of the luminaire in the direction toward the point being shaded. The main difficulty is that the direction toward the point being shaded is not constant with respect to the chosen position on the luminaire. Consider equation 6.7. To generate (u, v)'s distributed according to the magnitude of L_e in a manner analogous to that presented for the BRDF we must be able to evaluate the normalization term:

$$\rho(\mathbf{x}) = \int_0^1\!\!\int_0^1 L_e(u, v, \kappa(u, v), \lambda(u, v))dudv \tag{8.18}$$

Given this we can then search the cumulative probability function using the same algorithm, provided we can evaluate

$$g(\mathbf{x}) = \iint L_e(u, v, \kappa(u, v), \lambda(u, v))dudv \tag{8.19}$$

where the limits of integration are the quadratic subdivisions of our search algorithm. Recasting $L_e()$ into wavelet terms this integral becomes[1]:

$$\begin{aligned} g(\mathbf{x}) = \; & \sum_l \sum_m \sum_{\nu=1}^{15} f_{l,m,\nu} \iint \Gamma^2_{l,m_\kappa,\nu}(\kappa(u, v)) \Gamma^3_{l,m_\lambda,\nu}(\lambda(u, v)) \\ & \times \Gamma^0_{l,m_u,\nu}(u) \Gamma^1_{l,m_v,\nu}(v)dudv \end{aligned} \tag{8.20}$$

Note that unlike the case of the BRDF none of the basis functions can be removed from the integral, since the directional terms are functions of u and v. This sufficiently complicates the integral that the simple search described in section 8.5 cannot be used as the cost of evaluation is too high. It would be more useful to simply generate more sample points from a uniform distribution to reduce variance.

8.7 Results

Searching using the algorithm presented above can be slow—a great many terms might be examined. There are two ways to reduce the amount of work required to generate the reflection rays. By thresholding the wavelet representation of the BRDF the number of terms that need to be examined can be reduced. The result is

[1] The selector arguments to Γ change from equation 8.17 because we order a luminaire's parameters as u, v, κ, λ, while the BRDF parameters are ordered $\kappa_i, \lambda_i, \kappa_r, \lambda_r$ — the variables of integration correspond to different basis selectors

that the distribution of reflected rays is the same as some simplified version of the BRDF.

Figure 8.3 shows a ray-traced environment in which the direction of the reflected rays is determined using our method. The back wall is a Haar transformed Phong BRDF with an exponent of 100. The right hand image shows the same environment using naive random direction sampling. The images were both generated using four samples per pixel. The image on the left, with reflected rays generated using our scheme, has considerably less noise than the one on the right. The visual quality achieved in 4 samples per pixel on the reflected plane is superior to any other method we are aware of.

Figure 8.4, left, shows an environment illuminated by two of our fluorescent luminaires (Figure 5.1, center). Figure 8.4, right, shows the same environment illuminated by the sphere lamp (Figure 5.1, left). Again the images were generated at four samples per pixel. In both of these images the extra noise can be attributed to the variance accross the surface of the luminaire. Since generating samples based on the distribution of the emittance on the luminaire is not feasible we distributed samples geometrically over the luminaire and evaluated the emittance at the selected point.

8.8 Summary

In this chapter we have examined the integration of our BRDF and luminaire representations into a Monte Carlo ray tracing environment. The wavelet representation of the BRDF leads to a simple method for generating reflected directions based on the value of the BRDF that dramatically reduces the variance in the computation of reflections in Monte Carlo ray tracing. The method does not however extend to

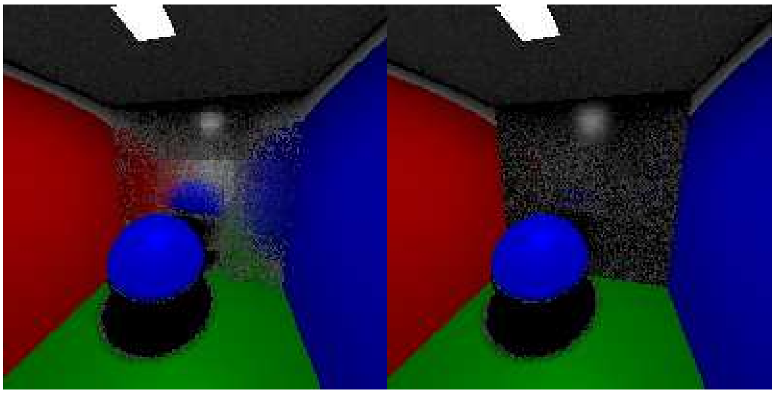


Figure 8.3: Ray-traced environment with reflected directions computed with (left) and without (right) our method. 4 samples per pixel.

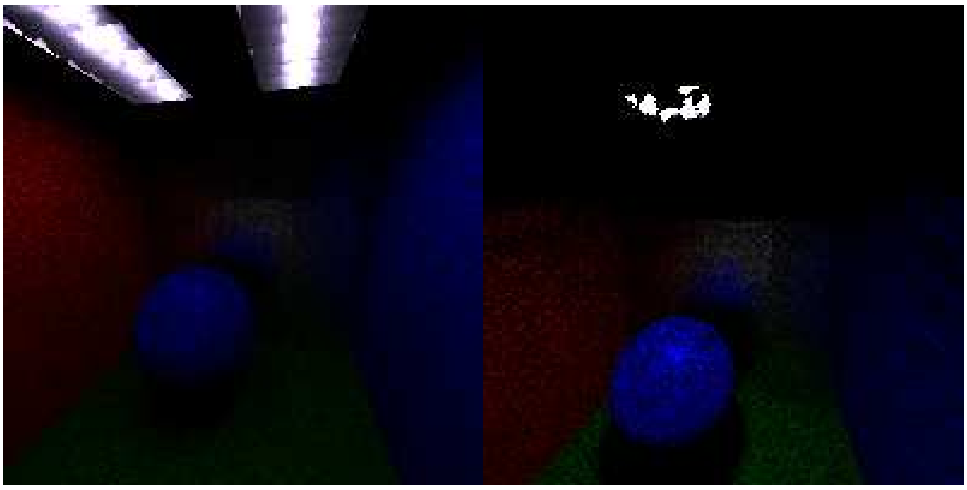


Figure 8.4: An environment illuminated by fluorescent lamps, and another illuminated by the sphere lamp.

sampling luminaires, as the computation is confounded by geometric terms that do not allow the separation of the integrals involved and so their efficient solution.

Chapter 9

Conclusion

This dissertation has presented a number of representations and algorithms relating to problems of local illumination when using light distribution fields. The bidirectional reflection distribution function and luminaires describe the interaction or emission of light with or from an object. The driving problem is that the data sets for BRDFs and for luminaires are very large and require some form of compression to use them effectively. The unifying feature of this work is an efficient representation of four dimensional data sets parameterized by positional and angular components using the non-standard wavelet decomposition. The decomposition leads to efficient algorithms for reconstructing BRDFs and luminaires at single points, for filtering textures, for generating importance-weighted samples of these functions, and for evaluating local shading computations.

We have shown that wavelets can be used to represent BRDFs efficiently and accurately. The representation allows the compression of many BRDFs to less than 5% and frequently less than 1% without appreciable degradation in the resulting shaded images. The point evaluation method runs in $O(\log n)$ time where there are

n^4 original samples.

Using a similar representation we have shown a method of compressing a representation of light flux through a boundary in space. This allows us to efficiently represent luminaires as well as the large number of views of objects required for light field rendering applications. An interactive application shows that these light field images can be reconstructed from their wavelet projected versions at interactive rates. We can use the same representation to describe the light incident on a plane.

Combining these two results and exploiting the sparseness of the wavelet representations of the BRDF and of the incident light, we can accurately evaluate the integral over the BRDF and incident hemisphere to generate accurate filtered lighting calculations for given reflected directions. Although not interactive on current hardware, the costs are roughly equivalent to evaluating the shading due to 10 point light sources. This shading algorithm is particularly well suited for use with Lucifer, a rendering system that propagates light through an environment, always representing light fluxes by their wavelet projections.

A related result uses the properties of the multi-dimensional wavelet transform to generate reflected directions for Monte Carlo path tracing applications. Given an incident light direction, it is possible to generate random reflected directions by treating the BRDF as a probability distribution, and generating the reflected ray according to the partial probability given by fixing the two incident variables. These representations can considerably reduce variance of Monte Carlo path tracing solutions to global illumination.

9.1 Future Directions

Throughout this work we have assumed that surface reflectance is only a function of incident and reflected directions. As was alluded to briefly in chapter 2 this is an oversimplification. In general the reflectance is also a function of incident and exiting positions on the surface. It would be interesting to add this complexity to our representation, but the costs due to the higher dimensionality of the resulting function overwhelm the computation. A more efficient representation is required.

A related area of interest is in curved surfaces. If sub-surface scattering is allowed then a curved surface will considerably complicate the shading computation. It may be important to accurately estimate the effect of the curvature on the exiting light directions, as this is essential to compute directly the shading of curved surface. One direction to take the investigation is towards light transfer. The process of reflection in general can be treated as a light transport operator. Instead of a purely geometric transfer function, as found in Lucifer for example, the transport could account for the interaction of the surface.

A further area of investigation is in the representation of sparse data used. In particular, our current data structures store all wavelet coefficients in a top-down fashion, forcing us to examine possibly unimportant coefficients before finding large, important coefficients nearer the bottom of the tree. Exploring data structures that would allow the coefficients to be accessed in order of importance will be critical in reducing the amount of work and in letting the algorithm be truly adaptive in the amount of work to perform in search of an approximate solution.

Appendix A

Introductory Radiometry

This appendix provides a brief introduction to radiometric quantities used in this dissertation. For more specific information the reader is encouraged to consult [CW93] which includes a good introduction to radiometry for computer graphics.

A.1 Light

Light is the part of the spectrum of electro-magnetic radiation that can affect the human sense of sight.

Electro-magnetic radiation is composed of a magnetic field wave and an electric field wave propagating at a right angle to each other. The main characteristic of a radiation is its wavelength (λ), the distance between two consecutive maxima (see Figure A.1). Another characteristic, directly related to the preceding, is its frequency (ν). It is related to the wavelength and the speed of the wave (c) by the relation: $c = \nu\lambda$. The units[1] of wavelengths are units of length, such as meters (m), nanometers ($nm = 10^{-9}m$) or angstroms ($A = 10^{-10}m$).

[1] We will use in our example the SI (Systeme International) units.

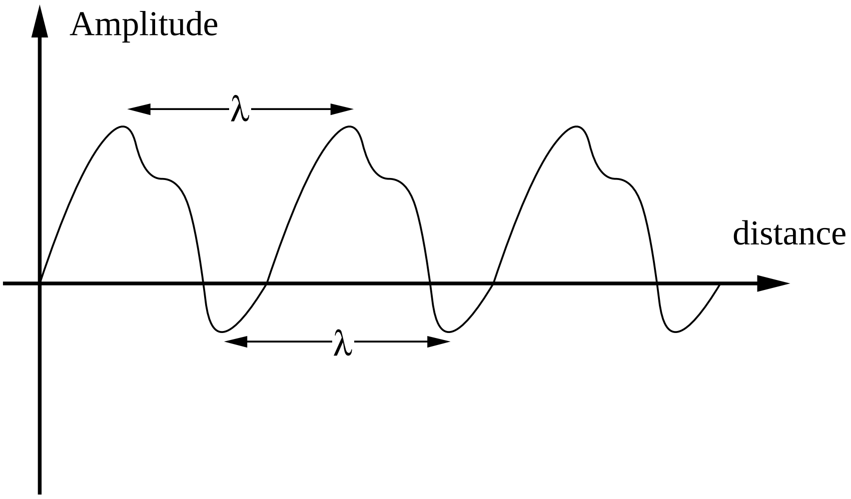


Figure A.1: Wavelength of a periodic phenomenon

The units of frequency are inverses of time units (such as s^{-1}), and the units of speed are units of lengths per unit of time (such as $m.s^{-1}$). Electromagnetic radiation ranges in wavelength from $10^{-12}m$ (gamma rays) to $10^{13}m$ (very low frequency radio waves). The visible part (again visible refers to the capabilities of a particular species, in the present case ours, and we assume every reader of this text is of the same species) lies between $4 \times 10^{-7}m$ and $7 \times 10^{-7}m$, which is from $400nm$ to $700nm$. Figure A.2 shows the visible spectrum and its immediate neighborhood, and the corresponding colors.

There are several other properties of light that are relevant to vision and computer graphics[2]. Light is a form of energy, so one way to characterize a light source is by the amount of energy it emits. The SI unit of energy is the Joule (J). The power, given in Watt (W), is the energy emitted per unit of time. A Watt is

[2] There are two parallel nomenclatures for the properties of light here, a physical one, which is concerned about the objective physical properties of the radiation, and a psychophysical one, which is concerned with the action of the light on our visual system. In this dissertation we are only concerned with the radiometric quantities, that is to say, the objective physical properties.

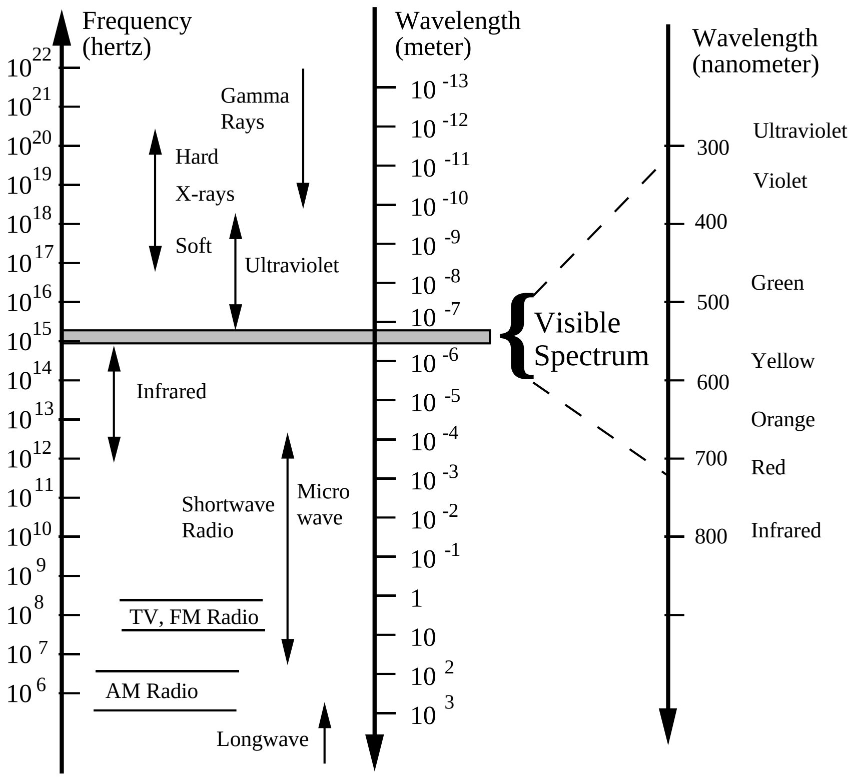


Figure A.2: The electro-magnetic and visible spectrum

thus a Joule per second. It is called a radiant flux in radiometry. A light source is normally more than a single point. The radiant exitance is the power emitted per unit of surface area of the source. The SI unit is the $W.m^{-2}$. Similarly one defines the amount of light received by an illuminated object in terms of the power received by unit area. It is called the irradiance, and has the same units as the exitance. To illustrate, let us consider a case where the light source can be assimilated to a point and the medium in which the light propagates does not absorb its energy. The wavefront of the light, which is the locus of points reached at some instant by the radiation emitted by the source at some earlier instant, is then a sphere. Since the total energy is conserved, the irradiance at some point decreases proportionally to the square of the distance from the source because the same total energy is spread over the area of the sphere which increases proportionally to the square of its radius (see Figure A.3). The power emitted is distributed all over the space around the light source. To characterize this distribution through space, one uses the radiant intensity, which is a power per unit of solid angle ($W.sr^{-1}$). One simplified way to define a solid angle is to imagine a cone (the section can be any shape) with its apex at the light source, and such that the intersection of that cone with a sphere of radius r has an area A. Then the solid angle defined by the cone is $\omega = A/r^2$ (see Figure A.3). The total solid angle around a point is then 4π. The energy density of a radiation is the energy per unit volume of space. (for instance in $J.m^{-3}$). It is then the intensity divided by the speed of propagation of the radiation (check for yourself that it agrees with the units). Table A.1 gathers these useful quantities and their units.

The speed of propagation of electro-magnetic radiation in a vacuum is one of the fundamental constant of nature, and is approximately equal to $3 \times 10^8 m.s^{-1}$.

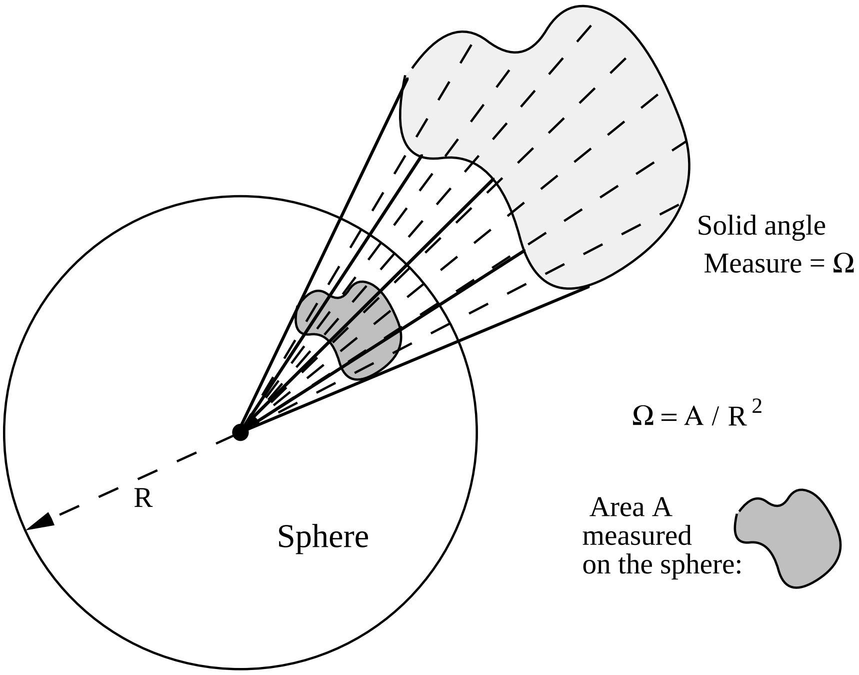


Figure A.3: Solid angle

Physical	Radiometric	Symbol	SI Units
Energy	Radiant Energy	U	J(joule)
Power	Radiant Flux	P	W(watt=$J.s^{-1}$)
Flux density (source)	Radiant Exitance Radiosity	B	$W.m^{-2}$
Flux density (receiver)	Irradiance	E	$W.m^{-2}$
Angular flux	Radiant Intensity	I	$W.sr^{-1}$
Angular flux density	Radiance	L	$W.m^{-2}.sr^{-1}$
	Spectral Reflectance Spectral Transmittance	ρ τ	dimensionless
Solid angle Frequency Wavelength		ω ν λ	sr(steradian) s^{-1} or Hz (Hertz) $meter$

Table A.1: Radiometric Quantities (in SI units)

This is the maximum speed realizable by any phenomenon capable of carrying matter or information in a given frame of reference. In media other than a vacuum, the speed of light is less, and a given material's index of refraction is the ratio between the speed of light in a vacuum and the speed of light in this material. This value depends on the wavelength of the light. Note that the speed of light in air is almost equal to the speed of light in a vacuum, and therefore the index of refraction of the air can be taken to be one for most purposes.

So far we have described light (and electro-magnetic radiation in general) in term of waves. Historically, light has first been thought of as a stream of corpuscles, traveling out in straight lines from light sources. Phenomena like reflection can of course be easily explained in this model. Later, about in the middle of the 17th century in Europe, physicists began to show that a wave model can easily explain reflection, as well as refraction and more complex phenomena. This model slowly became the standard one. But phenomena investigated later, such as the photoelectric effect, resisted explanation within the wave model. Modern physics now has adopted the position that light, as well as matter in general, can be described both in term of waves and in term of particles. In the case of light, the corresponding particle is the photon. The photon can be understood as the basic packet of radiation. For our purposes, all of this means that there is no harm in thinking of light either as wave or as a stream of corpuscles, and select the model most appropriate for the solution of the problem at hand. In particular, we will use the concept of rays of light. If you want to think about photons, rays are just the paths followed by the photons. If you prefer waves, a ray is an line drawn in the direction in which a wave is traveling.

Appendix B

Wavelets

This appendix provides an extremely brief introduction to wavelets, shamelessly lifted, with the author's permission, from [Lew95]. Much of the material is taken from Reissell [Rei94], although a similar and more easily obtainable treatment is contained in Daubechies [Dau92].

B.1 Fundamental Wavelet Properties

Wavelets are built from *scaling functions*, which we define by dilations and translations of a base scaling function $\phi(x)$ of the form:

$$\phi_{lm}(x) = 2^{-l/2}\phi(2^{-l}x - m)$$

each level l corresponds to a function space V_l, which is part of a nested sequence of subspaces $\ldots \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \ldots$ with these properties:

- the union of all V_l spans L^2
- $f(x) \in V_l \to f(x+k) \in V_l$

- $f(x) \in V_l \leftrightarrow f(2^l x) \in V_0$
- any $f(x) \in V_l$ has a unique representation as a linear combination of $\phi_{lm}(x)$'s

We define a wavelet function space W_l as composed of those functions that need to be added to a given space V_l to span the next finer space V_{l+1}: $V_{l+1} = V_l \oplus W_l$. The basis functions for W_l are also dilations and translations of a mother ("parent"?) wavelet $\psi(x)$:

$$\psi_{lm}(x) = 2^{-l/2}\psi(2^{-l}x - m)$$

Since $\phi(x) \in V_0$ and $V_0 \subset V_1$, we can write $\phi(x)$ as a linear combination of the basis functions $\phi(2x - m)$ for V_1:

$$\phi(x) = \sqrt{2}\sum_m h_m \phi(2x - m)$$

This also holds for ψ:

$$\psi(x) = \sqrt{2}\sum_m g_m \phi(2x - m)$$

These are the *dilation* or *refinement equations.* They are the essence of multiresolution analysis. Wavelet bases differ principally in their choices of $\{h_m\}$ (which turns out to determine $\{g_m\}$).

Let $P_l f$ be the projection of a function $f \in L^2$ into the subspace V_l:

$$P_l f(x) = \sum_m < f, \phi_{lm} > \phi_{lm}(x)$$

It can be shown

$$\| f - P_l f \| \leq C 2^{-lN_v} \sqrt{\sum_n \| f^{(n)} \|^2}$$

where N_v is the number of vanishing moments of the wavelets, i. e. for $n = 0, \ldots, N_v - 1$

$$\int x^n \psi(x) dx = 0$$

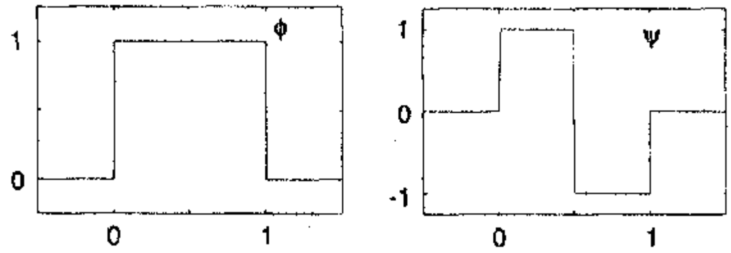


Figure B.1: The Haar smoothing function and wavelet.

B.2 Basis Functions

The simplest basis functions are the Haar bases. The smoothing functions are a constant over their range, and zero outside. The wavelets are simple step functions. Both are illustrated in figure B.1. They have the property that they are orthogonal. As far as this dissertation is concerned this means that the Haar smoothing and wavelet functions can be used both as analysis and reconstruction wavelets. This is not true in general. In a bi-orthogonal scheme the basis functions and their duals are different. Take for example the simplest linear spline. The analysis wavelet is a very complex function with broad support, whereas the reconstruction wavelet is simple with very limited support. The reconstruction wavelet is shown in figure B.2. This feature is important throughout this dissertation, where the width of the reconstruction wavelet is one of the major factors affecting the efficiency of the algorithms.

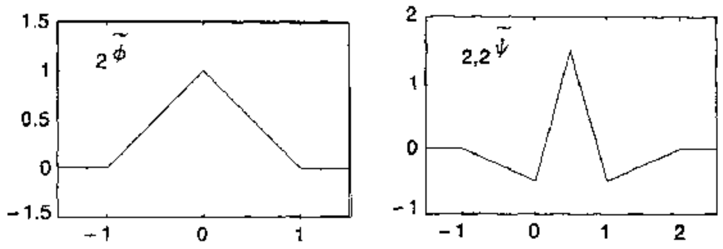


Figure B.2: The duals of the linear spline smoothing function and wavelet, used for reconstruction.

Bibliography

[Ama84] John Amanatides. Ray tracing with cones. In Hank Christiansen, editor, *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 129–135, July 1984.

[Arv94] James Arvo. The irradiance Jacobian for partially occluded polyhedral sources. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 343–350. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[Arv95] James Arvo. Applications of irradiance tensors to the simulation of non-lambertian phenomena. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 335–342. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.

[Ash92] Ian Ashdown. Near-field photometry: The helios approach. In *Graphics Interface '92 Workshop on Local Illumination*, pages 1–14, May 1992.

[Bli77] James F. Blinn. Models of light reflection for computer synthesized pictures. volume 11, pages 192–198, July 1977.

[BM93] Barry G. Becker and Nelson L. Max. Smooth transitions between bump rendering algorithms. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 183–190, August 1993.

[BN76] J. F. Blinn and M. E. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19:542–546, 1976.

[BS87] P. Beckmann and A. Spizzichino. *The scattering of electromagnetic waves from rough surf aces.* Artech House Inc, 1987.

[BW75] M. Born and E. Wolf. *Principles of Optics.* Pergamon, 1975.

[CG85] Michael F. Cohen and Donald P. Greenberg. The Hemi-Cube: A radiosity solution for complex environments. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19, pages 31–40, August 1985.

[CGIB86] Michael Cohen, Donald P. Greenberg, Dave S. Immel, and Philip J. Brock. An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications*, 6(3):26–35, March 1986.

[CMS87] Brian Cabral, Nelson Max, and Rebecca Springmeyer. Bidirectional reflection functions from surface bump maps. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 273–281, July 1987.

[CRMT91] Shenchang Eric Chen, Holly E. Rushmeier, Gavin Miller, and Douglass Turner. A progressive multi-pass method for global illumination. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 165–174, July 1991.

[CSSD96] Per H. Christensen, Eric J. Stollnitz, David H. Salesin, and Tony D. DeRose. Global illumination of glossy environments using wavelets and importance. *ACM Transactions on Graphics*, 15(1):37–71, January 1996. ISSN 0730-0301.

[CT81] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. volume 15, pages 307–316, August 1981.

[CT82] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics*, 1(1):7–24, January 1982.

[CW93] Michael F. Cohen and John R. Wallace. *Radiosity and Realistic Image Synthesis.* Academic Press Professional, San Diego, CA, 1993. Excellent book on radiosity algorithms.

[Dau92] Ingrid Daubechies. *Ten Lectures on Wavelets*, volume 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics.* SIAM, Philadelphia, PA, 1992.

[dC76] M. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice Hall, 1976.

[DF97] Joel DeYoung and Alain Fournier. Properties of tabulated bidirectional reflectance distribution functions. In *To appear, Proceedings of Graphics Interface '97*, May 1997.

[DLF96] Joel DeYoung, Paul Lalonde, and Alain Fournier. Acquiring and using realistic reflectance data in computer graphics images. In *Arkansas Computer Conference*, pages 77–82, 1996. Searcy, Arkansas: March 7–8, 1996.

[Fou92] Alain Fournier. Normal distribution functions and multiple surfaces. In *Graphics Interface '92 Workshop on Local Illumination*, pages 45–52, May 1992.

[GGSC96] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 43–54. ACM SIGGRAPH, Addison Wesley, August 1996. held in New Orleans, Louisiana, 04-09 August 1996.

[GH86] Ned Greene and Paul S. Heckbert. Creating raster omnimax images from multiple perspective views using the elliptical weighted average filter. *IEEE Computer Graphics and Applications*, 6(6):21–27, June 1986.

[Gla95] Andrew S. Glassner. *Priciples of Digital Image Synthesis.* Morgan Kaufmann, 1995.

[GMN94] Jay S. Gondek, Gary W. Meyer, and Jonathan G. Newman. Wavelength dependent reflectance functions. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 213–220. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[GSCH93] Steven J. Gortler, Peter Schroder, Michael F. Cohen, and Pat Hanrahan. Wavelet radiosity. In *Computer Graphics Proceedings, Annual Conference Series, 1993*, pages 221–230, 1993.

[GTGB84] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modelling the interaction of light between diffuse surfaces.

In *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 212–22, July 1984.

[GTS+97] Donald P. Greenbert, Kenneth E. Torrance, Peter Shirley, James Arvo, James A. Ferwerda, Sumanta Pattanaik, Eric Lafortune, Bruce Walter, and Sing Foo. A framework for realistic image synthesis. In *Computer Graphics (SIGGRAPH '97 Proceedings)*, volume 31, August 1997.

[Hec86] Paul S. Heckbert. Survey of texture mapping. *IEEE Computer Graphics and Applications*, 6(11):56–67, November 1986.

[HH64] J. M. Hammersley and D. C. Handscomb. *Monte Carlo Methods.* Wiley, New York, 1964.

[HK93] Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 165–174, August 1993.

[Hou91] Caroline Houle. Light source modelling. M.sc. thesis, Department of Computer Science, University of Toronto, 1991.

[HSA91] Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 197–206, July 1991.

[HTSG91] Xiao D. He, Kenneth E. Torrance, Francois X. Sillion, and Donald P. Greenberg. A comprehensive physical model for light reflection. In

Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 175–186, July 1991.

[ICG86] David S. Immel, Michael F. Cohen, and Donald P. Greenberg. A radiosity method for non-diffuse environments. In David C. Evans and Russell J. Athay, editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 133–142, August 1986.

[Ins86] American National Standards Institute. ANSI standard nomenclature and definitions for illuminating engineering,. ANSI/IES RP-16-1986, Illuminating Engineering Society, 345 East 47th Street, New York, NY 10017, June 1986.

[Jen95] Henrik Wann Jensen. Importance driven path tracing using the photon map. In *Eurographics Rendering Workshop 1995*. Eurographics, June 1995.

[Jen96] Henrik Wann Jensen. Global illumination using photon maps. In *Seventh Eurographics Workshop on Rendering*, pages 22–30, Porto, Portugal, June 1996.

[Kaj85] James T. Kajiya. Anisotropic reflection models. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19, pages 15–21, July 1985.

[Kaj86] James T. Kajiya. The rendering equation. In David C. Evans and Russell J. Athay, editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 143–150, August 1986.

[Knu73] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, Reading, MA, 1973.

[Lew93] Robert Lewis. Making shaders more physically plausible. In Michael F. Cohen, Claude Puech, and Francois Sillion, editors, *Fourth Eurographics Workshop on Rendering*, pages 47–62. Eurographics, June 1993. held in Paris, France, 14–16 June 1993.

[Lew95] Robert R. Lewis. Light-driven global illumination: Work in progress. In *Proceedings of the Sixth Western Computer Graphics Symposium (Skigraph)*, 1995.

[Lew96] Robert R. Lewis. Wavelet radiative transfer and surface interaction. In *Proceedings of the Seventh Western Computer Graphics Symposium*, pages 73–83, 1996.

[LF96] Robert R. Lewis and Alain Fournier. Light-driven global illumination with a wavelet representation of light transport. In *Seventh Eurographics Workshop on Rendering*, Porto, Portugal, June 1996.

[LH96] Marc Levoy and Pat Hanrahan. Light field rendering. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 31–42. ACM SIGGRAPH, Addison Wesley, August 1996. held in New Orleans, Louisiana, 04-09 August 1996.

[LW95] Eric P. Lafortune and Yves D. Willems. A 5D tree to reduce the variance of monte carlo ray tracing. In *Eurographics Rendering Workshop 1995*. Eurographics, June 1995.

[NKON90] Eihachiro Nakamae, Kazufumi Kaneda, Takashi Okamoto, and Tomoyuki Nishita. A lighting model aiming at drive simulators. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 395–404, August 1990.

[NON85] T. Nishita, I. Okamura, and E. Nakamae. Shading models for point and linear sources. *ACM Transactions on Graphics*, 4(2):124–146, April 1985.

[NRH$^+$77] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. Geometric considerations and nomenclature for reflectance. Monograph 161, National Bureau of Standards (US), October 1977.

[PAI97] Dinesh Pai. Personal communication July 1997.

[PA91] Pierre Poulin and John Amanatides. Shading and shadowing with linear light sources. *Computers and Graphics*, 15(2):259–265, 1991.

[PF90] Pierre Poulin and Alain Fournier. A model for anisotropic reflection. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 273–282, August 1990.

[Pho75] Bui-T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, June 1975.

[Pou91] Pierre Poulin. An extended shading model for linear light sources. In *Proceedings of the 1991 Western Computer Graphics Symposium*, pages 31–35, April 1991.

[PTVF92] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (2nd* *ed.)*. Cambridge University Press, Cambridge, 1992. ISBN 0-521-43108-5.

[Rei94] Leena-Maija Reissell. Multiresolution and wavelets. In *SIGGRAPH '94 Course Notes, Wavelets and Their Applications in Computer Graphics*, 1994.

[SAWG91] Francois X. Sillion, James R. Arvo, Stephen H. Westin, and Donald P. Greenberg. A global illumination solution for general reflectance distributions. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 187–196, July 1991.

[SDS95] Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. Wavelets for computer graphics: A primer. *IEEE Computer Graphics and Applications*, 15(3), 1995.

[SH92] Robert Siegel and John R. Howell. *Thermal Radiation Heat Transfer.* Hemisphere Publishing Corporation, Washington, D.C., 3rd edition, 1992.

[SHW92] Steve Shafer, Glen Healey, and Larry Wolff. *Physics Based Vision Principles and Practice: Radiometry.* A K Peters, 1992.

[SS95] Peter Schröder and Wim Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 161–172. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.

[SWZ96] Peter Shirley, Chang Yaw Wang, and Kurt Zimmerman. Monte carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, January 1996. ISSN 0730-0301.

[TR67] T. S. Trowbridge and K. P. Reitz. Average irregularity representation of a roughened surfaces for ray reflection. *Journal of Optical Society of America*, 65(3), 1967.

[TS66] K. E. Torrance and E. M. Sparrow. Polarization, directional distribution, and off-specular peak phenomena in light reflected from roughened surfaces. *Journal of Optical Society of America*, 56(7), 1966.

[TS67] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. *Journal of Optical Society of America*, 57(9), 1967.

[Ved93] Christophe Vedel. Computing illumination from area light sources by approximate contour integration. In *Proceedings of Graphics Interface '93*, pages 237–244, Toronto, Ontario, Canada, May 1993. Canadian Information Processing Society.

[VG84] C. P. Verbeck and D. P. Greenberg. A comprehensive light source description for computer graphics. *IEEE Computer Graphics and Applications*, 4(7):66–75, July 1984.

[VG95] Eric Veach and Leonidas J. Guibas. Optimally combining sampling techniques for monte carlo rendering. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 419–428.

ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.

[War83] D. R. Warn. Lighting controls for synthetic images. volume 17, pages 13–21, July 1983.

[War92a] Gregory J. Ward. Measuring and modeling anisotropic reflection. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 265–272, July 1992.

[War92b] Gregory J. Ward. Towards more practical reflectance measurements and models. In *Graphics Interface '92 Workshop on Local Illumination*, pages 15–21, May 1992.

[WAT92] Stephen H. Westin, James R. Arvo, and Kenneth E. Torrance. Predicting reflectance functions from complex surfaces. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 255–264, July 1992.

[Whi80] Turner Whitted. An improved illumination model for shaded display. *Communications of the ACM*, 23(6):343–349, June 1980.

[Wil83] Lance Williams. Pyramidal parametrics. In *Computer Graphics (SIGGRAPH '83 Proceedings)*, volume 17, pages 1–11, July 1983.

[WRC88] Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 85–92, August 1988.