EUROGRAPHICS 2003 / P. Brunet and D. Fellner
(Guest Editors)

Volume 22 (*2003*), *Number 3*

Hierarchical Higher Order Face Cluster Radiosity for Global Illumination Walkthroughs of Complex Non-Diffuse Environments

Enrico Gobbetti, Leonardo Spanò and Marco Agus

CRS4, Visual Computing Group, VI Strada Ovest, Z.I. Macchiareddu, C.P. 94, I-09010 Uta (CA), Italy
{gobbetti|leonardo|magus}@crs4.it – http://www.crs4.it/vic

Abstract

We present an algorithm for simulating global illumination in scenes composed of highly tessellated objects with diffuse or moderately glossy reflectance. The solution method is a higher order extension of the face cluster radiosity technique. It combines face clustering, multiresolution visibility, vector radiosity, and higher order bases with a modified progressive shooting iteration to rapidly produce visually continuous solutions with limited memory requirements. The output of the method is a vector irradiance map that partitions input models into areas where global illumination is well approximated using the selected basis. The programming capabilities of modern commodity graphics architectures are exploited to render illuminated models directly from the vector irradiance map, exploiting hardware acceleration for approximating view dependent illumination during interactive walkthroughs. Using this algorithm, visually compelling global illumination solutions for scenes of over one million input polygons can be computed in minutes and examined interactively on common graphics personal computers.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture and Image Generation; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism.

1. Introduction

Interactive computer graphics systems for realistically visualizing geometrically complex models are essential components of a number of practical applications, including evaluation, design, and training systems. The subtle lighting effects simulated by global illumination are of primary importance to make the virtual world representation look realistic. Finite element methods have established themselves in the industry as one of the methods of choice for simulating global illumination, mainly because their results are well suited, at least in the standard diffuse case, for interactive inspection in virtual reality simulators [1]. The ability to perform interactive walkthroughs of global illumination solutions including large tessellated models and glossy effects remains, however, a challenging open problem.

In this paper, we present a higher order extension of the face cluster radiosity technique that overcomes certain limitations of previous approaches. It combines face clustering, multiresolution visibility, vector radiosity, and higher order bases with a modified progressive shooting iteration to rapidly produce visually continuous solutions with limited memory requirements. In particular, since the method focuses on smoothly representing vector irradiance rather than radiosity, its memory and time complexity are practically independent from the input model size. The output of the method is a vector irradiance map that partitions input models into areas where global illumination is well approximated using the selected basis. The programming capabilities of modern commodity graphics architectures are exploited to render illuminated models directly from the vector irradiance map, exploiting hardware acceleration for computing view dependent illumination during interactive walkthroughs. Using this algorithm, global illumination solutions for scenes of over one million input polygons can be computed in minutes and examined interactively on common graphics personal computers.

The rest of the paper is organized as follows. An overview of the related work is presented in section 2. Then, hierarchical higher order face cluster radiosity is introduced in section 3. Section 4 discusses our prototype implementation and the preliminary results obtained. The paper concludes with a summary and a view of current and future work.

2. Related work

Volume clustering for hierarchical radiosity. The most successful radiosity technique for dealing with complex scenes is currently hierarchical radiosity [2]. The algorithm constructs a hierarchical representation of the form factor matrix by adaptively subdividing planar patches according to a user-supplied error bound. By treating interactions between distant patches at a coarser level than those between nearby patches, the algorithm reduces the cost from quadratic to linear in the number of sub-patches used. However, since an initial transport link has to be computed from each of the original patches to all others, the cost is also quadratic in the number of input polygons, which is the major bottleneck for highly tessellated scenes. Volume clustering methods [3, 2, 4] combat this problem by grouping input patches into volume clusters. Handling the light incident on a cluster is, however, a difficult problem and all presented solutions are more suitable to handling unorganized sets of polygons than highly tessellated models [5, 6, 7]. It is difficult to obtain continuously shaded surfaces, since interpolating scalar irradiances across volumes does not lead to good results because of the varying orientations of surfaces within the cluster [6]. At the same time, pushing irradiances to leaves on-the-fly [8, 3, 9] makes it difficult to construct higher order representations of polygon irradiances, makes the method complexity dependent on input model size, and drastically reduces the memory locality of the solution phase.

Hierarchical radiosity on simplified and multiresolution models. Mesh simplification techniques can be adopted to structure the data at different levels of detail [10]. A number of authors have recognized the potential of these techniques for handling large tessellated surfaces in radiosity. Rushmeier et al. [11] demonstrated the use of simplified models in radiosity. Greger et al. [10], showed how to apply the results of a simulation on a simplified scene to a more detailed version of the same scene through the use of irradiance volumes. Both methods force the user to select the complexity of the model before the simulation. Dumont and Bouatouch [12] recently improved this result, presenting a hierarchical radiosity algorithm that works on multiresolution meshes, picking the level of simplification appropriate to each transfer of radiosity between solution elements through the use of macrofacets. However, their technique requires touching all the input polygons during the push phase, which is prohibitively memory and time expensive for models where the geometric detail is much finer than the illumination complexity. Willmott and Heckbert [5] presented a hierarchical radiosity algorithm that focuses on vector irradiance rather than radiosity. Since vector irradiance conserves directional information, the push-to-leaves phase is avoided, and the method memory and time complexity are made independent from the input mesh complexity. The method is currently limited to handling a single irradiance vector per cluster, which leads to "blocky" solutions or fine subdivisions. As for volume clusters, the classic smoothing post-pass is difficult to apply, and re-evaluating visibility at the input polygon level is prohibitively expensive for highly tessellated scenes. For this reason, Willmott [13] proposes a final post-processing stage in which irradiance vectors are recomputed at the corners of each node throughout the hierarchy and interpolated at each input model vertex for computing radiosity. Our work improves over this method by using higher order bases during the solution, leading to better error control and reduced refinement. The virtual mesh approach [14] is similar to ours, as it tries to decouple illumination information from surface representation by using geometric abstractions. Their work, however, focuses mainly on parametric surfaces, while ours is more targeted to highly tessellated models.

Linkless hierarchical radiosity. A major problem with classic hierarchical radiosity methods is the necessity to store all links because all of them are reused in each gathering iteration, which imposes a considerable overhead that limits the size of scenes that can be handled by the method. This is a particularly severe problem with higher order techniques, since the number of coefficients per link grows with the square of the number of element basis functions. For this reason, a number of authors have proposed a shooting iteration scheme together with hierarchical radiosity with clustering [15, 16, 17, 18, 14]. Since the number of shooting links for every iteration decreases exponentially, the penalty for not storing, but recomputing some of the links is much smaller than it is in the case of gathering. We also adopt this approach, and propose a modified shooting scheme that also reduces storage requirements at the element level. Particle tracing and density estimation approaches [19, 20] tackle the complexity problems by separating the global transport from the local lighting reconstruction. They are an alternative to the finite element techniques discussed here, that potentially provide a more accurate simulation at the expense of walkthrough speed.

Non–diffuse global illumination and interactive walkthroughs. The radiosity method can be used to precompute view independent radiosity values which can be viewed interactively. Extensions have been proposed for view dependent illumination in the early 90s [21, 22], but they did not provide interactive rates. Recently, interactive walkthroughs of global illumination have become possible by caching of ray-traced samples [23, 24, 25, 26] or by efficiently updating partially precomputed radiosity solutions [27, 28, 29]. These approaches are very promising, but visualization of results still requires considerable CPU effort, especially when the view changes rapidly. They are therefore currently more applicable to interactive lighting design applications than walkthroughs of large tessellated scenes. Alternatively, an approximate representation of the global illumination can be precomputed and viewed interactively exploiting graphics hardware acceleration, as in, e.g., [30, 31, 32, 28, 33]. We also adopt this approach,

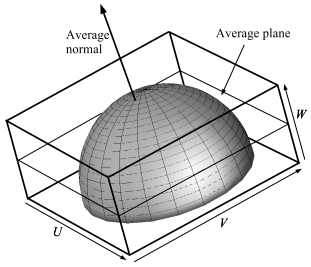

Figure 1: *Face cluster.*

proposing an approximation of glossy global illumination well suited to current programmable graphics hardware.

3. Hierarchical Higher Order Face Cluster Radiosity

3.1. Hierarchical Data Structure

As in the original face cluster radiosity algorithm [5], highly tessellated geometric models are represented with a face cluster hierarchy that has the original model polygons as leaves. Each cluster in the hierarchy groups a set of connected faces and behaves like a geometric object on its own, answering queries regarding its geometry (e.g. bounding volume, normal, total area, projected area) and attributes (e.g. reflectance, emission). Currently, each face cluster is represented by an oriented bounding box (see figure 1) with the local z axis aligned with the area averaged normal of the contained surface and the x and y axis assigned by a rotating caliper algorithm that minimizes the box volume. Hierarchy construction is done in a preprocessing step on an object by object basis using a greedy algorithm based on the method of Garland et al. [34] that we have extended to handle vertex attributes as in our earlier simplification tool [35] .

Since face clusters do not in general represent planar surfaces with constant material attributes, all queries return average, minimum, and maximum expected values. In particular, we employ the following expressions, due to Willmott [13]:

- Normal-projected area of cluster i: $A_i^{(n)} = \|\sum_k A_k \mathbf{n}_k\|$ where A_k is surface area of face k and $\mathbf{n}_k$ is the normal of face k;
- Minimum projected area of cluster i in direction $\mathbf{r}$: $\left[A_i^{(vis)}(\mathbf{r})\right] = A_i^{(n)} (\mathbf{n}_i \cdot \mathbf{r})_+$ where A_i is the total surface area of the cluster and $\mathbf{n}_i$ is the area averaged normal of the surface;
- Maximum projected area of cluster i in direction $\mathbf{r}$: $\left[A_i^{(vis)}(\mathbf{r})\right] = \sum_{j=1}^{3} (\mathbf{u}_j \cdot \mathbf{r})_+ D_j^+ + \sum_{j=1}^{3} (-\mathbf{u}_j \cdot \mathbf{r})_+ D_j^-$ where $\mathbf{u}_j$ is the j-th local axis of the oriented box, $D_j^+ = \sum_k A_k (\mathbf{n}_k \cdot \mathbf{u}_j)_+$ and $D_j^- = \sum_k A_k (-\mathbf{n}_k \cdot \mathbf{u}_j)_+$ are samples of the projected area of the cluster in the six principal directions of the oriented box, computed by traversing all cluster faces during hierarchy construction.

3.2. Higher-Order Vector Radiosity Approximation

The radiance distribution $L(\mathbf{x},\mathbf{z})$ in a non participating environment composed of general reflectors and emitters is described by the the following integral equation:

$$\begin{aligned} L(\mathbf{x},\mathbf{z}) &= L^e(\mathbf{x},\mathbf{z}) \\ &+ \int_A f_r(\mathbf{x},\mathbf{y},\mathbf{z}) L(\mathbf{y},\mathbf{x}) V(\mathbf{x},\mathbf{y}) G(\mathbf{x},\mathbf{y}) dA_y \end{aligned} \quad (1)$$

where $L(\mathbf{x},\mathbf{z})$ is the radiance at point $\mathbf{x}$ reflected towards point $\mathbf{z}$, $L^e(\mathbf{x},\mathbf{z})$ is the emitted radiance at point $\mathbf{x}$ reflected towards point $\mathbf{z}$, $V(\mathbf{x},\mathbf{y})$ is one if point $\mathbf{x}$ is visible from point $\mathbf{y}$ and zero otherwise, $f_r(\mathbf{x},\mathbf{y},\mathbf{z})$ is the bidirectional reflectance distribution function (BRDF) at $\mathbf{x}$ for light incident from $\mathbf{y}$ and reflected towards $\mathbf{z}$, $G(\mathbf{x},\mathbf{y}) = \frac{((\mathbf{y}-\mathbf{x})\cdot\mathbf{n}_\mathbf{x})_+ ((\mathbf{x}-\mathbf{y})\cdot\mathbf{n}_\mathbf{y})_+}{\pi\|\mathbf{x}-\mathbf{y}\|^4}$ is the geometry factor, and the integral is over the surface A of all objects of the environment.

For computational efficiency reasons, we make the simplifying assumption that the overall energy distribution in the environment is still well approximated when assuming that all surfaces emit and reflect light uniformly in the environment, i.e., that the BRDF $f_r(\mathbf{x},\mathbf{y},\mathbf{z})$ can be replaced by an average diffuse reflectivity $\rho(\mathbf{x})$, and that the emitted radiance $L^e(\mathbf{x},\mathbf{z})$ is well approximated by using an average diffuse emittance $B^e(\mathbf{x})$. This is true for the diffuse and moderately glossy materials common in architectural walkthrough applications, where directional effects mostly appear in the form of surface highlights. Under the diffuse assumption, equation 1 simplifies to

$$B(\mathbf{x}) = B^e(\mathbf{x}) + \rho(\mathbf{x}) \int_A B(\mathbf{y}) V(\mathbf{x},\mathbf{y}) G(\mathbf{x},\mathbf{y}) dA_y \quad (2)$$

The integrand of this equation represents the irradiance $E(\mathbf{x},\mathbf{y})$ at point $\mathbf{x}$ due to light emitted at point $\mathbf{y}$, and can be expressed in terms of two vector quantities:

$$E(\mathbf{x},\mathbf{y}) = (\mathbf{n}_\mathbf{x} \cdot \mathbf{E}(\mathbf{x},\mathbf{y}))_+$$

where $\mathbf{E}(\mathbf{x},\mathbf{y})$ is the irradiance vector at point $\mathbf{x}$ on the receiver due to point $\mathbf{y}$ on the emitter. The irradiance vector is parallel to the vector connecting $\mathbf{x}$ to $\mathbf{y}$ and is related to the radiosity of point $\mathbf{y}$ by $\mathbf{E}(\mathbf{x},\mathbf{y}) = \mathbf{m}(\mathbf{x},\mathbf{y}) B(\mathbf{y})$, where the transport vector $\mathbf{m}(\mathbf{x},\mathbf{y}) = V(\mathbf{x},\mathbf{y}) \frac{((\mathbf{x}-\mathbf{y})\cdot\mathbf{n}_\mathbf{y})_+}{\pi\|\mathbf{y}-\mathbf{x}\|^4} (\mathbf{y}-\mathbf{x})$ expresses the geometric relationship between $\mathbf{x}$ and $\mathbf{y}$.

The face cluster radiosity method approximates equation 2 by discretising the environment into face clusters A_j and by assuming, when computing energy transfer, that all points j within an emitting cluster are close together and far from the receiver [5]. The irradiance vector at a point $\mathbf{x}$ can thus be approximated by

$$\mathbf{E}_\mathbf{x} = \sum_j \int_{Aj} \mathbf{m}(\mathbf{x},\mathbf{y}) b(\mathbf{y}) dA_\mathbf{y} \quad (3)$$

and equation 2 thus becomes:

$$B(\mathbf{x}) = B^e(\mathbf{x}) + \rho(\mathbf{x})\mathbf{n}_\mathbf{x} \cdot \mathbf{E}_\mathbf{x} \quad (4)$$

The derivation of a higher-order finite element method for solving this equation follows closely that of the standard scalar radiosity[36, 37]. This equation can be solved approximately by assuming that the radiosity $B(\mathbf{x})$ on patch i can be well approximated by a linear combination $B(\mathbf{x}) = \sum_{i,\alpha} B_{i,\alpha}\Phi_{i,\alpha}(\mathbf{x})$ of a set of non-overlapping orthogonal basis functions $\Phi_{i,\alpha}$ defined on patch i. With this approximation, equation 4 becomes:

$$\mathbf{E}_\mathbf{x} \approx \sum_{j,\beta} B_{j,\beta} \left(\int_{A_j} \mathbf{m}(\mathbf{x},\mathbf{y})\Phi_{j,\beta}(\mathbf{y}) dA_\mathbf{y} \right) \quad (5)$$

$$\sum_{i,\alpha} B_{i,\alpha}\Phi_{i,\alpha}(\mathbf{x}) \approx \sum_{i,\alpha} B^e_{i,\alpha}\Phi_{i,\alpha}(\mathbf{x}) + \rho(\mathbf{x})\mathbf{n}_\mathbf{x} \cdot \mathbf{E}_\mathbf{x} \quad (6)$$

Following the Galerkin approach, we take the inner product of the left and right side of this equation with each basis function $\Phi_{i,\alpha'}$, obtaining a set of linear equations from which to compute the unknown irradiance vectors and radiosities:

$$\mathbf{K}_{i,\alpha;j,\beta} = \frac{\int_{A_i} \Phi_{i,\alpha}(\mathbf{x}) \int_{A_j} \mathbf{m}(\mathbf{x},\mathbf{y})\Phi_{j,\beta}(\mathbf{y}) dA_\mathbf{y} dA_\mathbf{x}}{\int_{A_i} \Phi_{i,\alpha}(\mathbf{x})^2 dA_\mathbf{x}} \quad (7)$$

$$\mathbf{E}_{i,\alpha} = \sum_{j,\beta} \mathbf{K}_{i,\alpha;j,\beta} B_{j,\beta} \quad (8)$$

$$B_{i,\alpha} = B^e_{i,\alpha} + \rho_i \mathbf{n}_i \cdot \mathbf{E}_{i,\alpha} \quad (9)$$

where ρ_i is the average reflectance of patch i and $\mathbf{n}_i$ is the average normal of patch i. These equations revert to the scalar Galerkin radiosity equations in case of perfectly planar elements, and revert to face cluster radiosity equations when using constant bases for both irradiance and radiosity. Since the method produces as output an irradiance vector distribution, it is possible to exploit it to quickly render illuminated geometry as the viewpoint changes, providing a visually compelling approximation of non-diffuse reflectance when generating output colors. This is done by evaluating equation 1 for a viewpoint $\mathbf{z}$ and the single directional light $\mathbf{E}(\mathbf{x}) = \sum_{i,\alpha} \mathbf{E}_{i,\alpha}\Phi_{i,\alpha}(\mathbf{x})$, which gives:

$$L(\mathbf{x},\mathbf{z}) \approx L^e(\mathbf{x},\mathbf{z}) + f_r(\mathbf{x},\mathbf{x}+\mathbf{E}(\mathbf{x}),\mathbf{z})(\mathbf{n}_\mathbf{x} \cdot \frac{\mathbf{E}(\mathbf{x})}{\pi})_+ \quad (10)$$

This shading equation is amenable to hardware implementation on commodity programmable graphics hardware. Even though glossy reflections are limited to the final stage of any illumination path and evaluated for a single dominant direction, the approximation quality is sufficient to provide compelling visual results and to convey important information on scene lighting and surface finish.

3.3. Integration and Hierarchical Refinement

3.3.1. Integration and visibility estimation

As for most current radiosity systems, we compute the coupling coefficients of equation 7 by numerical integration, using ray tracing to compute the visibility part of the kernel. The visibility queries involved in this process are often the most time consuming part of a radiosity simulation; at the same time, spatial subdivision methods for accelerating those queries are often requiring large amounts of memory. In this work, the multiresolution face cluster structure is used to speed-up visibility queries, using a multiresolution visibility method similar to the one used for volume clustering[38, 4]. The visibility query routine starts at the top face cluster of each potential occluder and descends into the face cluster hierarchy until the query ray is proved outside the current oriented box or the estimated projected shadow size of the current face cluster on the receiver is smaller than a given threshold. At this point we estimate the opacity of the occluder and stop the recursion. Since we are dealing with large thin clusters containing connected components, we do not use equivalent extinction coefficients, but, rather, estimate the opacity by the ratio of the visible projected area of the cluster in the direction of the ray and the projected area of the box. The size threshold used for stopping the recursion is chosen as a function of the distance between cubature nodes on the receiving cluster. This adaptive multiresolution visibility approach has the advantage of contributing to limiting core memory used, since face clusters will be accessed during visibility testing only when their size is comparable to that of the solution elements. By limiting the precision of the computation, we also reduce memory needs.

3.3.2. Transfer error estimation

Instead of estimating the errors on the propagation coefficients, we base refinement on a direct estimation of the error on the energy reflected by the receiver, using a BFA-weighted approach. The rationale for using a radiosity based refiner, even though the algorithm focuses on vector irradiance, is that it is radiosity that is finally displayed and perceived by the user. We use a number of sample points on the receiver and compare at each of those points the difference between the expected bounds on the radiosity at the sample point computed by direct integration and the value interpolated using the element basis. To ensure subdivision of side-on clusters, we distribute sample points on the entire cluster (and not only on the average plane). After computing the transfer coefficients using equation 7, upper and lower bounds on the BFA factors are estimated at each control point $\mathbf{x}$ using the bounds on the projected areas:

$$\lceil BFA_{ij}(\mathbf{x}) \rceil = \max_{\mathbf{y}\in A_j} \frac{B_j(\mathbf{y})V(\mathbf{x},\mathbf{y}) \left\lceil A_i^{(vis)}(\mathbf{y}-\mathbf{x}) \right\rceil \left\lceil A_j^{(vis)}(\mathbf{x}-\mathbf{y}) \right\rceil}{\pi \|\mathbf{y}-\mathbf{x}\|^2}$$

$$\lfloor BFA_{ij}(\mathbf{x}) \rfloor = \min_{\mathbf{y}\in A_j} \frac{B_j(\mathbf{y})V(\mathbf{x},\mathbf{y}) \left\lfloor A_i^{(vis)}(\mathbf{y}-\mathbf{x}) \right\rfloor \left\lfloor A_j^{(vis)}(\mathbf{x}-\mathbf{y}) \right\rfloor}{\pi \|\mathbf{y}-\mathbf{x}\|^2}$$

These bounds are compared with the value obtained from the coupling coefficients:

$$BFA'_{ij}(\mathbf{x}) = A_i \mathbf{n}_\mathbf{x} \cdot \left(\sum_{\alpha\, j,\beta} \mathbf{K}_{i,\alpha;j,\beta}\Phi_{i,\alpha}(\mathbf{x})B_{j,\beta} \right)$$

to estimate the maximum transfer error factor:

$$\Delta BFA_{ij} = \max_{\mathbf{x}\in A_i} \left\{ \begin{array}{l} \left| BFA'_{ij}(\mathbf{x}) - \lceil BFA_{ij}(\mathbf{x}) \rceil \right| \\ \left| BFA'_{ij}(\mathbf{x}) - \lfloor BFA_{ij}(\mathbf{x}) \rfloor \right| \end{array} \right\}$$

The transfer error used for refinement is then obtained with:

$$\delta_{i,j} = \lceil \rho_i \rceil \, \Delta BFA_{i,j}$$

3.3.3. Self transfer error estimation

Since face clusters are not perfectly planar, surfaces cannot be assumed to be perfectly convex and it is therefore necessary to handle self interaction. Following Willmott [13], an upper bound on the self form factor of a face cluster may be estimated by comparing the total visible external area to the surface area of the cluster, i.e. $\lceil F_{i,i} \rceil = 1 - \frac{A_i^{(n)}}{A_i}$ and $\lfloor F_{i,i} \rfloor = 1 - \frac{\sum_{k=1}^{3} D_k^+ + D_k^-}{A_i}$. We improve this bound by tracking during the clustering process whether a cluster is convex, and setting to zero the self form factor estimate in that case. Given an estimate of the self form factor, we estimate the error on self transfer by

$$\delta_{i,i} = \lceil \rho_i \rceil \, \lceil B_i(\mathbf{x}) \rceil \left(\lceil F_{i,i} \rceil - \lfloor F_{i,i} \rfloor \right) A_i$$

3.3.4. Push-Pull

A key step in every hierarchical radiosity algorithm is the push-pull phase, in which the information gathered at the different levels of detail is combined in a single multiresolution representation. For higher-order basis functions the coefficients for the push-pull operation, that depend purely on the relative geometry of the element and its children, are computed by taking the product of the basis functions. In the face cluster radiosity method, the relative geometry is not constant, and the push-pull coefficient matrix has to be recomputed at each level of the hierarchy from the parent-child transform. While in our method irradiance vectors and radiosity could be (and in general will be) represented using different bases, we assume that the set of basis functions used for radiosity is a subset of the set of basis functions used for irradiance vector. We thus have to compute and store a single coefficient matrix of size N^2, where N is the number of irradiance vector coefficients, since the coefficient matrix for radiosity is a submatrix of the coefficient matrix for irradiance. The computation is done at the sub-element creation time.

Pushing vector irradiance coefficients requires particular care, since vector irradiance is valid only in a single half-space. The approach we are currently taking is to push vector irradiance only when its first coefficient (relative to the constant basis) is in the positive half-space of the sub-element.

3.4. A Practical Solution Method

The techniques described above make it possible to extend face cluster radiosity with higher order bases. Our algorithm aims at rapidly producing decent quality illuminated models

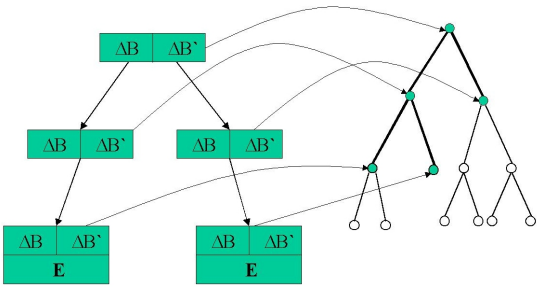

Figure 2: *Element hierarchy and original face cluster hierarchy. The solver operates only on the element hierarchy and on the clusters directly referenced by them.*

with limited memory footprint. Our main design decisions and their rationale are the following:

- radiosity and irradiance vectors may be represented with different bases. Since radiosity coefficients are used by the method only when elements act as emitters, while irradiance vectors are also used to produce local illumination detail when displaying the result, it makes sense to use a lower order basis for radiosity than for irradiance vectors. This would contribute to reduce memory usage with little degradation of visual results; a minimum storage solution for producing smooth results is therefore to use a constant basis for radiosity (one coefficient) and a non-product linear basis (three coefficients) for irradiance vectors;
- storing the coupling coefficients between two patches is typically extremely memory intensive for higher order radiosity. This is particularly true for vector radiosity, since each coupling coefficient is a transport vector (and not a scalar). For this reason, we have decided to avoid storing links, and therefore to use a shooting technique;
- vector irradiances are heavier than radiosities and are only used for accumulating contributions from other elements and at the leaves of the solution hierarchy to store the illumination result. By carefully ordering energy exchanges, we can accumulate irradiance into a temporary vector, which would thus be stored only at the leaves of the solution hierarchy. This would save 50% of the memory required for irradiance vectors.

The algorithm that we have derived from these design decisions is described in the following section.

3.4.1. Hierarchical radiosity shooting algorithm

Each solution element i stores the current unshot radiosity $\Delta B_{i,\alpha}$, the next iteration's unshot radiosity $\Delta B'_{i,\alpha}$, and a list of potential shooters, i.e., the elements that are candidates for transferring light to the element during the current iteration. The algorithm is structured in a way that the vector irradiance $\mathbf{E}_{i,\alpha}$ needs only be stored at the leaves of the solution hierarchy (see figure 2).

At the beginning of the algorithm, a top level solution

element is created for each of the top-level face clusters of the scene, with unshot radiosity initialized to the emittance, next iteration unshot radiosity initialized to zero, and an empty list of potential shooters. Multiple instances of the same model are possible. In that case, multiple top-level solution elements would reference the same face-cluster. At each iteration step, the algorithm starts by initializing each of the top level element's list of potential shooters with the other top-level elements that have a positive unshot radiosity and are facing toward the potential receiver. The list of potential shooters is then used in the multiresolution light transport phase. In this phase, the hierarchy of each of the top-level solution elements is traversed top-down to transport light from the potential shooters to the receivers. At each element i in the hierarchy, the unshot vector irradiance $\Delta\mathbf{E}_i$ is computed by summing the unshot vector irradiance of the parent with the unshot vector irradiance coming from the potential shooters list. The algorithm cyclically extracts a potential shooter j from the list until the list becomes empty. The coupling coefficients $\mathbf{K}_{i,\alpha\,j,\beta}$ and the error $\delta_{i,j}$ are computed. If the accuracy of the light transport is considered acceptable, the unshot vector irradiance $\Delta\mathbf{E}_i$ is incremented by $\sum_{j,\beta}\mathbf{K}_{i,\alpha;j,\beta}\Delta B_{j,\beta}$. Otherwise, the algorithm decides to compute the transport at a finer resolution. If the emitter is selected for refinement, the sub-elements of the emitter that are facing toward the receiver are inserted into the receiver's potential shooter list and will be treated later during the same iteration. Otherwise, the emitter is inserted into the list of potential shooters of the receiver's sub-elements that are facing toward it and will be treated later during the top-down element traversal. Self-link refinement is handled similarly by updating the potential shooters lists of the sub-elements in case of subdivision. When the potential shooters list is exhausted, $\Delta\mathbf{E}_i$ contains the unshot vector irradiance of the environment that is transferred directly to element i or higher up in the solution hierarchy. If element i is a leaf, the vector irradiance $\mathbf{E}_{i,\alpha}$ is incremented by $\Delta\mathbf{E}_i$ and the next iteration's unshot radiosity $\Delta B'_{i,\alpha}$ is set to $(1-F_{i,i})\rho_i\mathbf{n}_i\cdot\Delta\mathbf{E}_{i,\alpha}$. Otherwise, light transport is recursively applied to the sub-elements, and the next iteration's unshot radiosity $\Delta B'_{i,\alpha}$ is computed by pulling the unshot radiosity of the sub-elements. At the end of each iteration, the current ΔB values are set to those collected into $\Delta B'$, and $\Delta B'$ is cleared. The algorithm terminates when the (infinite) norm of ΔB falls below a user-defined threshold.

The major components of the method are summarized in algorithm 1.

```
SOLVE():
    for each top-level face cluster i
        create a top-level element i
        ΔB_{i,α} ← (B^e_i, 0, 0, .., 0) ΔB'_{i,α} ← 0 E_{i,α} ← 0
    repeat
        for each top level element i
            for each top level element j ≠ i
                ASSIGN-SHOOTER(i, j)
                TRANSPORT-LIGHT(i,0)
            for each top level element i
                ΔB_{i,α} ← ΔB'_{i,α} ΔB'_{i,α} ← 0
    until convergence

ASSIGN-SHOOTER(i, j):
    if ΔB_i ≠ 0 and element i is facing element j
        push j into shooters_i

TRANSPORT-LIGHT(i,ΔE_up):
    ΔE_tmp ← pushcoefficients(ΔE_up)
    while shooters_i not empty
        pop j from shooters_i
        compute coupling coefficients K_{i,α j,β} and error δ_{i,j}
        switch ORACLE(i, j, δ_{i,k})
            case subdivide i: for each child k of i:
            ASSIGN-SHOOTER(k, j)
            case subdivide j: for each child k of j:
            ASSIGN-SHOOTER(i,k)
            case else: ΔE_{tmp,α} ← ΔE_{tmp,α} + Σ_{j,β} K_{i,α;j,β} ΔB_{j,β}
    if SELF-ORACLE(i)
        for each child j of i
            for each child k of i, k ≠ j
                ASSIGN-SHOOTER(j,k)
    if i is a leaf
        ΔB'_{i,α} ← (1 − F_{i,i})ρ_i n_i · ΔE_{tmp,α} E_{i,α} ← E_{i,α} + ΔE_{tmp,α}
    else
        ΔB'_{i,α} ← 0
        for each child j of i
            TRANSPORT-LIGHT(j,ΔE_tmp)
            ΔB'_{i,α} ← ΔB'_{i,α} + pullcoefficients(ΔB'_{j,α})
```

Algorithm 1: *Hierarchical Higher Order Vector Radiosity*

3.5. Hardware Accelerated Solution Display

The output of the method is a vector irradiance map that partitions input models in areas where global illumination has a good approximation using the selected irradiance basis. Using equation 10, this map can be used to quickly render illuminated geometry as the viewpoint changes, taking into account non-diffuse reflectance when generating output colors. This technique, requiring only local data and a fixed small number of instructions, lends itself well to current commodity programmable graphics hardware.

Given the accuracy and program complexity limitations of current pixel pipelines, we have implemented vector irradiance to color conversion in a vertex program. The conversion includes radiance computation using the modified Phong BRDF [39] and simple tone mapping based on Reinhard's photographic tone reproduction operator [40]. This vertex shading approach is high quality enough for highly tessellated objects, which usually have more vertices than pixels for most viewpoints. The complete vertex program to implement this approach, expressed in the Cg language [41], is provided in Appendix A. As you can see, for efficiency reasons, vertex shaders are specialized by basis.

4. Implementation and Results

An experimental software library and a renderer application supporting the hierarchical higher order face cluster radiosity algorithm described in this paper has been implemented and tested on Linux, Silicon Graphics IRIX and

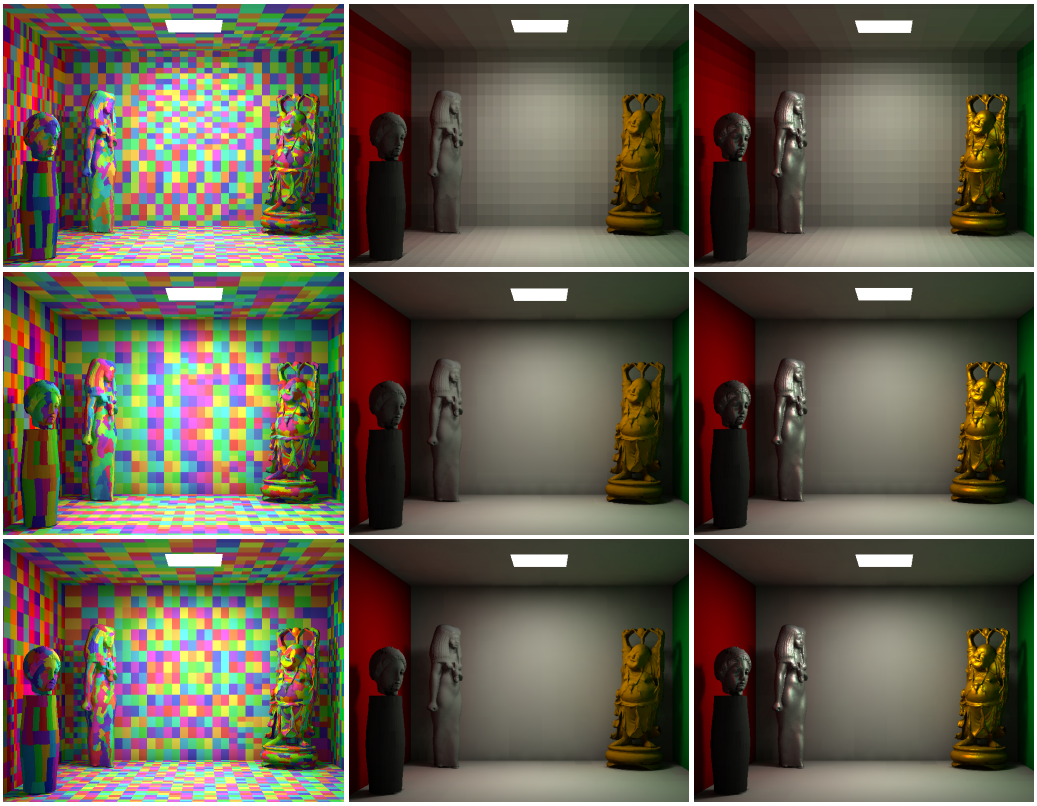

Figure 3: *Higher order face cluster radiosity solutions for a test scene of 1.5M input polygons. Solution time (10 iterations with 10^{-5} relative transfer accuracy) is in all cases under two minutes. First column: solution leaf clusters; second column: diffuse BRDF; third column: glossy BRDF. First row: constant basis; second row: linear basis; third row: quadratic basis.*

Windows NT machines. The software supports combinations of constant, linear, bilinear, quadratic, and cubic bases for representing radiosity and vector irradiance functions. The bases used were obtained by Gram-Schmidt orthogonalization of the functions $1, u, v, uv, u^2, v^2, u^3, u^2v, uv^2, v^3$ on the horizontal plane of the cluster's bounding box. On machines with graphics boards supporting the OpenGL `ARB_Vertex_Program` extension, global illumination solutions with glossy reflection effects can be examined in real-time thanks to hardware acceleration.

Each multiresolution model is stored using a face cluster table, a triangle table (with three vertex indices per triangle), and a vertex table with three coordinates per entry. Materials are stored at the level of clusters in the form of minimum, maximum, and area averaged emittance and reflectivity. Face clusters and triangles are sorted to permit direct sequential access. Using our current implementation, that does not employ particular compression schemes, the memory required for a face cluster node is 110 bytes, while a triangle and a vertex require 12 bytes each using 32 bits integer and floating point values. The memory required for a clustered geometric model of N faces is thus, assuming $2N$ clusters and $N/2$ vertices, of about $238N$ bytes. Only the parts of the model that participate to the solution will need to be swapped into core memory. To support the shooting algorithm, a solution element has to store a push-pull matrix, two unshot radiosities and the references to the two subelements and to the associated face cluster. Vector irradiances are stored only at the leaf elements. The size of a solution element is thus $12+24N_b+4N_e^2$ bytes for an internal element and $12+24N_b+4N_e^2+36N_e$ for a leaf element, where N_b is the number of radiosity coefficients per element and N_e is the number of irradiance coefficients per element.

In the test case discussed here, global illumination is computed for an indoor scene typical of those seen in the global illumination literature (see figure 3). The scene represents a closed rectangular room ($6\text{m} \times 5\text{m} \times 4\text{m}$) lit by a single area light (power $= 109.25\text{W}$) and containing three high resolution scanned models with subtle geometric details and a polygonized implicit surface (the pedestal). Using the quadric-based surface simplification method [42] we produced multiple versions of the scene with the same material properties and polygon counts ranging from 160K to 1.5M faces. The statues exhibit glossy surface properties (Phong

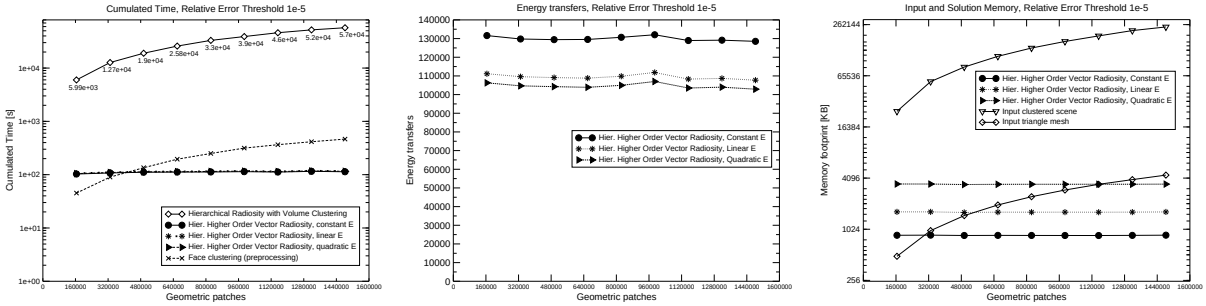

Figure 4: *Left: Performance of hierarchical radiosity with volume clustering and higher order face cluster radiosity as model complexity increases. Center: Number of energy transfers for hierarchical constant, linear, and quadratic face cluster radiosity as model complexity increases. Right: Memory requirements for input world, face hierarchy, and solution hierarchy.*

exponent=10), while the rest of the model is made of pure Lambert reflectors. The average scene reflectivity is 0.67. Much of the light in the room derives from secondary illumination and shadows are cast from and onto face clustered models, providing a good test of complex interreflection.

We compared the performance of the technique presented in this paper with the standard hierarchical radiosity with volume clustering based on the Gauss-Seidel iteration, as implemented in the widely available Renderpark software. These results were collected on a Linux PC with a AMD Athlon XP 1600MHz, 2GB RAM, and a NVIDIA GeForce4 Ti4600 graphics board. A common parameterization was used for all simulation runs: in particular, we used a 10^{-5} relative transfer error threshold, a degree 5 (7 nodes) integration rule on receivers, a degree 4 (6 nodes) integration rule on the emitter, and a visibility in quadrature technique. Simulation was stopped after 10 iterations. In our software, we always used a constant basis for the emitter and varied the receiver basis from constant (1 function/element) to linear (3 functions/elements) to quadratic (6 functions/element). An extended analysis is available elsewhere [43].

As expected, our method is constant time, taking 103 s to 115 s for a complete simulation, while the volume clustering method complexity grows linearly from 1*h*37 to 15*h*50 with the input model polygon count (see figure 4 left). Preprocessing time for the face clustering method ranges from 45*s* to 464*s*. It is negligible with respect to the volume clustering solution times and may, moreover, be amortized among multiple solution runs and multiple reuses of the same clustered model. The better results of the face clustering method are largely due to the fact that volume clusters do not provide a good fit for oriented surface regions, forcing the algorithm to transfer light between elements far down in the cluster tree to reach acceptable accuracy. Moreover, the need of a push-to-leaves phase heavily impacts on the memory locality of the scalar radiosity method. By contrast, the number of leaves that participate in the solution for the face clustering method is small and independent from input data size (from 5800 leaf elements for the quadratic solution to 6200 leaf elements for the constant one). The memory requirements for the method are illustrated in figure 4 right. The clustered model footprint grows linearly with the input polygon count, while solution memory stays constant. The quadratic basis is the more costly, but still requires less than 3.5MB for storing the solution hierarchy. The required working set for the simulation is in all cases under 10MB. Figure 4 center compares the number of energy transfers to compute a global illumination solution with the three irradiance bases used for the test. As you can see, much as the leaf element count, the number of energy transfers is independent from the geometric complexity and significantly diminishes as the basis order increases. Using higher order bases during the solution process, and not only for smoothing out the solution in a post-processing step as in [13], is therefore performance effective.

Figures 3 and 5 show the computed solutions, that can be inspected in real time on standard graphics PC. The accompanying video further illustrates the quality and performance of the method with interactive solution and inspection sequences. The importance of view dependent glossy reflection effects for perceiving both material type and surface shape is evident. Notice, in particular, the complex indirect reflections from the colored walls on the statues.

5. Conclusions and Future Work

We have presented an algorithm for simulating global illumination in scenes composed of highly tessellated objects with diffuse or moderately glossy reflectance. The solution method is a higher order extension of the face cluster radiosity technique. Our benchmarks demonstrate that, using this algorithm, visually compelling global illumination solutions for scenes of over one million input polygons with a good range of glossy BRDF can be computed in minutes and examined interactively on common graphics personal computers. While the displayed solution is not the result of a

Figure 5: *Selected close-up frames from an interactive rendering session with constant radiosity basis and linear irradiance basis. Notice shadow casting and view-dependent effects due to glossy reflection of indirect incoming light.*

full global illumination simulation, since glossy reflections are limited to the final stage of any illumination path, its visual quality is not very far from what provided by ray-tracing post-processing algorithms commonly in use in state-of-the-art architectural lighting systems. We thus believe that our approach has great promise, since it can be used to generate low to moderate quality solutions for glossy environments, that are suitable for interactive viewing. A detailed analysis of the approximation error introduced by our method is an important area for future work.

Our current work is concentrating on improving the implementation of the prototype solver and renderer and on evaluating the effect of the various accuracy and material parameters on rendering quality and speed. By decoupling visibility from transfer coefficient computation, as in visibility mask and adaptive shadow sampling approaches, we expect to take further advantage of the approximation capabilities of higher order bases. In order to further extend the range of BRDF, we also plan to evaluate more accurate representations of incoming light than the compact unidirectional vector representation employed in this paper. The announced availability for the near future of next generation graphics boards with programmable full floating point pixel pipelines will enable the evaluation of local shading per pixel. By moving vertex computation to the pixel level, we expect a speed improvement for highly tessellated scenes and increased quality for objects with low polygon counts.

Acknowledgments

The authors would like to thank Cyberware and the Stanford Graphics Group for making the scanned models available to the research community. This research is partially supported by the DIVERCITY project (EU-IST-13365), funded under the European FP5/IST program. We also acknowledge the contribution of Sardinian regional authorities.

References

1. A. Scheel, M. Stamminger, and H.-P. Seidel, "Grid based final gather for radiosity in complex environments", *Computer Graphics Forum (Proceedings of Eurographics 2002)*, **21**(3), (2002). 1

2. F. Sillion, G. Drettakis, and C. Soler, "A Clustering Algorithm for Radiance Calculation in General Environments", in *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)* (P. M. Hanrahan and W. Purgathofer, eds.), (New York, NY), pp. 196–205, (1995). 2

3. B. Smits, J. Arvo, and D. Greenberg, "A clustering algorithm for radiosity in complex environments", in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pp. 435–442, (1994). 2

4. S. Gibson and R. J. Hubbold, "Efficient hierarchical refinement and clustering for radiosity in complex environments", *Computer Graphics Forum*, **15**(5), pp. 297–310 (1996). 2, 4

5. A. J. Willmott, P. S. Heckbert, and M. Garland, "Face cluster radiosity", in *Rendering Techniques '99* (D. Lischinski and G. W. Larson, eds.), Eurographics, pp. 293–304, (1999). 2, 3

6. J. M. Hasenfratz, C. Damez, F. Sillion, and G. Drettakis, "A practical analysis of clustering strategies for hierarchical radiosity", in *Computer Graphics Forum (Proc. Eurographics '99)*, pp. C–221–C–232, (Sept. 1999). 2

7. G. Müller, S. Schäfer, and W. D. Fellner, "Automatic creation of object hierarchies for radiosity clustering", in *Computer Graphics Forum* (D. Duke, S. Coquillart, and T. Howard, eds.), vol. 19(4), pp. 213–221, Eurographics Association, (2000). 2

8. F. X. Sillion, "A Unified Hierarchical Algorithm for Global Illumination with Scattering Volumes and Object Clusters", *IEEE Transactions on Visualization and Computer Graphics*, **1**(3), pp. 240–254 (1995). 2

9. P. H. Christensen, D. Lischinski, E. J. Stollnitz, and D. H. Salesin, "Clustering for glossy global illumination", *ACM Transactions on Graphics*, **16**(1), pp. 3–33 (1997). 2

10. G. Greger, P. Shirley, P. M. Hubbard, and D. P. Greenberg, "The irradiance volume", *IEEE Computer Graphics and Applications*, **18**(2), pp. 32–43 (1998). 2

11. H. E. Rushmeier, C. Patterson, and A. Veerasamy, "Geometric simplification for indirect illumination calculations", in *Proc. Graphics Interface '93*, (Toronto, Ontario), pp. 227–236, (May 1993). 2

12. R. Dumont and K. Bouatouch, "Combining hierarchical radiosity with LODs", in *Proc. IASTED International Conference on Graphics and Imaging*, (Nov. 2000). 2

13. A. J. Willmott, *Hierarchical Radiosity with Multiresolution Meshes*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, (Nov. 2000). 2, 3, 5, 8

14. L. Alonso, F. Cuny, S. Petitjean, J.-C. Paul, S. Lazard, and E. Wies, "The virtual mesh: A geometric abstraction for efficiently computing radiosity", *ACM Transactions on Graphics*, **20**(3), pp. 169–201 (2001). 2

15. M. Stamminger, H. Schirmacher, P. Slusallek, and H.-P. Seidel, "Getting rid of links in hierarchical radiosity", in *Computer Graphics Forum* (D. Duke, S. Coquillart, and T. Howard, eds.), vol. 17(3), pp. 165–174, Eurographics Association, (1998). 2

16. R. Dumont, K. Bouatouch, and P. Gosselin, "A progressive algorithm for three point transport", in *Computer Graphics Forum* (D. Duke, S. Coquillart, and T. Howard, eds.), vol. 18(1), pp. 41–56, Eurographics Association, (1999). 2

17. X. Granier and G. Drettakis, "Controlling memory consumption of hierarchical radiosity with clustering", in *Proceedings of the Conference on Graphics Interface (GI-99* (I. S. MacKenzie and J. Stewart, eds.), (Toronto, Ontario), pp. 58–65, (June 2–4 1999). 2

18. F. Cuny, L. Alonso, and N. Holzschuch, "A novel approach makes higher order wavelets really efficient for radiosity", in *Computer Graphics Forum (Eurographics 2000)* (M. Gross and F. R. A. Hopgood, eds.), vol. 19(3), pp. 99–108, (2000). 2

19. B. Walter, P. M. Hubbard, P. Shirley, and D. F. Greenberg, "Global illumination using local linear density estimation", *ACM Transactions on Graphics*, **16**(3), pp. 217–259 (1997). 2

20. H. W. Jensen, *Realistic Image Synthesis Using Photon Mapping*. Natick, MA: A. K. Peters, (2001). 2

21. L. Aupperle and P. Hanrahan, "A hierarchical illumination algorithm for surfaces with glossy reflection", in *Computer Graphics Proceedings, Annual Conference Series, 1993*, pp. 155–162, (1993). 2

22. S. E. Chen, H. E. Rushmeier, G. Miller, and D. Turner, "A progressive multi-pass method for global illumination", *Computer Graphics*, **25**(4), pp. 165–174 (1991). 2

23. B. Walter, G. Drettakis, and S. Parker, "Interactive rendering using render cache", in *Rendering Techniques '99* (D. Lischinski and G. W. Larson, eds.), Eurographics, pp. 19–30, (1999). 2

24. M. Simmons and C. H. Séquin, "Tapestry: A dynamic mesh-based display representation for interactive rendering", in *Rendering Techniques '00*, Eurographics, pp. 329–340, (2000). 2

25. M. Stamminger, J. Haber, H. Schirmacher, and H.-P. Seidel, "Walkthroughs with corrective texturing", in *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)* (B. Peroche and H. Rushmeier, eds.), (New York, NY), pp. 377–388, (2000). 2

26. P. Tole, F. Pellacini, B. Walter, and D. P. Greenberg, "Interactive global illumination in dynamic scenes", *ACM Transactions on Graphics*, **21**(3), pp. 537–546 (2002). 2

27. G. Drettakis and F. X. Sillion, "Interactive update of global illumination using a line-space hierarchy", *Computer Graphics*, **31**(Annual Conference Series), pp. 57–64 (1997). 2

28. W. Stürzlinger and R. Bastos, "Interactive rendering of globally illuminated glossy scenes", in *Eurographics Rendering Workshop 1997* (J. Dorsey and P. Slusallek, eds.), (New York City, NY), pp. 93–102, Eurographics, (June 1997). 2

29. X. Granier and G. Drettakis, "Incremental updates for rapid glossy global illumination", *Computer Graphics Forum*, **20**(3), (2001). 2

30. M. Stamminger, P. Slusallek, and H.-P. Seidel, "Interactive walkthroughs and higher order global illumination", in *Modeling, Virtual Worlds, Distributed Grapics* (D. W. Fellner, ed.), pp. 121–128, Gesellschaft für Informatik, (1995). 2

31. B. Walter, G. Alppay, E. P. F. Lafortune, S. Fernandez, and D. P. Greenberg, "Fitting virtual lights for non-diffuse walkthroughs", in *SIGGRAPH 97 Conference Proceedings* (T. Whitted, ed.), Annual Conference Series, pp. 45–48, ACM SIGGRAPH, (Aug. 1997). 2

32. M. Stamminger, A. Scheel, X. Granier, F. Perez-Cazorla, G. Drettakis, and F. X. Sillion, "Efficient glossy global illumination with interactive viewing", *Computer Graphics Forum*, **19**(1), pp. 13–25 (2000). 2

33. P.-P. Sloan, J. Kautz, and J. Snyder, "Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments", *ACM Transactions on Graphics*, **21**(3), pp. 527–536 (2002). 2

34. M. Garland, A. Willmott, and P. S. Heckbert, "Hierarchical face clustering on polygonal surfaces", in *ACM Symposium on Interactive 3D Graphics*, pp. 49–58, (2001). 3

35. E. Bouvier and E. Gobbetti, "TOM – Totally Ordered Mesh: a multiresolution structure for time-critical graphics applications", *International Journal of Image and Graphics*, **1**(1), pp. 115–134 (2001). 3

36. H. R. Zatz, "Galerkin radiosity: A higher order solution method for global illumination", in *SIGGRAPH 93 Conference Proceedings* (J. T. Kajiya, ed.), Computer Graphics Proceedings, Annual Conference Series, pp. 213–220, ACM SIGGRAPH, (Aug. 1993). 4

37. P. Bekaert and Y. D. Willems, "HIRAD: A hierarchical higher order radiosity code", in *12th Spring Conference on Computer Graphics* (W. Purgathofer, ed.), pp. 213–227, Comenius University, Bratislava, Slovakia, (June 1996). 4

38. F. Sillion and G. Drettakis, "Feature-Based Control of Visibility Error: A Multiresolution Clustering Algorithm for Global Illumination", in *Computer Graphics Proceedings, Annual Conference Series, 1995 (ACM SIGGRAPH '95 Proceedings)*, pp. 145–152, (1995). 4

39. R. R. Lewis, "Making shaders more physically plausible", *Computer Graphics Forum*, **13**(2), pp. 109–120 (1994). 6

40. E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic tone reproduction for digital images", *ACM Transactions on Graphics*, **21**(3), pp. 267–276 (2002). 6

41. NVIDIA Corporation, *Cg Toolkit User's Manual*, 1.0 ed., (Dec. 2002). 6

42. M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics", in *SIGGRAPH 97 Conference Proceedings* (T. Whitted, ed.), Annual Conference Series, pp. 209–216, ACM SIGGRAPH, (Aug. 1997). 7

43. L. Spanò and E. Gobbetti, "An empirical evaluation of hierarchical higher order face cluster radiosity", Tech. Rep. CRS4 TR/, CRS4, Center for Advanced Studies, Research, and Development in Sardinia, Cagliari, Italy, (Feb. 2003). 8

Appendix A: Cg vertex shader

```
struct a2v: application2vertex {
  float4 Position: POSITION;
  float3 Normal  : NORMAL;
  float4 Kd      : TEXCOORD0; // rgb = diffuse  refl., a = alpha
  float4 Ks      : TEXCOORD1; // rgb = specular refl., a = shininess
  float3 Ke      : TEXCOORD2; // rgb = emittance
};

struct v2f: vertex2fragment {
  float4 HPOS    : HPOS;
  float4 COL0    : COL0;
};

// Interpolation bases
float f_1  (float2 uv)          { return 1; }
float f_u  (float2 uv)          { return 1.7320508*(2*uv.x-1); }
float f_v  (float2 uv)          { return 1.7320508*(2*uv.y-1); }
float f_uv (float fu, float fv) { return fu*fv; }
float f_u2 (float fu)           { return 1.1180340*(fu*fu-1); }
float f_v2 (float fv)           { return 1.1180340*(fv*fv-1); }
float f_u3 (float fu, float fu2){ return 1.1385501*(fu*(fu2-0.8944272)); }
float f_u2v(float fu2, float fv){ return fu2*fv; }
float f_uv2(float fu, float fv2){ return fu*fv2; }
float f_v3 (float fv, float fv2){ return 1.1385501*(fv*(fv2-0.8944272)); }

// return x^q if x>0, 0 otherwise
float specular(float x, float q) { return lit(x,x,q).z; }

float4 radiance(float3 position,
                float3 normal,
                float3 eye,
                float3 Ke, float4 Kd, float4 Ks,
                float3 Er, float3 Eg, float3 Eb) {
  float4 result;
  float3 r = reflect(normalize(position - eye), normal);
  result.rgb =
    Ke +
    float3(max(dot(normal,Er),0.0f),
           max(dot(normal,Eg),0.0f),
           max(dot(normal,Eb),0.0f)) *
    (Kd.rgb + Ks.rgb * float3(specular(dot(r,normalize(Er)),Ks.a),
                              specular(dot(r,normalize(Eg)),Ks.a),
                              specular(dot(r,normalize(Eb)),Ks.a)));
  result.a = Kd.a;
  return result;
}

float4 tone_map(float4 c,
                float  key_over_Lw_avg,
                float  Lw_white_squared) {
  float4 result;
  float Lw_c = 0.27 * c.r + 0.67 * c.g + 0.06 * c.b;
  result.rgb =
    key_over_Lw_avg *
    (1.0 + key_over_Lw_avg * Lw_c / Lw_white_squared) /
    (1.0 + key_over_Lw_avg * Lw_c) *
    c.rgb;
  result.a = c.a;
  return result;
}

v2f hhovr_constant_shader(a2v IN,
                          uniform float4x4 PVM,
                          uniform float4x4 M2E,
                          uniform float3   eye,
                          uniform float3   Er[1],
                          uniform float3   Eg[1],
                          uniform float3   Eb[1],
                          uniform float2   tonemap_params) {
  v2f OUT;
  float3 Er_uv = Er[0];
  float3 Eg_uv = Eg[0];
  float3 Eb_uv = Eb[0];
  OUT.COL0 = tone_map(radiance(IN.Position.xyz, IN.Normal, eye,
                               IN.Ke, IN.Kd, IN.Ks,
                               Er_uv, Eg_uv, Eb_uv),
                      tonemap_params[0],
                      tonemap_params[1]);

  OUT.HPOS = mul(PVM, IN.Position);
  return OUT;
}

v2f hhovr_linear_shader(a2v IN,
                        uniform float4x4 PVM,
                        uniform float4x4 M2E,
                        uniform float3   eye,
                        uniform float3   Er[3],
                        uniform float3   Eg[3],
                        uniform float3   Eb[3],
                        uniform float2   tonemap_params) {
  v2f OUT;
  float2 uv = mul(M2E, IN.Position).xy;
  float fu = f_u(uv);
  float fv = f_v(uv);
  float3 Er_uv = Er[0] + fu*Er[1] + fv*Er[2];
  float3 Eg_uv = Eg[0] + fu*Eg[1] + fv*Eg[2];
  float3 Eb_uv = Eb[0] + fu*Eb[1] + fv*Eb[2];
  OUT.COL0 = tone_map(radiance(IN.Position.xyz, IN.Normal, eye,
                               IN.Ke, IN.Kd, IN.Ks,
                               Er_uv, Eg_uv, Eb_uv),
                      tonemap_params[0],
                      tonemap_params[1]);
  OUT.HPOS = mul(PVM, IN.Position);
  return OUT;
}

// And so on for bilinear, quadratic, and cubic shader...
```