Wavelet Radiative Transport and Surface Interaction

Robert R. Lewis, *Student Member, IEEE*, and Alain Fournier

Abstract

Recently, there has been considerable interest in the representation of radiance in terms of wavelet basis functions. We will present a coordinate system called *Nusselt coordinates* which, when combined with wavelets, considerably simplifies computation of radiative transport and surface interaction. It also provides straightforward computation of the physical quantities involved.

We show how to construct a discrete representation of the radiative transport operator $\mathcal{T}$ involving inner products of smoothing functions, discuss the possible numerical integration techniques, and present an application. We also show how surface interaction can be represented as a kind of matrix product of the wavelet projections of an incident radiance and a bidirectional reflectance distribution function (BRDF).

1 Introduction

In computer graphics, we use illumination, the study of how light interacts with matter to produce visible scenes, to produce photo-realistic images. Illumination is typically decomposed into two sub-areas: local and global.

Local illumination describes the interaction of light with a single, small volume or surface element with given incident and viewing directions. Figure 1 shows the typical geometry and nomenclature for local illumination studies.

Global illumination describes how light is distributed in a scene: a collection of objects, including light sources, immersed in a given medium. Global illumination solutions must consider multiple reflections. Global illumination solutions are built on top of local illumination solutions. Fournier [6] describes their interrelation further.

In this paper, we will advance a new approach to an illumination solution that is intermediate between local and global illumination. Using wavelets, we are able to treat the interaction between two surfaces and the interaction of a surface with a radiation field in a source-to-destination model that applies to whole surfaces, not just small elements. We are continuing work to extend this to a fully global solution (see [13]).

Wavelets are relatively recent additions to the rendering toolkit. They were first used by Gortler et al. [10] and Schröder et al. [16] to solve the radiosity equations. Schröder and Hanrahan [15] and Christensen et al. [2] extended this work to radiance itself, applying them to non-diffuse situations. What we present here may be considered a further development of that work.

2 Radiative Transfer and Surface Interaction

Let us first discuss some of the basics of how light is represented. The fundamental quantity is *radiance*, the amount of power passing in a given direction though a given surface per unit area (perpendicular to the direction of travel) per unit solid angle[1]. Radiance at a point **P**

[1]In this report, we will take radiance to be monochromatic and assume we can construct a polychromatic image by combining independently computed monochromatic images. We also ignore polarization.

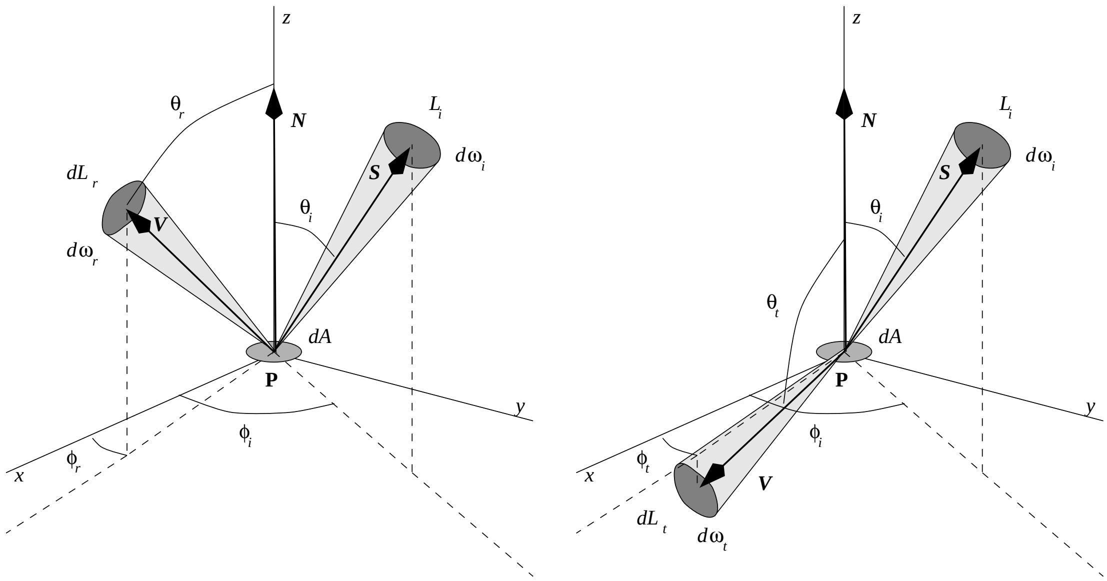

Figure 1: Geometry

in a direction $\mathbf{S}$ is usually represented by a function $L(\mathbf{P}, \mathbf{S})$.

Radiance's most useful property is its invariance: In a non-participating medium, the radiance given off at a point $\mathbf{P_o}$ on a surface in a direction $\mathbf{S}$ is constant until reaching another surface. We can express this as the action of a transport operator $\mathcal{T}$ acting on a source radiance L_s to produce a destination radiance L_d:

$$L_d(\mathbf{P}, \mathbf{S}) = \mathcal{T} L_s = L_s(\mathbf{P}_o(\mathbf{P}, \mathbf{S}), \mathbf{S}) \tag{1}$$

This principle underlies raytracing.

We take the fundamental equation describing surface interaction to be (cf. Foley, et al. [5] or Cohen and Wallace [3]):

$$L(\mathbf{V}) = L_e(\mathbf{V}) + \int_{\Omega^R_{\mathbf{N}}} f_r(\mathbf{S}^{+\prime}, \mathbf{V})\, L_i(\mathbf{S}^{+\prime}) \left|\mathbf{N} \cdot \mathbf{S}^{+\prime}\right| d\omega_i' + \int_{\Omega^T_{\mathbf{N}}} f_t(\mathbf{S}^{-\prime}, \mathbf{V})\, L_i(\mathbf{S}^{-\prime}) \left|\mathbf{N} \cdot \mathbf{S}^{-\prime}\right| d\omega_i' \tag{2}$$

where L is the total radiance given off of a surface with normal $\mathbf{N}$, L_e is the surface emissivity, L_i is the incident radiance, $\Omega_{\mathbf{N}}^R$ is the reflection hemisphere (contains $\mathbf{N}$, the "viewing" direction $\mathbf{V}$, and the "positive source" direction $\mathbf{S}^+$), $\Omega_{\mathbf{N}}^T$ is the transmission hemisphere (opposite $\Omega_{\mathbf{N}}^R$, containing the "negative source" direction $\mathbf{S}^-$), f_r is the bidirectional reflectance distribution function (BRDF), and f_t is the bidirectional transmittance distribution function (BTDF). Note that we use the prime ("$\prime$") to indicate bound variables of integration and, since it symbolically factors out of (2), we ignore the spatial variation of the radiances and distribution functions.

We represent combined reflection and transmission surface interaction by the integral operator $\mathcal{S}$:

$$L_r = L_e + \mathcal{S} L_i \tag{3}$$

2.1 Radiance in Nusselt Coordinates

If we confine our discussion to surfaces, we can assume a parameterization for $\mathbf{P}$ of (x, y). $\mathbf{S}$ is then typically represented in polar and azimuthal coordinates (θ, ϕ) according to the local frame of reference.

Consider the x, y, and z direction cosines corresponding to a direction (θ, ϕ):

$$\mu_x = \sin\theta\,\cos\phi \quad \mu_y = \sin\theta\,\sin\phi \quad \mu_z = \cos\theta \tag{4}$$

We take (μ_x, μ_y) to be an alternative parameterization of direction.

It is convenient in what follows for all variables to vary between the extrema of 0 and 1, so let make a change of the directional variables from (μ_x, μ_y) to (κ, λ):

$$\kappa = \frac{\mu_x + 1}{2} \qquad \lambda = \frac{\mu_y + 1}{2} \tag{5}$$

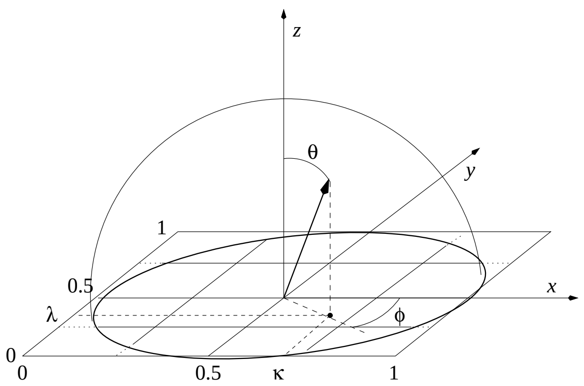

Figure 2: Nusselt Coordinates

Figure 2 shows the relation between (θ, ϕ) and (κ, λ) graphically. It also shows the *directional limiting circle* defined by $\mu_x^2 + \mu_y^2 = 1$.

To convert integration over (θ, ϕ) to integration over (κ, λ), the determinant of the Jacobian is:

$$\left| \frac{\partial(\theta, \phi)}{\partial(\kappa, \lambda)} \right| = \frac{4}{|\cos\theta \sin\theta|} \tag{6}$$

so, assuming L_i is zero for directions outside the directional limiting circle, (2) becomes

$$L = L_e + 4 \int_0^1 \int_0^1 f_r(\mathbf{S}^{+\prime}, \mathbf{V})\, L_i(\mathbf{S}^{+\prime}) + f_t(\mathbf{S}^{-\prime}, \mathbf{V})\, L_i(\mathbf{S}^{-\prime})\, d\kappa_i'\, d\lambda_i' \tag{7}$$

That the integral no longer contains trigonometric functions should come as no surprise. We have simply used a differential form of the "Nusselt analog" ([14], but see Cohen and Wallace [3] for a description in English): the amount of power per unit area transferred from a differential solid angle $d\omega_i$ is proportional to $d\kappa_i\, d\lambda_i$, the area of the surface that the projection of $d\omega_i$ on a unit sphere subtends. For this reason, we refer to κ and λ as "Nusselt coordinates".

We also note that, since $\mu_x^2 + \mu_y^2 + \mu_z^2 = 1$, we can express $\mathbf{S}^+$, $\mathbf{S}^-$, and $\mathbf{V}$ all unambiguously in terms of their respective incident and reflected κ's and λ's, since each vector is defined only over a hemisphere, not the whole directional sphere. Simply put, it is always clear which sign to attach to the square root.

Other ways to parameterize the directional component of a radiance distribution are possible. Light fields (as in Levoy and Hanrahan [12]) and lumigraphs (as in Gortler, et al. [9]), are very promising approaches for display purposes. Christensen, et al. [2] use a combination of a gnomonic projection and "stretch" to map directions to the unit square. None of these approaches, however, leads to the simplification of surface interaction that (7) demonstrates.

2.2 Radiance Representation

What are the characteristics of a four-dimensional radiance distribution $L(x, y, \kappa, \lambda)$? The easiest way to visualize this is as "light through a window" where the position of an observer on a window is (x, y) and he or she is looking in direction (κ, λ). For a fixed direction, the resulting two-dimensional projection is a parallel projection[2]. For a fixed position, the distribution in (κ, λ) would be a "fisheye" view. In both cases, the result is an image, so we can deal with those radiance distributions as we deal with images.

Radiance at a point on a surface is a potentially discontinuous, generally non-analytic function. We can approximate it with a finite element expansion with N_f degrees of freedom:

$$L(x, y, \kappa, \lambda) = \sum_{j=1}^{N_f} b_j B_j(x, y, \kappa, \lambda) \tag{8}$$

Choices for the basis functions B_i include box discretization (*a la* Fournier et al.'s FIAT [7]), Fourier, discrete cosine, spherical harmonics, orthogonal polynomials, and wavelets. We are particularly interested in wavelets because, unlike the other bases listed, their basis functions are of limited support and they can represent discontinuities compactly. They are also capable of considerable compression.

[2]The special case $(\kappa, \lambda) = (\frac{1}{2}, \frac{1}{2})$ is an orthographic projection.

3 Wavelet Radiance Properties

Appendix A summarizes the properties of one-dimensional wavelets. In this section, we will describe multidimensional wavelets and apply them to radiative transport and surface interaction.

Apart from compression, representing radiance in terms of a wavelet basis with direction expressed in Nusselt coordinates makes several calculations of relevance to illumination computation easier. Notice that these all act directly on wavelet coefficients themselves and do not require an inverse wavelet transform.

3.1 Multidimensional Wavelets

For D-dimensional coordinates

$$\mathbf{q} = (q_1, q_2, \ldots, q_D), \tag{9}$$

we can define a set of multidimensional wavelet basis functions indexed by a *standard multiresolution index*

$$\mathbf{j} = (\nu^j, l_1^j, m_1^j, l_2^j, m_2^j, \ldots, l_D^j, m_D^j) \tag{10}$$

where ν^j determines the combination of one dimensional smoothing and wavelet functions:

$$B_{\mathbf{j}}(\mathbf{q}) = \begin{cases} \phi_{l_1^j m_1^j}(q_1)\phi_{l_2^j m_2^j}(q_2)\ldots\phi_{l_D^j m_D^j}(q_D) & \nu^j = 0 \\ \psi_{l_1^j m_1^j}(q_1)\phi_{l_2^j m_2^j}(q_2)\ldots\phi_{l_D^j m_D^j}(q_D) & \nu^j = 1 \\ \phi_{l_1^j m_1^j}(q_1)\psi_{l_2^j m_2^j}(q_2)\ldots\phi_{l_D^j m_D^j}(q_D) & \nu^j = 2 \\ \vdots & \\ \psi_{l_1^j m_1^j}(q_1)\psi_{l_2^j m_2^j}(q_2)\ldots\psi_{l_D^j m_D^j}(q_D) & \nu^j = 2^D - 1 \end{cases} \tag{11}$$

We refer to the special case of $\nu^j = 0$ as the "pure smoothing" component, as the corresponding basis function is made up of only smoothing functions.

We can apply (A10) multi-dimensionally. If we have $\mathbf{j}$ as in (10) and define $\mathbf{k}$ as

$$\mathbf{k} = (\nu^k, l_1^k, m_1^k, l_2^k, m_2^k, \ldots, l_D^k, m_D^k) \tag{12}$$

then

$$\left\langle \tilde{B}_{\mathbf{j}} \mid B_{\mathbf{k}} \right\rangle_{\mathbf{q}} = \delta_{\nu^j \nu^k} \prod_{d=1}^{D} \delta_{l_d^j l_d^k} \delta_{m_d^j m_d^k} \tag{13}$$

(Note the use of the subscripted inner product described in Section A.3.) $\tilde{B}_{\mathbf{j}}$ is identical to $B_{\mathbf{j}}$ as defined in (11), but with the primal wavelets and smoothing functions replaced by their duals.

This is the "standard" multidimensional Cartesian product basis. It is also possible to constrain $l_1^j = l_2^j = \ldots = l_D^j \equiv l^j$, resulting in the so-called "nonstandard" basis. In general multidimensional (especially image-oriented) applications, as cited in Daubechies [4] and in Schröder et al. [16], the nonstandard bases are preferred because of their "square" support.

In this paper, we are considering 4-dimensional, nonstandard basis functions, so let us enumerate the coordinates with the 4-vector $\mathbf{q} = (u, v, \kappa, \lambda)$ and the basis functions with a *nonstandard multiresolution index* $\mathbf{j} = (\nu, l, m_u, m_v, m_\kappa, m_\lambda)$.

Using (A1), (A3), (A5), (A6) and (11), all the basis functions at level l of the pyramid can be written in terms of the $\nu = 0$ basis functions at level $l+1$:

$$B_{\nu l \mathbf{m}}(\mathbf{q}) = 4 \begin{cases} \sum_{\mathbf{m}'} h_{m'_u} h_{m'_v} h_{m'_\kappa} h_{m'_\lambda} B_{0(l+1)(2\mathbf{m}+\mathbf{m}')}(\mathbf{q}) & \nu = 0 \\ \sum_{\mathbf{m}'} g_{m'_u} h_{m'_v} h_{m'_\kappa} h_{m'_\lambda} B_{0(l+1)(2\mathbf{m}+\mathbf{m}')}(\mathbf{q}) & \nu = 1 \\ \sum_{\mathbf{m}'} h_{m'_u} g_{m'_v} h_{m'_\kappa} h_{m'_\lambda} B_{0(l+1)(2\mathbf{m}+\mathbf{m}')}(\mathbf{q}) & \nu = 2 \\ \vdots & \\ \sum_{\mathbf{m}'} g_{m'_u} g_{m'_v} g_{m'_\kappa} g_{m'_\lambda} B_{0(l+1)(2\mathbf{m}+\mathbf{m}')}(\mathbf{q}) & \nu = 15 \end{cases} \tag{14}$$

where $\mathbf{m}' \equiv (m'_u, m'_v, m'_\kappa, m'_\lambda)$.

So for any function f,

$$\left\langle f \mid B_{\nu l \mathbf{m}} \right\rangle_q = \sum_{\mathbf{m}'} W_{\nu \mathbf{m}'} \left\langle f \mid B_{0(l+1)(2\mathbf{m}+\mathbf{m}')} \right\rangle_q \tag{15}$$

where $W_{\nu\mathbf{m}'}$ is (4 times) a product of smoothing and wavelet coefficients.

3.2 Irradiance

Irradiance is defined as

$$E(x, y) = \int_{\Omega^R_{\mathbf{N}}} L_i(x, y, \theta', \phi') \; |\mathbf{N} \cdot \mathbf{S}'| \; d\omega'_i. \tag{16}$$

Again making use of (6), we have

$$E(x, y) = 4 \int_0^1 \int_0^1 L_i(x, y, \kappa', \lambda') \, d\kappa'_i \, d\lambda'_i. \tag{17}$$

The limits of the both integrations are 0 and 1, but if L_i is zero outside the directional limiting circle (see Figure 2), we can safely extend the integration limits to $-\infty$ and $+\infty$.

This allows us to say that if

$$L_i(x, y, \kappa, \lambda) = \sum_{\mathbf{j}} b_{\mathbf{j}} B_{\mathbf{j}}(x, y, \kappa, \lambda) \tag{18}$$

then

$$E(x, y) = 4 \sum_{\mathbf{j}} b_{\mathbf{j}} \left\langle B_{\mathbf{j}} \mid 1 \right\rangle_{\kappa,\lambda} \tag{19}$$

is the wavelet representation of the irradiance.

The inner products on the right hand side are usually easy to compute in tabular form, if not analytically, making particular use of (A4) to eliminate many coefficients.

3.3 Power Flux

The power flux passing through an area A is defined as

$$\Phi = \int_A \int_{\Omega^R_{\mathbf{N}}} L_i(x, y, \theta', \phi') \; |\mathbf{N} \cdot \mathbf{S}'| \; d\omega'_i dA = \int_A E(x, y) \, dA \tag{20}$$

If we have a spatial parameterization $(u, v) \leftrightarrow (x, y)$ that maps the unit square to A and back, then we have

$$\Phi = \int_0^1 \int_0^1 E(x(u, v), y(u, v)) \left| \frac{\partial(x, y)}{\partial(u, v)} \right| dudv. \tag{21}$$

If, as above, we take $E(x, y) = 0$ for (x, y) outside of A and if we extend (18) to include this parameterization:

$$L_i(x, y, \kappa, \lambda) = \sum_{\mathbf{j}} b_{\mathbf{j}} B_{\mathbf{j}}(u(x, y), v(x, y), \kappa, \lambda), \tag{22}$$

then we can incorporate (19) to get the wavelet representation of flux:

$$\Phi = 4 \sum_{\mathbf{j}} b_{\mathbf{j}} \left\langle B_{\mathbf{j}} \mid \left| \frac{\partial(x, y)}{\partial(u, v)} \right| \right\rangle_q . \tag{23}$$

3.4 Transport Between Surfaces

We represent radiance as

$$L(\mathbf{q}) = \sum_{\mathbf{k}} b_{\mathbf{k}} B_{\mathbf{k}}(\mathbf{q}) \tag{24}$$

where

$$b_{\mathbf{k}} = \left\langle L \mid \tilde{B}_{\mathbf{k}} \right\rangle_{\mathbf{q}} \tag{25}$$

and $\mathbf{k}$ is defined as in (12).

Radiance travels from a source point $\mathbf{q}_s$ to a destination point $\mathbf{q}_d$. If we have a mapping of $\mathbf{q}_s \to \mathbf{q}_d$, we can compute

$$L_d(\mathbf{q}_d) = \sum_{\mathbf{k}} b^d_{\mathbf{k}} B_{\mathbf{k}}(\mathbf{q}_d) \tag{26}$$

where

$$b^d_{\mathbf{k}} = \left\langle L_s(\mathbf{q}_s(\cdot)) \mid \tilde{B}_{\mathbf{k}} \right\rangle_{\mathbf{q}_d}$$
$$= \sum_{\mathbf{j}} b^s_{\mathbf{j}} T_{\mathbf{jk}}, \tag{27}$$

$\mathbf{j}$ is defined as in (10), and we define geometry-dependent "transport coefficients"

$$T_{\mathbf{jk}} \equiv \left\langle B_{\mathbf{j}}(q_s(\cdot)) \mid \tilde{B}_{\mathbf{k}} \right\rangle_{\mathbf{q}_d}. \tag{28}$$

Using the multidimensional refinement shown in (14), given $T_{(0 l_j \mathbf{m}_j)\mathbf{k}}$ on level l_j, we can compute all coefficients on the coarser level above it in the pyramid:

$$T_{(\nu_j (l_j - 1)\mathbf{m}_j)\mathbf{k}} = \sum_{\mathbf{m}'} W_{\nu_j \mathbf{m}'} T_{0 l_j (2\mathbf{m}_j + \mathbf{m}')\mathbf{k}} \tag{29}$$

and given $T_{\mathbf{j}(0 l_k \mathbf{m}_k)}$ on level l_k, we can compute

$$T_{\mathbf{j}(\nu_k (l_k - 1)\mathbf{m}_k)} = \sum_{\mathbf{m}''} W_{\nu_k \mathbf{m}''} T_{\mathbf{j}(0 l_k (2\mathbf{m}_k + \mathbf{m}''))} \tag{30}$$

where $\mathbf{m}_j = (m^j_u, m^j_v, m^j_\kappa, m^j_\lambda)$ and $\mathbf{m}_k = (m^k_u, m^k_v, m^k_\kappa, m^k_\lambda)$. This means that we can compute all transport coefficients strictly in terms of pure smoothing components:

$$T_{\mathbf{jk}} = \sum_{\mathbf{m}'} \sum_{\mathbf{m}''} W_{\nu_j \mathbf{m}'} W_{\nu_k \mathbf{m}''} T_{(0 l_j (2\mathbf{m}_j + \mathbf{m}'))(0 l_k (2\mathbf{m}_k + \mathbf{m}''))} \tag{31}$$

3.5 Surface Interaction

Using (11), let us define a mixed primal-dual, four-dimensional, nonstandard wavelet basis:

$$F_{\mathbf{j}}(\kappa_s, \lambda_s, \kappa_r, \lambda_r) \begin{cases} \tilde{\phi}_{l^j m^i_\kappa}(\kappa_s) \tilde{\phi}_{l^j m^i_\lambda}(\lambda_s) \phi_{l^j m^j_\kappa}(\kappa_r) \phi_{l^j m^j_\lambda}(\lambda_r) & \nu^j = 0 \\ \tilde{\psi}_{l^j m^i_\kappa}(\kappa_s) \tilde{\phi}_{l^j m^i_\lambda}(\lambda_s) \phi_{l^j m^j_\kappa}(\kappa_r) \phi_{l^j m^j_\lambda}(\lambda_r) & \nu^j = 1 \\ \tilde{\phi}_{l^j m^i_\kappa}(\kappa_s) \tilde{\psi}_{l^j m^i_\lambda}(\lambda_s) \phi_{l^j m^j_\kappa}(\kappa_r) \phi_{l^j m^j_\lambda}(\lambda_r) & \nu^j = 2 \\ \vdots & \\ \tilde{\psi}_{l^j m^i_\kappa}(\kappa_s) \tilde{\psi}_{l^j m^i_\lambda}(\lambda_s) \psi_{l^j m^j_\kappa}(\kappa_r) \psi_{l^j m^j_\lambda}(\lambda_r) & \nu^j = 15 \end{cases} \tag{32}$$

where

$$\mathbf{j} = (\nu_j, l_j, m^i_\kappa, m^i_\lambda, m^j_\kappa, m^j_\lambda) \tag{33}$$

So $F_{\mathbf{j}}(\kappa_s, \lambda_s, \kappa_r, \lambda_r)$ is $B_{\mathbf{j}}(\kappa_s, \lambda_s, \kappa_r, \lambda_r)$ with dual scaling functions and wavelets substituted for the primal scaling functions and wavelets in the incident directional components only.

If we then represent the BRDF in Nusselt coordinates with this basis:

$$f_r(\kappa_s, \lambda_s, \kappa_r, \lambda_r) = \sum_{\mathbf{j}} f^r_{\mathbf{j}} F_{\mathbf{j}}(\kappa_s, \lambda_s, \kappa_r, \lambda_r) \tag{34}$$

and

$$L_i(x, y, \kappa_s, \lambda_s) = \sum_{\mathbf{k}} b_{\mathbf{k}} B_{\mathbf{k}}(x, y, \kappa_s, \lambda_s) \tag{35}$$

where

$$\mathbf{k} = (\nu_k, l_k, m^k_u, m^k_v, m^k_\kappa, m^k_\lambda) \tag{36}$$

then, applying (7) and again requiring either f_r or L_i (or both) to vanish outside the directional limiting circle (thus allowing us to extend our integration to $(-\infty..\infty)$), the reflected radiance is

$$\begin{aligned} L_r &= 4 \int_0^1 \int_0^1 f_r(\mathbf{S}^{+\prime}, \mathbf{V})\, L_i(\mathbf{S}^{+\prime})\, d\kappa'_s\, d\lambda'_s \\ &= 4 \sum_{\mathbf{j}} \sum_{\mathbf{k}} f^r_{\mathbf{j}} b_{\mathbf{k}} \langle F_{\mathbf{j}} \mid B_{\mathbf{k}} \rangle_{\kappa_s, \lambda_s} \end{aligned} \tag{37}$$

Making use of biorthogonality via (13), we derive a wavelet representation of the post-interaction radiance:

$$L_r = \sum_{\mathbf{n}} b^r_{\mathbf{n}} B_{\mathbf{n}}(x, y, \kappa_s, \lambda_s) \tag{38}$$

where

$$\begin{aligned} b^r_{\mathbf{n}} &= b^r_{\nu l m_u m_v m_\kappa m_\lambda} \\ &= 4 \sum_{m^i_\kappa, m^i_\lambda} f^r_{\nu l m^i_\kappa m^i_\lambda m_\kappa m_\lambda} b^i_{\nu l m_u m_v m^i_\kappa m^i_\lambda}. \end{aligned} \tag{39}$$

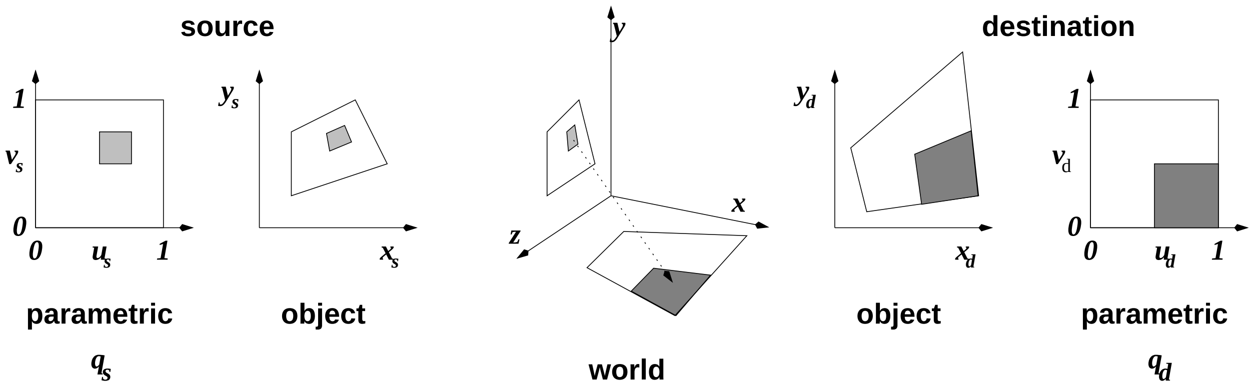

Figure 3: Coordinate Systems. The gray quadrilaterals represent the support of source and destination bases.

If both $\mathbf{f}^r$ and $\mathbf{b}^i$ have sparse representations, we can efficiently calculate $\mathbf{b}^r$ by traversing the shorter representation.

We can do the same thing with a BTDF and so represent general surface interactions: reflection, refraction, and transmission.

4 Application

Let us apply some of the concepts of the previous section to a classic illumination problem: the transport of radiation between two arbitrarily-oriented polygons.

4.1 Coordinate Systems

To establish the $\mathbf{q}_s \leftrightarrow \mathbf{q}_d$ mapping we need in order to transport radiance from source to destination, we must deal with several coordinate systems, as shown in Figure 3: source parametric, source object, world, destination object, and destination parametric. Obviously, source object to destination object is best done with a conventional affine transform (with projection), but there are several choices possible for the parametric $\leftrightarrow$ object mappings: rectilinear, perspective, and bilinear. All of these are local to the sending or receiving surface.

4.1.1 Rectilinear

This is the simplest possible mapping:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} W & 0 \\ 0 & H \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{40}$$

W and H are the dimensions of a bounding rectangle. If the object is a rectangle, an obvious strategy is to choose coordinates in which W is its width and H is its height. This will ensure that the (inverse) transformed object completely fills the unit square. Otherwise, or in the more general case of an arbitrarily-sided polygon, when computing $T_{\mathbf{jk}}$ we must clip the (square) support of $B_{\mathbf{j}}$ (source) or $B_{\mathbf{k}}$ (destination) against the polygon when integrating.

A major advantage of the rectilinear mapping is the ease of computation of the Jacobian determinant:

$$\left| \frac{\partial(x,y)}{\partial(u,v)} \right| = WH \tag{41}$$

which makes the power flux computation shown in (23) very easy:

$$\Phi = 4WH \sum_{\mathbf{j}} b_{\mathbf{j}} \left\langle B_{\mathbf{j}} \mid 1 \right\rangle_q . \tag{42}$$

especially in the case of Haar wavelets:

$$\Phi = 4WHb_{\mathbf{0}}. \tag{43}$$

4.1.2 Perspective

This mapping allows us to represent the more general quadrilaterals without the need to clip:

$$\begin{bmatrix} xw \\ yw \\ w \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{44}$$

where the A_{ij}'s are easily-determined functions of the quadrilateral vertices.

A straight line in perspective parametric coordinates $au + bv + c = 0$ transforms to a straight line $a'x + b'y + c'$ in object coordinates. This means that a quadrilateral in object coordinates will map to a quadrilateral in parametric coordinates and vice versa. This has favorable implications for transport coefficient computation that we will discuss below.

One drawback of a perspective mapping is that if the quadrilateral approaches degeneracy (a triangle, for instance), the appearance of a uniform grid in parametric space becomes increasingly nonuniform.

The power flux computation of (23) is not as easy as in the rectilinear case, but may still be analytically done for basis functions B_{j} with closed-form representations, such as splines.

4.1.3 Bilinear

As with 2-D perspective, this mapping also allows us to represent quadrilaterals without clipping. If the quadrilateral is defined by four points $\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$ (in CCW order), the customary bilinear mapping applies:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1-v & v \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 & \mathbf{p}_1 \\ \mathbf{p}_3 & \mathbf{p}_2 \end{bmatrix} \begin{bmatrix} 1-u \\ u \end{bmatrix} \tag{45}$$

Unlike the perspective case, we can treat triangles as degenerate quadrilaterals, albeit with some irregular object space meshing. However, this mapping is not without its own drawbacks. While the parametric-to-object mapping is straightforward, the inverse object-to-parametric mapping is, in fact, double valued: a given (x, y) usually has two solutions (u, v), one of them inside the unit rectangle, one outside. In order to distinguish the two cases, we must clip in object space before inversion.

A more serious drawback is that straight lines are not, in general, preserved. The inverse projection of a straight line $ax + by + c = 0$ into parametric space is, in general, a hyperbola. This also has implications for transport coefficient computation: it complicates determination of the limits of integration.

Again, while the still more complicated Jacobian of this transform makes the power flux computation of (23) more difficult, it may still be done for choices of $B_{\mathbf{j}}$ with closed-form representations, such as splines.

4.2 Choice of Wavelet

Except where noted, our discussions in Appendix A and Section 3 did not depend on any particular choice of wavelet. For implementation purposes, we have to choose one. There are several good reasons for choosing Haar wavelets.

As we mentioned in Section A.6, the fast wavelet transform can be performed in $O(N)$ time, but if we allow for a varying dimensionality D and wavelet basis, it is easy to see from (14) that the complexity is actually $O(W_h^D N)$ where W_h is the maximum width of the $\{h_j\}$ and $\{\tilde{h}_j\}$ (and, consequently, $\{g_j\}$ and $\{\tilde{g}_j\}$) coefficient sets.

For this reason, as the dimensionality increases, the rapidly-increasing operation count makes narrower filters more and more desirable, even though wider filters generally have better approximation properties. Since Haar is the narrowest possible wavelet filter ($W_h = 2$), it seems a wise strategy to make any multidimensional efforts first with Haar and move to wider bases later if Haar proves unsatisfactory.

An additional advantage of Haar wavelets over the others is the simplification of the calculation of the transport coefficients, irradiance and power flux. As (29) and (30) have shown, these coefficients can be computed entirely in terms of pure smoothing calculations. A four-dimensional Haar pure smoothing basis is a function that is constant ($= 4^l$) within a hypercube and zero outside of it. The resulting transport coefficients are volume integrals of the overlap between such a hypercube in destination parametric space and the object which is a projection of a hypercube in source parametric space into the destination space[3].

4.3 Problems with Transport Coefficient Computation

As the preceding section suggests, using Haar wavelets turns transport coefficient computation into volume integral evaluation. There are two practical problems that complicate the computation of that volume.

4.3.1 Some Source Points Do Not Project Into Destination Space

The source hypercube defines a range of positional and directional coordinates $\mathbf{q}_s$. Not all of these coordinates may map to points in the destination plane, much less the destination quadrilateral. This complicates any attempt at direct evaluation of the transport integrals.

[3]This may not be such a great simplification. After all, any arbitrarily complex 3-dimensional integral may be trivially turned into a 4-dimensional volume integral!

4.3.2 The Projected Hypercube Has Curved Sides

Even if all points in the source hypercube map to the destination plane, the nature of the resulting volume, is not trivial. Needless to say, the projection of a source parametric hypercube into destination parametric space is not a hypercube. If, however, we could choose coordinate systems such that the source hypercube mapped to a polytope in destination space, we could take advantage of computational geometric techniques to first clip it against the destination hypercube and then compute the volume of the resulting polytope. Unfortunately, this is not possible because the source hypercube does not project to a polytope.

Consider the coordinate systems described in Section 4.1. Even if we choose rectilinear parametric $\leftrightarrow$ object mappings, the source object to destination object transform involves a projection. Hence, at best, the $q_s \rightarrow q_d$ mapping has a nonlinear dependence on the directional components.

As a result, the projection of the source hypercube into destination parametric space has curved sides. Furthermore, the curvature is such that we cannot guarantee that the convex hull of the polytope formed by projecting the 16 corners of the source hypercube into destination space contains the hypervolume[4].

4.4 Integration Techniques

For these reasons, we must resort to multidimensional numerical integration schemes. Before considering candidate techniques, we first make an observation about the dimensionality required for numerical integration.

[4]We have not fully pursued the possibility of an approximation of the projected hypercube by its convex hull here.

4.4.1 Reducing the Dimensionality

As (28) indicates, the computation of transport coefficients is intrinsically four-dimensional: two directional integrals and two positional integrals. If we take the two outermost integrals over direction and restrict our discussion to pure Haar smoothing components ($\nu_j = \nu_k = 0$, as (31) permits), it is then evident that

$$T_{\mathbf{jk}} = 4^{l_j + l_k} \iint_{S_{\mathbf{k}}} G_{\mathbf{j}}(\kappa'_d, \lambda'_d)\, A_{\mathbf{jk}}(\kappa'_d, \lambda'_d)\, d\kappa'_d\, d\lambda'_d \tag{46}$$

$S_{\mathbf{k}}$ is the (square) directional support of $\tilde{B}_{\mathbf{k}}$. $G_{\mathbf{j}}(\kappa_d, \lambda_d)$ is a geometric function which is equal to one if a ray from the destination plane projected backwards along the (κ_d, λ_d) direction reaches the source plane and falls within the directional support of $B_{\mathbf{j}}$. Otherwise, it is zero. $A_{\mathbf{jk}}(\kappa_d, \lambda_d)$ is the area of the intersection of the spatial support of $B_{\mathbf{k}}$ in destination parametric space with the projection (in the κ_d, λ_d direction) to destination parametric space of the spatial support of $B_{\mathbf{j}}$.

We are now in a position to evaluate the coordinate mappings given in Section 4.1 to see which of them makes $A_{\mathbf{jk}}(\kappa_d, \lambda_d)$ easy to compute. All the mappings transform lines of constant u or v to lines in object space, so any of them would work for the source parametric to source object mapping. Only rectilinear and perspective mappings, however, transform arbitrary lines in object space to lines in parametric space. If we choose either of them for our destination object to destination parametric mappings, computation of $A_{\mathbf{jk}}(\kappa_d, \lambda_d)$ amounts to clipping the projected quadrilateral to the spatial support of $B_{\mathbf{k}}$ using a conventional polygon clipping algorithm and computing the resulting area. This allows us to reduce the dimensionality that we need to integrate numerically from four to two.

4.4.2 Numerical Quadrature

Regardless of the number of dimensions, numerical integration techniques are all based on some form of quadrature:

$$\int f(\mathbf{x}) d\mathbf{x} \approx \sum_{i=1}^{N_{samp}} w_i f(\mathbf{x}_i) \tag{47}$$

where N_{samp} is the number of samples. Techniques differ principally in their choices of weights w_i and sample points $\mathbf{x}_i$. Zwillinger [17] provides an extensive survey of these. The ones we have chosen to evaluate are:

- *trapezoidal*: a regular grid approximating f linearly in each dimension
- *Romberg*: a multilevel (Richardson) extrapolation (in terms of the grid spacing) of trapezoidal results
- *Monte Carlo*: $\mathbf{x}_i$'s chosen from a (possibly stratified) pseudo-random sequence[5]
- *Halton*: similar to Monte Carlo, but using *quasi-random* numbers generated according to number theoretical considerations
- *Hammersley*: an alternative quasi-random method

While pseudo-random and quasi-random techniques are generally preferred for multidimensional quadrature, we include trapezoidal and Romberg techniques to explore the two-dimensional case, where Monte Carlo has error of order $O(N_{\text{samp}}^{-1/2})$[6] and the trapezoidal rule has error of order $O(N_{\text{samp}}^{-3/2})$. We must be careful to use these error estimates cautiously, however, since our integrand, $G(\kappa_d, \lambda_d)\, A_{\mathbf{jk}}(\kappa_d, \lambda_d)$ contains discontinuities that error analyses do not account for.

[5] Glassner [8] is a good overview of pseudo- and quasi-random integration methods in image synthesis.

[6] This is true regardless of the dimensionality of the problem and is one of the appeals of Monte Carlo integration.

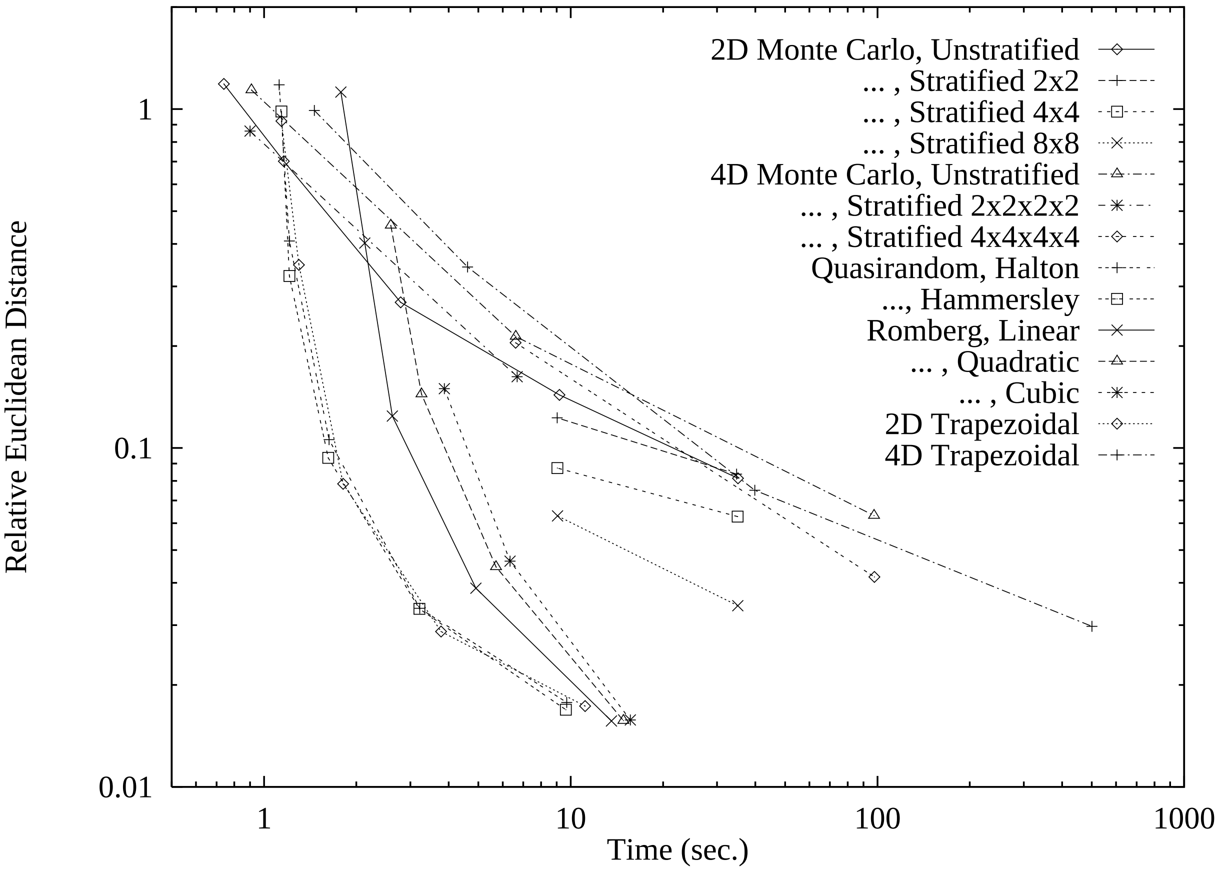

Figure 4: Relative Euclidean Distance vs. Time for Different Integration Techniques and Numbers of Samples

4.4.3 Comparison

To evaluate the accuracies of the various methods, we have applied them to a test problem: computing the set of all $T_{\mathbf{jk}}$ for a given $\mathbf{j}$ and geometry. To analyze the results we can treat this set as a vector in a K-dimensional space, where $K = 16^{l_{\max}+1}$ and $l_{\max}$ is the maximum level we have coefficients for. (I.e., we have K values of $\mathbf{k}$.)

To compare results against a reference, we create a reference vector $T^{\mathrm{ref}}_{\mathbf{jk}}$ using an extremely long (25 CPU-minutes) integration time and then adopt two metrics. The first, which we call the *relative Euclidean distance* (hereafter, RED) is the Euclidean distance (L^2 norm) of $T_{\mathbf{jk}}$ from $T_{\mathbf{jk}}^{\text{ref}}$ divided by the magnitude of $T_{\mathbf{jk}}^{\text{ref}}$:

$$\frac{\sqrt{\sum_{\mathbf{k}} \left(T_{\mathbf{jk}} - T_{\mathbf{jk}}^{\text{ref}}\right)^2}}{\sqrt{\sum_{\mathbf{k}} \left(T_{\mathbf{jk}}^{\text{ref}}\right)^2}}. \tag{48}$$

Figure 4 shows how the RED varies with integration time[7] and N_{samp} for the various techniques. Note that we allow the degree of extrapolation for Romberg integration to vary from linear to quadratic to cubic. The times shown are for an IBM Model RS/6000 POWERserver 560 workstation.

The second metric, which we call *relative maximum departure* (hereafter, RMD), is more conservative. It is the maximum absolute difference between $T_{\mathbf{jk}}$ and $T_{\mathbf{jk}}^{\text{ref}}$ (L_∞ norm) divided by the maximum absolute value of $T_{\mathbf{jk}}^{\text{ref}}$:

$$\frac{\max_{\mathbf{k}} \left|T_{\mathbf{jk}} - T_{\mathbf{jk}}^{\text{ref}}\right|}{\max_{\mathbf{k}} \left|T_{\mathbf{jk}}^{\text{ref}}\right|}. \tag{49}$$

Figure 5 shows how the RMD varies for the same parameters as Figure 4.

From these plots, we can draw several conclusions:

- The two figures are qualitatively similar: a method that does well by one metric generally does well by the other. This gives us some confidence that these metrics are valid.
- Stratification improves Monte Carlo results. (This comes as no surprise.)
- The time required for Monte Carlo methods being linear in N_{samp}, they all approximate the expected $O(N_{\text{samp}}^{-1/2})$ behaviour.

[7]Rather than N_{samp}, time is the appropriate abscissa here. Since the time required to evaluate the integrand varies with dimensionality, fewer samples do not necessarily imply faster computation.

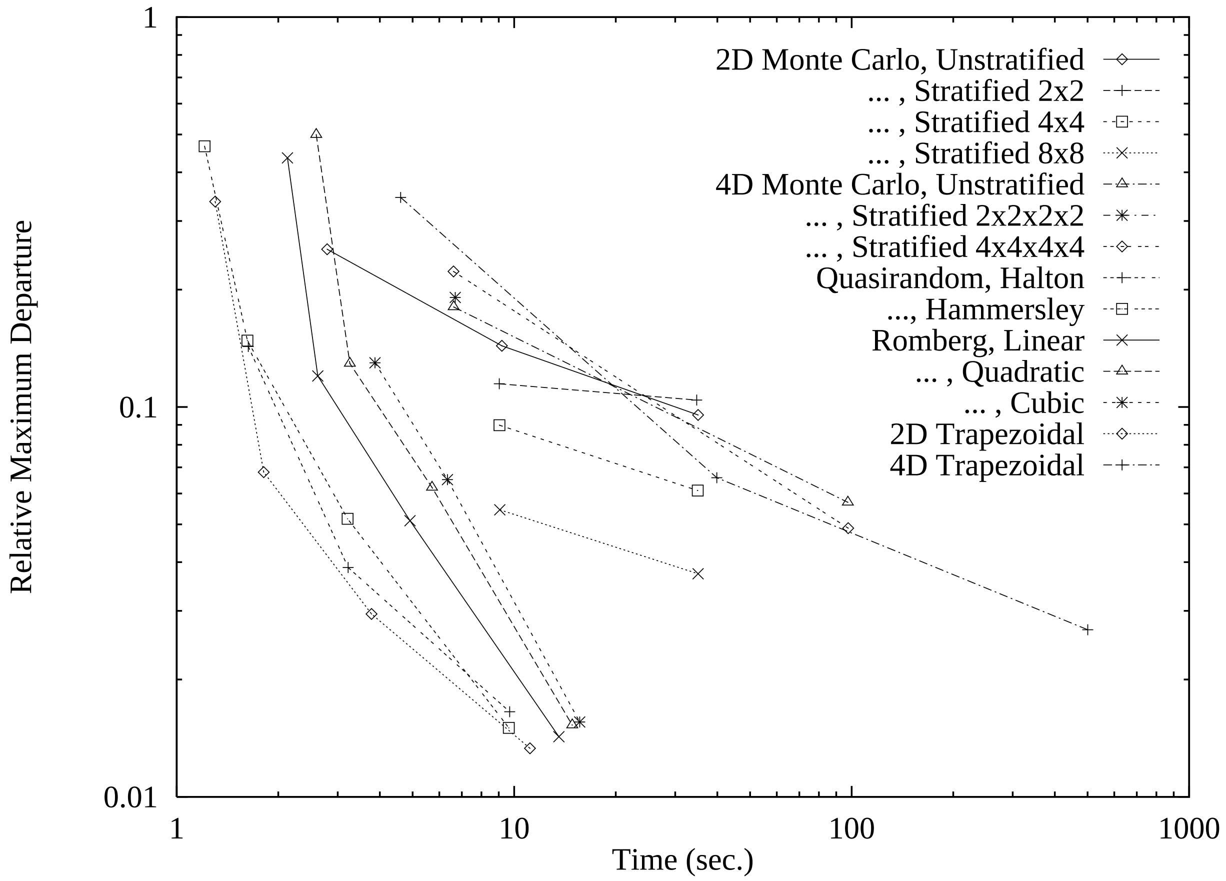

Figure 5: Relative Maximum Departure vs. Time for Different Integration Techniques and Numbers of Samples

- For a given integration time, the 4D trapezoidal method does generally worse than the other methods.

- Increasing the degree of extrapolation for Romberg (2D) integration generally makes matters worse. This is presumably a result of the discontinuities in the integrand we discussed above.

- Quasi-random methods give similar and comparatively good results. This reinforces the findings of Keller [11].

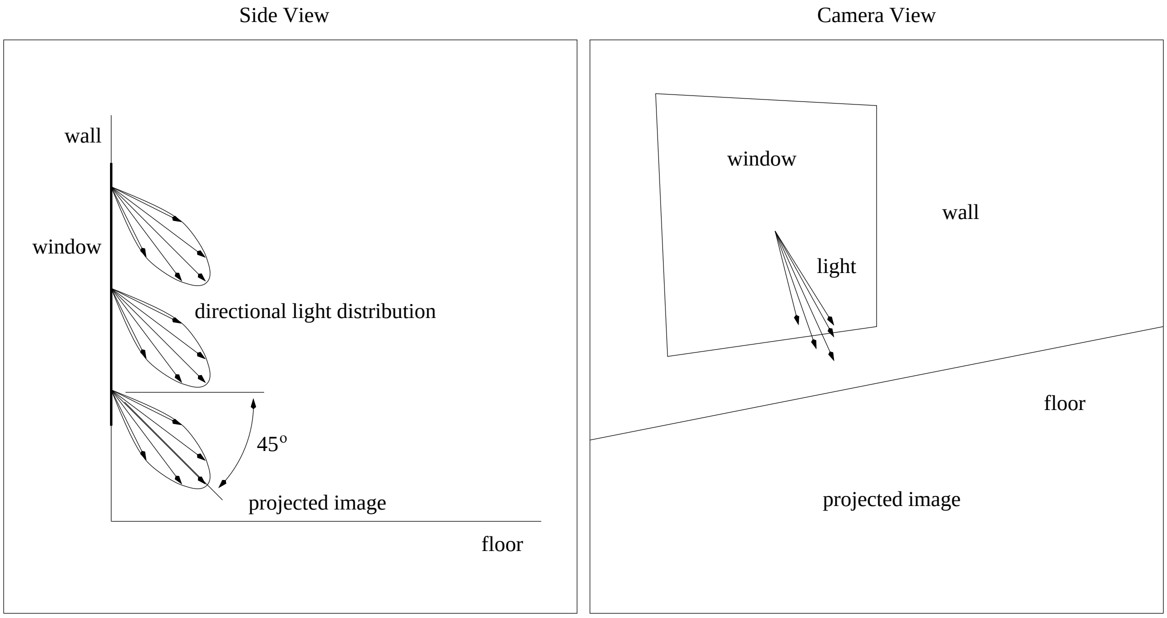

Figure 6: Geometry Used for Example of Transport

The most surprising observation, however, is how well the straightforward 2D trapezoidal rule does. For the RED metric, it is comparable with the best of the other methods, quasi-random, and for the RMD metric it does noticeably better for short integration times.

For this reason, we are favouring use of the 2D trapezoidal method in transport coefficient computation. It remains to be seen, however, if other methods are possible which might be more suitable for an integrand with such discontinuities as we have.

4.5 Example of Transport

For a test configuration, we imagine light shining through the square stained glass window shown in Figure 11. As illustrated in Figure 6, the incident light shines down with a distribution peaking at an angle of 45° from the horizontal, diffused by the glass according to a distribution proportional to the 4th power of the cosine of the angle between the propagation direction and the peak direction. The light is transported to the floor and is collected there. Figure 12 shows the complete, inversely-transformed 4D results. Each of the small images represents the spatial variation of radiance in the fixed direction given by the image's position in the matrix. Figure 13 is a detailed view of the brightest part of Figure 12. This computation of $32 \times 32 \times 32 \times 32$ coefficients (in red, green, and blue channels) compressed by 95% required 12.4 hours of CPU time on an IBM RS/6000 POWERserver 560 workstation.

Needless to say, this is impractical for a single frame, but the result is reusable. Figure 14 shows several frames generated with an otherwise conventional raytracer modified to treat a wavelet radiance distribution as a "4-D texture". Each frame is generated from a different camera position. Given the diffuse nature of the problem, the resulting images look adequate and contain none of the "noise" common to the usual Monte Carlo approach to this sort of problem.

5 Compression Strategies

Representing a maximum level of resolution $l_{\max}$ on a surface requires $16^{l_{\max}+1}$ possible wavelet coefficients. Clearly, compression is called for. In Section A.7, we describe how wavelet properties provide an L^2-optimal compression strategy – thresholding low-magnitude coefficients. In this case, the data was single-channel and single dimensional (i.e., each wavelet coefficient's support was unique).

For wavelet radiance, we are dealing with multichannel data (typically RGB) in four dimensions. We not only want to guarantee a good approximation of the data with the compressed coefficients, but also efficient use of storage and fast reconstruction. Let us consider the import of these practical considerations and give an example of compression. We will also discuss how compression affects transport coefficient storage.

5.1 Multichannel Grouping

As we mentioned in Section 2, we have been treating data monochromatically throughout this paper. In practical applications, however, we must evaluate (27) for all three (or however many) channels. Having assumed a non-participating medium, the transport coefficients $T_{\mathbf{jk}}$ are achromatic. Evaluation of (27) simply means multiplying each element of a (now) 3-vector $b^s_{\mathbf{j}}$ by the same value of $T_{\mathbf{jk}}$ and accumulating the result in another 3-vector $b^d_{\mathbf{k}}$.

In the absence of the need for compression, it would be convenient to group all channels of $b^s_{\mathbf{j}}$ into a single group, rather than create a separate representation for each channel. In the presence of compression, however, each channel has its own threshold. If one component is above its threshold but the other two are below theirs, saving the latter is a waste of storage. This would be mitigated, however, if there were a high degree of correlation between the magnitudes of the coefficients. Certainly, a wavelet representation of white light given off by a luminaire displays a high degree of correlation. When this light is reflected off a surface of varying spectral reflectivity, however, that correlation will be diminished. By how much depends on the nature of the surface.

We have two choices here: to group or not to group multichannel data. For the time being, we have chosen the former – believing that there is sufficient correlation in most situations to justify grouping. This definitely requires further study.

5.2 Hashing Coefficients

Given a sparse set of wavelet radiance coefficients $\{b_{\mathbf{k}}\}$, we need to store them in a way that facilitates the mapping of the nonstandard multiresolution index $\mathbf{k} \rightarrow b_{\mathbf{k}}$ needed to perform the transport operation (27). The obvious way to do this is with a hash table. Not knowing the set of destination indices $\{\mathbf{k}\}$ in advance prohibits perfect hashing, so it is necessary that

$l_{\max}$	1	2	3	4	5	6	7	8	9	10
Number of Bits	9	14	18	23	27	31	35	40	44	48

Table 1: Number of Bits Required to Store a Nonstandard Multiresolution Index as a Function of the Maximum Resolution Level

the hashing scheme allows for collisions and that the hash table entries contain $\mathbf{k}$ as well as $b_{\mathbf{k}}$ values. Nonstandard multiresolution indices contain six integer values:

- the basis selector $\nu \in \{0 \ldots 15\}$, as used in (11),
- the level $l \in \{0 \ldots l_{\max}\}$, where $l_{\max}$ is the maximum depth of resolution,
- a vector of four offset values $\mathbf{m}$ with each component $m_i \in \left\{0 \ldots 2^l - 1\right\}$ (for Haar).

The total number of bits required is then $4 + \lceil \log_2(l_{\max} + 1) \rceil + 4l_{\max}$. Table 1 shows this quantity for several possible values of $l_{\max}$. If we represent the coefficients as 32-bit floating point values, then for any value of $l_{\max} > 6$ we would use more storage for indices than for coefficients. If, for efficiency of index computations, we use standard 8-bit variables for ν and l and 16- or 32- bit variables for the elements of $\mathbf{m}$, the situation is even worse. This clearly inefficient use of storage prompts us to examine alternatives.

5.3 Hashing Nodes Instead of Coefficients

In one dimension, a node in the wavelet pyramid contains a single wavelet coefficient. In four dimensions, such a node has one pure wavelet ($\nu = 15$) coefficient and fourteen mixed wavelet/smoothing ($\nu \in \{1 \ldots 14\}$) coefficients. (Recall that the pure smoothing ($\nu = 0$) coefficient may be reconstructed from the node's ancestors and does not need to be kept at the node.) All coefficients at a given node correspond to basis functions with the same

(Haar) or similar (other bases) support.

We might therefore expect to find a higher degree of correlation in magnitude between coefficients that belong to the same node and coefficients that do not. This suggests that we can reduce the index storage overhead by hashing entire nodes rather than individual coefficients.

5.4 Example of Radiance Compression

Figure 15 shows the effect of compressing nodes of the wavelet representation of the radiance by differing ratios prior to transport in the above example. It is evident that considerable compression is possible. Although beyond the scope of this paper, more sophisticated reconstruction techniques could be applied to remove the artifacts visible for high compression.

5.5 Storing Transport Coefficients

The above application, radiative transport between two polygons, does not require the storage of transport coefficients. In other situations, however, storing them may be desirable. In this section we will outline the considerations to be made in deciding whether or not they should be stored, and if so, how?

Like form factors in radiosity, the transport coefficients are dependent only upon geometry. If the same geometry occurs in several different instances, precomputing the transport coefficients means they only have to be computed once.

Counterbalancing the argument for storing coefficients is the observation that, assuming wavelet compressibility, $b^s_{\mathbf{j}}$ is sparse. This means that only those elements of $T_{\mathbf{jk}}$ for which we have nonzero $b^s_{\mathbf{j}}$ values need to be computed.

If we decide to store transport coefficients, we face another problem: $T_{\mathbf{jk}}$ is large. If $l_{\max}$ is the maximum level of resolution on the source surface, there are $16^{l_{\max}+1}$ possible values of $\mathbf{j}$ (for Haar). Assuming the same resolution is required for the destination surface, $T_{\mathbf{jk}}$ would require $256^{l_{\max}+1}$ elements. If represented as a conventional array, $l_{\max} = 3$ would exhaust a 32-bit address space!

Fortunately, $T_{\mathbf{jk}}$ is also sparse in this sense: for a given $\mathbf{j}$, the number of destination basis functions $B_{\mathbf{k}}$ that it affects is much smaller than the number of possible values of $\mathbf{k}$. If we do decide to put $T_{\mathbf{jk}}$ in a lookup table, that table will have to be sparse, in which case the nodal compression strategies we discussed in Section 5.3 would also apply to stored transport coefficients.

6 Conclusions

We have shown how a wavelet representation in Nusselt coordinates leads to simplified computation of radiative transport and surface interaction. The results are applicable to any wavelet basis, although the Haar basis provides for an easier implementation, and they apply to any surface whose characteristics can be represented with a BRDF or BTDF. We have also presented an efficient computational scheme for transport coefficients using a combined geometric and numerical algorithm and discussed compression strategies for radiance and transport coefficients.

We are continuing work with improved transport coefficient integration techniques and plan to take further advantage of wavelet representations, such as knowledge of the destination's reflective properties, to reduce the amount of computation required and allow us to go to finer resolutions. We are incorporating wavelet radiative transport and surface interaction into the global illumination scheme described in [13].

A One-Dimensional Wavelet Properties

In this appendix, we will review some of the properties of wavelets that make them particularly suitable for the representation of radiance. The standard reference on wavelets is Daubechies [4], from which much of this appendix is derived. We will confine ourselves here to one-dimensional wavelets. Section 3.1 will extend these properties to multiple dimensions.

A.1 Scaling Functions and Wavelets

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

$$\phi_{lm}(x) = 2^{l/2}\phi(2^l x - m) \tag{A1}$$

each level l corresponds to a function space V_l, which is part of a nested sequence of subspaces $\ldots \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \ldots$[8].

Scaling functions have the property that

$$\int_{-\infty}^{+\infty} \phi(x)dx \neq 0 \tag{A2}$$

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 then dilations and translations of a *mother wavelet* $\psi(x)$:

$$\psi_{lm}(x) = 2^{l/2}\psi(2^l x - m) \tag{A3}$$

Wavelets have the property

$$\int_{-\infty}^{+\infty} \psi(x)dx = 0 \tag{A4}$$

[8]This notation is a departure from that of Daubechies [4] that is more suitable for discrete data: an increasing subscript of V corresponds to an increasing number of samples.

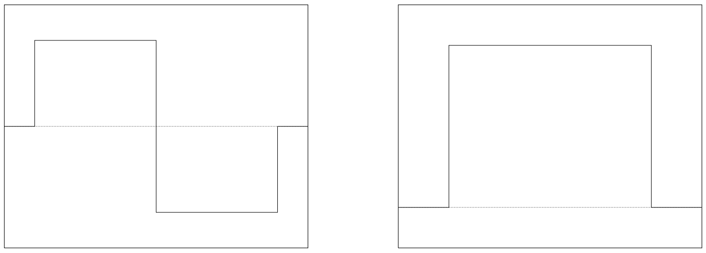

Figure 7: Haar Wavelet and Smoothing Functions. Haar wavelets are the simplest possible wavelet and are the most commonly used wavelets in graphics. They have only one vanishing moment.

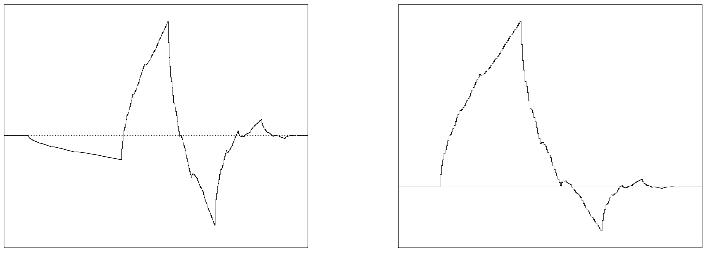

Figure 8: Daubechies-4 Wavelet and Smoothing Functions. Daubechies wavelets were the first compact orthogonal wavelets discovered. This particular wavelet has two vanishing moments.

Figures 7-9 show some commonly-used wavelets and their corresponding smoothing functions.

A.2 Multiresolution Refinement Equations

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 $\phi(2x - j)$ for V_1:

$$\phi(x) = \sqrt{2} \sum_j h_j \phi(2x - j) \tag{A5}$$

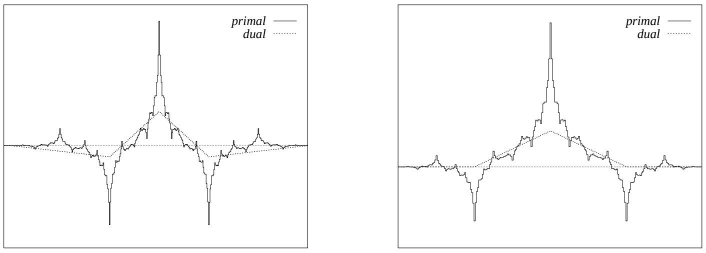

Figure 9: Linear Spline Wavelet And Smoothing Functions. These wavelets are biorthogonal and have two vanishing moments. Unfortunately, while the dual wavelets and smoothing functions have a simple closed form, this is not true for the primals.

This also holds for ψ:

$$\psi(x) = \sqrt{2} \sum_j g_j \phi(2x - j) \tag{A6}$$

These are the *dilation* or *refinement equations.* Wavelet bases differ principally in their choices of $\{h_j\}$ (which determines $\{g_j\}$).

Using the enumerated bases:

$$\begin{aligned} \phi_{lm}(x) &= \sqrt{2} \sum_j h_j \phi_{l+1,2m+j}(x) \\ \psi_{lm}(x) &= \sqrt{2} \sum_j g_j \phi_{l+1,2m+j}(x) \end{aligned} \tag{A7}$$

A.3 Orthogonal Wavelets

We define the inner product of two functions f and g with respect to a x:

$$\langle f \mid g \rangle_x \equiv \int_{-\infty}^{+\infty} f(x') g(x') dx' \tag{A8}$$

(Using x as a subscript is non-traditional, but will become useful when we consider multidimensional wavelets.)

Some wavelets are orthogonal:

$$\langle \psi_{lm} \mid \psi_{l'm'} \rangle_x = \delta_{ll'} \delta_{mm'} \tag{A9}$$

But these have undesirable features: except for Haar, they are not symmetric and they do not include useful functions like splines.

A.4 Biorthogonal Wavelets

We can construct biorthogonal bases by using four functions instead of two: wavelets ϕ_{lm} and $\tilde{\phi}_{lm}$ and smoothing functions ψ_{lm} and $\tilde{\psi}_{lm}$. These are defined defined, respectively, by four sets of coefficients: $\{h_j\}$, $\{\tilde{h}_j\}$, $\{g_j\}$, and $\{\tilde{g}_j\}$. $\{h_j\}$ determines $\{\tilde{g}_j\}$ and $\{\tilde{h}_j\}$ determines $\{g_j\}$. If done consistently, the primal and dual components are entirely interchangeable.

For these,

$$\left\langle \psi_{lm} \mid \tilde{\psi}_{l'm'} \right\rangle_x = \delta_{ll'} \delta_{mm'} \tag{A10}$$

In the rest of this section, we'll assume the more general biorthogonality, since we can always treat orthogonal wavelets as a special case of biorthogonal wavelets.

A.5 Wavelet Projections and Approximation

Let us discuss the ability of a wavelet representation to approximate an arbitrary function f. Let $\mathcal{P}_l f$ be the projection of a function $f \in L^2$ into the subspace V_l:

$$\mathcal{P}_l f(x) = \sum_m \left\langle f \mid \tilde{\phi}_{lm} \right\rangle_x \phi_{lm}(x) \tag{A11}$$

It can be shown

$$\| f - \mathcal{P}_l f \|_2 \leq C 2^{-lN_v} \sqrt{\sum_n \| f^{(n)} \|^2} \tag{A12}$$

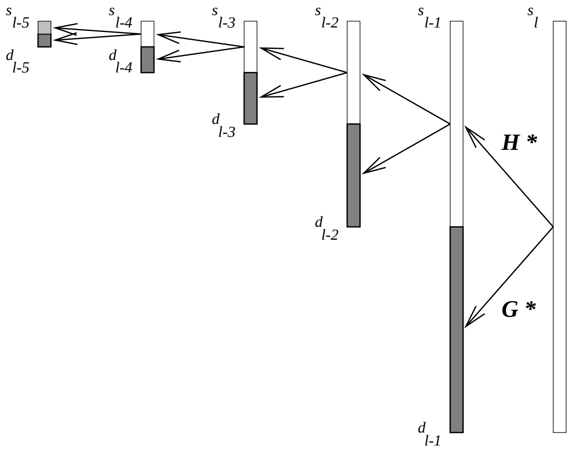

Figure 10: The Fast Wavelet Transform

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

$$\int x^n \tilde{\psi}(x) dx = \left\langle x^n \mid \tilde{\psi}_{00} \right\rangle_x = 0 \tag{A13}$$

What (A12) means is that we can always decrease the L^2 error of a wavelet approximation by increasing the number of levels l or by choosing wavelets with a higher number of vanishing moments.

A.6 The Fast Wavelet Transform

The fast wavelet transform, first developed by Beylkin, et al. [1], starts with a set of data $s_{lm}, m = 0 \ldots N-1$, where $N = 2^l$. We treat these as coefficients of ϕ_{lm} and can compute a wavelet transform in $O(N)$ operations. Figure 10 shows this schematically: starting on the right, each step operates on half the data of the previous step and there are l steps. This is another advantage of wavelet methods over Fourier techniques, which typically require $O(N \log N)$ operations.

The transform leaves us with $2^l - 1$ wavelet coefficients (dark gray in Figure 10) and 1 smoothing coefficient (light gray). The hierarchical arrangement of these coefficients is sometimes referred to as the "wavelet pyramid".

A.7 Wavelet Compression

Even with the wavelet pyramid, we are still dealing with N coefficients. We have not saved any storage (yet).

It can be shown that if f is of the form

$$f(x) = \sum_{l,m} w_{lm} \psi_{lm}(x) \tag{A14}$$

and we approximate it by

$$F(x) = \sum_{w_{lm} \in \tilde{W}} w_{lm} \psi_{lm}(x) \tag{A15}$$

where the $\tilde{W}$ is a subset of the set of coefficients $\{w_{lm}\}$, that

$$\| f - F \|^2 = \sum_{w_{lm} \notin \tilde{W}} |w_{lm}|^2 \tag{A16}$$

This gives us a convenient error metric. It also tells us that the optimal compression scheme discards the coefficients with smaller magnitudes first – a thresholding process.

References

[1] G. Beylkin, Ronald R. Coifman, and V. Rokhlin. Fast wavelet transforms and numerical algorithms i. *Communications on Pure and Applied Mathematics*, 44:141–183, 1991.

[2] Per Henrik 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.

[3] Michael F. Cohen and John R. Wallace. Radiosity and realistic image synthesis. Academic Press Professional, San Diego, CA, 1993. Excellent book on radiosity algorithms.

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

[5] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics, Principles and Practice, Second Edition.* Addison-Wesley, Reading, Massachusetts, 1990. Overview of research to date.

[6] Alain Fournier. From local to global illumination and back. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 127–136, Dublin, Ireland, June 1995. Springer-Verlag.

[7] Alain Fournier, Eugene Fiume, Marc Ouellette, and Chuan K. Chee. Fiat lux. Technical Report 90-1, Dynamic Graphics Project, University of Toronto, January 1990.

[8] Andrew S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann, 1995.

[9] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In Holly Rushmeier, editor, *Proceedings of SIGGRAPH '96 (New Orleans, LA, August 4-9, 1996)*, Computer Graphics Proceedings, Annual Conference Series, pages 43–54. ACM SIGGRAPH, ACM Press, August 1996.

[10] 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.

[11] Alexander Keller. Quasi-monte carlo radiosity. In *Seventh Eurographics Workshop on Rendering*, Porto, Portugal, June 1996.

[12] Marc Levoy and Pat Hanrahan. Light field rendering. In Holly Rushmeier, editor, *Proceedings of SIGGRAPH '96 (New Orleans, LA, August 4-9, 1996)*, Computer Graphics Proceedings, Annual Conference Series, pages 31–42. ACM SIGGRAPH, ACM Press, August 1996.

[13] 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.

[14] W. Nusselt. Grapische bestimmung des winkelverhaltnisses bei der warmestrahlung. *Zeitschrift des Vereines Deutscher Ingenieure*, 72(20):673, 1928. historical interest.

[15] Peter Schroder and Pat Hanrahan. Wavelet methods for radiance computations. In *Fifth Eurographics Workshop on Rendering*, pages 303–311, Darmstadt, Germany, June 1994.

[16] Peter Schroeder, Steven Gortler, Michael Cohen, and Pat Hanrahan. Wavelet projections for radiosity. In Michael F. Cohen, Claude Puech, and Francois Sillion, editors, *Fourth Eurographics Workshop on Rendering*, pages 105–114. Eurographics, June 1993. held in Paris, France, 14–16 June 1993.

[17] Daniel Zwillinger. *Handbook of Integration.* Jones and Bartlett Publishers, Inc., Boston, MA, 1992.

 Robert R. Lewis is a doctoral candidate in Computer Science at the University of British Columbia. He holds a B. S. (1974) in Physics from Harvey Mudd College, an M. A. (1979) in Astronomy from the University of California at Berkeley, and an M. S. (1989) in Computer Science from the Oregon Graduate Institute of Science and Technology.

His research interests include rendering, computational geometry, and numerical methods.

 Alain Fournier is a Professor in the Department of Computer Science at the University of British Columbia and co-director of *Imager*, its computer graphics laboratory.

His research interests include the modelling of natural phenomena, shape modelling, surface structure modelling, filtering, global and local illumination.

Figure 11: A Stained Glass Window. This is the spatial component of radiance for the test configuration.

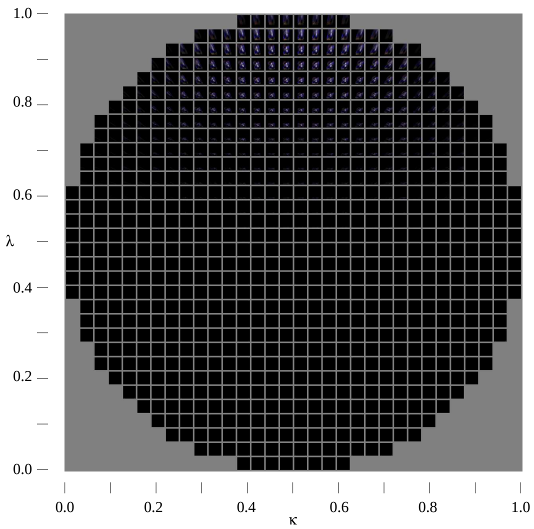

Figure 12: Four-Dimensional Results of Wavelet Radiative Transport

Figure 13: Detailed View of the Brightest Part of Figure 12

Figure 14: Example of Wavelet Transport. The resulting wavelet representation of radiance shown in Figure 12 can be sampled from different directions.

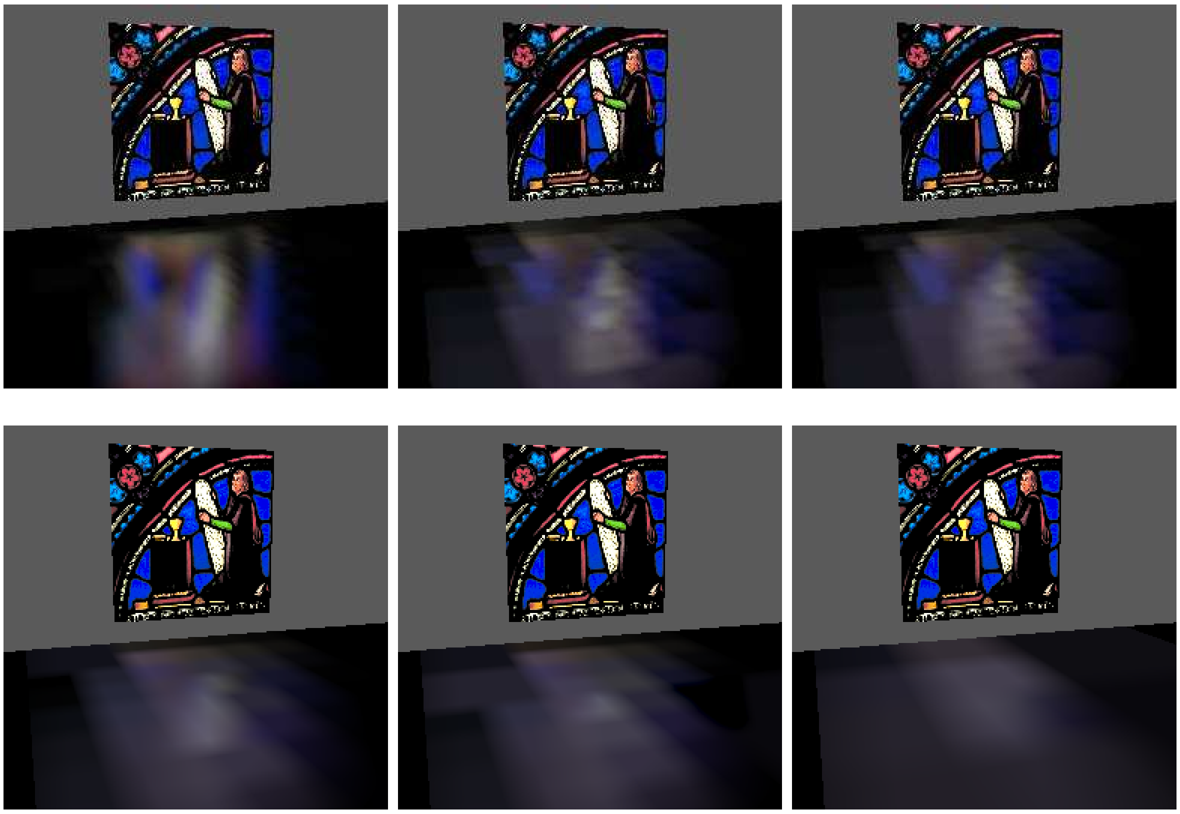

Figure 15: Effect of Compressing the Wavelet Radiance Distribution by (left-to-right and top-to-bottom) 2:1, 10:1, 20:1, 100:1, 200:1, and 1000:1