# A Meshless Hierarchical Representation for Light Transport

Jaakko Lehtinen
MIT CSAIL
TKK

Matthias Zwicker
University of California,
San Diego

Emmanuel Turquin
Grenoble University
INRIA

Janne Kontkanen
PDI/DreamWorks

Frédo Durand
MIT CSAIL

François X. Sillion
INRIA
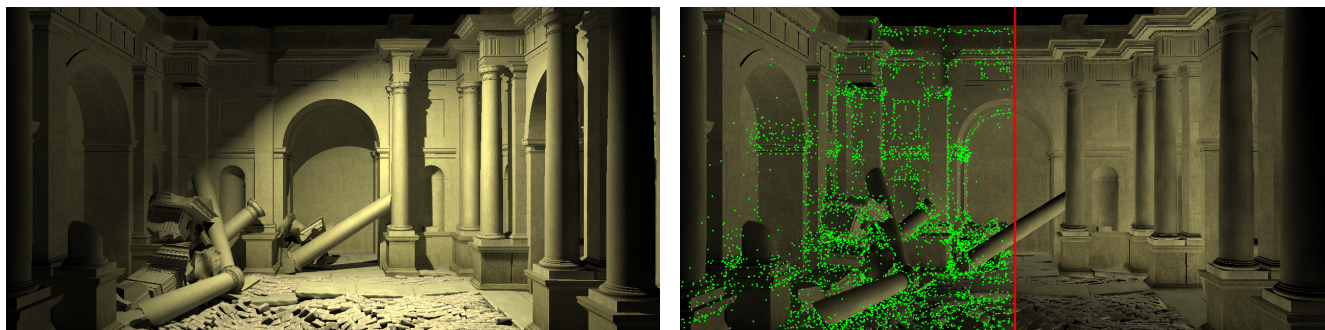Grenoble University

Timo Aila
NVIDIA Research

**Figure 1: Left:** Real-time global illumination on a static 2.3M triangle scene. Both the light and the viewpoint can be moved freely at 7-21 frames per second after a little less than half an hour of precomputation on a single PC. **Right:** The indirect illumination expressed in our meshless hierarchical basis (emphasized for visualization). Green dots represent non-zero coefficients.

## Abstract

We introduce a meshless hierarchical representation for solving light transport problems. Precomputed radiance transfer (PRT) and finite elements require a discrete representation of illumination over the scene. Non-hierarchical approaches such as per-vertex values are simple to implement, but lead to long precomputation. Hierarchical bases like wavelets lead to dramatic acceleration, but in their basic form they work well only on flat or smooth surfaces. We introduce a hierarchical function basis induced by scattered data approximation. It is decoupled from the geometric representation, allowing the hierarchical representation of illumination on complex objects. We present simple data structures and algorithms for constructing and evaluating the basis functions. Due to its hierarchical nature, our representation adapts to the complexity of the illumination, and can be queried at different scales. We demonstrate the power of the new basis in a novel precomputed direct-to-indirect light transport algorithm that greatly increases the complexity of scenes that can be handled by PRT approaches.

**Keywords:** Global illumination, meshless basis functions, precomputed radiance transfer, scattered data

## 1 Introduction

Both precomputed light transport (PRT) techniques and finite element methods produce global illumination solutions that can be viewed interactively with a changing viewpoint. In addition, PRT techniques support changing illumination. To achieve view-independence, they represent the lighting solutions in object space using basis functions defined on the surfaces of the geometry. Non-hierarchical bases, such as piecewise linear per-vertex values or texture maps, have the advantage of simplicity, but the lack of hierarchy leads to long precomputation times and necessitates sophisticated compression algorithms in the PRT setting to achieve good runtime performance [Sloan et al. 2003; Kristensen et al. 2005]. Hierarchical function bases have a number of desirable properties for light transport [Gortler et al. 1993; Kontkanen et al. 2006] but they are harder to construct on complex real-world geometry. For instance, wavelet hierarchies cannot be efficiently constructed on surfaces that do not consist of large piecewise smooth surfaces.

We build on a strategy that has proved powerful for single-image offline rendering: the decoupling of the illumination representation from the surface representation using point samples [Ward et al. 1988; Jensen 1996]. Most of these ray-tracing approaches focus on the ability to cache data in a Monte-Carlo integration context. In contrast, free-viewpoint techniques such as PRT are more related to finite elements [Lehtinen 2007] and thus require a finite set of basis functions on scene surfaces. Furthermore, scalability necessitates a hierarchical representation (Fig. 1). In addressing these challenges, this paper makes the following contributions:

- A meshless hierarchical basis to represent lighting on complex surfaces of arbitrary geometric representation.
- A 5D Poisson-disk distribution generation algorithm that only requires the ability to ray-cast the geometry.
- A method for rendering directly from the meshless basis on the GPU.

- A direct-to-indirect precomputed light transport technique for local light sources that supports both complex geometry and free viewpoints with significantly less precomputation than previous techniques.

While this paper focuses on precomputed direct-to-indirect light transport, our representation is fully general and can be used in many other PRT and global illumination applications. In addition, our meshless representation decouples illumination simulation algorithms from the underlying geometric representation.

## 1.1 Related Work

**Hierachical Function Spaces for Light Transport.** Hierarchical techniques [Hanrahan et al. 1991; Gortler et al. 1993; Sillion 1995; Christensen et al. 1996; Willmott et al. 1999] use multiresolution function spaces such as wavelets to dramatically accelerate finite-element global illumination. Recent work addresses the problems of defining hierarchical function bases on mesh surfaces [Holzschuch et al. 2000; Lecot et al. 2005], but constructing efficient hierarchical bases on arbitrary geometry remains a challenge.

Dobashi et al. [2004] build a hierarchy over surfels and describe a hierarchical radiosity algorithm for point-based models. In contrast, we seek to define smooth basis functions at all scales over the surfaces of arbitrary scenes.

**Precomputed Radiance Transfer.** PRT techniques precompute linear transfer operators that encode how illumination affects the appearance of surface points. These methods have concentrated on the representation of the angular component of lighting and outgoing light using, e.g., spherical harmonics [Sloan et al. 2002], wavelets [Ng et al. 2003], zonal harmonics [Sloan et al. 2005], and radial basis functions on the sphere [Tsai and Shih 2006; Green et al. 2006]. In the spatial domain, these techniques usually precompute the solutions in a non-hierarchical finely-tessellated vertex basis, and compress the data afterwards [Sloan et al. 2003]. Some techniques support local light sources [Kristensen et al. 2005; Kontkanen et al. 2006]. We extend these approaches by introducing a hierarchical basis that is independent of the scene representation and provides dramatic speedup of the precomputation.

**Decoupling Lighting from Geometry.** Point sampling has allowed ray-tracing techniques to cache illumination at a sparse set of points, so that illumination can be interpolated from samples that store irradiance or radiance [Ward et al. 1988; Křivánek et al. 2005; Arikan et al. 2005], or reconstructed using density estimation when photons are stored [Jensen 1996]. These approaches are usually not hierarchical.

Lightcuts [Walter et al. 2005; Walter et al. 2006] build a hierarchy over point samples and adaptively select the appropriate level for efficient synthesis of images with complex illumination, motion blur, and depth of field. The technique is hierarchical with respect to both the point light sources (senders) and geometry (receivers). In their hierarchical image relighting technique, Hašan et al. [2006] precompute the light transport from a scattered cloud of points in the scene into the image. In contrast, we seek to enable view-independent precomputed direct-to-indirect transport. An important difference between our approach and these point-based methods is that our points are used for defining basis functions that represent output lighting, whereas the above methods use point samples to represent an integrand.

**Meshless Finite Elements.** Meshless methods have been applied to computer animation [Desbrun and Cani 1996; Müller et al. 2003; Müller et al. 2004; Pauly et al. 2005] and mesh generation [Ohtake et al. 2005]. In numerical simulation, meshless or mesh-free methods have been used to define basis functions, but usually in a non-hierarchical manner [Belytschko et al. 1996; Liu 2002].

Some recent meshless methods solve the radiative transfer equation in participating media [Hiu 2006]. These approaches deal with smooth media without boundaries and do not employ hierarchies.

**Scattered Data Approximation.** The construction of our basis functions is inspired by scattered data interpolation. Floater and Iske [1996] describe a hierarchical interpolation method using radial basis functions (RBFs), while Fasshauer [2002] proposes a multilevel moving least squares (MLS) scattered data approximation technique. We build on their techniques and define hierarchical basis functions that are particularly well-suited for expressing lighting functions and light transport on the surfaces of a 3D scene. In contrast to radial basis functions, our formulation has the advantage that no large linear systems need to be solved.

## 1.2 Overview

We describe a novel hierarchical function basis for representing illumination in a way that is independent of the underlying geometric representation. For this, we rely on point samples scattered across the scene. The illumination is stored and computed only at those points but can be evaluated anywhere using a scattered data approximation scheme. Furthermore, we define a hierarchy over the point samples and use it to define hierarchical basis functions where each level stores the difference to the next level. We show that our construction corresponds to a projection operator. This means that it can be used for any projection method such as a global illumination solver or pre-computed light transport.

The basis shares many of the advantages of wavelets, but without the difficulties associated to meshing and parameterizing the surfaces. Just as with wavelets, the representation tends to be sparse and many of the finer-level coefficients are negligible, in particular where the illumination is smooth. This also means that illumination can be efficiently approximated at a coarser level when, for example, gathering light coming from a distant complex object.

We describe a novel view-independent precomputed direct-to-indirect light transport algorithm that capitalizes on the advantages of the basis, allowing the efficient rendering of dynamic indirect illumination on models of high complexity. The direct illumination is projected into our basis, yielding a vector of coefficients. We precompute a transfer matrix that maps such a direct illumination vector into indirect lighting, also represented in our basis. Indirect illumination is then rendered from the basis, while direct illumination is rendered using standard real-time techniques. In contrast to most previous methods that compute a transfer matrix densely before compressing it, we compute the matrix by hierarchical refinement, thereby directly leveraging sparsity.

## 2 Meshless Hierarchical Basis Functions

### 2.1 Meshless Approximation

Our hierarchy is defined using multiple levels of point distributions with increasing density (Fig. 3). We denote the points on level $l$ by $X_l = \{p_j\}_{j=N_{l-1}+1}^{N_l}$, $l = 0, \ldots, L$. $N_l$ is the number of points on all levels up to (and including) level $l$, $N_{-1} = 0$, and $L$ is the finest level. Notice that we enumerate the points across all levels using a single index.

Suppose we want to approximate a function $f(p)$ of surface location $p$. We first construct a function $F_0$ that approximates function values at the points on the coarsest level using a Shepard [1968] scheme. The Shepard approximation $F_0(p)$ at an arbitrary point $p$ is defined as

$$F_0(p) = \sum_{j=1}^{N_0} f_j \frac{w_j(p)}{\sum_{k=1}^{N_0} w_k(p)}, \qquad (1)$$

where we denote the function value $f(p_j)$ at a point $p_j$ by $f_j$. The $w_j(p)$ are *weight functions* or *kernels* associated with the points.
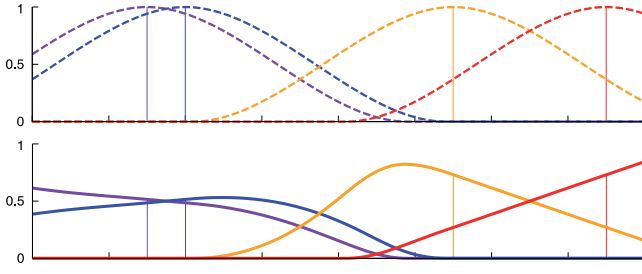
**Figure 2: Top:** A 1D illustration of weight functions associated to a scattered set of points. Point locations are indicated by vertical lines. **Bottom:** The respective basis functions resulting from Shepard approximation: the shapes of the basis functions (y-axis) adapt to varying point density. The basis functions always sum to one.

Typically, they depend only on the distance between $p_j$ and $p$, and they smoothly decrease with increasing distance. We discuss our specific choice for these weight functions in more detail in Section 2.2. We now define

$$B_j(p) := \frac{w_j(p)}{\sum_{k=1}^{N_0} w_k(p)}, \qquad (2)$$

and notice that this leads to a basis expansion (see Figure 2)

$$F_0(p) = \sum_{j=1}^{N_0} B_j(p) f_j, \qquad (3)$$

where the $f_j$ are the coefficients for the basis functions $B_j$. By associating a weight function $w_j$ to each sample point, we have created a linear finite-dimensional function space.

The idea behind multilevel approximation is to progressively add detail by introducing the points from finer levels, in a manner similar to the lifting scheme in second-generation wavelets [Sweldens 1997]. At the points on finer levels, we approximate the *difference* between the approximation from the previous levels and the function values at the new points. For example, at level $l + 1$, our complete approximation is

$$F_{l+1}(p) = F_l(p) + \tilde{f}_l(p), \quad \text{with}$$

$$\tilde{f}_l(p) = \sum_{j=N_l+1}^{N_{l+1}} B_j(p)\left(f_j - F_l(p_j)\right), \qquad (4)$$

The function $\tilde{f}_l$ is the Shepard approximation of the difference between the $l$th-level approximation $F_l$ and the function values at the points on level $l + 1$. The basis functions for points on finer levels are defined analogously to Equation (2), with the normalization performed among the basis functions on the current level only. We can express the approximation at level $l$ as

$$F_l(p) = F_0(p) + \tilde{f}_0(p) + \ldots + \tilde{f}_{l-1}(p). \qquad (5)$$

Finally, we may rewrite the multilevel scheme as a regular basis expansion with new basis coefficients $\alpha_j$:

$$F_l(p) = \sum_{j=1}^{N_i} \alpha_j B_j(p), \qquad (6)$$

where the $\alpha_j$ are defined as

$$\alpha_j = \begin{cases} f_j, & 0 < j \leq N_0, \\ f_j - F_0(p_j), & N_0 < j \leq N_1, \\ \vdots \\ f_j - F_{L-1}(p_j), & N_{L-1} < j \leq N_L. \end{cases} \qquad (7)$$

A practical computation of the approximation coefficients $\alpha_j$ starts from the coarsest level, where the $\alpha_j$ correspond to the function values $f_j$ at the points $p_j$. On each finer level $l$ the coefficients encode the difference between the function values and the approximation $F_{l-1}$ resulting from coefficients on all previous levels. As a result, finer level coefficients tend towards zero in smooth regions. Similarly to simple wavelet compression, the coefficients with small absolute values may be truncated to zero. Figure 3 illustrates our multilevel approximation.

**Neighbors, Parents, and Children.** Consider a basis function $B_i$ that belongs to level $l$. We call all other basis functions $B_n$ on the same level for which $B_i(p_n) \neq 0$ the *neighbors* of $B_i$. We call all functions $B_j$ on level $l + 1$ for which $B_i(p_j) \neq 0$ the *children* of $B_i$. Correspondingly, the *parents* of $B_i$ are all the functions $B_k$ on level $l - 1$ for which $B_k(p_i) \neq 0$. These parent and child relationships both form a directed acyclic graph (DAG). While this contrasts with the tree wavelets most often employed in graphics applications, the structure is straightforward and leads to simple algorithms.

**Pull-Push Approximation.** The coarse-to-fine approximation converges visually once it reaches a fine enough level so that the sample points are denser than the local Nyquist frequency of the function $f$. However, intermediate reconstructions from the basis are not necessarily properly bandlimited versions of $f$ because of aliasing caused by undersampling at the previous levels. Sometimes such bandlimiting can be desirable to guarantee that the intermediate levels carry deviations from local averages (see Section 3). We can compute such an expansion easily in a pull-push manner. The function $f$ is first sampled on all the points across all levels. Then, starting from the second-finest level, each basis function coefficient is set to the weighted average of function values in its children. The procedure continues level-by-level to the coarsest level, where each basis function now carries the weighted average of all the points under its support. Using these modified coarsest-level coefficients, the coefficients for finer levels are recomputed as in Equation (7). Instead of point samples, the previously computed averages are used as the values $f_j$.

**Efficient Reconstruction.** To reconstruct a value from the basis, we need to quickly determine all the basis functions whose weight functions are non-zero at the query point. To facilitate this, we store the points on each level into a separate R-Tree [Guttman 1984], which is basically a bounding volume hierarchy. Each basis function is represented by a sphere, and the bounding volumes of the nodes are axis-aligned boxes. Querying this tree is a simple binary tree traversal. After the basis functions are known, the reconstruction proceeds by evaluating Equation (6) with the basis functions defined through Equation (2). We chose the R-tree over the kD tree since in a bounding volume hierarchy each node splits the data into exactly two without replication.

**Tensor Product Bases.** Our spatial basis can be easily used for representing directionally-varying quantities such as the full 4D radiance field. This is achieved by taking a tensor product of the spatial basis with a directional basis such as the spherical harmonics. The approximation coefficients $\alpha_i$ are then not scalars but vectors of directional basis coefficients.

## 2.2 Weight Functions

In addition to the surface location, illumination functions such as irradiance depend strongly on the local surface orientation. To accommodate this, our weighting combines a spatial fall-off kernel and a factor based on the differences between normals. Concretely, the points $p_j$ we use consist of a 3D position $x_j$ and the corresponding surface normal $n_j$, i.e., $p_j = (x_j, n_j) \in \mathbb{R}^3 \times S^2$. Each point is also assigned a radius $r_j$ whose value depends on the level of hi-

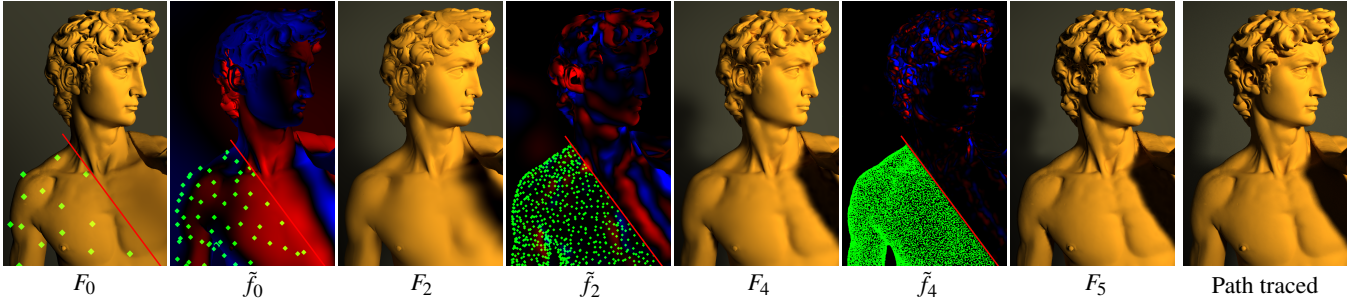| $F_0$ | $\tilde{f}_0$ | $F_2$ | $\tilde{f}_2$ | $F_4$ | $\tilde{f}_4$ | $F_5$ | Path traced |

**Figure 3:** An illustration of hierarchical meshless approximation of incident irradiance, including both direct and indirect illumination. Starting from the coarse approximation $F_0$, successive reconstructions $F_{2,4,5}$ show refinement towards the path traced reference. Levels 1,3 have been omitted due to space constraints. Images $\tilde{f}_{0,2,4}$ denote the differences encoded by successive hierarchy levels (red denotes positive and blue negative). Green dots in the coarse reconstruction and difference images denote the points that define that basis on each level.

erarchy the point belongs to: the higher the level, the smaller the radius. The choice of radii is discussed in Section 2.3. The weight function $w_j$ associated with $p_j$ is is defined as

$$w_j(p) = w_j(x,n) = K_j(\text{dist}_j(x)) \max\left(0, n \cdot n_j\right)^2, \qquad (8)$$

with

$$K_j(r) = K(\frac{r}{r_j}), \quad K(r) = \begin{cases} 2r^3 - 3r^2 + 1, & r \leq 1, \\ 0 & r > 1, \end{cases} \qquad (9)$$

and    $\text{dist}_j(x) = \|x - x_j + F_{\text{squash}}\langle x - x_j \cdot n_j\rangle n_j\|. \qquad (10)$

The polynomial kernel $K_i$ ramps smoothly from 1 at $r = 0$ to 0 at distance $r = r_j$, which implies that the corresponding basis function has compact support (see top row in Figure 2). The second term gives more weight to points oriented similarly to $p_j$, and in particular, the weight is zero for points backfacing w.r.t $p_j$. The distance $\text{dist}_j(\cdot)$ is defined so that its iso-surfaces around $x_j$ form ellipsoids whose axis aligned with $n_j$ is a factor $1/(1 + F_{\text{squash}})$ smaller than the other two. This means that the distance increases faster when $x$ deviates from the plane tangent to $x_j$. We use $F_{\text{squash}} = 2$. This kind of distance is called a Mahalanobis distance.

While the weight functions are defined on a 5D domain, the basis functions are always evaluated at surface points that have unique normals. They thus form a 2D basis on the surfaces. The interpolation weights in irradiance caching [Ward et al. 1988] resemble ours in this respect. Basis functions are visualized in Figure 4. The influence of an individual weight function is not restricted to a single continuous surface patch. Instead, it contributes to all surfaces that intersect its support in $\mathbb{R}^3 \times S^2$. However, finer levels correct leaks from coarser levels as illustrated in Figure 3.

## 2.3 Point placement

We now describe an algorithm for distributing point samples across the scene in a way that is minimally dependent on the surface representation. We identify the following requirements: First, the sample points have to be located on surfaces that will be viewed by the user, and not on surfaces that are completely enclosed (inside solid structures in the model), and second, the basis functions on each level should uniformly cover the scene.

The first criterion is non-trivial to fulfill on real-world models, because they often contain out-of-sight solid structures within other structures. We solve this problem by requiring the user to provide one valid viewpoint in the scene. We then define all surface locations from where light can reach this viewpoint as *valid*. We compute a point-sampled representation of all such surfaces by ray tracing importance particles backwards from the user-specified viewpoint, letting the particles reflect about in the scene, and always

depositing a particle on a surface at an intersection. We disregard the first few bounces (typically, 3 bounces out of 30) to make the resulting distribution reasonably uniform and independent of the starting location. (Small non-uniformities in the distribution are of no consequence in the subsequent steps.) In the following, we will refer to these points as *candidates*.

The second requirement is fulfilled by distributing the samples according to a Poisson disk distribution that follows the 5D distance induced by our weight functions. We define the distance as

$$D(p,q) = \frac{\min(\text{dist}_p(x_q), \text{dist}_q(x_p))}{\max\left(0, n_p \cdot n_q\right)^2}, \qquad (11)$$

i.e., we symmetrize the distances $\text{dist}_p(\cdot)$ and $\text{dist}_q(\cdot)$ used in the weight functions and divide by the cosine term. This distance has the desired property that two points can be placed arbitrarily close in 3D if their normals are in a $\geq 90$ degree angle. We use the candidate points from the importance tracing step in a simple dart throwing algorithm to construct a hierarchical Poisson disk distribution that respects $D(p, q)$. The hierarchical construction begins with an initial disk radius $R = R_{\text{coarse}}$. Then for each remaining level we halve $R$ and repeat the process [Mitchell 1991]. Since the different levels of hierarchy are independent, the points on previous levels are not included in the test. We add new points on each level until the dart thrower fails 2000 times in a row. More candidates are lazily generated by the importance tracing algorithm when necessary.

In Shepard approximation, the weight functions need to overlap substantially to ensure smoothness. In our direct-to-indirect transport application, we thus need to have fine basis functions overlap sufficiently to get a smooth final reconstruction. On the other hand, such overlap on the coarser levels would result in large basis functions that do not capture illumination variation well. We set $r_i = CR$, where $C$ varies linearly from 1.5 on the coarse level to 2.5 on the finest level. The initial radius $R_{\text{coarse}}$ and the number of levels are user-specified constants. Typically, basis construction takes a few minutes on a multi-million triangle model.

When representing indirect illumination in the basis, we want the supports of the basis functions to correspond to the expected variation in illumination, i.e., coarse points should be placed in regions that have high visibility to their surroundings, and tighter locations should get filled in by points on finer levels. We accommodate this by an additional acceptance criterion: we compute the harmonic mean distance to nearby geometry over the hemisphere for the candidate points that pass the Poisson disk criterion using rays of length $3R$. If the mean distance is low ($< R/6$), the candidate is rejected.

## 2.4 Mathematical Discussion

**Projection Operator Notation.** The approximation coefficients $\alpha_i$ are linear functions of the input function $f$. To see this, consider
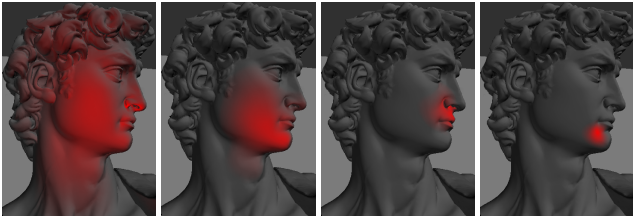
**Figure 4:** A visualization of a basis function on four different levels of hierarchy. Red color denotes magnitude.

the coarse-to-fine approximation scheme: The coarse coefficients are just point samples of the function, and the subsequent levels depend on earlier ones in a linear fashion. It is thus appropriate to write the meshless approximation as a linear operator $\mathcal{P}$ that acts on $f$ and produces the basis expansion in Equation (6), denoted by $\mathcal{P}f$. The pull-push approximation algorithm corresponds to a different linear operator $\mathcal{P}^{\uparrow\downarrow}$. From an abstract point of view, this ties our approximation technique in with other projection operators, such as $L_2$ projection, that have been widely used in light transport algorithms, e.g., [Gortler et al. 1993; Christensen et al. 1996].

Our approximation schemes are not exact projectors, because applying the procedure again to an approximated function $\mathcal{P}f$ does not produce exactly the same result. We could make $\mathcal{P}$ an interpolatory projection by using singular weight functions. However, interpolations are very sensitive to noise in $f_j$, whereas our approximation smooths the data and is thus more forgiving visually.

**Connections.** The basis functions on the finer levels do not directly correspond to wavelets because they are non-negative functions that have nonzero integrals. Thus, in wavelet parlance, our basis more resembles an overcomplete pyramid of scaling functions. The resemblance is not exact, since our hierarchy levels are not nested. Our construction is also akin to the Laplacian pyramid [Burt and Adelson 1983], although with irregular sampling. Furthermore, the connections between our basis and second-generation wavelets [Sweldens 1997] offer an interesting avenue for future exploration.

# 3 Hierarchical Direct-to-Indirect Transfer

We describe a novel meshless direct-to-indirect transport technique [Kristensen et al. 2005; Kontkanen et al. 2006; Hašan et al. 2006] based on our meshless hierarchical representation. The main idea is to precompute a finite transport matrix that maps discretized direct illumination from the surfaces to a basis expansion for indirect illumination. The final image is obtained by compositing the basis expansion for indirect illumination with direct illumination which is rendered using traditional real-time techniques.

## 3.1 Precomputed Direct-to-Indirect Transport

Denoting direct incident illumination on the scene surfaces by $E^{\leftarrow}$, the rendering equation may be written in its incident form [Arvo et al. 1994] as

$$L^{\leftarrow} = \mathcal{T}L^{\leftarrow} + E^{\leftarrow}. \quad (12)$$

The solution $L^{\leftarrow}$ denotes incident radiance over the surfaces, including indirect illumination. This continuous equation may be discretized by applying a basis projector $\mathcal{P}$ to both $L^{\leftarrow}$ and $E^{\leftarrow}$ [Atkinson 1997]

$$\mathcal{P}L^{\leftarrow} = \mathcal{P}\mathcal{T}\mathcal{P}L^{\leftarrow} + \mathcal{P}E^{\leftarrow}. \quad (13)$$

This equation is a discrete linear system that forms the basis of finite element global illumination algorithms such as radiosity and a large colletion of its variants, e.g., [Gortler et al. 1993; Christensen et al. 1996]. Because our approximation scheme can be written as

a projector, our representation directly applies in any scenario that can be written in the form (13).

Because the output of the projector $\mathcal{P}$ lies in the finite-dimensional subspace spanned by the basis functions, the operator $\mathcal{P}\mathcal{T}\mathcal{P}$ can be represented by a finite matrix $\mathbf{T}$ that describes the effect of the transport operator $\mathcal{T}$ on the basis functions. Its columns are simply coefficient vectors that describe the one-bounce illumination that results from a single emitting basis function at a time. Denoting the solution coefficients by $\mathbf{l}$ and the emission coefficients $\mathbf{e} = \mathcal{P}E^{\leftarrow}$, this becomes

$$\mathbf{l} = \mathbf{T}\mathbf{l} + \mathbf{e} \quad \Leftrightarrow \quad \mathbf{l} = (\mathbf{I} - \mathbf{T})^{-1}\mathbf{e}. \quad (14)$$

The indirect part of the solution is given by $\mathbf{i} := \mathbf{l} - \mathbf{e}$, i.e., the final direct-to-indirect transport operator is

$$\mathbf{S} := (\mathbf{I} - \mathbf{T})^{-1} - \mathbf{I}, \quad \text{so that} \quad \mathbf{i} = \mathbf{S}\mathbf{e}. \quad (15)$$

The goal of precomputed direct-to-indirect transport is to precompute the operator $\mathbf{S}$ and apply it at runtime to a changing incident illumination vector. The crucial advantage of employing hierarchical function bases, such as ours, is that the solution matrix $\mathbf{S}$ will be *sparse* in the spirit of wavelet radiosity [Gortler et al. 1993]. We also note that all other types of view-independent PRT algorithms, such as those that use distant illumination [Sloan et al. 2002], can be discretized along similar lines [Lehtinen 2007], and thus our basis can be applied in various other PRT scenarios.

Our meshless direct-to-indirect transport technique implements the above equations directly. As described below, we first precompute the matrix $\mathbf{T}$ by an adaptive hierarchical algorithm, and then use it for computing $\mathbf{S}$ by the Neumann series in the spirit of Kontkanen et al. [2006]. We represent the indirect illumination by vector irradiance [Willmott et al. 1999] and the direct illumination as scalar irradiance. We use fewer levels of hierarchy for representing the senders (the direct illumination) than for the receivers, since only the receivers are visualized directly on screen. Direct illumination is rendered using shadow mapping. Because we let all sender basis functions potentially link to every receiving basis function, our discretization is a so-called *standard operator decomposition*.

## 3.2 Precomputation

The aim of the precomputation stage is to find and compute the significant entries of the matrix $\mathbf{T}$ without exhaustively considering all pairs of senders and receivers. The entries $T_{ij}$ are called *transport links* between a receiver $B_i$ and a sender $B_j$, i.e., each link denotes the vector irradiance at the receiver due to unit scalar irradiance at the sender.

We compute the links from *all* senders to a given receiver $B_i$ by a single hemispherical gathering integral. This allows many link computations to share the same ray cast operations. Specifically:

$$T'_{ij} = \int_{\Omega(i)} B_j(r(x_i, \omega)) \rho(r(x_i, \omega)) \bar{\omega} \, d\omega. \quad (16)$$

Here $r(x_i, \omega)$ is the ray-cast operation that returns the closest point to $x_i$ in direction $\omega$, $\rho$ is the diffuse reflectance, and $\bar{\omega}$ is a 3-vector that points to direction $\omega$. These integrals give the vector irradiance at node $x_i$ from all sender basis functions $B_j$ that can see $x_i$. However, if the receiver $B_i$ is not on the coarsest level, the contributions of the parent receiver functions have to be subtracted out, since $B_i$ encodes a difference according to the coarse-to-fine approximation scheme. Thus, the final link is, according to Equation (7),

$$T_{ij} = T'_{ij} - \sum_{k \in \text{all-parents}(i)} T_{kj} B_k(x_i), \quad (17)$$

where "all-parents" means that all levels above the level to which $B_i$ belongs have to be considered.

> ▷ Push the coarsest basis functions onto a queue Q.
>
> ▷ **while** Q is not empty
>
> > ▷ Pop a receiver basis function $B_i$ off the front of Q.
> >
> > ▷ Compute the links from all senders to $B_i$ according to Equations (16) and (17).
> >
> > ▷ Examine each link $T_{ij}$ for current receiver $B_i$:
> >
> > > ▷ **if** the sender $B_j$ is on the coarsest level:
> > > Accept link $T_{ij}$ if the magnitude $\|T_{ij}\| > \varepsilon$.
> > > ▷ **if** the sender $B_j$ is on a finer level:
> > > Let $M = \max_k \|T_{ij} - T_{ik}\|$, where $k \in \text{neighbors}(j)$.
> > > Accept the link if $M > \varepsilon$ and $\|T_{ij}\| > \varepsilon$.
> >
> > ▷ **if** any of the links $T_{ij}$ were kept:
> > Push all the children of $B_i$ in the end of Q.

**Figure 5:** Pseudocode for the adaptive hierarchical algorithm for computing the single-bounce operator **T**.

**Adaptive refinement of the single-bounce operator.** We precompute the operator **T** using a receiver-centric refinement strategy. This differs from earlier techniques that consider pairs of links at a time only [Hanrahan et al. 1991; Kontkanen et al. 2006]. Figure 5 provides pseudocode. The refinement algorithm works its way down the receiver hierarchy one level at a time. This has the desirable effect that all the links needed for the differencing in Equation (17) have either been computed or marked as insignificant. The receivers are kept in a queue that guarantees the correct order of descent onto finer levels. A single receiver function at the front of the queue is examined at a time, and the links from all senders to the receiver are computed all at once by Equations (16) and (17). The links are then examined in order to see which ones are actually needed (see below). If any links to the current receiver are kept, its children are pushed onto the queue if they are not already in it.

We employ the following criteria for deciding which links to keep. For coarsest-level senders we consider only the magnitude of the link. If the magnitude is negligible, we conclude that the effect of this sender has been correctly accounted for by the higher levels of receiver hierarchy, and the link is thrown away. An additional criterion is employed for finer-level senders: we look at the *variation* in links between the sender function and the neighboring senders, and decide to keep the link if the variation is greater than a threshold. We refer the reader to Appendix A for discussion.

After the refinement stops, the final operator **S** is computed by the Neumann series with sparse matrix-matrix products as $\mathbf{S} = \mathbf{T} + \mathbf{T}^2 + \mathbf{T}^3 + \dots$ We compute the second and subsequent bounces ($\mathbf{T}^2$ and onwards) using only the coarsest functions as receivers. This is justified, since second-bounce indirect illumination is typically very smooth. Negligible coefficients, i.e., links that have $\|S_{ij}\| < \varepsilon$, are truncated to maintain sparseness at each bounce.

### 3.3 Runtime rendering

We use a single pointlike spotlight as the source of direct illumination in our prototype implementation. When the light moves, the coefficients for direct incident illumination are recomputed in a pull-push manner to ensure approximate bandlimiting: $\mathbf{e} = \mathcal{P}^{\uparrow\downarrow} E^{\leftarrow}$. Visibility from the light source to the sample points is determined using a ray tracer, but any other direct illumination approach could be used as well. The vector irradiance coefficients for indirect illumination are then computed as $\mathbf{i} = \mathbf{S}\,\mathbf{e}$, using all the non-zero links $S_{ij}$ and all emitters for which $e_j > \varepsilon$. Processing one sender at a time rather than one receiver at a time means that links that carry no energy in the current frame can be skipped. Both the projection of direct illumination and the multiplication by the transport matrix are computed on the CPU. The indirect vector irradiance specified by the coefficients **i** is rendered on the GPU into an offscreen buffer (see Section 4.1), turned to irradiance by dotting with the surface

normal, and composited with direct illumination which is rendered using standard shadow mapping. Finally, the result is modulated by the surface albedo. Note that normal mapping is easily supported because the indirect solution is represented as vector irradiance.

## 4 Implementation and Results

### 4.1 GPU Renderer

Our GPU renderer visualizes the approximation directly from the meshless representation. The implementation resembles earlier GPU splatting methods, e.g., deferred splatting [Guennebaud et al. 2004] and radiance cache splatting [Gautron et al. 2005].

We utilize deferred shading, and in the first pass the geometry of the scene is rendered into two buffers that contain the positions and normals of the visible surfaces. Then, as optimizations, we perform view frustum culling and occlusion queries using the nodes of the R-Tree that hold the basis functions. The granularity of queries was chosen so that each query represents roughly 1000 basis functions.

For each level, all the spatial basis functions on that level are passed to pixel shaders as quads whose size is conservatively determined from the projection of the function's bounding sphere. Whenever a bounding sphere intersects the viewport, we render the corresponding function using a screen-sized quad. We send the basis functions as points that are expanded to quads by the geometry shader unit. The approximation coefficients $\alpha_i$ are uploaded as a dynamic vertex buffer. A pixel shader then evaluates the weight according to Equation (8) for each covered pixel, and the weights and approximation coefficients are accumulated into separate buffers. After a level has been rendered, the accumulated coefficients are normalized using the accumulated weights, and the result is added to the final image.

**Flattening.** When visualizing a hierarchical approximation directly, the overall performance of the GPU renderer is limited by the accumulation of basis functions because the hierarchical basis consists of several overlapping layers. Because of this, we resample the indirect illumination solution **i** into a single-level meshless basis constructed from the same points as the hierarchical basis before display. This significantly reduces overdraw. Similar strategies are employed by previous wavelet algorithms: they perform simulation in wavelet space and resample the solution into a flat basis for display. It should be noted that the resampling can trivially be done into any basis, e.g., piecewise linear vertex functions.

### 4.2 Results

All tests were run on a PC with a quad-core 2.4GHz Intel Q6600 CPU, NVIDIA GeForce 8800Ultra, and 2GB of physical memory. All images use a simple global tone mapping operator [Reinhard et al. 2002]. Figures 1 and 7 show screenshots from a real-time visualization where both the viewpoint and the local light source can be moved freely. In our test scenes, runtime performance varies between 12 and 25 frames per second (FPS) in $1280 \times 720$ resolution when the light is stationary, and between 6 and 9 FPS when the light is moving. When the light is moving, the CPU is used for computing direct illumination (between 8 and 70ms for our scenes), applying the transport operator (5 to 54ms) and flattening the solution for display (21 to 33ms). All these stages could potentially be implemented on the GPU for added efficiency.

Table 1 provides precomputation statistics for our test scenes. The precomputation simulated four bounces of light. Only a small fraction of the potential links were computed in the precomputation step thanks to our hierarchical adaptive operator refinement algorithm. The number of potential links is the number of sender basis functions times the number of receiver basis functions. The threshold $\varepsilon$ can be used for controlling the balance between speed, memory consumption and the fidelity of the result. It was set to 0.025 in all tests, which results in reasonably good quality indirect

| Scene | #Triangles | #Links | #Visibility rays | #Hierarchy levels (rcvr/sndr) | Basis Functions | | Timing breakdown (m:s) | | | Memory consumption |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Level 0 | Total | Adaptive refinement | Indirect Bounces | Total time | |
| **Sponza** | 66k | 973k (0.032%) | 92M | 6/3 | 339 | 192k | 18:18 | 7:56 | 26:14 | 77.4 MB |
| **Great Hall** | 2.3M | 1.3M (0.052%) | 86M | 6/3 | 176 | 177k | 25:58 | 2:58 | 28:56 | 91.0 MB |
| **Jade Palace** | 5.1M | 1.3M (0.038%) | 103M | 6/3 | 317 | 221k | 27:47 | 7:32 | 35:19 | 119.6 MB |

**Table 1:** Direct-to-indirect light transport precomputation statistics from three test scenes.

illumination in our scenes. The memory consumption includes the data structures for both the meshless basis and the transport links. The Great Hall and Jade Palace scenes include detail that would be very difficult to parameterize using wavelets. The Jade Palace is a scene from the animated feature film Kungu Fu Panda (courtesy of DreamWorks Animation), demonstrating our ability to work with production geometry without hand-tuning the mesh. The majority of precomputation time is spent in adaptive refinement. The refinement time is dominated by basis function evaluation in Equation (16), management of links, and evaluating the differences by Equation (17). In all, they account for roughly 90% of the time.

**Comparisons.** Our meshless approach shows all its advantages in this application. Compared to Kristensen et al. [2005], we achieve two orders of magnitude faster precomputation on the Sponza Atrium scene thanks to the hierarchy. Also, our method is not limited to omni-directional light sources. While our approach is conceptually similar to the technique of Kontkanen et al. [2006], we can handle much more intricate geometry since we do not rely on wavelet parameterization of the surfaces. Hašan et al. [2006] also precompute direct-to-indirect transport due to local light sources. Their work and ours both use a hierarchy of point samples for representing the direct illumination on surfaces (the sender hierarchy). The difference is that their gather cloud is a piecewise constant wavelet hierarchy over the point samples themselves, and it does not define a function basis over all surfaces like ours does. Furthermore, since their receivers are image pixels, the solution can only be viewed from a fixed position, while our receivers are hierarchical surface basis functions that allow arbitrary viewpoints.

We have shown that performing the operator precomputation directly in the meshless hierarchical basis is advantageous, and that rendering from a meshless representation is also reasonably fast when the solution is first resampled into a flat basis. However, the discrete operator itself could be resampled into a traditional non-hierarchical basis. This would allow compression and rendering using earlier techniques [Sloan et al. 2003; Kristensen et al. 2005].

# 5 Discussion and Conclusions

## 5.1 Algorithm Portability

All the algorithms presented in the previous sections rely minimally on the geometric representation of the scene. The basis construction algorithm relies on ray tracing functionality only, and reconstruction from the basis is not dependent on the representation at all, because the weights can be evaluated without notion of a surface. In addition to ray tracing, any meshless light transport algorithm based on our representation requires the ability to determine how the intersection point reflects incident illumination, as for example in Equation (16). These operations are readily available from any geometric representation, and can be easily wrapped in a small user-provided interface class. Thanks to the lean interface, all the code related to both construction and evaluation of basis functions and the actual light transport algorithm become independent of the surface representation, and thus light transport algorithms formulated in terms of the meshless basis are automatically portable between surface representations.

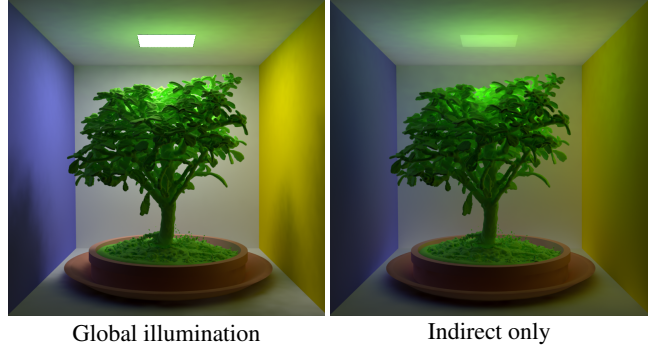To demonstrate portability, we have implemented a simple adap-



Global illumination        Indirect only

**Figure 6:** A scene containing various surface representations, rendered with a meshless hierarchical radiosity plugin to PBRT. The box is a mesh, the bonsai is an isosurface from a volumetric dataset and the pot is made out of implicit surfaces.

tive meshless progressive radiosity algorithm [Lehtinen et al. 2007] in the freely available PBRT renderer [Pharr and Humphreys 2004]. The solver automatically works with any surface representation supported by PBRT. Figure 6 displays a radiosity solution on a scene that contains a volumetric isosurface, quadrics, and a mesh.

## 5.2 Limitations

Since the coarse basis functions have large supports, they may leak to unwanted locations, for example, through a wall between adjacent rooms. The hierarchical solver subsequently has to do extra work to counter the effect. Furthermore, our smooth basis functions cannot exactly represent sharp discontinuities. While the normal term in Equation (8) handles sharp corners in geometry, other discontinuities such as sharp shadow boundaries may occasionally be problematic. The approximation correctly adapts to such boundaries, but can still exhibit some ringing artifacts as the finer basis functions cannot immediately counter the coarse functions leaking over the discontinuity, and many levels of hierarchy are thus required. It is probably possible to enhance our approximation procedure with techniques that respect discontinuities better, e.g., edge-aware illumination interpolation [Bala et al. 2003]. However, we feel that almost any realistic application would render direct lighting using other methods and represent only the smooth indirect component of illumination in our basis, as we have done in our direct-to-indirect transport algorithm. This separation is common in all kinds of global illumination algorithms.

In our direct-to-indirect PRT application, we use the same basis functions for both the senders and receivers. However, the properties that make a basis efficient for the senders are different from those that are desirable for receivers. For the senders, a deep hierarchy with few basis functions at the lower levels is beneficial to be able to approximate distant illumination more aggressively and with fewer links. Receivers are different, since light leaks may require refinement that would be unnecessary with a finer initial distribution. It would be interesting to explore the use of different bases for senders and receivers in the fashion of some prior relighting techniques to better capture these non-symmetric requirements.
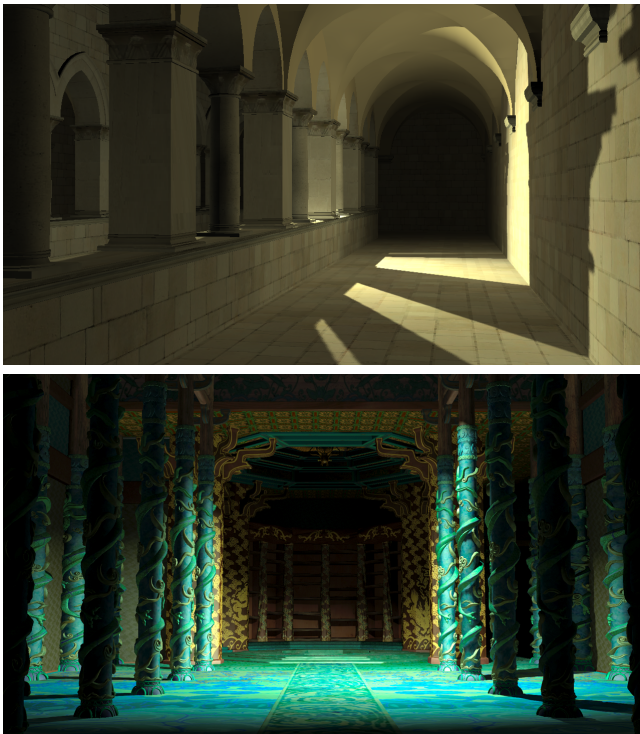
**Figure 7:** *Precomputed direct-to-indirect light transport. Both the light and the camera may be moved freely at runtime.* **Top:** *Sponza Atrium (66k triangles).* **Bottom:** *Jade Palace from the movie Kung Fu Panda (courtesy of DreamWorks Animation, 5.1M triangles).*

## 5.3 Summary

We have presented a hierarchical meshless basis for light transport computations. The basis is suitable for complex geometry, and it is only weakly connected to surface representations. It can thus be used for implementing view-independent hierarchical global illumination algorithms that are portable between surface types and existing codes. We applied our representation to direct-to-indirect precomputed light transport, and described a hierarchical algorithm that supports complex geometry at significantly less precomputation than previous techniques. We believe that our general representation will enable the formulation of many more, highly versatile global illumination methods. For example, we expect our representation to easily generalize to participating media.

## Acknowledgments

## A Appendix

**Details on link thresholding.** All the sender basis functions are non-negative even on the finer levels, and hence their integrals do not vanish, as would be the case with wavelets. Thus the matrix **T** only exhibits numerical sparsity in its columns, not rows. Due to the pull-push approximation algorithm used for the direct illumination, the finer sender functions encode differences to local averages, and thus the averages of the sender coefficients between neighboring functions will be close to zero. Now, if the links from a given receiver to a number of neighboring senders are similar, the net effect of these senders will be small, and the links do not need to be kept. On the other hand, if the links vary significantly, this tells of either occlusion in between the receiver and senders, or that the senders are near the receiver, in which case the links are necessary.

## References

ARIKAN, O., FORSYTH, D. A., AND O'BRIEN, J. F. 2005. Fast and detailed approximate global illumination by irradiance decomposition. *ACM Trans. Graph. 24*, 3, 1108–1114.

ARVO, J., TORRANCE, K., AND SMITS, B. 1994. A framework for the analysis of error in global illumination algorithms. In *Proc. ACM SIGGRAPH 94*, ACM Press, 75–84.

ATKINSON, K. 1997. *The Numerical Solution of Integral Equations of the Second Kind.* Cambridge University Press.

BALA, K., WALTER, B. J., AND GREENBERG, D. P. 2003. Combining edges and points for interactive high-quality rendering. *ACM Trans. Graph. 22*, 3, 631–640.

BELYTSCHKO, T., KRONGAUZ, Y., ORGAN, D., FLEMING, M., AND KRYSL, P. 1996. Meshless methods: An overview and recent developments. *Comput. Methods Appl. Mech. Eng.*, 139, 3–48.

BURT, P., AND ADELSON, E. 1983. The Laplacian pyramid as a compact image code. *IEEE Trans. Comm. 31*, 4, 532–540.

CHRISTENSEN, P. H., STOLLNITZ, E. J., SALESIN, D. H., AND DEROSE, T. D. 1996. Global illumination of glossy environments using wavelets and importance. *ACM Trans. Graph. 15*, 1, 37–71.

DESBRUN, M., AND CANI, M.-P. 1996. Smoothed particles: A new paradigm for animating highly deformable bodies. In *Proc. Eurographics Workshop on Computer Animation and Simulation*, 61–76.

DOBASHI, Y., YAMAMOTO, T., AND NISHITA, T. 2004. Radiosity for point-sampled geometry. In *Proc. 12th Pacific Conference on Computer Graphics and Applications*, 152–159.

FASSHAUER, G. E. 2002. Matrix-free multilevel moving least-squares methods. *Approximation Theory X: Wavelets, Splines, and Applications*, 271–281.

FLOATER, M., AND ISKE, A. 1996. Multistep scattered data interpolation using compactly supported radial basis functions. *J. Comput. Applied Math. 73*, 65–78.

GAUTRON, P., KŘIVÁNEK, J., BOUATOUCH, K., AND PATTANAIK, S. 2005. Radiance cache splatting: A GPU-friendly global illumination algorithm. In *Proc. Eurographics Symposium on Rendering*, 55–64.

GORTLER, S. J., SCHRÖDER, P., COHEN, M. F., AND HANRAHAN, P. 1993. Wavelet radiosity. In *Proc. SIGGRAPH 93*, 221–230.

GREEN, P., KAUTZ, J., MATUSIK, W., AND DURAND, F. 2006. View-dependent precomputed light transport using nonlinear gaussian function approximations. In *Proc. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 7–14.

GUENNEBAUD, G., BARTHE, L., AND PAULIN, M. 2004. Deferred Splatting . *Comp. Graph. Forum 23*, 3, 653–660.

GUTTMAN, A. 1984. R-trees: A dynamic index structure for spatial searching. In *Proc. ACM SIGMOD 84*, 47–57.

HANRAHAN, P., SALZMAN, D., AND AUPPERLE, L. 1991. A rapid hierarchical radiosity algorithm. In *Computer Graphics (Proc. SIGGRAPH 91)*, 197–206.

HAŠAN, M., PELLACINI, F., AND BALA, K. 2006. Direct-to-indirect transfer for cinematic relighting. *ACM Trans. Graph. 25*, 3, 1089–1097.

HIU, L. H. 2006. Meshless local Petrov-Galerkin method for solving radiative transfer equation. *Thermophysics and Heat Transfer 20*, 1, 150–154.

HOLZSCHUCH, N., CUNY, F., AND ALONSO, L. 2000. Wavelet radiosity on arbitrary planar surfaces. In *Proc. Eurographics Workshop on Rendering*, 161–172.

JENSEN, H. W. 1996. Global illumination using photon maps. In *Proc. Eurographics Workshop on Rendering*, 21–30.

KONTKANEN, J., TURQUIN, E., HOLZSCHUCH, N., AND SILLION, F. X. 2006. Wavelet radiance transport for interactive indirect lighting. In *Proc. Eurographics Symposium on Rendering 2006*, 161–171.

KRISTENSEN, A. W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Precomputed local radiance transfer for real-time lighting design. *ACM Trans. Graph. 24*, 3, 1208–1215.

KŘIVÁNEK, J., GAUTRON, P., PATTANAIK, S., AND BOUATOUCH, K. 2005. Radiance caching for efficient global illumination computation. *IEEE Trans. Vis. Comput. Graph. 11*, 5, 550–561.

LECOT, G., LÉVY, B., ALONSO, L., AND PAUL, J.-C. 2005. Master-element vector irradiance for large tessellated models. In *Proc. GRAPHITE 05*.

LEHTINEN, J., ZWICKER, M., KONTKANEN, J., TURQUIN, E., SILLION, F. X., AND AILA, T. 2007. Meshless finite elements for hierarchical global illumination. Tech. Rep. TML-B7, Helsinki University of Technology.

LEHTINEN, J. 2007. A framework for precomputed and captured light transport. *ACM Trans. Graph. 26*, 4, 13:1–13:22.

LIU, G. 2002. *Mesh-free methods*. CRC Press.

MITCHELL, D. P. 1991. Spectrally optimal sampling for distributed ray tracing. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, vol. 25, 157–164.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 154–159.

MÜLLER, M., KEISER, R., NEALEN, A., PAULY, M., GROSS, M., AND ALEXA, M. 2004. Point based animation of elastic, plastic and melting objects. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 141–151.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans. Graph. 22*, 3, 376–381.

OHTAKE, Y., BELYAEV, A., AND SEIDEL, H.-P. 2005. 3D scattered data interpolation and approximation with multilevel compactly supported RBFs. *Graph. Models 67*, 3, 150–165.

PAULY, M., KEISER, R., ADAMS, B., DUTRÉ, P., GROSS, M., AND GUIBAS, L. J. 2005. Meshless animation of fracturing solids. *ACM Trans. Graph. 24*, 3, 957–964.

PHARR, M., AND HUMPHREYS, G. 2004. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann.

REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. 2002. Photographic tone reproduction for digital images. *ACM Trans. Graph. 21*, 3, 267–276.

SHEPARD, D. 1968. A two-dimensional interpolation function for irregularly-spaced data. In *Proc. 23rd ACM national conference*, 517–524.

SILLION, F. X. 1995. A unified hierarchical algorithm for global illumination with scattering volumes and object clusters. *IEEE Trans. Vis. Comput. Graph. 1*, 3, 240–254.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proc. SIGGRAPH 2002*, 527–536.

SLOAN, P.-P., HALL, J., HART, J., AND SNYDER, J. 2003. Clustered principal components for precomputed radiance transfer. *ACM Trans. Graph. 22*, 3, 382–391.

SLOAN, P.-P., LUNA, B., AND SNYDER, J. 2005. Local, deformable precomputed radiance transfer. *ACM Trans. Graph. 24*, 3, 1216–1224.

SWELDENS, W. 1997. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal. 29*, 2, 511–546.

TSAI, Y.-T., AND SHIH, Z.-C. 2006. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Trans. Graph. 25*, 3, 967–976.

WALTER, B., FERNANDEZ, S., ARBREE, A., BALA, K., DONIKIAN, M., AND GREENBERG, D. P. 2005. Lightcuts: A scalable approach to illumination. *ACM Trans. Graph. 24*, 3, 1098–1107.

WALTER, B., ARBREE, A., BALA, K., AND GREENBERG, D. P. 2006. Multidimensional lightcuts. *ACM Trans. Graph. 25*, 3, 1081–1088.

WARD, G. J., RUBINSTEIN, F. M., AND CLEAR, R. D. 1988. A ray tracing solution for diffuse interreflection. In *Computer Graphics (Proc. SIGGRAPH 88)*, vol. 22, 85–92.

WILLMOTT, A., HECKBERT, P., AND GARLAND, M. 1999. Face cluster radiosity. In *Proc. Eurographics Workshop on Rendering*.