Wavelet Radiosity: Wavelet Methods for Integral Equations

Peter Schröder
California Institute of Technology

1 Introduction

In this chapter we will explain how wavelets can be used to solve integral equations. The example we use is an important integral equation in graphics, the radiosity equation. The radiosity equation governs the transport of light between surfaces under the assumption that all reflection occurs isotropically. The resulting integral equation is linear and can be analyzed as a linear operator. Since wavelets can be used as bases for function spaces, linear operators can be expressed in them. If these operators satisfy certain smoothness conditions—as radiosity does—the resulting matrices are approximately sparse and can be solved asymptotically faster if only finite precision is required of the answer.

We develop this subject by first introducing the Galerkin method which is used to solve integral equations. Applying the method results in a linear system whose solution approximates the solution of the original integral equation. This discussion is kept very general. In a subsequent section the realization of linear operators in wavelet bases is discussed. There we will show why the vanishing moment property of wavelets results in (approximately) sparse matrix systems for integral operators satisfying certain kernel estimates. After these foundations we change gears and describe some techniques recently introduced in the radiosity literature. A technique, known as Hierarchical Radiosity, is shown to be equivalent to the use of the Haar basis in the context of solving integral equations. Treating this example in more detail allows us to fill many of the mathematical definitions with geometric intuition. Finally we discuss the implementation of a particular wavelet radiosity algorithm and the construction of an oracle function which is crucial for a linear time algorithm.

In general we will concentrate on the arguments and intuition behind the use of wavelet methods for integral equations and in particular their application to radiosity. Many of the implementation details will be deliberately abstracted and they can be found by the interested reader in the references ([31, 17, 20]).

1.1 A Note on Dimensionality

The final application of the developments in this chapter will be to the problem of radiosity in 3D, i.e., the light transport between surfaces in 3D. Consequently all functions will be defined over 2D parameter domains. Initially we will discuss only 1D parameter domains to simplify the exposition. The chief advantage of this reduction in dimensionality lies in the fact that many quantities, which we have to manipulate, have a number of subscripts or superscripts which is directly proportional to the dimensionality of the problem. It is easy to loose sight of the essential ideas unless we limit ourselves to the 1D domain case. The 1D domain case corresponds to what is known as flatland radiosity [21], i.e., the exchange of light between line segments in the plane. Aside from dimensionality there is no essential difference between the integral equations governing 3D or flatland radiosity. Where appropriate we will be explicit about the changes necessary to go to 2D domains. In general the differences are limited to more indices to manipulate, or in the case of a program, more array dimensions to iterate over.

2 Galerkin Methods

Galerkin methods are a class of algorithms designed to solve integral equations of a particular kind [12]. In this section we begin with an introduction to the radiosity equation as a particular example of an integral

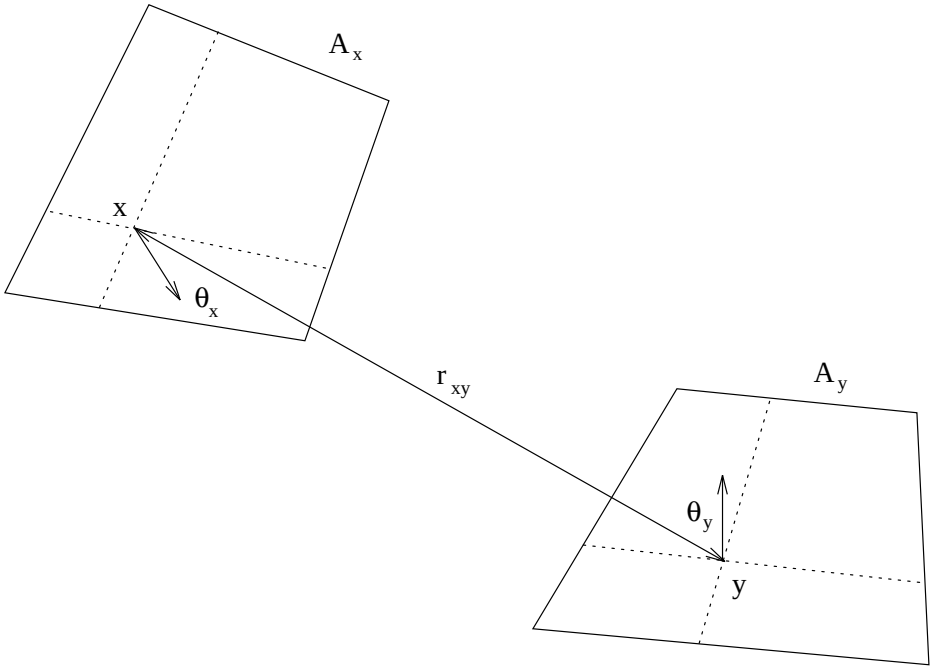


Figure 1: Geometry for the transport between two surfaces. θ denotes the angles that the vector connecting two points on the surfaces (x and y) makes with the local surface normals.

equation which can be solved efficiently with a Galerkin method. This is followed by a detailed description of the quantities which need to be computed when applying a Galerkin scheme to such an equation.

2.1 The Radiosity Equation

The computation of radiosity, i.e., power per unit area $[\frac{W}{m^2}]$, on a given surface is a widely used technique in computer graphics to solve for the illumination in an environment. Radiosity is governed by an integral equation which arises from a more general integral equation known as the rendering equation [23] when one assumes that all reflection occurs isotropically. Solving the underlying integral equation exactly is not possible in general. Thus numerical approximations must be employed leading to algorithms which are generally very expensive. The fundamental reason for the high cost of numerical approximations is that all surfaces in a given scene can potentially influence all other surfaces via reflection.

Radiosity $B(y)$ is a function defined over all surfaces $\mathbf{M}^2 \subset \mathbf{R}^3$ which make up a given scene. It is governed by a Fredholm integral equation of the second kind

$$B(y) = B^e(y) + \rho(y) \int_{\mathbf{M}^2} G(x,y) B(x)\, dx, \tag{1}$$

which describes radiosity as a sum of an emitted part (light sources) and the product of irradiance, computed by the integral, multiplied with the local reflectance $\rho(y)$, i.e., the fraction of light reemitted. Irradiance accounts for radiosities originating at all other surfaces weighted by a geometry term $G(x,y) = c \cos\theta_x \cos\theta_y r_{xy}^{-d} V(x,y)$ consisting of the cosines made by the local surface normals with a vector connecting two points, a normalization constant c, the distance r_{xy} between the two points, and a visibility function whose value is in $\{0,1\}$ depending whether the line between the two surface points x and y is obscured or unobscured respectively (see Figure 1). The points x and y are functions of some parameter. For flatland radiosity the parameter domain is 1D with $d = 1$, and the normalization constant $c = 1/2$. For full 3D radiosity the domain is 2D, $d = 2$ and the normalization constant is $c = \pi^{-1}$. In all the following derivations d, c, and x and y will be defined according to their context (1D or 2D).

In the context of integral equations we refer to the function G as the *kernel* of the integral operator

$\mathcal{G}(f) = \int G(x,.)f(x)\,dx$. Using operator notation we can express the equation to be solved as

$$(I - \rho\mathcal{G})B = B^e.$$

This particular integral operator has a number of properties which will be important later on. G is singular because the r factor in its denominator becomes zero for surfaces that touch. Nonetheless $\mathcal{G}$ is a bounded operator and closed on all L^p spaces [3]. We are mostly interested in its action on L^2, i.e., the space which contains all finite energy functions. Since ρ is strictly less than one for physically realistic environments we also know that the spectral radius of $\rho\mathcal{G}$ is strictly less than one, insuring the convergence of various iterative schemes. In particular we can compute, at least formally, the solution to the operator equation by a Neumann series

$$B = (I - \rho\mathcal{G})^{-1}B^e = \sum_{i=0}^{\infty}(\rho\mathcal{G})^i B^e = B^e + (\rho\mathcal{G})B^e + (\rho\mathcal{G})^2 B^e \ldots,$$

which gives the solution as the sum of directly emitted light, light that bounced through the environment once, twice, and so forth. While not a practical prescription for computation as such, it is nonetheless a basis for a number of algorithms to compute the solution to such operator equations. In particular in our case the physical system is generally so damped (small ρ) and the falloff is so quick (r^2 in 3D) that iterative schemes need to compute only a few terms in the above series until (numerical) convergence.

The task then is to find an efficient representation of both B and $\rho\mathcal{G}$ which facilitates the computation of terms in the Neumann series. In what follows we will assume that ρ is piecewise constant so that we only have to concentrate on the realization of $\mathcal{G}$. This is an often made assumption, but it is not necessary [14]. Once again we make it to simplify our exposition.

2.2 Projections

A canonical solution technique for integral equations such as the radiosity equation (1) is the weighted residual method [12], often referred to as finite elements. Historically radiosity algorithms were derived from power balance arguments [15, 28] and only recently [21] was the traditional mathematical framework brought to bear on the radiosity problem. However, all previous algorithms can be analyzed in the framework of the weighted residual method. For example, Classical Radiosity (CR) [15, 28] can be analyzed as a finite element method using piecewise constant basis functions.

A Galerkin method is an instance of a weighted residual method in which the original operator equation is *projected* into some subspace. We then seek an approximation $\hat{B}$ of B in this subspace such that the residual

$$r(y) = \hat{B}(y) - B^e(y) - \rho(y)\int_{\mathbf{M}^2} G(x,y)\hat{B}(x)\,dx,$$

i.e., the difference between the left and right hand sides of the original integral equation with $\hat{B}$ in place of B is orthogonal to the chosen subspace. To understand the projection of our operator $\mathcal{G}$ into a subspace we first consider writing the operator with respect to a basis for the entire space.

Let $\{N_i\}_{i\in\mathbf{Z}}$ be some basis for L^2. Using this basis the radiosity function B is characterized by a set of coefficients b_i such that $B(x) = \sum_i b_i N_i(x)$. The coefficients b_i can be found by projecting the function B with respect to the dual basis $\{\tilde{N}_i\}_{i\in\mathbf{Z}}$ which is defined by the property

$$\langle N_i, \tilde{N}_j\rangle = \int N_i(x)\tilde{N}_j\,dx = \delta_{ij}.$$

Using this fact we can write

$$B(x) = \sum_i b_i N_i(x) = \sum_i \langle B, \tilde{N}_i\rangle N_i(x).$$

Since the dual basis is a basis as well—whose dual is the original (primal) basis—we can also write

$$B(x) = \sum_j \tilde{b}_j \tilde{N}_j(x) = \sum_j \langle B, N_j\rangle \tilde{N}_j(x).$$

From the study of linear operators we know that a linear operator is fully specified if only we know its action on a basis set. In our case the resulting vectors are $\{\mathcal{G}(N_j)\}_{j\in\mathbf{Z}}$. These vectors, living in our space, are subject to being described with respect to our basis as well, leading us to consider

$$G_{ij} = \langle \mathcal{G}(N_j), \tilde{N}_i \rangle.$$

Arranging these coefficients in a tableaux we arrive at an infinite sized matrix equation which represents the original integral operator

$$\forall i: \; b_i = b_i^e + \rho_i \sum_j G_{ij} b_j. \tag{2}$$

The coefficients of this system are integrals of the form

$$G_{ij} = \int\int G(x,y) N_j(x) \tilde{N}_i(y)\, dx\, dy. \tag{3}$$

These coefficients are often called couplings or interactions to remind us that they have the physical interpretation of measuring how much one basis function physically interacts or couples with another across the integral operator. Note that the well known form factors of CR arise as $F_{ij} = G_{ij}$ when $\tilde{N}_i = \chi_{A_i}/A_i$ and $N_j = \chi_{A_j}$ ($\chi_A(x)$ is the function which takes on the value 1 for $x \in A$ and 0 otherwise).

In practice we have to deal with finite sized matrices. This corresponds to ignoring all but a finite sub-square of the infinite matrix, or said differently, the use of a finite basis. Doing this we in effect fix some subspace $V \subset L^2$ spanned by the corresponding finite basis. There are many choices for V (and its associated basis). For example one choice is the space of functions piecewise constant over fixed intervals, and one basis for that space is the set of "box" functions. Other examples are spaces spanned by "hat" functions or B-splines of higher orders. It is important to remember the difference between a choice of subspace and a choice of basis for this subspace. Once we make a choice of subspace, e.g., all functions which are piecewise linear, we still have considerable freedom in choosing a basis for this space. In particular we will consider wavelet bases.

When choosing a *finite* primal basis $\{N_j\}_{j=1,\ldots,n}$ and associated dual basis $\{\tilde{N}_i\}_{i=1,\ldots,n}$ we need to be careful as to the spaces specified by these. The subspace span$\{N_j\}$ is not necessarily the same as the space span$\{\tilde{N}_i\}$. If they are the same we say that $\{N_j\}$ and $\{\tilde{N}_i\}$ are *semi-orthogonal* and in particular they are *orthogonal* if $N_j = \tilde{N}_j$. In either of these cases we still have a Galerkin technique. The more general case arises when we consider biorthogonal bases $\{N_j\}$ and $\{\tilde{N}_i\}$ in which case we have a Petrov-Galerkin method. In what follows we will quietly ignore this distinction and collectively refer to the resulting algorithms as Galerkin methods.

Once we have chosen finite subsets $\{N_j\}_{j=1,\ldots,n}$ and $\{\tilde{N}_i\}_{i=1,\ldots,n}$ of our basis we have in effect restricted the integral equation to a subspace. To analyze these restrictions further we define the projection operator for this space by

$$\hat{B} = P_V B = \sum_{i=1}^{n} \langle B, \tilde{N}_i \rangle N_i.$$

Since the span of the primal basis is not necessarily the same as the span of the dual basis, we have $P_V \neq P_{\tilde{V}}$.

Limiting the original integral equation to this subspace we arrive at

$$(I - \rho P_V \mathcal{G} P_V) B = P_V B^e,$$

which is now characterized by a finite linear system $(G_{ij})_{i,j=1,\ldots,n}$. In this way we have reduced our task to one of solving a finite linear system in order to find coefficients b_i for a function which is an approximation to the actual solution. The quality of this approximation depends on the approximation properties of the space V. Generally these spaces contain piecewise polynomial functions up to some order $M-1$. In this case it is easy to see that the error in our computed answer $|\hat{B} - B|$ can be[1] $O(h^M)$, where h is the sidelength of some discretization imposed on the original geometry. There are other sources of error due to imprecise geometry or boundary conditions for example, which we will not consider here (for a careful analysis of these see Arvo *et al.*[3]).

[1] If the numerical techniques employed properly account for the singularity in the kernel function.

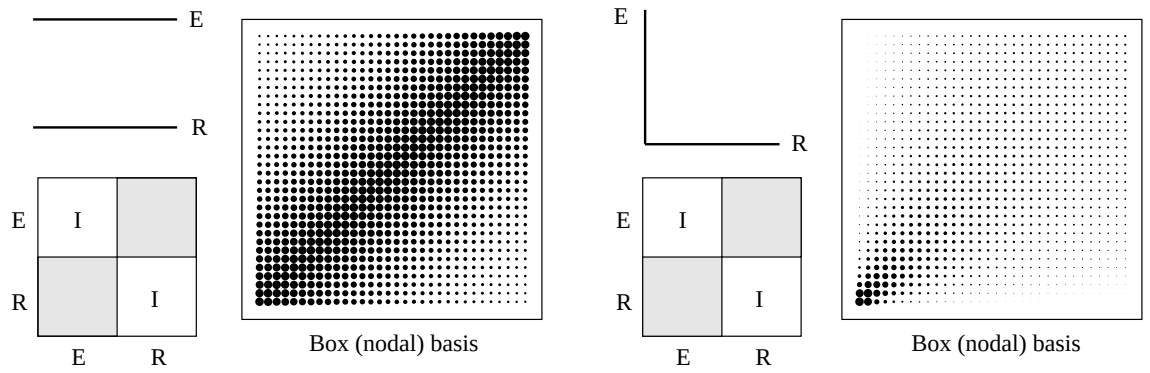


Figure 2: Two simple environments in flatland, two parallel line segments (left), and two perpendicular line segments (right), and the resulting matrix of couplings using piecewise constant functions. (Adapted from [31].)

Since wavelets can be used as bases for function spaces it makes sense to consider them in the context of a Galerkin method to solve an integral equation. In the next section we turn to a detailed study of $P_V \mathcal{G} P_V$ and the associated coefficients G_{ij} in the case that the space V is some space V_j in a multiresolution analysis and the basis set $\{N_i\}_{i=1,\ldots,n}$ is a wavelet basis.

3 Linear Operators in Wavelet Bases

In the previous section we showed why the object $P_V \mathcal{G} P_V$ is central to our study. This projected version of the integral operator $\mathcal{G}$ has some special properties which wavelets can exploit to yield efficient algorithms.

Consider CR which uses piecewise constant functions at some (finest) level V_L of meshing. Two examples of the resulting matrices $(G_{ij})_{i,j=1,\ldots,32}$ are illustrated in Figure 2. The figure shows two flatland radiosity scenarios. On the left is the flatland environment of two parallel line segments (upper left hand corner; denoted E and R). The resulting matrix of $(I - \rho\mathcal{G})$ has a block diagonal form. The diagonal blocks are identity matrices while one of the off diagonal blocks is shown enlarged. The size of dots is proportional to the magnitude of the coupling coefficient G_{ij}. Similarly on the right we see the resulting matrix for an environment with two line segments meeting in a corner, for which the domain contains the singularity. Notice how smooth and coherent the resulting matrices are. This is due to the smoothness of the kernel function itself. Suppose now we treat these matrices as pictures and apply a lossy wavelet compression to them. We can expect a high compression ratio while maintaining a good representation of the matrix, i.e., incurring only a small error in our computed answer. This is the essence of the use of wavelet bases for integral operators with smooth kernels (such as radiosity).

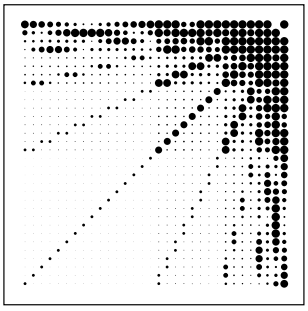
To understand the meaning of a lossy compression scheme in the context of linear algebra computations we need to connect the wavelet transform of a picture (matrix) to a vector space basis change. Since the Galerkin method uses projections we define projection operators for a multiresolution hierarchy. For the space V_i we define

$$P_i = \sum_{k=0}^{2^i-1} \langle\, . \,, \tilde{\phi}_{i,k} \rangle \, \phi_{i,k},$$

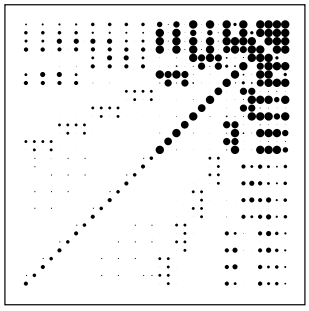
while the wavelet spaces W_i have projection operators

$$Q_i = P_{i+1} - P_i = \sum_{k=0}^{2^i-1} \langle\, . \,, \tilde{\psi}_{i,k} \rangle \, \psi_{i,k}.$$

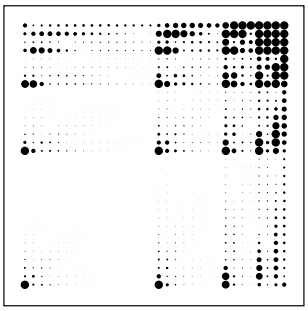
Armed with these we describe—in the context of linear algebra—the first version of a wavelet transform, which is known as the *standard basis*.



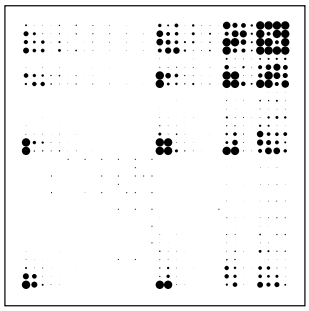
Haar basis (1 vanishing moment)



Flatlet basis (2 vanishing moments)



Haar basis (1 vanishing moment)



Flatlet basis (2 vanishing moments)

Figure 3: Coupling matrices for two flatland environments (see Figure 2) expressed in wavelet bases. The top row shows the coupling matrix for two parallel line segments expressed in the Haar basis (top left) and in the $\mathcal{F}_2$ (Flatlet) basis [17] (top right), which has 2 vanishing moments but remains piecewise constant at the finest level. The bottom row shows the same bases applied to the form factor matrix for two perpendicular lines segments. (Adapted from [31].)

3.1 Standard Basis

As we saw earlier there are alternative ways of writing some finest resolution space V_L using wavelets. Writing $V_L = V_0 + \sum_{i=0}^{L-1} W_i$ corresponds to writing the projection operator as $P_L = P_0 + \sum_{i=0}^{L-1} Q_i$. Using this identity we have

$$\begin{aligned} P_L \mathcal{G} P_L &= (P_0 + \sum_{i=0}^{L-1} Q_i)\mathcal{G}(P_0 + \sum_{i=0}^{L-1} Q_i) \\ &= P_0\mathcal{G}P_0 + \sum_{i=0}^{L-1} P_0\mathcal{G}Q_i + \sum_{i=0}^{L-1} Q_i\mathcal{G}P_0 + \sum_{i,l=0}^{L-1} Q_i\mathcal{G}Q_l. \end{aligned}$$

This decomposition corresponds to a particular two dimensional basis construction. Given a one dimensional wavelet basis $\{\phi_0, \psi_{i,k}\}$, $i = 0, \ldots, L-1$, $k = 0, \ldots, 2^i - 1$ we can build a two dimensional basis via a tensor product construction $\{\tilde{\phi}_0, \tilde{\psi}_{i,k}\} \times \{\phi_0, \psi_{l,m}\}$, $i, l = 0, \ldots, L-1$, $k = 0, \ldots, 2^i - 1$, and $m = 0, \ldots, 2^l - 1$. This is often referred to as the standard realization of the integral operator [6].

The pyramid algorithms that were mentioned earlier for transforming a function of a single variable between a basis of V_L and the bases in $V_0 + \sum_{i=0}^{L-1} W_i$ can be applied to matrices (functions of two variables). In particular the standard decomposition corresponds to applying such a pyramid transform to all rows (transforming the right hand side P_L) followed by a transform of all row transformed columns. This transformation of the coefficients corresponds exactly to a change of basis as is often done with matrices for various reasons. The remarkable property of the change to the wavelet basis is that it can be performed in time proportional to the number of basis functions, $O(n^2)$. In general expressing a matrix of size $O(n^2)$ with respect to another basis entails a transform of cost $O(n^3)$.

Figure 3 shows the effects of transforming form factor matrices expressed originally in the piecewise constant nodal basis (see Figure 2) into different wavelet bases. On the left the Haar basis was used, while on the right the Flatlet basis with two *vanishing moments* [17] was used. The top row gives matrices for the example of two parallel line segments, while the bottom row shows the case of two perpendicular line segments. Notice how many of the coefficients are small in magnitude (small disks). As the number of vanishing moments increases from one to two (left to right) we can observe many more entries becoming small. This demonstrates for two particular cases how more vanishing moments lead to more (approximate) sparsity in the matrices. In the next section we will explain why vanishing moments are so important for the compression (sparsification) of matrices which arise from smooth integral operators.

3.2 Vanishing Moments

We begin with the definition of vanishing moments. A function ψ is said to have M vanishing moments if its projection against the first M monomials vanishes

$$\langle \psi, x^i \rangle = 0 \;\; i = 0, \ldots, M-1.$$

The Haar wavelet for example has 1 vanishing moment. Other wavelets can be constructed to have more vanishing moments.

To see why this leads to small coefficients in general consider some function $f \in L^2$. Suppose we want to write it with respect to a wavelet basis. The coefficients of such an expansion can be found by taking inner products against the dual basis functions

$$f(x) = \sum_{i,j} \langle f, \tilde{\psi}_{i,j} \rangle \psi_{i,j}.$$

We want to show that for smooth f many of the coefficients $f_{i,j} = \langle f, \tilde{\psi}_{i,j} \rangle$ are small. If f is smooth we can apply Taylor's theorem to expand it about some point x_0 (for simplicity, let $x_0 = 0$) to get

$$f(x) = \sum_{i=0}^{M-1} \frac{f^{(i)}(0)}{i!} x^i + \frac{f^{(M)}(\xi)}{M!} x^M,$$

for some $\xi \in [0, x]$. Now consider computing $f_{i,j}$. To simplify the argument we consider the inner product necessary to compute $f_{0,0}$, i.e., the inner product with $\tilde{\psi}$ (all others being related by translations and scalings). Suppose that the dual basis functions have vanishing moments, then we can bound the resulting coefficient as follows

$$\begin{aligned}
|f_{0,0}| &= \left| \int f(x) \tilde{\psi}(x)\, dx \right| \\
&= \left| \int \frac{f^{(M)}(\xi)}{M!} x^M \tilde{\psi}(x)\, dx \right| \\
&\leq \left| \frac{f^{(M)}(\xi)}{M!} \right| I_x^M \int |\tilde{\psi}(x)|\, dx \\
&= C_M \sup_{\xi \in I_x} \left| f^{(M)}(\xi) \right| I_x^M, \qquad (4)
\end{aligned}$$

where I_x is the size of the interval of support of $\tilde{\psi}$. From this bound we can see that the associated coefficient will be small whenever either I_x is small or the M^{th} derivative of f is small. Similar arguments can be made for functions of more than one variable, for example the kernel function of an integral operator.

This bound allows us to argue that many of the entries in a matrix system arising from an integral operator will be very small and can be ignored, leading to a sparse matrix system. Recall that integral operators led to linear systems whose coefficients are integrals of the kernel function against the chosen basis functions (primal as well as dual). In the case of radiosity this led to the G_{ij} (Equation 3). Suppose that the basis functions for the integral operator are chosen to be wavelets and that these wavelets (both primal and dual) have vanishing moments. If G is smooth then many of the G_{ij} will be quite small because of the vanishing moment property, and can be ignored without incurring too much error. Below we will make this argument mathematically precise.

3.3 Integral Operators and Sparsity

In a paper published in 1991 Beylkin *et al.*[6] showed that for a large class of integral operators the resulting linear system, when using wavelet bases, is approximately sparse. More specifically they showed that for a class of integral operators satisfying certain kernel estimates and any $\epsilon > 0$ a $\delta(\epsilon)$ exists such that all but $O(n \log n)$ of the matrix entries will be below δ and can be ignored without incurring more than an error of ϵ in the computed answer.

The requirements on the kernel of the integral operator are given as estimates of "falloff away from the diagonal"

$$\begin{aligned} |K(x,y)| &\leq \frac{1}{|x-y|^d} \\ |\partial_x^M K| + |\partial_y^M K| &\leq \frac{C_M}{|x-y|^{d+M}}, \end{aligned} \tag{5}$$

for some $M > 0$, and $K : \mathbf{R}^d \times \mathbf{R}^d \to \mathbf{R}$ the kernel function of the integral operator in question. Note that the kernel G of the radiosity integral operator satisfies a falloff property of this type if we replace $|x-y|$ with r. Since the parameterizations which we use for our surfaces are well behaved (bounded derivative) this distinction from the classical case does not matter. Examining the matrices in Figure 3 we can immediately see the $O(n \log n)$ structure. There are approximately $\log n$ bands visible, each of length approximately equal to n. This is particularly noticeable in the case of two parallel lines and the Haar basis (upper left in Figure 3). We will not give a proof here, but give a geometric argument for the case of radiosity later on. The geometric argument is equivalent to the mathematical proof (for the radiosity operator), but provides more intuition.

Beylkin *et al.*[6] proceeded to analyze the $\log n$ dependence in the number of non-negligible entries in the matrix and showed that by decoupling all the scales it is possible to reduce the number of needed entries to $O(n)$ (for certain classes of operators). It is interesting to note that the original Hierarchical Radiosity (HR) algorithm [20] (see below) already gave a proof of the $O(n)$ complexity based purely on geometric arguments using a construction which does decouple the scales in a way very close to the Beylkin *et al.* argument. This so called *non-standard construction* is also the basis of later wavelet radiosity work [17, 31]. We will describe this construction next.

3.4 Non-Standard Basis

We saw earlier how the decomposition $P_L = P_0 + \sum_{i=0}^{L-1} Q_i$ applied to $P_L \mathcal{G} P_L$ on both sides resulted in a realization of $\mathcal{G}$ in the wavelet basis. The resulting sum consisted of terms involving all possible combinations of subspaces $\{P_0, Q_i\}_{i=0,\ldots,L-1}$ on either side of $\mathcal{G}$. Said differently, the operator was expressed as a sum of contributions between subspaces at *all* resolutions. To remove this coupling across levels we use a telescoping sum argument to write

$$\begin{aligned} P_L \mathcal{G} P_L &= P_0 \mathcal{G} P_0 + \sum_{i=0}^{L-1} (P_{i+1} \mathcal{G} P_{i+1} - P_i \mathcal{G} P_i) \\ &= P_0 \mathcal{G} P_0 + \sum_{i=0}^{L-1} Q_i \mathcal{G} P_i + \sum_{i=0}^{L-1} P_i \mathcal{G} Q_i + \sum_{i=0}^{L-1} Q_i \mathcal{G} Q_i, \end{aligned}$$

using the fact that $P_{i+1} = P_i + Q_i$ and rewriting each summand in turn as

$$\begin{aligned} P_{i+1} \mathcal{G} P_{i+1} - P_i \mathcal{G} P_i &= (P_i + Q_i) \mathcal{G} (P_i + Q_i) - P_i \mathcal{G} P_i \\ &= P_i \mathcal{G} Q_i + Q_i \mathcal{G} P_i + Q_i \mathcal{G} Q_i. \end{aligned}$$

The main difference to the earlier decomposition is the fact that the subspaces occurring on either side of $\mathcal{G}$ in the final sums all have the same index, i.e., only spaces at the *same* level interact. This is referred to as the *non-standard realization*, since it corresponds to a realization of the operator in a basis which requires an over representation for the functions to which the operator is applied. The over representation occurs because for each i both P_i and Q_i occur on either side of $\mathcal{G}$. However, the total number of functions that

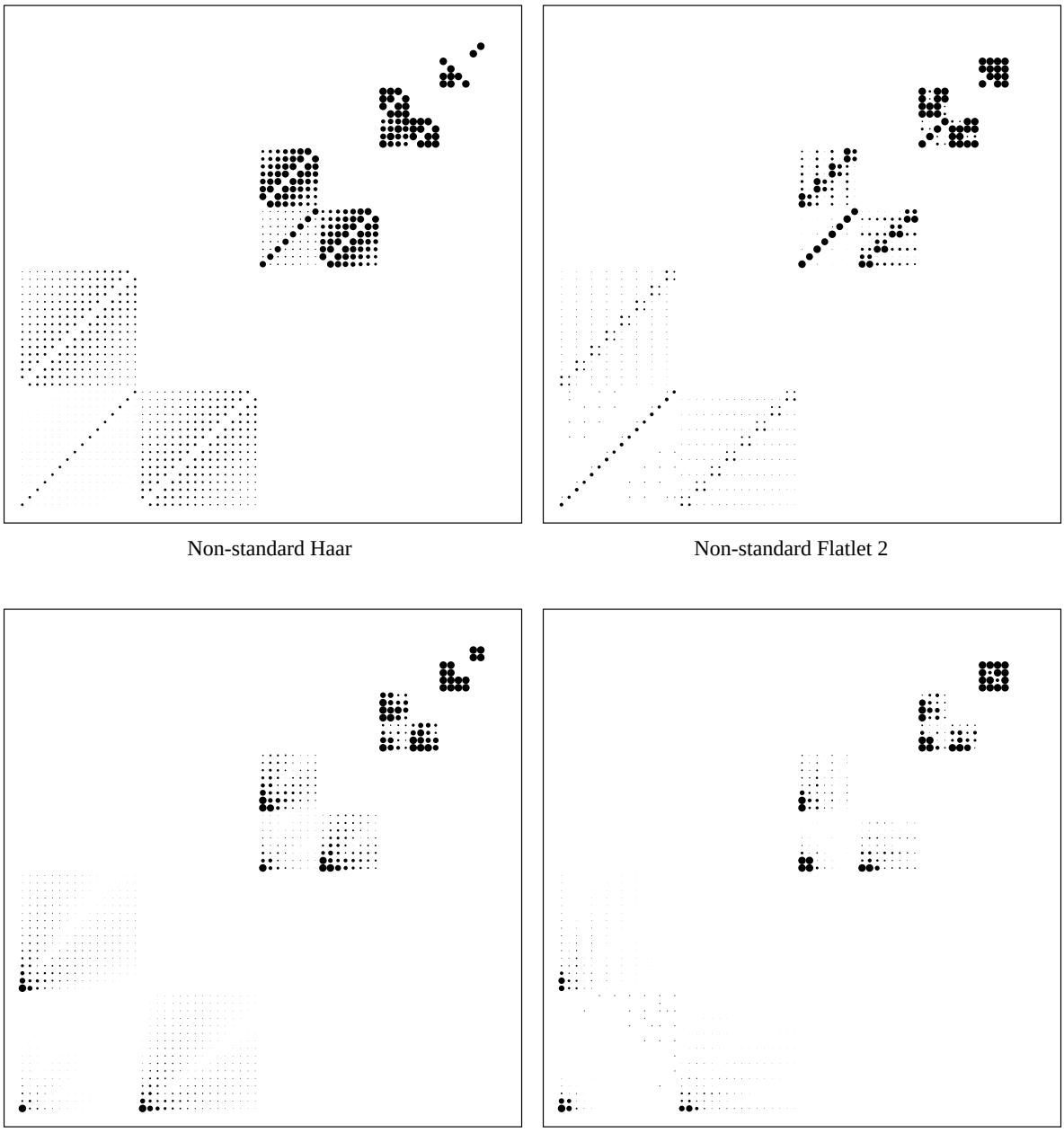


Figure 4: Coupling matrices for two flatland environments (see Figure 2) expressed in wavelet bases using the non-standard operator realization. The top row shows the coupling matrix for two parallel line segments expressed in the Haar basis (top left) and in the $\mathcal{F}_2$ basis [17] (top right). The bottom row shows the same bases applied to the coupling matrix for two perpendicular line segments. (Adapted from [31].)

occur is still only n^2, but they cannot be written as a cross product of one dimensional bases. This set of functions, $\{\tilde{\phi}_0\phi_0, \tilde{\phi}_{i,m}\psi_{i,j}, \tilde{\psi}_{i,m}\phi_{i,j}, \tilde{\psi}_{i,m}\psi_{i,j}\}$, $i = 0, \ldots, L-1$, and $j, m = 0, \ldots, 2^i - 1$, is also referred to as the *non-standard basis*.

Figure 4 shows the non-standard realizations of the operators for the two flatland environments considered earlier (Figure 2). Each level consists of three blocks. The sets of triples consist of the $Q_i\mathcal{G}Q_i$ block in the lower left, the $P_i\mathcal{G}Q_i$ block in the upper left and the $Q_i\mathcal{G}P_i$ block in the lower right. The empty quadrant would have corresponded to $P_i\mathcal{G}P_i$, however this is the block that the recursion (telescoping sum) occurs on. This last observation also suggests how to transform a matrix from the nodal basis into the non-

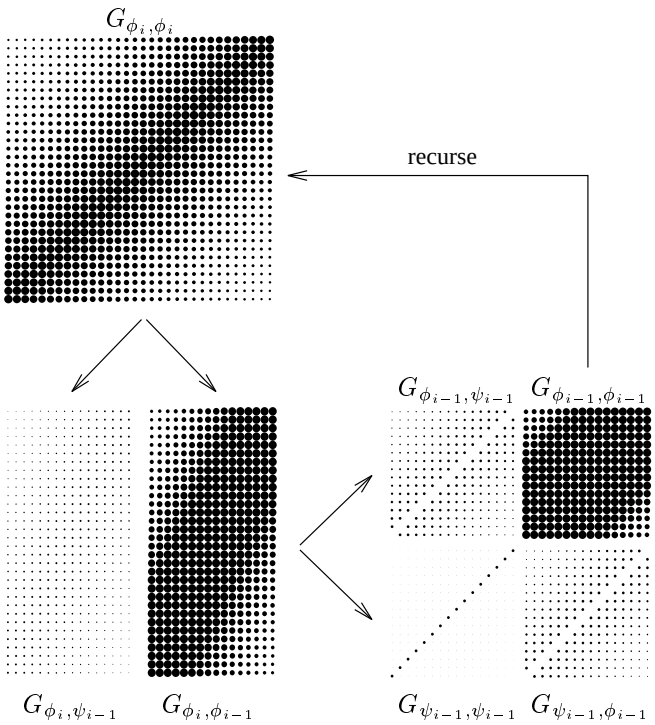


Figure 5: The 2D non-standard Pyramid Algorithm is applied to coupling coefficients taken from the flatland radiosity environment consisting of two parallel line segments. One step in the transform is shown. (Adapted from [17].)

standard realization. Instead of performing complete pyramid transforms on each row, followed by complete transforms on each column, the non-standard realization can be achieved by interleaving the individual transforms. First all rows are split into high pass and low pass bands (a single level application of the two scale relation), then all columns are subjected to a single level transform. Now recurse on the low pass/low pass quadrant $P_i\mathcal{G}P_i$ (see Figure 5). When writing this out as a matrix suitable for matrix/vector multiplies the matrices in Figure 4 result.

4 Wavelet Radiosity

Wavelet Radiosity (WR) was first introduced by Gortler *et al.*[17] and Schröder *et al.*[31]. Their algorithm unifies the benefits of higher order Galerkin Radiosity (GR) [21, 22, 41, 40] and HR [20]. HR was the first method to fully exploit a multilevel hierarchy to gain an asymptotic improvement in the efficiency of radiosity computations. It also corresponds directly to the use of a Haar wavelet basis for radiosity.

In the next section we first give a quick review of GR to motivate the desire to extend the ideas of HR to higher order basis functions. This latter extension was realized with the use of wavelets. Approaching the description of the final algorithm in this way also allows us to argue the correctness of the method with very direct geometric means.

4.1 Galerkin Radiosity

GR, first introduced by Heckbert [21, 22] aims to increase the order of basis functions used in radiosity algorithms. In this context CR [15, 28] is seen to be a Galerkin method using piecewise constant functions. The original goal of applying higher order Galerkin methods to radiosity was to improve the quality of the answers computed, as well as the efficiency of the computations. In particular using higher order basis functions allows the use of much coarser meshes than CR required while still meeting a requested error bound. In his original work Heckbert applied these ideas in a flatland environment using piecewise

linear basis functions. More recently Troutman and Max [40] and Zatz [41] have applied higher order basis functions to the computation of 3D radiosity. Zatz in particular has pushed the ideas to their extreme by leaving many surfaces unmeshed. Instead he increased the polynomial order of the basis functions so that the radiosity even over large surfaces, such as entire walls, could be computed with high accuracy without any subdivision.

4.2 Hierarchical Radiosity

The first use of hierarchies was made by Cohen *et al.*[10] who introduced a two level hierarchy known as sub-structuring. They observed that a fine subdivision was only necessary on the receiver of a transport of light, while a coarser subdivision was sufficient on the source. Since the roles of receivers and sources are reversible a two level hierarchy over each geometric primitive resulted. These ideas were developed further in a paper by Hanrahan *et al.*[20]. They introduced HR, which applied some arguments from the n-body literature [2, 19, 5] to CR. In their approach a possibly very deeply nested subdivision hierarchy was imposed on every primitive. Light transport was allowed to occur throughout these hierarchies. They showed that to within some user selectable error tolerance a linear number of interactions amongst all possible interactions was sufficient to compute an accurate answer. Because the algorithms up to that point always used a quadratic number of interactions HR improved the performance of radiosity computations considerably.

4.2.1 A Note on Performance Analyses

To put these two techniques and their respective advantages into perspective we need to look at their costs. Given k input surfaces, say polygons[2], any one of the above algorithms will use some number of basis functions n defined over the totality of input surfaces. For example in the case of CR the surfaces are typically subdivided into many elements with each element carrying an associated constant basis function (whose support is exactly the element itself). In this case n elements correspond to n basis functions. Similarly for higher order Galerkin methods we will probably do some meshing into elements as well, albeit not as fine a mesh. Each resulting element will then typically carry some number of basis functions. For example, if we are using piecewise linear basis functions each surface (2D) element will typically have four basis functions associated with it. For each parameter axis we need two basis functions (constant and linear) and we have two parameter axes for a total of four combinations. In general an $M-1$ order piecewise polynomial basis will have M^2 basis functions defined over each (2D) element. Counting in this manner it makes sense to talk about n basis functions in total for n/M^2 elements.

Once we have a set of n basis functions the Galerkin method will give rise to a linear system relating all of these basis functions with each other resulting in a system of size $O(n^2)$ (see Equation 2). This linear system needs to be solved to find the coefficients of all the basis functions. Using some iterative solver the solution cost is proportional to $O(n^2)$. Our linear systems are very well behaved due to the r^{-d} falloff in the kernel of the operator. As a result, iterative schemes typically converge within very few iterations.

GR, by going to higher order bases, manages to decrease n and thus get efficiency gains. Even though the number of bases per element increases (M^2) the number of elements necessary for a given overall accuracy falls faster for a net gain. To see why this is, we use the fact that a Galerkin method using a piecewise polynomial basis of order $M-1$ will have an accuracy of $O(h^M)$[3]. Where h gives the sidelength of the elements in the mesh [12, 24]. To make this concrete, suppose we are willing to allow an error proportional to $1/256$. Using piecewise constant basis functions, h would have to be on the order of $1/256$ to meet our goal. Now consider piecewise linear functions. In this case h would only need to be on the order of $\sqrt{1/256} = 1/16$. So even though the number of basis functions per element goes up, we still come out ahead. In the case of flatland there are two linear basis functions per element and we go from $n = 256$ to $n = 2 \cdot 16$ bases total. In 3D radiosity where we have $2 \cdot 2$ linear basis functions per element n goes from 256^2 down to $(2 \cdot 16)^2$ basis functions overall.

We have seen that for n basis functions we have $O(n^2)$ interactions in general. It is also immediately

[2] To simplify our exposition we will stick to polygons, in particular quadrilaterals. However, there is no fundamental mathematical limitation preventing us from using more general parametric surfaces such as bicubic or triangular patches, for example.

[3] This assumes that the singularity in the kernel function is treated correctly. If this is not done the method will have much worse behavior.

clear on an intuitive level that not all interactions are equally important. HR makes this statement precise and takes advantage of it to reduce the number of interactions, which need to be computed, to $O(n)$. For example, "far" interactions do not need as much subdivision as "close" interactions. The exact argument as to why $O(n)$ elements are enough will be given below. However, even if we can make statements about the number of elements generated during meshing, and how they will interact, we still need to consider at least one interaction between each pair of the original set of incoming surfaces. Consequently the work of an HR algorithm will be $O(k^2 + n)$. Even though there still is a k^2 dependence we will often have $n >> k$ resulting in significant savings. Note that in a case in which the original set of k surfaces is presented premeshed as n elements HR will be reduced to CR. Thus it will perform no worse, but in practice often dramatically better, than CR. We will take up the issue of the k^2 dependence in the last section when we consider clustering.

4.3 Algorithms

All radiosity algorithms have roughly two components for purposes of this discussion. These can be described as setting up the equations, i.e., *computing* the entries of the linear system, and *solving* the linear system. The latter typically invokes some iterative solution scheme, for example Jacobi or Gauss Seidel iteration [38], or Southwell relaxation [18]. In actual implementations these two phases are often intermixed, for example when refining a subdivision mesh (adding basis functions) during successive iterations. Nonetheless we can distinguish these two fundamental operations in our algorithms. Since iterating, i.e., performing row updates, or matrix/vector multiplies is conceptually straightforward we will first focus on the aspect of setting up the equations.

The simplest version of a wavelet radiosity algorithm would compute the initial matrix of coupling coefficients at some finest level V_L (see Figure 2), followed by the transformation of this matrix into the non-standard form (see Figure 4). Eliminating all entries less than some threshold would leave us with a sparse system for which $O(n)$ solution techniques exist. The major disadvantage of this algorithm is the cost of setting up the initial set of equations. Computing the full matrix to begin with consumes $O(n^2)$ time. Recall that our eventual goal is an $O(n)$ algorithm. The only way to achieve this goal is to compute only the entries in the transformed matrix which will be larger than the allowed threshold. The difficulty is that it is not a-priori clear where these entries are.

The HR algorithm addressed this concern in an elegant way which we now turn to. Studying this example gives us a somewhat unusual approach to the non-standard wavelet basis, since only scaling functions appear in the formulation of HR. The advantage of this approach is that it has a clear and provably correct way to enumerate just those interactions which are above the threshold. In the process it provides a constructive, geometric proof for the $O(n)$ claims of general wavelet methods for certain integral operators. In a later section we will relate the HR construction back to the more general theory, but first we give a detailed exposition of HR.

4.3.1 Hierarchical Radiosity

HR considers the possible set of interactions in a recursive enumeration scheme. We want to insure that every transport, i.e., every surface interacting with other surfaces, is accounted for once and only once. Physically speaking we want to neither miss power, nor introduce it into the simulation multiple times. To do this we call the following procedure for every input surface with every other input surface as a second argument (once again we consider the problem over 1D domains)

```
ProjectKernel( Element i, Element j )
    error = Oracle( i, j );
    if( Acceptable( error ) || RecursionLimit( i, j ) )
        G_ij = Quadrature( i, j );
    else
        if( PreferredSubdivision( i, j ) == i )
            ProjectKernel( LeftChild( i ), j );
            ProjectKernel( RightChild( i ), j );
        else
            ProjectKernel( i, LeftChild( j ) );
```

```
ProjectKernel( i, RightChild( j ) );
```

This procedure consists of several parts which we discuss in turn.

First we call a function `Oracle`, which is capable of estimating the error across a proposed interaction between elements `i` and `j`. If this estimated error satisfies the predicate `Acceptable`, the required coefficient is created by calling a quadrature routine which evaluates the integral of Equation 3. We have in effect created an entry in the matrix system, as well as implicitly decided on a particular basis function. Even if the error is not acceptable yet, resource limitations may require us to terminate the recursion. This predicate is evaluated by `RecursionLimit`. For example, we may decide that we cannot afford to subdivide input elements to a size smaller than some minimum. Of course the hope is that this predicate will never be the cause of recursion termination.

If the error is too high we recurse by subdividing, i.e., by going to a finer space V_{j+1} over the particular element. Typically we will find that the benefit in terms of error reduction is not equal for the two elements in question. For example, one element might be much larger than the other and it will be more helpful to subdivide the larger one in order to reduce the overall error. This determination is made by `PreferredSubdivision` and a recursive call is initiated on the child interactions which arise from splitting one of the parent elements. For 2D elements there would typically be four recursive calls each, not two. The preferred element would be split into four children (quadrants).

As mentioned earlier, the process of iterating and subdividing is not typically separated in a real implementation. For example, we could imagine that the predicate `Acceptable` takes into account the brightness of the sender (brightness refinement [20]) and maybe the importance of the receiver (importance refinement [37]) vis-a-vis some global error threshold ϵ. The error threshold may itself become smaller upon successive iterations (multigridding [20]), creating a fast but inaccurate solution first and using it as the starting point for successive solutions with lesser error. Any of these techniques we might refer to as refinement. Thus we will typically reexamine interactions created in an earlier iteration when iterating again.

In an implementation this is easily done by keeping a list of all G_{ij} created and calling a modified version of `ProjectKernel` on these before the next iteration. If none of the parameters which influence `Acceptable` has changed, `ProjectKernel` would simply return; otherwise it would delete the interaction G_{ij} because it has too much error and replace it with a set of finer interactions. This would correspond to replacing some set of basis functions (and their interactions) with a new and finer set of basis functions (and their interactions).

From the structure of the recursion, it is clear that every transport will be accounted for once and only once. The remaining task is to show that for a strictly positive amount of allowable error[4] we will create only a linear number of interactions amongst all the (implicit in the subdivision) basis functions created. Furthermore we need to show that the function `Oracle` can be implemented in an efficient way.

4.3.2 Bounding the Error

We proceed by analyzing the function `ProjectKernel` more closely to understand how many recursive calls it will generate. Again in order to streamline the presentation we first analyze the case of radiosity defined over 1D domains (flatland radiosity). When we used the name `ProjectKernel` we already anticipated one meaning of the G_{ij} coefficients which we will now use to analyze the relationship between allowed error and number of interactions necessary.

Recall the definition of G_{ij} (Equation 3). We may interpret the G_{ij} as expansion coefficients of G as follows

$$\begin{aligned} G_{ij} &= \int\int G(x,y)\, N_j(x)\tilde{N}_i(y)\, dx\, dy \\ &= \langle\langle G, N_j\rangle, \tilde{N}_i\rangle \\ G(x,y) &\approx \hat{G}(x,y) = \sum_{i,j=1}^{n} G_{ij}\tilde{N}_j(x)N_i(y). \end{aligned}$$

[4] Imagine `Acceptable` always returns `False`. In this case the recursion would always bottom out and in fact all n bases at the finest level of meshing, as determined by `RecursionLimit` would interact, resulting in n^2 interactions.

In other words, computing some set of G_{ij} is equivalent to approximating the function $G(x, y)$ with a projected version $\hat{G}(x, y)$.

Using the fact that the radiosity integral operator is bounded and strictly less than 1 for physically realistic reflectances [3], the error in our computed solution $\hat{B}$ can be bound vis-a-vis the actual solution B as

$$|\hat{B} - B| \leq C_{\mathcal{G}} |\hat{G} - G|,$$

where the norms are between functions, and C is some constant associated with the input (geometry, reflectances, and right hand side), but independent of the basis functions used. Clearly given a user selectable $\epsilon > 0$ the error in the computed solution can be forced below ϵ by making $\hat{G}$ sufficiently close to G.

So far we only have a global statement on the error. We next need to show that we can keep the global error, in some suitable error norm, under control by making local decisions. There are many possible ways to derive such bounds. We consider only a very simple one, not necessarily the tightest. Observe that the difference between $\hat{G}$ and G is simply given by all the terms in the infinite expansion of G which are not accounted for by the finite number of terms used for $\hat{G}$. In a wavelet expansion all the coefficients in this difference have a level number associated with them. Now consider the 1-norm of these coefficients within each level and the sup norm accross levels. We would like to argue that the resulting norm will fall off as we consider more and more levels. Away from the singularity this follows easily since there even the 2-norm of the coefficients will fall off as $2^{-i(\alpha+n/2)}$ ($n = 2$ in flatland and $n = 4$ in 3D radiosity), with α the local Lipschitz exponent and i the level number. Note that this is sufficient even for discontiuities due to visibility where $\alpha = 0$. Only the behavior at the singularity actually forces us to use the 1-norm. This follows from the fact that the form factor will stay constant ($\alpha = -d$), but the throughput (1-norm), i.e., area times formfactor, will fall off exponentially with the level number at the singularity. Consequently any strategy which considers the 1-norm within each level and stops refining, i.e., going deeper, when some threshold has been undercut (the sup-norm estimate is satisfied) will be capable of insuring that the resulting error in the solution is below some desirable ϵ.

Now we already know that the simplest version (no brightness, importance, or multigridding refinements) of the function `Acceptable` is a comparison of `error` against δ.

4.3.3 Bounding the Number of Interactions

Suppose now that we stay in the framework of CR in so far that we only allow constant basis functions (as HR does [20]) and that we simply set $\hat{G} = G(x_0, y_0)$ where x_0 and y_0 are the midpoints of the respective intervals (areas) we are considering. In the language of wavelets our scaling functions are "box" functions and the associated wavelet is the Haar function. Using the fact that

$$|G(x, y)| < \frac{C}{r^d},$$

we get, over the support of two elements I_x and I_y, which do not intersect

$$\begin{aligned} |\hat{G} - G| &\leq \int_{I_y} \int_{I_x} |G(x_0, y_0) - G(x, y)| \, dx \, dy \\ &\leq C \, I^2 \left(\frac{I}{r} \right)^{d+1}, \end{aligned}$$

through an application of the mean value theorem. I denotes the length of the maximum edge of any of the involved domains (two 1D domains in flatland, four 1D domains in 3D). The bound given above is small whenever the ratio of sizes to distances is small. In particular it will fall as a second power (even faster in 3D) of the ratio of the largest edge length to the distance. From this follow two observations

1. I always needs to be less than r to get the bound below our error threshold;

2. the involved elements should be of comparable size, since nothing is gained by making one smaller but not the other.

Below we will see that this is enough to argue that for every element I_x there are a constant number of elements I_y which satisfy these criteria.

The bound given above is only useful when $r > 0$. When the two elements meet, a more careful analysis must be applied. The difficulty arises because of the r^{-d} nature of the radiosity kernel. In other words, the bound given above holds everywhere so long as we exclude an arbitrarily small region around the intersections of any of the elements. To deal with these remaining regions we need the boundedness of our original operator. For this small region around the intersection set $\hat{G} = 0$ to get

$$|\hat{G} - G| = |G| \leq C I_y F_{I_y, I_x}$$

(in 3D the factor I_y is replaced by A_y). Since the form factor $F_{I_y, I_x} \leq 1$ we can force $|\hat{G} - G|$ below any desired threshold by making I_y (A_y respectively) small enough.

Taking both bounds together we can make the distance between G and its approximation arbitrarily small by making the ratio of size to distance small or, when we are at the singularity, by simply making the size itself small. The region over which we have to employ the second bound can be made arbitrarily small, and with it the bound itself. For sake of our argument we allocate $\epsilon/2$ of our total allowed error to the regions touching the singularity and continue to consider only the case of elements which are separated. Their error must now be kept below $\epsilon/2$, for a total of the given ϵ.

Given that we have a remaining error budget of $\epsilon/2$ we need to show that for this allowable error any recursive call will create at most a constant number of calls to the function `Quadrature`. From the above error bound we see that an interaction will be created whenever the size to distance ratio is smaller than some threshold. How many elements can there be for which this is true? To answer this question we interpret the size to distance ratio geometrically as a measure of angle subtended. In other words, this ratio is proportional to the angle that one element subtends from the point of view of the other element.

On the initial call to `ProjectKernel` there can at most be k elements (the original input surfaces) less than this threshold (hence the k^2 in the overall performance analysis). Suppose that some of those initial input surfaces are too large, i.e., their angle subtended is above our threshold. These surfaces will result in recursive calls. How many can there be? Since the total angle subtended above a given element is bounded there can at most be a constant number of elements larger than some minimum on any given recursive call. Suppose that at the next recursion level, due to subdivision, all of these elements have fallen below the threshold. In this case they all interact with our element, i.e., this element interacts with a constant number of other elements. Suppose instead that not all elements have fallen below the threshold due to the subdivision. Once again, there can be at most a constant number of such "too-large" elements.

In either case each element—below the top level call to `ProjectKernel`—interacts at most with a constant number of other elements. This means that the total number of interactions created due to recursive calls is proportional to the total number of elements. The constant of proportionality is a function of the problem definition and error requested, but not of the discretization itself.

4.3.4 Oracle

From the above arguments, we have seen that the function `Oracle` can be implemented by estimating the ratio of size to distance, or in the vicinity of the singularity, simply the size itself. In the case of radiosity with constant basis functions, measuring the ratio is particularly simple since it is given by the point to finite area form factor, a quantity for whose computation many formulas are known (see for example [34] or [27]). This was the oracle used in the original HR algorithm [20]. For higher order methods a simple form factor estimate is sufficient to argue the asymptotic bounds, but does not take full advantage of the information present. There are other, more direct methods to estimate the quantity $|\hat{G} - G|$ discussed in the next section.

4.3.5 Higher Orders

Consider again the argument used above to show that HR constructs only a linear number of interactions. There was nothing particular in the argument which ties it to constant basis functions. Suppose we wish to employ a Galerkin scheme with higher order basis functions. In this case each interaction between two elements entails a number of quadratures. For constant basis functions there was simply one coefficient G_{ij}

for elements i and j. We will continue to use the indexing G_{ij}, but think of the quantity G_{ij} as consisting of an array of numbers describing all the possible coupling terms over the given elements due to higher order basis functions. For example, in the case of piecewise linear basis functions we have two basis functions along each dimension. In flatland G_{ij} now consists of $2 \cdot 2$ couplings and in 3D G_{ij} has $2^2 \cdot 2^2$ numbers associated with it. If $M-1$ is the order of basis functions used we will abstract $M \cdot M$ (flatland) and $M^2 \cdot M^2$ (3D) couplings respectively into G_{ij}.

The basic reasoning of the recursion count argument still holds. $|\hat{G} - G|$ is still the quantity which needs to be kept below some $\delta(\epsilon)$, however $\hat{G}$ is not constant anymore. The form factor argument to measure angle subtended does not take full advantage of the power of higher order basis functions. However, it is still sufficient to argue the asymptotic bound. In practice we will of course want to take advantage of the higher order nature of the basis functions. One way to do this is to have the function `Oracle` use an estimate of the G_{ij} to construct a polynomial and measure how well this polynomial interpolates the real kernel G over the support of the two elements in question. This type of oracle was employed in the case of wavelet radiosity [17, 31] and estimates the quantity $|\hat{G} - G|$ directly.

4.3.6 Iterative Solvers

As pointed out earlier there are two parts to a complete algorithm, setting up the equations, and solving them. Above we described how to set up the equations and argued why there are $O(k^2 + n)$ interactions total for any given finite accuracy requirement. To complete the algorithm we need the iteration function. This function corresponds to the matrix/vector multiply in an iterative solver. In HR this was referred to as `Gather`, a function which moves radiosity from element `j` across G_{ij} to element `i`, multiplying it with the factor G_{ij} (the form factor for constant basis functions). Once this has occurred we still need a function referred to as `PushPull` in [20].

For each input surface (element) `i`, `ProjectKernel` is called with all other input surfaces (elements) `j`. As pointed out above, the choice of interactions G_{ij} actually created corresponds to an implicit choice of basis functions. Consequently when `ProjectKernel` was called on, say `i` and $\mathtt{j}_0$, versus `i` and $\mathtt{j}_1$, different basis functions may have been constructed on `i` for those two calls. Put differently, irradiance at a surface will be computed at different levels of the hierarchy, due to different sources. These incoming irradiances need to be consolidated.

Consider the function `PushPull` as proposed in Hanrahan *et al.*[20]. Irradiance of a parent in the subdivision hierarchy is added to the children on a downward pass, while on an upward pass the radiosity at a parent is the area average of the radiosity at the children

```
PushPull( Element i )
    if( !Leaf( i ) )
        i.children.E += i.E; //Push
        ForAllChildren( i.c )
            PushPull( i.c );
        i.B = AreaAverage( i.children.B ); //Pull
    else
        i.B = i.Be + ApplyRho( i.E );
```

where we used the symbols `B` to denote radiosity, `E` to denote irradiance, and `Be` for the emitted part of radiosity.

The summing of irradiance on the way down follows immediately from the physical meaning of irradiance. The irradiance at a given element is the sum of all the irradiances received at the element itself and all its ancestor elements. The area averaging on the way up follows directly from the definition of constant radiosity, which is a density quantity per area.

How to extend this `PushPull` reasoning to the higher order hierarchical algorithm briefly outlined above is not immediately clear. This is where wavelets come in since they not only generalize the notion of higher order hierarchical basis sets, but also the attendant notions of pushing (pyramid down) and pulling (pyramid up) throughout such a hierarchy.

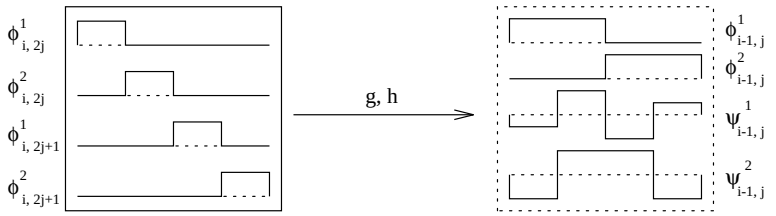


Figure 6: The $\mathcal{F}_2$ wavelet construction. $\mathcal{F}_2$ bases have two different wavelet functions. Both of them have two vanishing moments (from [17]).

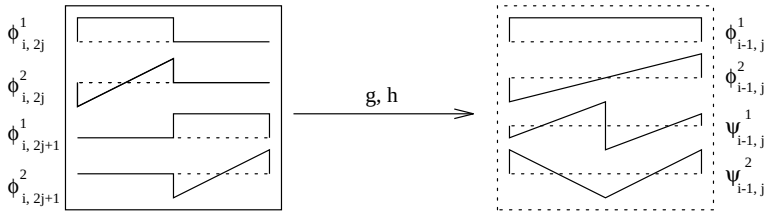


Figure 7: The $\mathcal{M}_2$ wavelet construction whose scaling functions are the first two Legendre polynomials. Both of the wavelet functions (lower right) have two vanishing moments (from [17]).

4.4 $O(n)$ Sparsity

The abstract mathematical proof of the $O(n)$ sparsity claim for certain integral operators given by Beylkin *et al.*[6] is the exact analog of the constructive geometric argument we gave above for the $O(n)$ claim of HR. The main difference is that the abstract proof argues that *all but* $O(n)$ entries in the resulting matrix system are below the threshold, while HR argues the complement: *only* $O(n)$ entries are above the threshold.

In HR we argued that for a given allowable error of ϵ we can permit some amount of error (δ) across each link and that there would only be a linear number of such links. In fact we used scaling functions (piecewise polynomial) as basis functions. Saying that there is an error of δ for one such approximation is equivalent to saying that the associated wavelet coefficient is less than δ (for sufficiently smooth kernels). Recall that the wavelet coefficient measures the difference between one level of approximation and the next finer level (recursive splitting) of approximation.

While we used an "angle subtended" argument to limit the number of coefficients thusly created the classical falloff property (Equation 5) is the abstract analog of this geometric statement. Recall the bound we gave on the coefficients of ψ for a smooth function f (Equation 4). It bounds the magnitude by interval (area) size raised to the M^{th} power multiplied with the derivative. But for integral operators we have a bound on these derivatives of the form $|x-y|^{-d-M}$. In other words the coefficients are bounded by a power of a size (interval or area) to distance ($|x-y|$) ratio. The same argument we used earlier in the context of radiosity, except this time made on purely mathematical grounds with no reference to the surrounding geometry. In this way the classical argument of Beylkin *et al.* generalizes to other integral operators an idea that is perhaps more obvious in the geometrical context of graphics.

One issue remains. The abstract theory of integral operators has us use the scaling *and* wavelet functions to construct the sparse linear system. WR [17] (or higher order hierarchical methods) only use the scaling functions.

Consider again the Haar example. Suppose we are using the Haar basis for a non-standard realization of our operator (see Figure 4 left column). If we ignore all entries in the matrix less than some threshold we will be left with some set of entries corresponding to couplings between a mixture of scaling and wavelet functions. In the Haar case we can transform this set of couplings into a set of couplings involving *only* scaling functions by exploiting the two scale relationship. Simply replace all occurrences of $\psi_{i,j}$ with $2^{-1/2}\phi_{i+1,2j} - 2^{-1/2}\phi_{i+1,2j+1}$. The remaining set of couplings involves only scaling functions.

The reason the Haar basis allowed us to do this simplification lies in the fact that the scaling functions in the Haar system do not overlap. For more general wavelets there is overlap between neighboring functions. Consequently the above substitution, while still possible [16], is not as straightforward. The problem

arises with overlapping basis functions because some regions may be accounted for multiple times, in effect introducing the same power more than once into the system. The wavelets that were used in the original WR work [17, 31] did not suffer from this problem because they were tree wavelets. In a tree wavelet the filter sequences do not overlap.

The Haar basis is a tree wavelet basis. When trying to extend these ideas to more vanishing moments we have to allow more than one wavelet (and scaling) function over a given interval to keep the filter sequences from overlapping. In essence neighboring intervals are decoupled. This is not a classical construction because there are multiple generators of the MRA. WR used so called Flatlets, which are still piecewise constant, but combine more than two box functions to increase the number of vanishing moments (Figure 6 shows the shape of Flatlets with two vanishing moments). Another set of wavelets explored for WR was introduced by Alpert [1] under the name multi-wavelets. Over each interval a set of Legendre polynomials up to some order $M-1$ is used and a wavelet hierarchy is imposed. Here too, neighboring intervals decouple giving multi-wavelets the tree property as well (see Figure 7 for a multi-wavelet with two vanishing moments). Details regarding these functions in the context of WR can be found in [17, 31].

Using tree wavelets and the substitution of all wavelet functions by sequences of scaling functions leads to an obvious simplification of the code and follows naturally from the historical development. It also results in a straightforward procedure to enumerate all "important" couplings and circumvents all issues associated with boundaries. Instead of specializing a wavelet constructed for the real line to one usable for an interval the multi-wavelets and Flatlets have the interval property right from the start.

There are other practical issues which are taken up in the original papers and the interested reader is referred to them for more details ([17, 31]). For example, in some wavelet constructions only the primal (or dual) bases have vanishing moments. Recall that the G_{ij} (Equation 3) had both primal and dual bases under the integral sign. If only one of these has vanishing moments, say the primal basis, it is desirable to use a projection into the dual basis on the left hand side of the original operator, $P_{\tilde{V}}\mathcal{G}P_V$. This was the case in [17, 31] for the so called Flatlets. Doing this requires a basis change back to the primal basis after each iteration of the operator. This is easily absorbed into the `PushPull` procedure, though.

5 Issues and Directions

In our treatment so far we have deliberately left out a number of issues arising in a real implementation for purposes of a clear exposition of the general principles. We now turn to some of these issues as well as to a discussion of extensions of the basic ideas.

5.1 Tree Wavelets

Both HR and WR used scaling functions which do not maintain continuity across subdivision boundaries. While convergence of the computed answers is assured in some weighted error norm, there is nothing in the algorithm which will guarantee continuity between adjacent elements. This has undesirable consequences for purposes of displaying the computed answers. Discontinuities in value or even derivative lead to visually objectionable artifacts (e.g., Mach bands).

These discontinuities arose from a desire to use tree wavelets. Recall that in classical wavelet constructions with more than 1 vanishing moment the supports of neighboring scaling functions overlap. In this way continuity between neighboring mesh elements up to some order (depending on the wavelet used) can be assured. Two difficulties arise if one wants to use such wavelets: (A) They need to be modified at the boundary of the original patch since overlap onto the outside of the patch is not desirable (it is not even physically meaningful); (B) sparse representations, i.e., partially refined subdivisions, are difficult to build with such wavelets. To appreciate the latter point consider the scaling function associated with the subdivision child of some element. If the neighboring element does not get subdivided, i.e., does not have children itself, the former scaling function will again overlap a "niece" element which does not exist. Tree wavelets avoid both of these issues. Since they inherently live on the interval no overlap outside the interval or over "niece" elements, which do not exist, can occur. Furthermore every wavelet can be replaced immediately by a linear combination of its constituent scaling functions, resulting in a much streamlined program which only needs to deal with scaling functions. This convenience comes at a cost of higher storage. Whenever an element is subdivided we do not just simply add a single new coefficient to the representation of the kernel,

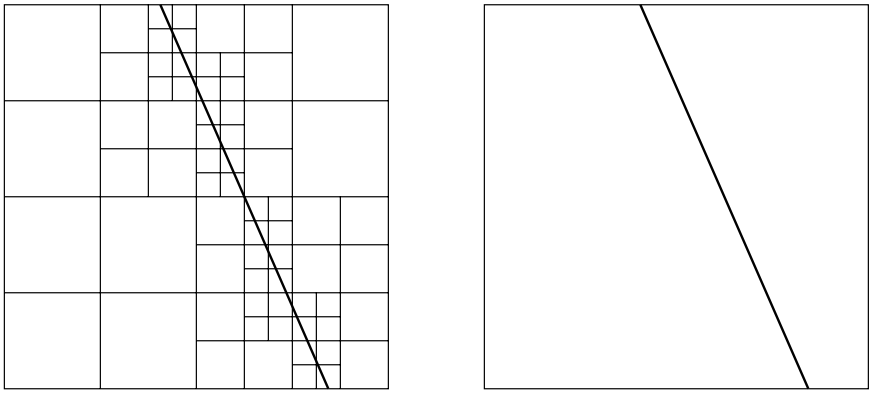


Figure 8: Subdivision around a feature line (bold diagonal line). On the left a restricted quadtree subdivision bracketing the feature line to within some grid resolution. On the right subdivision is induced immediately along the feature line, representing it perfectly and resulting in far fewer elements.

but rather a chunk. Consider 3D radiosity and wavelets with M vanishing moments. In this case every G_{ij} actually consists of $(M^2)^2$ coefficients (one for each possible combination of bases over the two elements). For cubic bases the refinement of an existing interaction into 4 child interactions results in $3 \cdot 256$ additional floating point coupling coefficients.

Preliminary experiments with classical wavelets for radiosity have recently been reported by Pattanaik and Bouatouch [29]. They used Coiflets [11] as well as interpolating scaling functions [13]. However, they ignored issues associated with the boundary of patches, the singularity, and only gave an algorithm which does uniform refinement when the error criterion is not met (resulting in an $O(n^2)$ algorithm).

Clearly more research is needed for an algorithm which uses overlapping scaling functions of higher regularity and properly addresses boundary and adaptive subdivision issues.

5.2 Visibility

The basic premise on which the sparsification arguments for integral operators rest is the smoothness of the kernel function. However, in the case of radiosity the kernel function contains a non-smooth component: the visibility function $V(x, y)$. Clearly the kernel is still piecewise smooth so the arguments certainly hold piecewise. Alternatively, the arguments can be approached with a notion of smoothness as defined in the Besov space sense. However, the complexity analysis is considerably more complicated. To our knowledge no such analysis has yet been performed. We hypothesize that the total number of coefficients will have a component which is in some sense proportional to the "length" of the discontinuity.

In practice two basic approaches have emerged to address the discontinuities in the kernel function. HR [20] and WR [31, 17] use regular quadtree subdivision of quadrilaterals. Thus they in effect resolve the resulting features in the computed radiosity function by approximating them with successively smaller rectangles (see the left side of Figure 8). Since the oracle is based on estimating how well the kernel is approximated by a low order polynomial over the support of the two elements in question, it will automatically "zoom" in on these feature boundaries. This follows trivially from the fact that the discontinuity in the kernel is not well approximated by a low order polynomial. Another approach has been put forward by Lischinski *et al.*[25]. They take the feature lines due to discontinuities in the visibility function explicitly into account with *discontinuity meshing*. Instead of using regular subdivision they introduce subdivisions along lines of discontinuities in the computed answer (see the right side of Figure 8). As a result they generate far fewer elements and discontinuity features are resolved exactly. The disadvantage of their approach lies in the considerably more complicated global visibility analysis necessary to find all such feature lines. Another difficulty arises from the fact that such an approach needs wavelets over irregularly subdivided domains. Lischinski *et al.*[25] stayed within the HR, i.e., constant basis function, framework. In this case the filter coefficients for `PushPull` are still just simple area ratios. Piecewise polynomial triangular elements could be accommodated as well in a straightforward extension of the use of multi-wavelets in [17]. The feasibility of this approach was recently examined by Bouatouch and Pattanaik [7]. Classical wavelets however, have only

recently been adapted to irregular meshes [26, 33] and they have not yet been applied to wavelet radiosity algorithms with explicit (or implicit) discontinuity meshing.

5.3 Radiance

The basic ideas behind HR and WR can also be applied to the computation of *radiance*, i.e., global illumination in the presence of reflection which is no longer uniform with respect to direction. In this case the physical quantity of interest has units $[\frac{W}{m^2 sr}]$ and the basic integral equation to solve becomes

$$L(y,z) = L^e(y,z) + \int_{\mathbf{M}^2} f_r(x,y,z)G(x,y)L(x,y)\,dx.$$

Here $L(y,z)$ is the unknown radiance function describing the flow of power from y to z, f_r is the bidirectional reflectance distribution function (BRDF), and G accounts for geometry as before (with $c = 1$). The BRDF gives the relation at y between incoming radiance from x and outgoing radiance towards z.

Aupperle and Hanrahan [4] were the first to give a hierarchical finite element algorithm for radiance computations. They extended their earlier work [20] in a straightforward manner by considering triple interactions from A_x via A_y towards A_z (as opposed to the case of radiosity with interactions from A_x towards A_y). The complexity arguments are similar to the ones we gave for the case of radiosity with the difference that the overall complexity is now $O(k^3 + n)$ since initially all triples of surfaces have to be accounted for. This work was extended to higher order multi-wavelet methods in [32].

In both of these approaches [4, 32] radiance was parameterized over pairs of surfaces. Christensen *et al.*[9] pursued a different avenue. They treated radiance as a function of a spatial and *directional* argument given by the corresponding integral equation

$$L(y,\omega_o) = L^e(y,\omega_o) + \int_{\mathbf{H}^2} f_r(\omega_i,y,\omega_o)\cos\theta_i L_i(y,\omega_i)\,d\omega_i,$$

where $L_i(y,\omega_i) = L(x,-\omega_i)$ is the incoming radiance at y from direction ω_i, which is identical to the outgoing radiance at some point x visible from y in the direction ω_i. The integration is now performed over the hemisphere of incoming directions. The chief advantage of this formulation is the fact that recursive coupling coefficient enumeration needs to consider only all *pairs* of input surfaces. As a basis they used the Haar basis for the spatial support. For the directional part of the domain they constructed a basis by parametrically mapping the Haar basis over the unit square onto the hemisphere. For a more detailed discussion of some of the differences between these two approaches the reader is referred to [32].

Computing solutions to the radiance integral equations is notoriously expensive due to the higher dimensionality of the involved quantities, 4D functions interacting across a 6D integral operator with 4D functions. Naive finite element methods are hopeless, but even hierarchical methods based on wavelets still require enormous amounts of space and time and more research is needed before these techniques become truly practical.

5.4 Clustering

In all our discussions so far we have only considered the intelligent *subdivision* of surfaces. Ironically the historical roots of HR lie in n-body algorithms [19], which are all about clustering, not subdivision. This difference moves most clearly into focus when considering the complexity analysis we gave earlier. There we argued that HR and WR have a complexity of $O(k^2 + n)$ where k is the number of input surfaces and n the number of elements they are meshed into. In order to remove the k^2 dependence the hierarchy of interactions must be extended "upward". A number of such clustering algorithms have recently appeared in the literature [30, 36, 35, 8].

The main difficulty with clustering in the context of radiosity is due to visibility. For example, the light emitted by a cluster of elements is not equal to the sum of the individual emissions. Similarly, the reflective behavior of a cluster is not uniform in all directions even though each individual reflection is uniform in the hemisphere above the respective surface.

Sillion [35] realizes clustering of surfaces by imposing an octree partition on the entire scene and treating all surfaces within one of the octree nodes as an approximate volume density. In the limit with surfaces

very small and uniform in each cube of the octree the resulting approximation is correct. The resulting datastructure can be built in time linear in the number of surfaces and the only modification to an existing HR solver is the introduction of volume elements characterized by their averaged behavior. As observed by Sillion even in the case of purely diffuse reflection the aggregate behavior of any volume is generally not diffuse (uniform in all directions). In order to account for this observation a correct system needs to be able to deal with directionally dependent quantities.

Smits *et al.*[36] give a clustering extension to HR with a complexity of $O(k \log k + n)$ by introducing higher level links between clusters of surfaces. The main task is to set up an error estimator usable by the oracle, which is conservative but tight for such links. They too deal only with isotropic approximations of clusters. Noting this deficiency Christensen *et al.*[8] give a clustering algorithm which addresses the more general radiance case. Each cluster is treated as a point source (and receiver) whose behavior is characterized as a function of direction with a small number of discretized directions. In this way the resulting algorithm is more closely related to the multipole based algorithm of Greengard [19] rather than a wavelet method.

All of the above clustering algorithms compute an approximation of the radiosity or radiance at such a coarse level that a final reconstruction step (also referred to as final gather) needs to be added to produce an acceptable looking final image. This final step is generally very expensive and better techniques are clearly desirable.

6 Conclusion

We have seen that the Galerkin method for integral equations gives rise to a linear system which needs to be solved to find an approximation to the original integral equation solution. The linear system has entries which are the coefficients of the kernel function itself with respect to some basis (standard or non-standard). As such they possess properties which derive directly from the kernel function itself. Using wavelets as basis functions the resulting matrix system is approximately sparse if the kernel function is smooth. A wide class of operators whose kernel functions satisfy certain "falling off with distance" estimates have the right properties. By ignoring all entries below some threshold the resulting linear system has only $O(n)$ remaining entries leading to fast solution algorithms for integral equations of this type. To realize an algorithm which is $O(n)$ throughout a function `Oracle` is needed to help enumerate the important entries in the matrix system.

HR was described in this context as an application of the Haar basis to the radiosity integral equation. We argued that HR needs only a linear number of interactions between elements to achieve an a-priori accuracy claim. The argument used geometric reasoning which corresponds exactly to the abstract arguments given by Beylkin *et al.*[6]. In this way we in effect gave a constructive, geometric proof of the sparsity claim for somewhat more benign operators than are treated in the general case. The development of these arguments led to a WR algorithm which has been shown to perform exceedingly well in practice under many circumstances [20, 31, 17, 14, 39].

The original method [31, 17] used tree wavelets (multi-wavelets and Flatlets) which simplify many implementation issues and are a natural extension from the historical development out of HR. As such the exploration of interesting basis functions from the wide variety of available wavelet bases has only begun and we look forward to further developments in this area.

References

[1] Alpert, B. A Class of Bases in L^2 for the Sparse Representation of Integral Operators. *SIAM Journal on Mathematical Analysis 24*, 1 (January 1993).

[2] Appel, A. An Efficient Program for Many Body Simulation. *SIAM Journal of Sci. Stat. Computing 6*, 1 (1985), 85–103.

[3] Arvo, J., Torrance, K., and Smits, B. A Framework for the Analysis of Error in Global Illumination Algorithms. In *Computer Graphics Annual Conference Series, 1994* (1994).

[4] Aupperle, L., and Hanrahan, P. A Hierarchical Illumination Algorithm for Surfaces with Glossy Reflection. In *Computer Graphics Annual Conference Series 1993* (August 1993), Siggraph, pp. 155–162.

[5] Barnes, J., and Hut, P. A hierarchical $O(n \log n)$ Force Calculation Algorithm. *Nature 324* (1986), 446–449.

[6] Beylkin, G., Coifman, R., and Rokhlin, V. Fast Wavelet Transforms and Numerical Algorithms I. *Communications on Pure and Applied Mathematics 44* (1991), 141–183.

[7] Bouatouch, K., and Pattanaik, S. N. Discontinuity Meshing and Hierarchical Multi-Wavelet Radiosity. In *Proceedings of Graphics Interface* (1995).

[8] Christensen, P. H., Lischinski, D., Stollnitz, E., and Salesin, D. Clustering for Glossy Global Illumination. TR 95-01-07, University of Washington, Department of Computer Science, February 1995. `ftp://ftp.cs.washington.edu/tr/1994/01/UW-CSE-95-01-07.PS.Z`.

[9] Christensen, P. H., Stollnitz, E. J., Salesin, D. H., and DeRose, T. D. Wavelet Radiance. In *Proceedings of the 5th Eurographics Workshop on Rendering* (Darmstadt, June 1994), pp. 287–302.

[10] Cohen, M. F., Greenberg, D. P., Immel, D. S., and Brock, P. J. An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications 6*, 3 (March 1986), 26–35.

[11] Daubechies, I. Orthonormal bases of compactly supported wavelets II: Variations on a theme. *SIAM J. Math. Anal. 24*, 2 (1993), 499–519.

[12] Delves, L. M., and Mohamed, J. L. *Computational Methods for Integral Equations.* Cambridge University Press, 1985.

[13] Donoho, D. L. Interpolating wavelet transforms. Preprint, Department of Statistics, Stanford University, 1992.

[14] Gershbein, R. S., Schröder, P., and Hanrahan, P. Textures and Radiosity: Controlling Emission and Reflection with Texture Maps. In *Computer Graphics Annual Conference Series, 1994* (1994).

[15] Goral, C. M., Torrance, K. E., Greenberg, D. P., and Battaile, B. Modelling the Interaction of Light between Diffuse Surfaces. *Computer Graphics 18*, 3 (July 1984), 212–222.

[16] Gortler, S. personal communication, 1993.

[17] Gortler, S., Schröder, P., Cohen, M., and Hanrahan, P. Wavelet Radiosity. In *Computer Graphics Annual Conference Series 1993* (August 1993), Siggraph, pp. 221–230.

[18] Gortler, S. J., Cohen, M. F., and Slusallek, P. Radiosity and Relaxation Methods; Progressive Refinement is Southwell Relaxation. Tech. Rep. CS-TR-408-93, Department of Computer Science, Princeton University, February 1993. To appear in IEEE Computer Graphics and Applications.

[19] Greengard, L. *The Rapid Evaluation of Potential Fields in Particle Systems.* MIT Press, 1988.

[20] Hanrahan, P., Salzman, D., and Aupperle, L. A Rapid Hierarchical Radiosity Algorithm. *Computer Graphics 25*, 4 (July 1991), 197–206.

[21] Heckbert, P. S. *Simulating Global Illumination Using Adaptive Meshing.* PhD thesis, University of California at Berkeley, January 1991.

[22] Heckbert, P. S. Radiosity in Flatland. *Computer Graphics Forum 2*, 3 (1992), 181–192.

[23] Kajiya, J. T. The Rendering Equation. *Computer Graphics 20*, 4 (1986), 143–150.

[24] Kondo, J. *Integral Equations.* Oxford Applied Mathematics and Computing Science Series. Kodansha, 1991.

[25] Lischinski, D., Tampieri, F., and Greenberg, D. P. Combining Hierarchical Radiosity and Discontinuity Meshing. In *Computer Graphics Annual Conference Series 1993* (August 1993), Siggraph, pp. 199–208.

[26] Lounsbery, M., DeRose, T. D., and Warren, J. Multiresolution Surfaces of Arbitrary Topological Type. Department of Computer Science and Engineering 93-10-05, University of Washington, October 1993. Updated version available as 93-10-05b, January, 1994.

[27] Moon, P. *The Scientific Basis for Illuminating Engineering*, dover, 1961 ed. McGraw-Hill, 1936.

[28] Nishita, T., and Nakamae, E. Continuous Tone Representation of Three-Dimensional Objects Taking Account of Shadows and Interreflection. *Computer Graphics 19*, 3 (July 1985), 23–30.

[29] Pattanaik, S. N., and Bouatouch, K. Fast Wavelet Radiosity Method. *Computer Graphics Forum 13*, 3 (September 1994), C407–C420. Proceedings of Eurographics Conference.

[30] Rushmeier, H. E., Patterson, C., and Veerasamy, A. Geometric Simplification for Indirect Illumination Calculations. *Proceedings Graphics Interface* (May 1993), 227–236.

[31] Schröder, P., Gortler, S. J., Cohen, M. F., and Hanrahan, P. Wavelet Projections For Radiosity. In *Fourth Eurographics Workshop on Rendering* (June 1993), Eurographics, pp. 105–114.

[32] Schröder, P., and Hanrahan, P. Wavelet Methods for Radiance Computations. In *Proceedings 5th Eurographics Workshop on Rendering* (June 1994), Eurographics.

[33] Schröder, P., and Sweldens, W. Spherical wavelets: Efficiently representing functions on the sphere. *Computer Graphics, (SIGGRAPH '95 Proceedings)* (1995). To appear, (`ftp://ftp.math.scarolina.edu/pub/imi_95/imi95_1.ps`).

[34] Siegel, R., and Howell, J. R. *Thermal Radiation Heat Transfer*. Hemisphere Publishing Corp., 1978.

[35] Sillion, F. Clustering and Volume Scattering for Hierarchical Radiosity Calculations. In *Proceedings of the 5th Eurographics Workshop on Rendering* (Darmstadt, June 1994), pp. 105–117.

[36] Smits, B., Arvo, J., and Greenberg, D. A Clustering Algorithm for Radiosity in Complex Environments. *Computer Graphics Annual Conference Series* (July 1994), 435–442.

[37] Smits, B. E., Arvo, J. R., and Salesin, D. H. An Importance Driven Radiosity Algorithm. *Computer Graphics 26*, 2 (August 1992), 273–282.

[38] Stoer, J., and Bulirsch, R. *Introduction to Numerical Analysis*. Springer Verlag, New York, 1980.

[39] Teller, S., Fowler, C., Funkhouser, T., and Hanrahan, P. Partitioning and Ordering Large Radiosity Computations. In *Computer Graphics Annual Conference Series 1994* (July 1994).

[40] Troutman, R., and Max, N. Radiosity Algorithms Using Higher-order Finite Elements. In *Computer Graphics Annual Conference Series 1993* (August 1993), Siggraph, pp. 209–212.

[41] Zatz, H. R. Galerkin Radiosity: A Higher-order Solution Method for Global Illumination. In *Computer Graphics Annual Conference Series 1993* (August 1993), Siggraph, pp. 213–220.