Meshless Finite Elements for Hierarchical Global Illumination

Jaakko Lehtinen[1,2] Matthias Zwicker[3] Janne Kontkanen[1,4] Emmanuel Turquin[5] François X. Sillion[5,6] Timo Aila[1,7]

[1]Helsinki University of Technology [2]Remedy Entertainment [3]University of California, San Diego
[4]PDI/DreamWorks [5]Grenoble University [6]INRIA [7]NVIDIA Research

Abstract

We introduce a meshless finite element framework for solving light transport problems. Traditional finite element methods use basis functions parameterized directly on the mesh surface. The creation of suitable parameterizations or clusterings requires pre-processing that is difficult, error-prone, and sensitive to the quality of input geometry. The resulting light transport solutions still tend to exhibit discontinuities, necessitating heuristic post-processing before visualization. Due to these problems finite element methods are rarely used in production.

The core idea of our approach is to use finite element basis functions induced by hierarchical scattered data approximation techniques. This leads to a mathematically rigorous recipe for meshless finite element illumination computations. As a main advantage, our approach decouples the function spaces used for solving the transport equations from the representation of the scene geometry. The resulting solutions are accurate, exhibit no spurious discontinuities, and can be visualized directly without post-processing, while parameterization, meshing and clustering problems are avoided. The resulting methods are furthermore easy to implement.

We demonstrate the power of our framework by describing implementations of hierarchical radiosity, glossy precomputed radiance transfer from distant illumination, and diffuse indirect precomputed transport from local light sources. Moreover, we describe how to directly visualize the solutions on graphics hardware.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: Global illumination, meshless finite elements, precomputed radiance transfer, radiosity

1 Introduction

Global illumination algorithms aim to generate realistic images of virtual scenes by solving the rendering equation [Kajiya 1986] that describes the transport of light within a scene. Two main variants exist: *Stochastic ray tracing techniques* solve the distribution of light as seen from viewpoints. The output is an estimate of the radiance seen through the pixels of an image grid. In contrast, *finite element techniques* produce an approximation to the global illumination solution as a radiosity or radiance distribution function defined on the surfaces of the scene. These functions are represented using basis expansions in some suitable function spaces. Finite element methods are particularly useful for precomputing operators that describe light transport in the scene; such operators allow interactive visualization of global illumination effects from dynamic lighting from arbitrary viewpoints. Other typical uses include precomputation of fixed lighting solutions (e.g. "lightmaps"), important in many interactive applications. The line between the two main categories is fuzzy, since stochastic computations may be employed in a finite element framework, and finite element solutions may be used in a ray tracing context, for example in "final gathering."

Since the introduction of radiosity in computer graphics by Goral et al. [1984], finite element techniques have been widely studied in both the original diffuse context as well as in glossy settings. Despite the theoretical beauty of hierarchical finite element techniques, their practical use has been impeded by a number of issues. Most importantly, the function spaces used by most finite element techniques for representing the radiance functions *rely on 2D parameterization or clustering of the underlying scene geometry*. This requires extensive pre-processing and high-quality input geometry, and any function space resulting from such parameterization or clustering has unwanted discontinuities that tend to be visible in the resulting images. Additionally, both the representation and computation of lighting are tightly coupled with the geometric representation of the scene. Given that the versatile and popular irradiance caching [Ward et al. 1988] and photon maps [Jensen 1996] abolish this coupling, we believe that a similar decoupling may be beneficial for finite element methods as well.

Figure 1: This global illumination visualization runs 7.2 FPS at 960×720 resolution. The direct-to-indirect transport operator was precomputed in 31 minutes for the Sibenik cathedral with Lucy and David statues (1.2M triangles) using our meshless finite element method. Per-pixel direct lighting was rendered using shadow maps. The viewpoint and light source can be freely moved at runtime.

Our fundamental idea is to represent the spatial variation in the lighting solutions using a form of *multilevel scattered data approximation* based on point samples of the lighting. Our approach is inspired by the multilevel interpolation and approximation techniques of Floater and Iske [1996] and Fasshauer [2002]. We use such a scheme to represent lighting functions on surfaces. Our approach defines a cascade of linear function spaces with increasing resolution. Conceptually much like wavelets, the finer spaces encode differences with respect to the coarser ones, such that we achieve the crucial property of *numerical sparsity* similarly to wavelets. The resulting discretization of the rendering equation is mathematically analogous to previous methods that employ function spaces defined on piecewise 2D domains.

The main advantage of our approach is its independence of the underlying representation of the scene geometry. The structure of our function spaces is solely determined by the distribution of scattered point samples and the associated approximation method. In particular, we do not require 2D parameterization of the domain,

nor grouping of the geometric input primitives at any stage. Therefore hierarchical algorithms based on our framework can exploit smoothness in the solution regardless of chart or cluster boundaries. Contrary to prior finite element techniques, our solutions exhibit no artificial discontinuities and thus require no post-processing prior to visualization. Our general approach is applicable to any geometric representation that allows the generation of scattered point samples, and supports ray queries.

In summary, we make the following contributions:

- A meshless multilevel scheme to represent lighting functions on surfaces (Section 4).
- A framework for meshless finite element methods for light transport simulation that is based on an operator discretization of the rendering equation (Section 5).
- Implementation of several global illumination algorithms as examples of our general framework (Section 6).

2 Related Work

Finite Element Methods for Light Transport. Finite element methods for light transport simulation, known in diffuse scenes as *radiosity* methods, were first introduced to computer graphics by Goral et al. [1984]. Since then, a vast amount of literature has been published on the subject.

A naive finite element discretization of the rendering equation leads to quadratic complexity in both time and space. Theoretically, hierarchical techniques [Hanrahan et al. 1991; Gortler et al. 1993; Christensen et al. 1996] manage to reduce the complexity from quadratic to linear. These algorithms represent the lighting distributions in multiresolution function spaces, e.g., using wavelets. The hierarchies exploit smoothness in the illumination function such that it can be represented accurately using only a small subset of the basis. Hierarchies can also be constructed by clustering geometric primitives [Sillion 1995]. Face Cluster Radiosity [Willmott et al. 1999] computes light transport between clusters of input faces. In addition, it is based on *vector* instead of scalar irradiance. This enables rendering of small scale surface variation without including the detailed geometry in the simulation loop.

The accuracy of finite element methods can be increased by employing higher order basis functions [Zatz 1993]. Many methods combine the concepts of hierarchical and higher order bases, most elegantly using wavelets [Gortler et al. 1993]. Wavelet radiosity has been extended by Holzschuch et al. [2000], who form the domain of a wavelet hierarchy on a planar collection of polygons instead of a single input face. Lecot et al. [2005] cluster the faces of complex input meshes and parameterize them on the unit square. They then define wavelets on the parameter square to represent lighting on the mesh. Face Cluster Radiosity can also be combined with higher order bases [Gobbetti et al. 2003]. However, the spatial function bases are defined in an approximate way on the bounding boxes surrounding the clusters.

Radiosity methods have also been generalized to include glossy reflection in addition to purely diffuse surfaces. Sillion et al. [1991] introduced spherical harmonics for representing the directional variation of radiance. Christensen et al. [1996] use a four-dimensional wavelet basis for storing spatially- and angularly-varying radiance distributions.

Our framework allows us to derive finite element algorithms with properties similar to the ones mentioned above, including hierarchical, higher order bases, and directional effects. The main difference between previous approaches and our framework is that we construct hierarchical bases from scattered points. Our approach does not require the construction of meshes specifically for light transport simulation, nor the parameterization of surface patches. Dobashi et al. [2004] presented a related algorithm for surfaces defined by Surfels [Pfister et al. 2000]. While their approach is indeed a meshless radiosity algorithm, it has significant shortcomings compared to our framework. First, they use the area-to-area formulation of radiosity, which forces them to estimate "effective areas" for each surface sample in an approximate fashion. Second, their approach still couples the representation of the radiosity function tightly with the geometric surface representation. Our framework does not have these issues.

Precomputed Radiance Transfer. Light transport algorithms that are based on a basis expansion of the radiance function have also become popular in the area of precomputed radiance transfer (PRT) [Sloan et al. 2002]. Initially developed using spherical harmonics, there are now variations of the basic technique based on wavelets [Ng et al. 2003], zonal harmonics [Sloan et al. 2005], Gaussians [Green et al. 2006], or spherical radial basis functions [Tsai and Shih 2006]. While many PRT methods are restricted to distant illumination, there are also extensions to local light sources [Kristensen et al. 2005; Kontkanen et al. 2006]. These methods store radiance on the surfaces, similarly to traditional radiosity techniques. In fact, PRT and radiosity methods share the same mathematical fundamentals [Lehtinen 2004]. Therefore, our framework applies to the PRT setting as well. A noteworthy advantage of our approach is that the representation of the radiance distribution is not tied to the discretization of the surface geometry.

Decoupling Lighting from Geometry. The concept of decoupling lighting from geometry has proven useful in rendering strategies that are not based on finite element methods. Variants of irradiance [Ward et al. 1988] and radiance caching [Krivanek et al. 2005] compute the lighting in a sparse set of points and reconstruct the continuous distribution using scattered data interpolation [Amidror 2002]. These caching techniques are fast and widely used in production rendering. Photon mapping [Jensen 1996] traces "photons" from light sources and stores them in a spatial data structure when they hit surfaces. The radiance at an arbitrary point can be computed using density estimation from the photons in a neighborhood. Density estimation requires computing the number of photons per area or volume, which can be problematic around corners and closely spaced surfaces. Nevertheless, photon mapping is a versatile, fast, and widely used technique.

The Lightcuts algorithm [Walter et al. 2005] computes bounded-error estimates of irradiance hierarchically to determine the appropriate representation of light sources at each pixel. In their image relighting technique, Hašan et al. [2006] precompute the light transport from a scattered cloud of points in the scene into the image. After evaluating direct lighting at these points, the transport operator is hierarchically applied to yield an image with indirect illumination. Multidimensional lightcuts [Walter et al. 2006] generalize the lightcuts approach to participating media, depth of field, and motion blur. The technique is hierarchical w.r.t. both the point light sources (senders) and geometry (receivers).

A crucial difference between our approach and these point-based methods is that our points represent basis functions. Further discussion about the connections is presented in Section 6.

Meshless Finite Elements. Meshless finite element methods, also known as meshfree methods, have become popular for a wide range of numerical simulation problems [Belytschko et al. 1996; Liu 2002]. As a common theme, meshless approaches express the solutions using basis functions that are constructed from non-uniformly distributed points. This is in contrast to traditional techniques that define basis functions on elements of tessellated domains. Perhaps the most related to our application are recent meshless methods for solving the radiative transfer equation [Hiu 2006] in the context of heat transfer. However, in stark contrast to graph-

ics, these methods deal with the distribution of radiation in a participating medium with non-reflecting (black) boundaries and simple geometries.

In the computer graphics community, meshless methods have been applied, for example, to the simulation of fluids [Desbrun and Cani 1996; Müller et al. 2003], deformable solids [Müller et al. 2004], fracture [Pauly et al. 2005], and mesh generation [Ohtake et al. 2005]. To the best of our knowledge, this paper is the first to apply this class of numerical techniques to the rendering equation.

Floater and Iske [1996] describe a hierarchical interpolation method for scattered data using radial basis functions. Our method builds on their results, and on Fasshauer's [2002] multilevel moving least squares (MLS) [Lancaster and Salkauskas 1981] technique. We refer the interested reader to the survey by Fasshauer [2007] for an overview of the many flavors of meshless methods and MLS.

3 Operator Formulation of Light Transport

The rendering equation is a linear integral equation that may be written as

$$L = \mathcal{T}L + E, \tag{1}$$

where L is the unknown radiance (or radiosity) distribution, $\mathcal{T}$ is the *transport operator*, and E describes the emission of light. This equation is infinite-dimensional in the sense that the unknown is a function, not a finite-dimensional vector, and the transport operator is an integral operator defined on an infinite-dimensional space of functions. In an abstract form, a finite element discretization of the rendering equation may be written as [Atkinson 1997; Schröder et al. 1994]

$$\mathcal{P}L = \mathcal{P}\mathcal{T}\mathcal{P}L + \mathcal{P}E. \tag{2}$$

Here $\mathcal{P}$ is a *projection operator* that projects a function onto a finite-dimensional subspace, which is spanned by a finite set of basis functions. We note that this framework applies both to diffuse and glossy transport, and the radiance distribution may represent either incident or outgoing radiance. The operator $\mathcal{P}\mathcal{T}\mathcal{P}$ is in fact a *finite* transport matrix as the output of $\mathcal{P}$ lies in a finite-dimensional space. Although the functions that result from applying $\mathcal{T}$ to the basis functions do generally not lie in the same space anymore, the subsequent application of $\mathcal{P}$ after $\mathcal{T}$ forces the result back there. Thus, the jth column of the transport matrix is merely the vector $\mathcal{P}\mathcal{T}B_j$ of projection coefficients, where B_j is the jth basis function of the approximating subspace.

In most well-founded finite element illumination techniques the operator $\mathcal{P}$ is defined through least-squares (L_2 or Hilbert space) projection. This is also known as the Galerkin method. In this work, however, we abandon the Hilbert space context. Instead we work with *operators defined through a scattered data approximation procedure*. These operators produce function expansions that approximate their argument at a set of scattered points, rather than lying closest to the argument in the least squares sense. This type of technique is also known as a meshless collocation method [Zhang et al. 2000].

4 Meshless Multilevel Approximation for Lighting Functions

In this section, we describe our technique for representing lighting functions on the surfaces of the scene. We follow the general outline given by Floater and Iske [1996] in their work on multilevel scattered data approximation, but use Shepard approximation [Shepard 1968] instead of radial basis functions. The resulting algorithm is a variant of Fasshauer's multilevel approximation scheme [2002]; he also studies the approximation properties of such bases.

We will use this technique in Section 5 for discretizing both the radiance functions and the transport operator, and for solving the resulting discrete rendering equation (Equation 2).

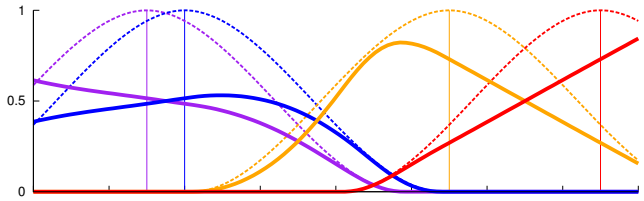


Figure 2: The vertical lines and associated dashed weight functions illustrate a sparse set of points. The respective basis functions resulting from Shepard approximation are shown using bold curves. As can be seen, the shapes of the basis functions (y-axis) adapt to varying point density (x-axis). The functions always sum to one.

4.1 Meshless Approximation

Scattered data approximation schemes take an irregularly sampled set of points with associated function values as input, and produce an approximating function that is defined everywhere in the domain as output. Although in the following we explain our approach for scalar valued functions, it is straightforward to extend it to a vector valued case.

Assume we have generated a number of sample points in the vicinity of the surfaces of our scene (the exact method will be covered in Section 6.1). We group the points into distinct sets of increasing density, which we call *levels*. We denote the points on level i by $X_i = \{p_j\}_{j=N_{i-1}+1}^{N_i}$, $i = 0,\ldots,L$. N_i is the number of points on all levels up to (and including) level i, $N_{-1} = 0$, and L is the finest level. Notice that we enumerate the points across all levels using a single index. The reader may think of the argument p as simply a 3D location but it is also possible to use higher-dimensional points, such as a combination of position and normal vector (see Section 4.2).

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 approximant $F_0(p)$ at an arbitrary point p is defined as

$$F_0(p) = \sum_{j=1}^{N_0} f_j \frac{w_j(p)}{\sum_{k=1}^{N_0} w_k(p)}, \tag{3}$$

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. 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 4.2. We now define

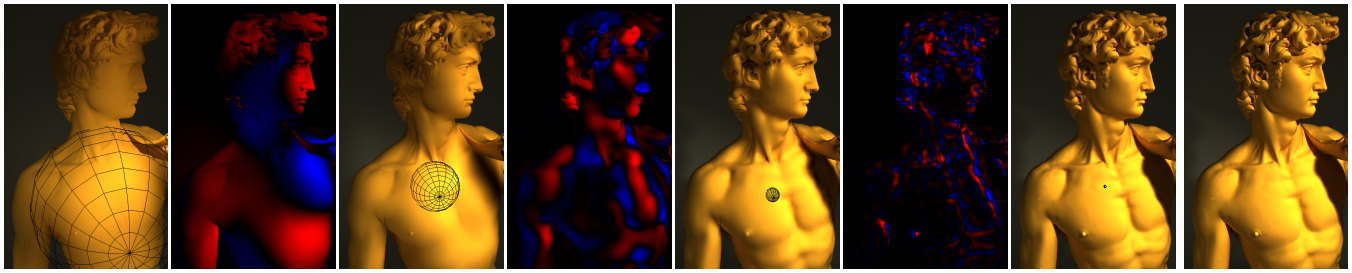
$$B_j(p) := \frac{w_j(p)}{\sum_{k=1}^{N_0} w_k(p)}, \tag{4}$$

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

$$F_0(p) = \sum_{j=1}^{N_0} B_j(p) f_j, \tag{5}$$

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. At the points of each finer level, we approximate the *difference* between the approximant from the previous levels and the function values at the new



F_0 $\tilde{f}_0$ F_2 $\tilde{f}_2$ F_4 $\tilde{f}_4$ F_6 Path traced

Figure 3: An illustration of hierarchical meshless approximation. The approximation F_0 is reconstructed from the coarsest level basis functions; the black sphere shows the support of one function. The difference images $\tilde{f}_{0,2,4}$ (red denotes positive and blue negative) and reconstructions $F_{2,4,6}$ show progressive refinement towards the path traced reference image. Levels 1,3,5 have been omitted due to space constraints. As can be seen, even the coarsest (and fastest) solutions show smooth behavior, unlike traditional finite element methods. The coarse basis functions do leak to areas that are in shadow but the finer levels counter the effect and crisp shadows appear.

points. For example, at level $i+1$, our complete approximation is

$$F_{i+1}(p) = F_i(p) + \tilde{f}_i(p), \quad \text{with}$$
$$\tilde{f}_i(p) = \sum_{j=N_i+1}^{N_{i+1}} B_j(p)\,\left(f_j - F_i(p_j)\right), \tag{6}$$

The function $\tilde{f}_i$ is the Shepard approximant of the difference between the ith-level approximation F_i and the function values at the points on level $i+1$. We can express the approximation at level i compactly as

$$F_i(p) = F_0(p) + \tilde{f}_0(p) + \ldots + \tilde{f}_{i-1}(p). \tag{7}$$

The basis functions for points on finer levels are defined analogously to Equation 4, i.e.,

$$B_j(p) := \frac{w_j(p)}{\sum_{k=N_{i-1}+1}^{N_i} w_k(p)}, \quad N_{i-1} < j \le N_i. \tag{8}$$

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

$$F_i(p) = \sum_{j=1}^{N_i} \alpha_j B_j(p), \tag{9}$$

where the α_j are defined as

$$\alpha_j = \begin{cases} f_j, & 0 < j \le N_0, \\ f_j - F_0(p_j), & N_0 < j \le N_1, \\ \vdots & \\ f_j - F_{L-1}(p_j), & N_{L-1} < j \le N_L. \end{cases} \tag{10}$$

A practical computation of the approximation coefficients α_j starts from the coarsest level, where the α_j correspond to the function values f_j at the points p_j. On each finer level i the coefficients encode the difference between the function values and the approximation F_{i-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.

Operator Notation We can also frame the above scattered data approximation procedure as a linear operator $\mathcal{P}$ acting on a function f that is defined everywhere in the domain. Applying $\mathcal{P}$ to f produces an approximation $\mathcal{P}f$ in the finite-dimensional function space spanned by our basis. Conceptually, the operator $\mathcal{P}$ may be viewed as a series of linear functionals or "evaluation stencils" $\delta_j(f)$ that "unroll" the dependence on coefficients from coarser levels in Equation 10. Each δ_j can be thought of as a function that computes a single coefficient α_j as a linear combination of point values of f, i.e., $\alpha_j = \delta_j(f)$. Derivation of explicit formulae is straightforward and we omit the details. The finite-dimensional approximation of f in our basis can then be written as

$$\mathcal{P}f = \sum_j \delta_j(f) B_j. \tag{11}$$

It is important to note that $\mathcal{P}$ is completely specified by the approximation points and weight functions, and does not depend on the function being approximated. Our $\mathcal{P}$ is not an exact projection. We could make $\mathcal{P}$ an interpolatory projection by using singular weights with $w_i(p_i) = \infty$, which leads to $B_i(p_j) = 0$ whenever $i \neq j$. This would enforce that $(\mathcal{P}f)(p_j) = p_j$, and thus $\mathcal{P}^2 = \mathcal{P}$. However, interpolations are very sensitive to noise in f_j, whereas approximations smooth the data slightly and are thus more forgiving visually.

4.2 Approximation Weights

In our context of light transport, the functions we want to approximate represent radiosity or radiance distributions on the surfaces of the scene. In addition to the surface location, these functions often depend strongly on the local surface orientation. To accommodate these requirements, our weighting combines a spatial fall-off kernel and a factor based on differences between normals.

Concretely, the points p_j include a position $x_j \in \mathbb{R}^3$ on a surface and the corresponding surface normal $n_j \in S^2$. Each point also carries a radius r_j, defined through k-nearest neighbor distance. This implies that an arbitrary point on the surface is covered by roughly kL basis functions when there are L levels in the hierarchy. Ideally k is set separately for each point according to the number of natural neighbors (i.e. Delaunay neighbors) the point has. Since we do not wish to mesh the points, we have chosen a safe value $k = 10$ that works well in practice. Using too small a number would lead to lack of support in some regions, and thus visible glitches in the resulting images. Too high a k would cause unnecessary low-pass filtering. The weight function is defined as

$$w_j(p) = K_j(\|x - x_j\|)\,\max(0, n \cdot n_j), \tag{12}$$

with

$$K_j(r) = K(\frac{r}{r_j}), \quad \text{and} \quad K(r) = \begin{cases} 2r^3 - 3r^2 + 1, & r \le 1, \\ 0 & r > 1. \end{cases} \tag{13}$$

The radially symmetric kernel K_j ramps smoothly from 1 at $r = 0$ to 0 at distance $r = r_j$, which implies that the corresponding basis function has compact support. 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 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 $\mathbb{R}^3 \times S^2$. While this may sound problematic, the finer levels will correct any leaks from coarser levels as illustrated in Figure 3.

5 Meshless Light Transport

We now have everything we need to realize the $\mathcal{PTP}$ discretization of the light transport operator given in Equation 2. First we will derive a meshless multilevel variant of radiosity, and then extend it to glossy surfaces. We describe the implementation and results of several algorithms based on our framework in Section 6.

5.1 Meshless Diffuse Transport

This section describes a meshless discretization of the diffuse transport operator. We choose to work with scalar incident irradiance instead of outgoing radiosity, since it is generally a smoother function due to possible high-frequency surface albedo variations [Gershbein et al. 1994]. Note that this choice is not dictated by our approximation framework.

Taking the abstract form (Equation 11) of the approximation operator $\mathcal{P}$ and expanding yields

$$\begin{aligned}\mathcal{PTP} &= \sum_j \delta_j (\mathcal{TP})\, B_j = \sum_j \delta_j \left(\mathcal{T} \left[\sum_k \delta_k(\cdot) B_k \right] \right) B_j \\ &= \sum_j \delta_j \left(\sum_k \delta_k(\cdot)(\mathcal{T}B_k) \right) B_j = \sum_j \sum_k \delta_k(\cdot)\, \delta_j (\mathcal{T}B_k)\, B_j \\ &= \sum_j \sum_k \delta_k(\cdot)\, \mathbf{T}_{jk} B_j, \quad (14)\end{aligned}$$

where the discretized transport operator $\mathbf{T}$ has entries $\mathbf{T}_{jk} := \delta_j(\mathcal{T}B_k)$, which are called *transport coefficients*. They describe the effect of sending a basis function B_k onto a receiving basis function B_j. The evaluation stencils δ_k are applied to the function being operated on; this produces the coefficients for the sending basis functions. Since, in principle, all basis functions are linked to all other basis functions, this is a so-called standard operator decomposition.

Computing the coefficients $\mathbf{T}_{jk}$ is simple. A single column of the matrix consists of the approximation coefficients of the function $\mathcal{T}B_k$, i.e., the irradiance cast onto the scene when basis function B_k is the sole emitter. Concretely, the single coefficient $\mathbf{T}_{jk}$ for a receiver j on the coarsest level is computed as

$$\mathbf{T}_{jk} = (\mathcal{T}B_k)(p_j) = \frac{1}{\pi} \int_{\Omega_{jk}} B_k(r(x_j,\omega))\, \rho(r(x_j,\omega))\, \lfloor \cos\theta \rfloor\, d\omega,$$

where $r(x,\omega)$ is the ray-casting function that returns the closest point to x in direction ω, Ω_{jk} is the solid angle subtended by the bounding sphere of B_k as seen from p_j, θ is the angle between n_j and ω, ρ is the diffuse reflectance, and $\lfloor\cdot\rfloor$ denotes clamp to zero from below.

For receivers j on finer levels, we also need the coefficients of j's parents[1] (for the same sender k) in order to perform the differencing according to Equation 10. This implies a natural coarse-to-fine refinement strategy. Clearly, the computation of coefficients requires the irradiance due to the sending basis functions only at a discrete set of points. Since the only integrals involved are over solid angles, our transport coefficients are independent of the underlying geometric representation – we never need to integrate over the surfaces.

[1] Basis functions on coarser levels that are non-zero at p_j.

The transport matrix $\mathbf{T}$ exhibits sparsity at the receivers analogously to operator discretizations in the usual hierarchical bases. Where the kernel of the transport operator $\mathcal{T}$ is smooth, the function $\mathcal{T}B_k$ is smooth with the consequence that many of its approximation coefficients are negligible. The resulting sparsity in indirect illumination is illustrated in Figure 4d. Sender sparsity is not directly visible in the form of negligible matrix entries, since our basis functions have non-zero integrals. However, since the finer sender functions carry details, i.e., local deviations from coarser approximations, the refinement oracle can skip refinement of a sender when smoothness is detected in the kernel. Analogously to wavelets, this leads to sparse operator representations (cf. Section 6.5).

5.2 Extension to Glossy Surfaces

The operator discretization in Equation 14 can be easily extended to capture directional variation in the radiance field. The spatially-scattered samples in the scene will then record coefficient vectors that define function expansions in terms of some directional basis, such as the spherical harmonics or spherical wavelets. This amounts to approximating the radiance function in a tensor product of a linear basis $\{\psi_i(\omega)\}_{i=1}^N$ for the directional variation and the spatial basis functions defined by our scattered approximation procedure. Concretely, a function $F(p,\omega)$ of both space and direction is approximated by

$$F(p,\omega) = \sum_i \sum_j \alpha_{ij}\, B_i(p)\, \psi_j(\omega). \quad (15)$$

The interpretation of α_{ij} is "the j:th directional coefficient associated with the spatial basis function B_i". To see how to compute the coefficients α_{ij} in Equation 15, let us first define a *directional projection operator* $\mathcal{P}_\psi$. This operator acts on a function $F(p,\omega)$ of space and direction. At each p it produces a vector of directional basis coefficients that approximate the directional variation of F at p, i.e.,

$$F(p,\omega) \approx \mathcal{P}_\psi F = \sum_j \alpha_j(p)\, \psi_j(\omega). \quad (16)$$

The output of $\mathcal{P}_\psi$ is thus a vector-valued function of p; the operator discretizes only the directional variation of F by "flattening" it into a finite-dimensional basis. To discretize also the spatial variation, we apply our scattered approximation operator $\mathcal{P}$ from the previous section to the vector-valued function $\mathcal{P}_\psi f$. This means computing the directional coefficient vectors at the spatial sample points p_j, and passing each of the components of these vectors through the scalar approximation framework described earlier.

A glossy transport operator $\mathcal{T}$ may be discretized using the above projector in a way exactly analogous to Equation 14, with the basis functions B_j replaced by the tensor product basis $B_i\,\psi_j$, and the projector $\mathcal{P}$ replaced by $\mathcal{P}\mathcal{P}_\psi$. Performing the same algebra results in the discretized transport operator $\mathbf{T}$ that has the entries

$$\mathbf{T}_{(ij)(kl)} := \delta_i \left(\mathcal{P}_\psi (\mathcal{T} B_k\, \psi_l) \right). \quad (17)$$

The index j refers to the jth component of the vector $\mathcal{P}_\psi(\mathcal{T}B_k\,\psi_l)$. These transport coefficients describe the effect of sending the basis function pair $B_k\,\psi_l$ to the receiving basis function pair $B_i\,\psi_j$.

It is not strictly necessary that the operator $\mathcal{P}_\psi$ produces projection coefficients in the rigorous meaning. For instance, while *vector irradiance* [Willmott et al. 1999] is not a basis projection, it can often meaningfully describe directional variation in the lighting incident onto a diffuse surface. Vector irradiance can be computed from the incident illumination using a special $\mathcal{P}_\psi$. It is also possible to discretize the senders and receivers using different projections $\mathcal{P}_\psi$. We will exercise this freedom in our indirect PRT technique in Section 6.5.

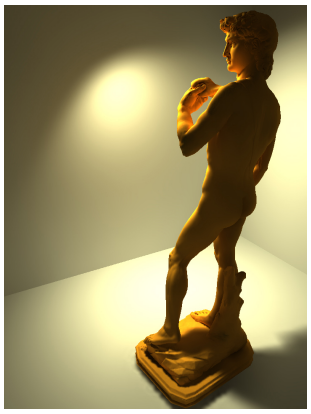
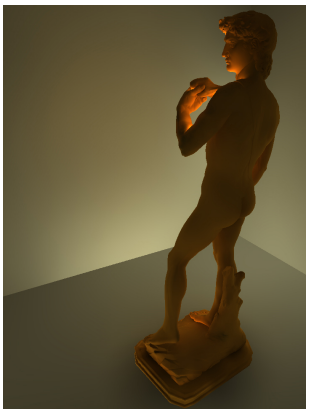
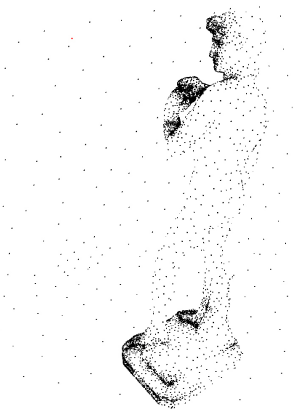
a)  b)  c)  d) 

Figure 4: Michelangelo's David. a) A meshless radiosity rendering of the statue. Note the complete absence of spurious discontinuities in the solution. c) The indirect component of the lighting. b) and d) The corresponding sparsity visualizations. The black points indicate non-zero coefficients, i.e., only their basis functions were touched during the radiosity simulation.

6 Applications and Results

We now demonstrate three applications of meshless light transport: hierarchical radiosity, glossy pre-computed radiance transfer from distant illumination, and indirect diffuse precomputed transport for local light sources. Before diving into particular algorithms, we will explain the point placement algorithm used in the tests, and briefly discuss the GPU renderer that was used for visualizing the results. All the tests were run on a PC with 2.4GHz Intel Core 2 Duo, NVIDIA GeForce 8800GTX, and 2GB of physical memory. All images use a linear tone map, and indirect illumination has been exaggerated for better visualization.

6.1 Point Placement

Our simple point placement algorithm uses rejection sampling to construct a hierarchical Poisson disk distribution on the surfaces of the triangles. Each point on the surface has an equal chance of getting proposed.

Our Poisson disk acceptance criterion uses Euclidean distances in $\mathbb{R}^3$. We modify the usual criterion so that the disk radius is scaled by the dot product of the normals. This has the desirable effect that two points can be arbitrarily close to each other if there is a ≥ 90 degree corner between them. The hierarchical construction begins with an initial disk radius R. We add new points to the coarsest level until the rejection sampler fails 1000 times in a row. Then for each remaining level we halve R and repeat the process. Note that the acceptance test also takes into account all the samples already created in the coarser levels. The initial radius and the number of levels are user-specified constants.

The point placement function is called separately for each *reconstruction group*. For example, Figure 1 has three groups: the cathedral, Lucy, and David. This allows us to specify a different point density for each group, and also prevents basis functions leaking between groups.

Additionally, we use a heuristic to deal with sub-optimally constructed geometry. A ray is cast from a proposed point to the direction of the normal vector. If the ray hits a backface, we conclude that the proposed point is inside geometry and thus the point should be rejected. Our point placement equipped with this simple heuristic can handle many typical geometric problems, such as holes, double polygons, and simple interpenetrations, but obviously cannot fix all scenarios, e.g. heavily interpenetrating geometry. More sophisticated placement methods are a potential avenue for future work.

Finally we store each level into a separate R-Tree [Guttman 1984] (basically a bounding volume hierarchy) in order to quickly find the non-zero basis functions during light transport simulation.

6.2 GPU Renderer

Our GPU renderer visualizes the approximations directly from the hierarchical representation. The implementation resembles earlier GPU splatting methods, e.g., deferred splatting [Guennebaud et al. 2004] or 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 of our hierarchical representation, all the spatial basis functions of that level are passed to pixel shaders as quads whose size is conservatively determined from the projection of the function's bounding sphere. The shaders then evaluate the weight according Equation 12 for each covered pixel, and the weights and weighted approximation coefficients are accumulated into separate surfaces. The function's depth is also conservatively estimated from the bounding sphere, and special care needs to be taken to avoid accidental near clipping. Whenever a bounding sphere intersects the viewport, we render the corresponding function using a screen-sized quad. After a level has been rendered, the accumulated coefficients are normalized using the accumulated weights, and the result is added to the final image. Optionally, the surface albedos need to be multiplied to the final image in order to convert incident irradiance to the final color.

The overall performance of the GPU renderer is limited by the accumulation of basis functions because the amount of overlap is fairly high (cf. Section 4.2).

6.3 Hierarchical Radiosity Implementation

We have implemented a progressive radiosity solver using our hierarchical meshless machinery. We chose a simple scalar radiosity algorithm to make it easy to judge the quality of the results. Both the direct and indirect lighting are represented in our hierarchical basis. We stress that this algorithm is only one possible choice and not dictated by our framework in any way.

The discretized radiosity equation that we solve is

$$\mathcal{P}b = \mathcal{P}\mathcal{T}\mathcal{P}b + \mathcal{P}e \quad \Leftrightarrow \quad \mathbf{b} = \mathbf{T}\mathbf{b} + \mathbf{e},$$

where b is the unknown solution and e is the basis expansion of direct illumination, $\mathbf{b}$ and $\mathbf{e}$ are their approximation coefficient vectors, and $\mathbf{T}$ is the discretized transport operator as defined in Section 5.1. We solve for $\mathbf{b}$ by evaluating the Neumann series, i.e., computing the lighting bounce by bounce, starting from emission:

1. $\mathbf{e} \leftarrow$ approximation coefficients of direct lighting, Eq. 10
2. $\mathbf{i}_0 \leftarrow \mathbf{e}, \quad \mathbf{b} \leftarrow \mathbf{e}$
3. **for each** bounce $j = 0$ to *MaxBounces*
4. $\quad \mathbf{i}_{j+1} \leftarrow \mathbf{T}\,\mathbf{i}_j$ // execute COMPUTE-BOUNCE
5. $\quad \mathbf{b} \leftarrow \mathbf{b} + \mathbf{i}_{j+1}$ // accumulate j:th bounce to solution

The vectors $\mathbf{i}_j$ denote $\mathbf{T}^j\mathbf{e}$, i.e., the approximation coefficients of the j:th bounce of light. The initial shooting stage on Line 1 emits the direct lighting to the scene. The COMPUTE-BOUNCE procedure on Line 4 takes the coefficients for the previous bounce as input, and produces new coefficients. The procedure starts by gathering the irradiance from the previous bounce onto the coarsest-level receiving basis functions (see explanation of GATHER below). The coarse functions are pushed onto a queue, which is used for adaptively refining the receivers where significant variation is detected by RECEIVER-ORACLE (explained below). The algorithm iterates until no refinement is deemed necessary.

The procedure GATHER works by estimating the hemispherical irradiance at the receiver by shooting a fixed number (512) of rays into the hemisphere above the receiver. The rays are in a fixed pattern drawn from a Hammersley sequence, rotated at each sample a random angle around the normal vector. At the points where the rays hit, we evaluate the irradiance from the basis functions using coefficients of the previous bounce. The level at which this reconstruction occurs is chosen adaptively by looking at the solid angles subtended by the bounding spheres of the basis functions at the hit point as seen from the receiver. This choice of sender level is similar in spirit to the hierarchical Monte Carlo radiosity technique of Bekaert et al. [1998]. The reconstruction is stopped on the earliest level where the average solid angle subtended by the bounding spheres fall below a threshold ε_Ω.

The function RECEIVER-ORACLE determines whether or not a receiver basis function B_k requires refinement into its children (basis functions on the next-finer level that are non-zero at p_k). On the coarsest level, the irradiance at p_k is compared to the irradiance at the neighboring (the ones that are non-zero at p_k) coarse basis functions. Variation above a threshold ε_E triggers refinement. On finer levels, the refinement is triggered solely according to the magnitude of the coefficient $\mathbf{i}_{j+1}(k)$, i.e., a high-magnitude is estimated to mean that there are details in the neighborhood that need to be resolved. The thresholds used in our tests are given in Table 1.

At finer receiver levels the coefficient computation first gathers irradiance at the receiver from the whole hemisphere using GATHER. It then reconstructs an irradiance value from parent basis functions in order to be able to perform the differencing that results in the final basis coefficient. Since the parent-child relationships of our basis functions do not form a tree, this function needs to check that the coefficients for all the parent basis functions have been computed already. If this is not the case, the function recursively computes the parent coefficients first. In the end, the newly-computed basis function is pushed onto the refinement queue.

Results We have tested our implementation in three scenes, shown in Figures 5 and 3. The statistics from the scenes are listed in Table 1. The images show that our technique produces high-quality results in scenes featuring geometry that would be difficult for traditional radiosity methods. Our method requires no pre-processing of the statues and the cathedral, and no post-processing of the result. Figure 3 demonstrates that our solution indeed visually converges towards a path traced reference. Both direct and indirect lighting are represented in our basis. As the refinement of the receivers proceeds from coarse to fine, the algorithm produces coarse solutions in a matter of seconds, which then get progressively refined. Most of the time (80-90%) is spent in evaluating the basis functions. In Sibenik, the kd-tree used by the ray tracer corresponds to 70% of the memory consumption, whereas in the more accurate David two thirds of the memory goes to storing the point hierarchy. We decided to use face cluster radiosity [Willmott et al. 1999] as a comparison method, because it is reasonably modern and Willmott's

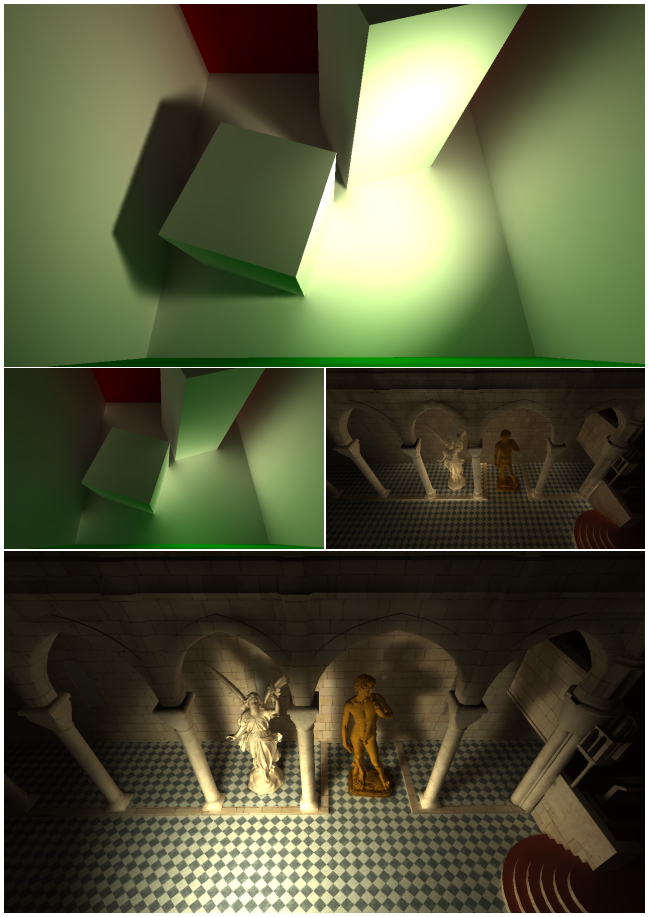


Figure 5: Radiosity images from two test scenes. Top: Cornell box final image. Bottom: Sibenik final image. Middle: the corresponding indirect-only images.

Figure 6: A comparison of our meshless solution (left) with face cluster radiosity (right). Equal time (3min) was given to both algorithms. The FCR image has more detail because it uses vector irradiance instead of scalar irradiance as our implementation. However, the FCR clusters are clearly visible on the shoulder, whereas our method does not suffer from any unwanted discontinuities.

Scene	#Triangles	#Samples generated	#Samples gathered to	#Visibility rays	#Hierarchy levels	#Samples at coarsest level	ε_E	ε_ω	Timing breakdown				Memory consumption
									Basis func. evaluation	Visibility rays	Misc	**Total time (s)**	
Sibenik	1.2M	583k	28k	19.2M	6	1595	0.04	$\pi/10$	266.9	45.8	5.6	318.3	116.3 MB
David	617k	1.6M	45k	25.9M	7	629	0.04	$\pi/10$	313.6	53.2	8.2	375.0	153.9 MB
Cornell	32	252k	10k	7.8M	6	310	0.01	$\pi/10$	65.9	3.6	2.1	71.6	19.0 MB

Table 1: Statistics from three radiosity test scenes.

implementation is publicly available[2]. Figure 6 shows an image quality comparison between our method and face cluster radiosity.

Discussion Our algorithm has some similarities with point-based methods (e.g. [Ward et al. 1988; Jensen 1996; Walter et al. 2005; Walter et al. 2006; Hašan et al. 2006]) that have no clear connection to finite elements. The most significant difference between our method and these point-based approaches is that our points represent basis functions. Therefore the lighting function is properly defined, not just at the points, but everywhere within the support of the basis functions. Compared to irradiance caching [Ward et al. 1988] our hierarchical refinement is based on the values of the function, whereas Ward inserts new points based on a heuristic that does not depend on the function values. Contrary to photon mapping [Jensen 1996] we do not need density estimation at any stage, and thus avoid the "boundary bias" issues. These techniques are also not hierarchical. Multidimensional lightcuts [Walter et al. 2006] is probably the closest to our technique in the sense that, like ours, the technique is hierarchical with respect to both senders and receivers, and also refines according to the function values. We believe it could be useful to combine their sophisticated oracle with our meshless finite element framework.

6.4 Glossy PRT from Distant Illumination

Precomputed radiance transfer methods, e.g. the method by Sloan [2002], allow interactive rendering of static scenes with dynamically changing, non-pointlike illumination with soft shadows and interreflections. This is achieved by parameterizing the emission of light in some low-dimensional linear space, and pre-computing lighting solutions that correspond to the degrees-of-freedom of this "emission space". Most of these methods are in fact finite element techniques where the rendering equation has been augmented by an operator that accounts for the constrained emissions [Lehtinen 2004]. Hence, our framework is directly applicable to precomputed transfer techniques.

As a proof of concept, we have implemented a meshless variant of the glossy PRT technique of Sloan et al. [2002], where a scene is lit by low-frequency distant illumination parameterized using 25-term Spherical Harmonics. Here we make use of the glossy operator discretization from Section 5.2. For any point on the surface, its *transfer matrix* $\mathbf{T}_p$ captures the effects of occlusions, interreflections and subsurface scattering on the light emitted by the spherical harmonics. We approximate the transfer matrices in our basis, i.e., finer levels now encode matrix differences. We leverage the multilevel nature of our basis by computing the transfer matrices adaptively, much like the RECEIVER-ORACLE procedure described in Section 6.3. We first compute matrices for the coarsest points. Then we examine the variation between the matrices of neighboring points; a variation in a matrix norm (we use $\|\cdot\|_1$) above a threshold $\varepsilon_\mathbf{T}$ triggers refinement. The refinement decision on finer levels is based on the matrix norm of the point alone.

When rendering a frame, we compute the approximation coefficients of outgoing radiance for each point (using CPU for ease of implementation). This amounts to computing $\mathbf{T}_p\mathbf{l}$ for each point and performing the final reflection, for which we employ the method of Kautz et al. [2002]. The coefficients that represent outgoing radiance are then fed into our GPU renderer.

[2]http://www.cs.cmu.edu/∼ajw/thesis-code/

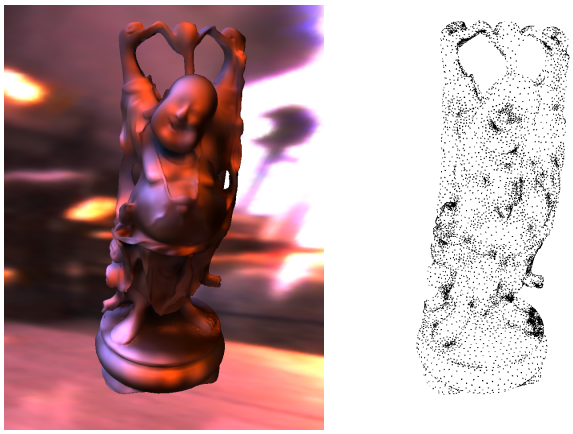


Figure 7: Left: A glossy PRT solution computed using meshless glossy operator discretization. Right: The black dots show where the transfer matrices were computed.

Results. Figure 7a shows a glossy Buddha model in the Grace cathedral lighting environment, rendered at 10.6 FPS in 720×960 resolution with interactive view and lighting. The model has 100k triangles and 50k vertices. Fig. 7b shows the sample points at which the adaptive precomputation chose to compute the transfer matrices. $\varepsilon_\mathbf{T}$ was set to 0.7, and the total number of matrices computed was 16270 (roughly one third of the number of vertices in the model), totaling at 116 MB. The precomputation took 8min 18s with 3 indirect bounces of lighting.

Discussion Conventional PRT techniques first compute a dense set of transfer matrices, typically one for each vertex of a mesh, and then compress them. In contrast, our method saves time and memory by adaptively refining the PRT solution during precomputation. In addition, our approach is independent of the tessellation of the meshes. It is an interesting avenue for future research to combine our technique with advanced compression schemes, e.g., the one proposed by Sloan et al. [2003].

6.5 Indirect Diffuse PRT for Local Light Sources

After the original radiosity technique of Goral et al. [1984], virtually all finite element global illumination techniques have abandoned the computation of the full matrix inverse that gives the illumination solution. Instead, these methods work with a known emission and an iterative solution scheme, similar to our radiosity technique in Section 6.3. However, two recent techniques [Kristensen et al. 2005; Kontkanen et al. 2006] turn the quickly evaluated direct lighting into indirect lighting using a precomputed operator. This allows real-time rendering of indirect illumination in static scenes with dynamic, local light sources and arbitrary viewpoints.

As a demonstration, we have implemented a meshless variant of the hierarchical technique of Kontkanen et al. We compute transport links that include multiple indirect bounces from sending basis functions onto receivers, so that unit scalar incident irradiance at the sender is turned into vector irradiance at the receiver. At runtime,

Scene	#Tris	#Links	#Rays	Mem. consump.	Precomp. time	Runtime FPS
Sibenik	1.2M	2.3M (0.5%)	234M	103 MB	31min 32s	7.2
Thinker	681k	668k (0.3%)	66M	30 MB	8min 12s	10.3

Table 2: Statistics from two indirect diffuse PRT test scenes.

the basis coefficients of direct scalar irradiance are determined by casting rays from the light source to the sending basis functions. Then, the precomputed transport operator is applied to these coefficients by following links from the senders, and accumulating the transported light onto the receivers. The result describes indirect illumination on the scene. These coefficients are then fed to the GPU renderer for visualization.

The precomputation stage of our algorithm is similar to that of Kontkanen et al. First, a sparse representation of the one-bounce operator $\mathbf{T}$ is computed by adaptive refinement; this step is basically equivalent to a hierarchical radiosity method that does not know the emission beforehand. Subsequent bounces are computed by evaluating the Neumann series $\mathbf{T}+\mathbf{T}^2+\mathbf{T}^3+\ldots$ of the powers of the one-bounce operator as described by Kontkanen et al.

The computation of the one-bounce operator proceeds as follows. For each coarse receiver, we compute initial links to coarse senders using a hemispherical gather. The links are then inspected one at a time. A refinement oracle decides whether or not the interaction must be refined, and at which end (sender or receiver). The refinement then proceeds recursively. Our oracle measures both the magnitude of the link, and the variation in the kernel function of $\mathcal{T}$ over the children of both the sending and receiving basis functions, so that deviation larger than a threshold triggers refinement. This procedure resembles the wavelet radiosity oracle of Gortler et al. [1993]. We limit the level onto which the senders can be refined because, after all, it is the receivers whose result is visualized directly on screen and therefore we can get away with reduced number of senders with little visual impact. We compute the coefficients for the senders by averaging 10 rays to avoid binary visibility transitions that would cause artifacts due to the small number of senders.

Results Table 2 gives statistics for two test scenes, Sibenik and Thinker. Figures 1 and 8 show screenshots from interactive visualization. Only 0.3–0.5% of the potential links were computed in the precomputation step, and that explains the fast precomputation times compared to previous methods [Kristensen et al. 2005; Kontkanen et al. 2006]. The precomputation simulated three bounces of light. The runtime memory consumption was 103 MB for the 1.2M triangle Sibenik, and the GPU renderer ran 7.2 FPS at 960×720 output resolution. Roughly half of the 122k basis functions were rendered per frame. The light source and viewpoint can be freely moved at runtime.

Discussion Our meshless approach shows all its advantages in this application. Compared to Kristensen et al. [2005], we achieve significantly faster precomputation by using an adaptive, hierarchical scheme, which avoids considering the link between every pair of basis functions. 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 complex geometry since we do not rely on wavelet parameterization of the surfaces of the scene. In comparison, Hašan et al. [2006] present an image relighting technique that also precomputes direct-to-indirect transport due to local light sources. They can produce compelling images with complex final-bounce reflections; however, this comes at the cost of a fixed viewpoint. Furthermore, their emission space consists of point sources (not basis functions) and they do not utilize hierarchies in the precomputation step.

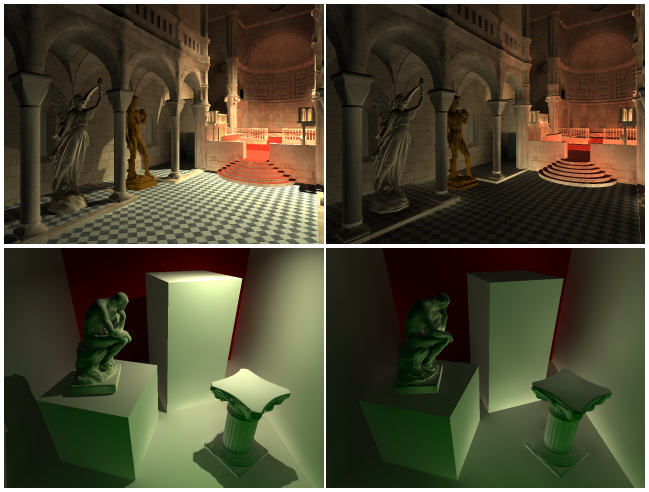


Figure 8: Screenshots from indirect diffuse PRT. Top: final and indirect-only images from Sibenik. Bottom: final and indirect-only from Thinker. Per-pixel direct lighting was rendered using shadow maps.

7 Conclusions and Future Work

We have presented a meshless finite element framework for hierarchical global illumination computations, and demonstrated simple and efficient algorithms that avoid many of the problems associated with traditional finite element methods by decoupling lighting from geometry. Our prototypes leave ample room for improvement and open up a number of exciting directions for future research. We believe that our general framework will enable the formulation of many more, highly versatile global illumination methods. For example, generalization to participating media should be easy.

Limitations As the supports of the coarse basis functions are large, they may leak to unwanted locations, for example, through a wall between adjacent rooms. The hierarchical solver will subsequently have to do extra work to counter the effect. Also, the surfaces should have consistently oriented normal vectors (e.g. no random flipping of triangles), because we use normals in our weighting functions. Furthermore, our smooth basis functions cannot exactly represent sharp discontinuities. While the normal term in Equation 12 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 exactly counter the coarse functions leaking over the discontinuity. It could be possible to enhance our approximation procedure with techniques that better obey discontinuities, e.g., bilateral filtering [Tomasi and Manduchi 1998]. However, a realistic application would probably render direct lighting using other methods and represent only the smooth indirect component of illumination in our basis – a common separation in all kinds of illumination algorithms.

Future Work Our weighting functions are simple and reasonably general, but we believe that it would be possible to define improved functions with smarter approximation behavior. Currently, the functions are radially symmetric. It would be interesting to investigate functions with anisotropic support that better follow the surface geometry.

Our proof-of-concept point placement could be further improved. For example, for scenes that have a lot of (disjoint) small-scale detail, such as plants or pebbles, the placement of points should not be restricted to surfaces. At least the points on coarser levels could be offset from the surface to better utilize coherence. We currently generate all points in a preprocess. Generating ad-

ditional points on demand could be useful for refining the lighting function in areas with high frequencies, such as shadow boundaries.

Our current GPU renderer visualizes the hierarchical representations directly. This approach would allow a straightforward implementation of adaptive level-of-detail rendering of the lighting. If that freedom is not desired, the runtime performance could be improved by abandoning the hierarchical representation after the approximation coefficients have been computed. One would keep the same set of points but flatten the hierarchy so that each point would store the function values instead of differences. Alternatively, the lighting could be easily resampled to textures, for instance.

Acknowledgments

The authors thank Tomas Akenine-Möller, Michael Garland, Nicolas Holzschuch, Andy Nealen, Jussi Räsänen, and Lauri Savioja for early reviews and comments; the Stanford Computer Graphics Laboratory for the statue models; Marko Dabrovic (www.rna.hr) for the Sibenik cathedral model; Olli Pelz-Hänninen for help on the cathedral model. Thinker model courtesy of the MIT CSAIL database. This research has been funded by Jaakko's generous NVIDIA Fellowship, the National Technology Agency of Finland, Anima Vitae, ATI Research Finland, Hybrid Graphics/NVIDIA Helsinki, Remedy Entertainment, and the Academy of Finland (Grant# 108 222).

References

AMIDROR, I. 2002. Scattered data interpolation methods for electronic imaging systems: a survey. *J Electr. Imag. 11*, 2, 157–176.

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

BEKAERT, P., NEUMANN, L., NEUMANN, A., SBERT, M., AND WILLEMS, Y. D. 1998. Hierarchical monte carlo radiosity. In *Proc. 9th Eurographics Workshop on Rendering*, 259–268.

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.

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.

FASSHAUER, G. E. 2007. *Meshfree Methods*. ch. Handbook of Theoretical and Computational Nanotechnology, *to appear*.

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

GERSHBEIN, R., SCHRÖDER, P., AND HANRAHAN, P. 1994. Textures and radiosity: Controlling emission and reflection with texture maps. In *Proc. SIGGRAPH 94*, 51–58.

GOBBETTI, E., SPANÒ, L., AND AGUS, M. 2003. Hierarchical higher order face cluster radiosity for global illumination walkthroughs of complex non-diffuse environments. *Comp. Graph. Forum 22*, 3, 563–572.

GORAL, C. M., TORRANCE, K. E., GREENBERG, D. P., AND BATTAILE, B. 1984. Modeling the interaction of light between diffuse surfaces. In *Computer Graphics (Proc. SIGGRAPH 84)*, 213–222.

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 Symposium in Interactive 3D Graphics*.

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. *Thermohysics and Heat Transfer 20*, 1, 150–154.

HOLZSCHUCH, N., CUNY, F., AND ALONSO, L. 2000. Wavelet radiosity on arbitrary planar surfaces. In *Proc. 11th Eurographics Workshop on Rendering*, 161–172.

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

KAJIYA, J. T. 1986. The rendering equation. In *Computer Graphics (Proc. SIGGRAPH 86)*, 143–150.

KAUTZ, J., SLOAN, P.-P., AND SNYDER, J. 2002. Fast, arbitrary brdf shading for low-frequency lighting using spherical harmonics. In *Proc. Eurographcis Workshop on Rendering 2002*, 301–308.

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.

KRIVANEK, 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.

LANCASTER, P., AND SALKAUSKAS, K. 1981. Surfaces generated by moving least squares methods. *Math. Comput.*, 37, 141–158.

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

LEHTINEN, J. 2004. *Foundations of Precomputed Radiance Transfer*. Master's thesis, Helsinki University of Technology.

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

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.

PFISTER, H., ZWICKER, M., VAN BAAR, J., AND GROSS, M. 2000. Surfels: surface elements as rendering primitives. In *Proc. SIGGRAPH 2000*, 335–342.

SCHRÖDER, P., GORTLER, S. J., COHEN, M., AND HANRAHAN, P. 1994. Wavelet projections for radiosity. *Comp. Graph. Forum 13*, 2, 141–151.

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

SILLION, F., ARVO, J., WESTIN, S., AND GREENBERG, D. P. 1991. A global illumination solution for general reflectance distributions. In *Computer Graphics (Proc. SIGGRAPH 91)*, 187–196.

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.

TOMASI, C., AND MANDUCHI, R. 1998. Bilateral filtering for gray and color images. In *Proc. ICCV 98*, 839–846.

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. 10th Eurographics Workshop on Rendering 99*.

ZATZ, H. R. 1993. Galerkin radiosity: a higher order solution method for global illumination. In *Proc. SIGGRAPH 93*, 213–220.

ZHANG, X., SONG, K. Z., LU, M. W., AND LIU, X. 2000. Meshless methods based on collocation with radial basis functions. *Computational Mechanics 26*, 4, 333–343.