Hierarchical Higher Order Face Cluster Radiosity

Enrico Gobbetti, Leonardo Span and Marco Agus

CRS4
Visualization and Virtual Reality Group
VI Strada Ovest, Z.I. Macchiareddu, C.P. 94, I-09010 Uta (CA), Italy
`http://www.crs4.it/vvr`

March 29, 2002

Abstract

We present an algorithm for simulating diffuse interreflection in scenes composed of highly tessellated objects. The 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 OpenGL register combiners extension can be used to render illuminated models directly from the vector irradiance map, exploiting hardware acceleration for computing vertex radiosity on commodity graphics boards.

Chapter 1

Introduction

Complex synthetic scenes composed of millions of graphics primitives are rapidly becoming commonplace in many application domains. Representative examples are the large triangle meshes that are currently produced by scanning Cultural Heritage artifacts and the large triangulated models generated by tessellating non-planar objects in architectural scenes. Handling such objects in the context of finite element global illumination simulations is a challenging task.

The most advanced finite element methods for computing global illumination solutions, currently variations of the hierarchical radiosity technique with volume clustering [CLSS97, GH96, Sil95, SAG94], permit scenes of only moderate complexity (tens of thousands of graphics primitives) to be simulated in times compatible with the constraints of a design tool (seconds or minutes for a good quality draft to ensure rapid design cycles) [Che90, WH97, DB00]. Unfortunately, the method is not well suited to scenes containing highly tessellated objects, because both of memory and time complexity constraints [WH97]. This fact has forced current radiosity systems to limit themselves to simple (or simplified) graphics scenes. In most cases, this limitation forces application users to manually remove details (e.g. by specifying fixed low polygon count tessellations of curved surfaces) from the designed scenes. The goal of this work is to contribute to removing this limitation, by employing multiresolution modeling techniques.

In this report, we present 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. 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 OpenGL register combiners extension can be used to render illuminated models directly from the vector irradiance map, exploiting hardware acceleration for computing vertex radiosity on the fly on commodity graphics boards.

The rest of the report 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 report concludes with a summary and a view of current and future work.

Chapter 2

Related work

2.1 Partitioning methods for radiosity

Radiosity simulation for complex building interiors cope with memory and computational problems by employing partitioning strategies [Air90, FTSK96, MBMD98, TFFH94]. These solutions rely on a preprocessing step, which consists in partitioning the model in 3D cells and computing a graph that expresses visibility relationships among these cells. Once this preprocessing is done, ordering strategies are used to optimize radiosity computations[TFFH94]. These strategies are particularly adapted to scenes with large convex occluders (e.g. flat walls), but do not address the problems caused by large tessellated objects. In this case, hierarchical techniques based on clustering approaches have to be employed to reduce the initial linking complexity. Partitioning methods are orthogonal to the clustering methods discussed in this report and may coexist in a single system.

2.2 Volume clustering for hierarchical radiosity

The most successful radiosity technique for dealing with complex scenes is currently hierarchical radiosity [SDS95]. The algorithm constructs a hierarchical representation of the form factor matrix by adaptively subdividing planar patches into sub-patches according to a user-supplied error bound. By treating interactions between distant patches at a coarser level than those between nearby patches, the algorithms 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 [SAG94, SDS95, GH96] combat this problem by grouping input patches into volume clusters. While volume clustering avoids the initial quadratic transport link step, handling the light incident on a cluster is a difficult problem and all presented solutions are more suitable to handling unorganized sets of polygons rather than highly tessellated models [WHG99, HDSD99, MSF00]. 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 [HDSD99]. At the same time, pushing irradiances to leaves on-the-fly [Sil95, SAG94, CLSS97], 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.

2.3 Hierarchical radiosity on simplified and multiresolution models

Mesh simplification techniques can be adopted to structure the data at different levels of detail [GSHG98]. These techniques have been used especially in the field of adaptive rendering, where the perceptual impact of a given object or component is used to select dynamically the appropriate level of representation. Different approaches for creating multiresolution models have been proposed, based either on the explicit representation of the graph of inferences of the atomic simplification steps, or on implicit regular decomposition rules which adapt well to regular meshes (e.g. gridded terrain models and tessellations of parametric patches). A number of authors have recognized the potential of these techniques for handling large tessellated surfaces in radiosity. Rushmeier et al. [RPV93] demonstrated the use of simplified models in radiosity. Greger et al. [GSHG98], 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 [DB00] 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 at the, which is prohibitively memory and time expensive for models where the geometric detail is much finer than the illumination complexity. Willmott and Heckbert [WHG99] 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 [Wil00] 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.

2.4 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 growths 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 [SSSS98, DBG99, GD99, CAH00, ACP^{+}01]. 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 level of elements.

Chapter 3

Hierarchical Higher Order Face Cluster Radiosity

3.1 Hierarchical Data Structure

As in the original face cluster radiosity algorithm[WHG99], 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 3.1) 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. [GWH01] that we have extended to handle vertex attributes as in our earlier simplification tool [BG01].

Since face clusters do not in general not 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 [Wil00]:

- Normal-projected area of cluster i:

$$A_i^{(n)} = \left\| \sum_k A_k \mathbf{n}_k \right\| \tag{3.1}$$

  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 $\mathbf{r}$:

$$\left\lfloor A_i^{(vis)}(\mathbf{r}) \right\rfloor = A_i^{(n)} \left(\mathbf{n}_i \cdot \mathbf{r}\right)_+ \tag{3.2}$$

  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 $\mathbf{r}$:

$$\left\lceil A_i^{(vis)}(\mathbf{r}) \right\rceil = \sum_{j=0}^{3} \left(\mathbf{u}_j \cdot \mathbf{r}\right)_+ D_j^+ + \sum_{j=0}^{3} \left(-\mathbf{u}_j \cdot \mathbf{r}\right)_+ D_j^- \tag{3.3}$$

  where $\mathbf{u}_j$ is the j-th local axis of the oriented box, $D_j^+ = \sum_k A_k \left(\mathbf{n}_k \cdot \mathbf{u}_j\right)_+$and $D_j^- = \sum_k A_k \left(-\mathbf{n}_k \cdot \mathbf{u}_j\right)_+$ are samples of the projected area of the cluster in the

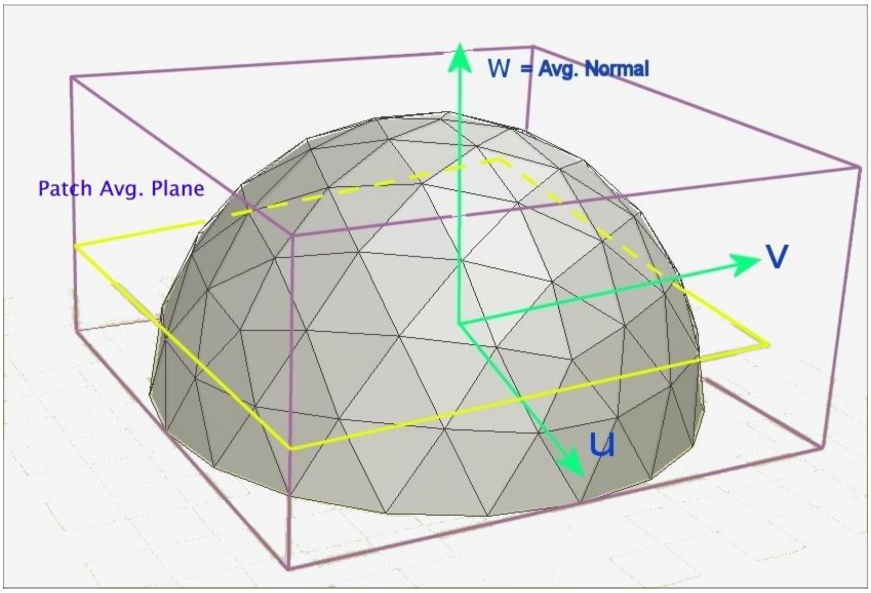


Figure 3.1: Face cluster

six principal direction of the oriented box, computed by traversing all cluster faces during hierarchy construction.

3.2 Higher-Order Vector Radiosity Approximation

The radiosity distribution $b(\mathbf{x})$ in an environment composed only of Lambertian diffuse reflectors and emitters is described by the following integral equation:

$$b(\mathbf{x}) = e(\mathbf{x}) + \rho(\mathbf{x}) \int_A E(\mathbf{x}, \mathbf{y}) dA_y \tag{3.4}$$

where $e(\mathbf{x})$is the diffuse emittance at point $\mathbf{x}$, $\rho(\mathbf{x})$is the diffuse reflectance at point $\mathbf{x}$, $E(\mathbf{x}, \mathbf{y})$ is the irradiance at point $\mathbf{x}$ due to the light emitted at point $\mathbf{y}$, and the integral is over the surface A of all objects of the environment. The irradiance $E(\mathbf{x}, \mathbf{y})$ can be expressed in terms of two vector quantities:

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

where $\mathbf{n_x}$ is the unit normal at point $\mathbf{x}$ and $\mathbf{E}(\mathbf{x}, \mathbf{y})$ 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 $\mathbf{r_{xy}}$ connecting $\mathbf{x}$ to $\mathbf{y}$ and is related to the radiosity of point $\mathbf{y}$ by

$$\mathbf{E}(x, y) = \mathbf{m}(\mathbf{x}, \mathbf{y}) b(\mathbf{y})$$

where the transport vector $\mathbf{m}(\mathbf{x}, \mathbf{y})$ expresses the geometric relationship between $\mathbf{x}$ and $\mathbf{y}$:

$$\mathbf{m}(\mathbf{x}, \mathbf{y}) = vis(\mathbf{x}, \mathbf{y}) \frac{(-\mathbf{r_{xy}} \cdot \mathbf{n_y})_+}{\pi \left\|\mathbf{r_{xy}}\right\|^4} \mathbf{r_{xy}}$$

The face cluster radiosity method approximates equation 3.4 by discretizing 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 [WHG99]. The irradiance vector at a point $\mathbf{x}$ can thus be approximated by

$$\mathbf{E}_\mathbf{x} = \sum_j \int_{Aj} \mathbf{m}(\mathbf{x},\mathbf{y}) b(\mathbf{y}) dA_\mathbf{y} \tag{3.5}$$

and equation 3.4 thus becomes:

$$b(\mathbf{x}) = e(\mathbf{x}) + \rho(\mathbf{x}) \mathbf{n}_\mathbf{x} \cdot \mathbf{E}_\mathbf{x} \tag{3.6}$$

The derivation of a higher-order finite element method for solving this equation follows closely that of the standard scalar radiosity[Zat93, BW96b]. 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 3.6 becomes:

$$\begin{aligned} \mathbf{E}_\mathbf{x} &\approx \sum_{j,\beta} b_{j,\beta} \left(\int_{A_j} \mathbf{m}(\mathbf{x},\mathbf{y}) \Phi_{j,\beta}(\mathbf{y}) dA_\mathbf{y} \right) && (3.7) \\ \sum_{i,\alpha} b_{i,\alpha} \Phi_{i,\alpha}(\mathbf{x}) &\approx \sum_{i,\alpha} e_{i,\alpha} \Phi_{i,\alpha}(\mathbf{x}) + \rho(\mathbf{x}) \mathbf{n}_\mathbf{x} \cdot \mathbf{E}_\mathbf{x} && (3.8) \end{aligned}$$

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 irradiances and radiosities:

$$\begin{aligned} \mathbf{K}_{i,\alpha;j,\beta} &= \frac{\int_{A_i} \Phi_{i,\alpha}(\mathbf{x}) \int_{A_j} \mathbf{m}(\mathbf{x},\mathbf{y}) \Phi_{j,\beta}(\mathbf{y}) dA_\mathbf{y} dA_\mathbf{x}}{\int_{A_i} \Phi_{i,\alpha}(\mathbf{x})^2 dA_\mathbf{x}} && (3.9) \\ \mathbf{E}_{i,\alpha} &= \sum_{j,\beta} \mathbf{K}_{i,\alpha;j,\beta} b_{j,\beta} && (3.10) \\ b_{i,\alpha} &= e_{i,\alpha} + \rho_i \mathbf{n}_i \cdot \mathbf{E}_{i,\alpha} && (3.11) \end{aligned}$$

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 the Willmott's face cluster radiosity equations when using constant bases for both irradiance and radiosity.

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 3.9 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[SD95, GH96]. 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 proven 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

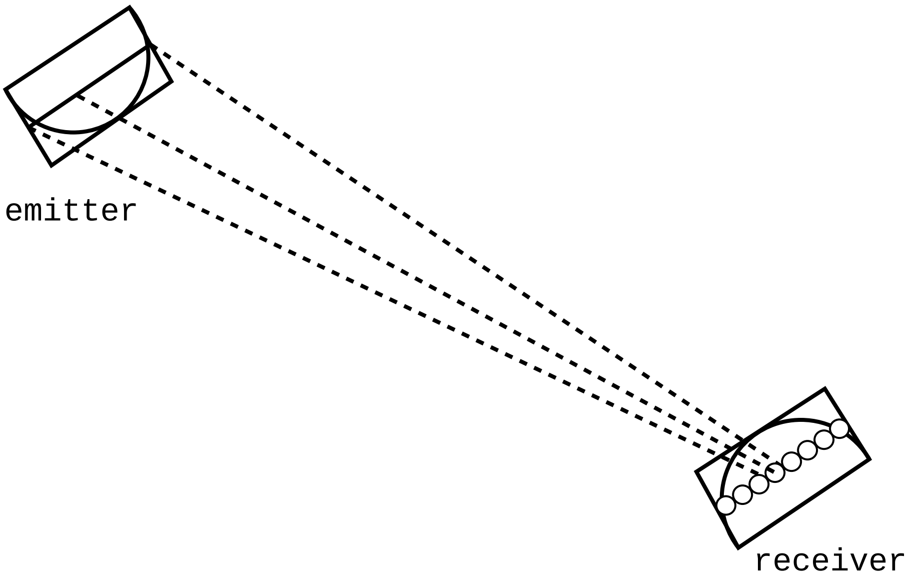


Figure 3.2: Multiresolution visibility geometry. The size threshold used for stopping the recursion is a function of the size of the emitter and the distance between sample points on the receiver.

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 choosen as a function of the distance between cubature nodes on the receiving cluster (see figure 3.2). 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 percieved by the user. As proposed by other authors (see, eg. Bekaert and Willems [BW96b, BW96a] and Cuny et al. [CAH00]), we use a number of control points on the receiver and compare at each of those points the difference between the expected bounds on the radiosity at the control point computed by direct integration and the value interpolated using the element basis and assume that the radiosity is well approximated on the emitter by the constant term $b_{j,0}$. After computing the transfer coefficients using equation 3.9, bounds on the FA factors are estimated at each control point $\mathbf{x}$ using the bounds on the projected areas:

$$\begin{aligned}
\lceil FA_i(\mathbf{x}) \rceil &= \max\nolimits_{\mathbf{y}\in A_j} \left\{ vis(\mathbf{x},\mathbf{y}) \frac{\left\lceil A_i^{(vis)}(\mathbf{r}_{xy}) \right\rceil \left\lceil A_j^{(vis)}(-\mathbf{r}_{\mathbf{xy}}) \right\rceil}{\pi \|\mathbf{r}_{\mathbf{xy}}\|^2} \right\} \\
\lfloor FA_i(\mathbf{x}) \rfloor &= \min\nolimits_{\mathbf{y}\in A_j} \left\{ vis(\mathbf{x},\mathbf{y}) \frac{\left\lfloor A_i^{(vis)}(\mathbf{r}_{xy}) \right\rfloor \left\lfloor A_j^{(vis)}(-\mathbf{r}_{\mathbf{xy}}) \right\rfloor}{\pi \|\mathbf{r}_{\mathbf{xy}}\|^2} \right\}
\end{aligned}$$

These bounds are compared with the value obtained by direct integration:

$$FA'_i(\mathbf{x}) = A_i \mathbf{n}_i \cdot \left(\sum_{\alpha\, j,\beta} \mathbf{K}_{i,\alpha;j,0} \Phi_{i,\alpha}(\mathbf{x}) \right)$$

to estimate the maximum transfer error factor:

$$\Delta FA_{i,j} = \max_{\mathbf{x} \in A_i} \left\{ \left| FA'_i(\mathbf{x}) - \lceil FA_i(\mathbf{x}) \rceil \right|, \left| FA'_i(\mathbf{x}) - \lfloor FA_i(\mathbf{x}) \rfloor \right| \right\}$$

The transfer error used for refinement is then obtained by:

$$\delta_{i,j} = b_{j,0} \lceil \rho_i \rceil \Delta FA_{i,j}$$

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 [Wil00], 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. $\lceil F_{i,i} \rceil = 1 - \frac{A_i^{(n)}}{A_i}$ $\lfloor F_{i,i} \rfloor = 1 - \frac{\sum_{k=0}^{3} D_k^+ + D_k^-}{A_i}$. We thus estimate the error on self transfer by

$$\delta_{i,i} = b_{i,0} \lceil \rho_i \rceil \left(\lceil F_{i,i} \rceil - \lfloor F_{i,i} \rfloor \right) A_i$$

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

- when using highly tessellated objects, the geometric detail is finer the the irradiance detail. Vector radiosity is beneficial in this case because it can work on a separate element hierarchy, avoiding pushing irradiances to the leafs of the geometric model during the solution phase; by using higher order elements we aim at producing smooth solutions while keeping the subdivision level of the solution hierarchy low;

- 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 irrradiance 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 wavelet 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 leafs 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 A hierarchical vector 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 transfering 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 leafs of the solution hierarchy (see figure 3.3).

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 elements's list of potential shooters with the other top-level elements that have a positive unshot radiosity and are facing towards 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 towards 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 towards 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 enviroment that is transfered

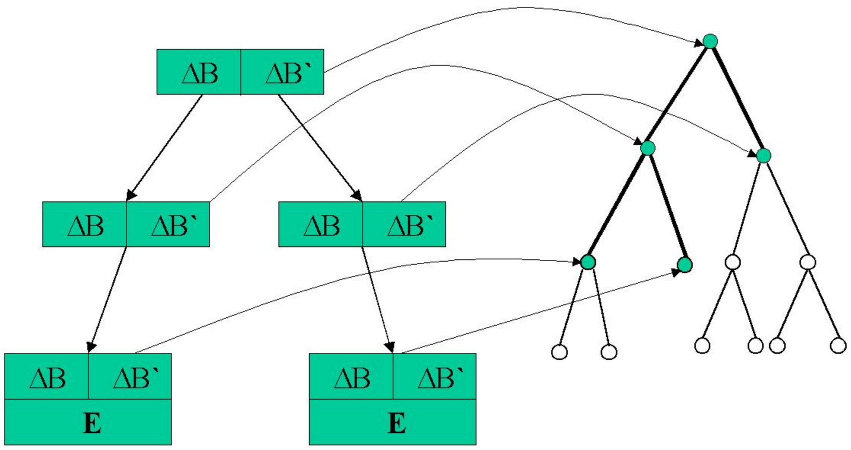


Figure 3.3: Element hierarchy and original face cluster hierarchy. The solver operates only on the element hierarchy and on the clusters directly referenced by them.

directly to element i or at coerser level 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.

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. In the case of constant, linear, and bilinear bases, the OpenGL register combiners extension can be used to render illuminated models directly from the vector irradiance map, exploiting hardware acceleration for computing vertex radiosity on commodity graphics boards. Higher-order bases can be rendered using this method by using subdivision to convert them to a linear representation.

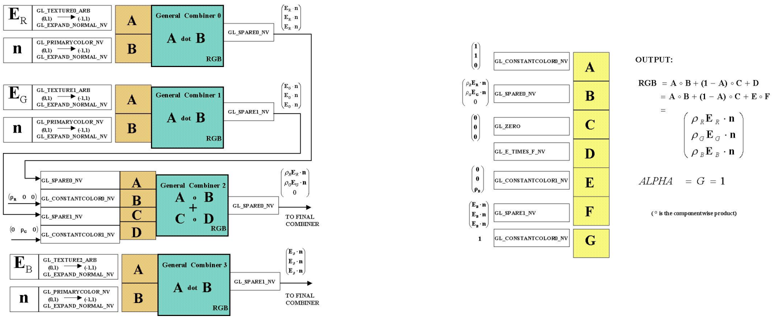
In our approach, the irradiance maps are simulated by three 2x2 textures (GL_TEXTURE0_ARB, GL_TEXTURE1_ARB, GL_TEXTURE2_ARB) associated to the leaf solution element bounding rectangle. Each texture contains the four corner vector irradiance associated to a single primary color. The texture matrix is used to define the transformation from model coordinates to irradiance map coordinates. The normal vector is mapped to GL_PRIMARY_COLOR_NV (using glColor3fv to communicate it to the graphics pipe) and RGB diffuse reflectances are mapped to GL_CONSTANT_COLOR02_NV, GL_CONSTANT_COLOR12_NV and GL_CONSTANT_COLOR1_NV respectively (alphas are all set to one). Normals and textures are linearly interpolated by the

Algorithm 1 Hierarchical Higher Order Face Cluster Radiosity

```
Solve():
    for each top-level face cluster $i$
        create a top-level element $i$
        $\Delta B_{i,\alpha} \leftarrow (e_i, 0, 0, .., 0)$ $\Delta B'_{i,\alpha} \leftarrow 0$ $\mathbf{E}_{i,\alpha} \leftarrow 0$
    repeat
        for each top level element $i$
            for each top level element $j \neq i$
                Assign-shooter($i$,$j$)
                Transport-light($i$,0)
            for each top level element $i$
                $\Delta B_{i,\alpha} \leftarrow \Delta B'_{i\ \alpha}$ $\Delta B'_{i,\alpha} \leftarrow 0$
    until convergence

Assign-Shooter($i$,$j$):
    if $\Delta B_i \neq 0$ and element $i$ is facing element $j$
        push $j$ into $shooters_i$

Transport-Light($i$,$\Delta E_{up}$):
    $\Delta \mathbf{E}_{tmp} \leftarrow pushcoefficients(\Delta \mathbf{E}_{up})$
    while $shooters_i$ not empty
        pop $j$ from $shooters_i$
        compute coupling coefficients $\mathbf{K}_{i,\alpha\,j,\beta}$ and error $\delta_{i,j}$
        switch Oracle($i$,$j$,$\delta_{i,k}$)
            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: $\Delta \mathbf{E}_{tmp,\alpha} \leftarrow \Delta \mathbf{E}_{tmp,\alpha} + \sum_{j,\beta} \mathbf{K}_{i,\alpha;j,\beta} \Delta B_{j,\beta}$
    if Self-Oracle($i$)
        for each child $j$ of $i$
            for each child $k$ of $i$, $k \neq j$
                Assign-Shooter($j$,$k$)
    if i is a leaf
        $\Delta B'_{i,\alpha} \leftarrow (1 - F_{i,i})\rho_i \mathbf{n}_i \cdot \Delta \mathbf{E}_{tmp,\alpha}$ $\mathbf{E}_{i,\alpha} \leftarrow \mathbf{E}_{i,\alpha} + \Delta \mathbf{E}_{tmp,\alpha}$
    else
        $\Delta B'_{i,\alpha} \leftarrow 0$
        for each child $j$ of $i$
            Transport-Light($j$,$\Delta E_{tmp}$)
            $\Delta B'_{i,\alpha} \leftarrow \Delta B'_{i,\alpha} + pullcoefficients(\Delta B'_{j,\alpha})$
```


$\mathbf{E}_R$
GL_TEXTURE0_ARB
(0,1) → (-1,1)
GL_EXPAND_NORMAL_NV
n
GL_PRIMARYCOLOR_NV
(0,1) → (-1,1)
GL_EXPAND_NORMAL_NV
A
B
General Combiner 0
A dot B
RGB
GL_SPARE0_NV
$\begin{pmatrix}\mathbf{E}_R\cdot\mathbf{n}\\ \mathbf{E}_R\cdot\mathbf{n}\\ \mathbf{E}_R\cdot\mathbf{n}\end{pmatrix}$
$\mathbf{E}_G$
GL_TEXTURE1_ARB
(0,1) → (-1,1)
GL_EXPAND_NORMAL_NV
n
GL_PRIMARYCOLOR_NV
(0,1) → (-1,1)
GL_EXPAND_NORMAL_NV
A
B
General Combiner 1
A dot B
RGB
GL_SPARE1_NV
$\begin{pmatrix}\mathbf{E}_G\cdot\mathbf{n}\\ \mathbf{E}_G\cdot\mathbf{n}\\ \mathbf{E}_G\cdot\mathbf{n}\end{pmatrix}$
GL_SPARE0_NV
$(\rho_R\ 0\ 0)$
GL_CONSTANTCOLOR0_NV
GL_SPARE1_NV
$(0\ \rho_G\ 0)$
GL_CONSTANTCOLOR1_NV
A
B
C
D
General Combiner 2
A ∘ B + C ∘ D
RGB
GL_SPARE0_NV
$\begin{pmatrix}\rho_R\mathbf{E}_R\cdot\mathbf{n}\\ \rho_G\mathbf{E}_G\cdot\mathbf{n}\\ 0\end{pmatrix}$
TO FINAL COMBINER
$\mathbf{E}_B$
GL_TEXTURE2_ARB
(0,1) → (-1,1)
GL_EXPAND_NORMAL_NV
n
GL_PRIMARYCOLOR_NV
(0,1) → (-1,1)
GL_EXPAND_NORMAL_NV
A
B
General Combiner 3
A dot B
RGB
GL_SPARE1_NV
$\begin{pmatrix}\mathbf{E}_B\cdot\mathbf{n}\\ \mathbf{E}_B\cdot\mathbf{n}\\ \mathbf{E}_B\cdot\mathbf{n}\end{pmatrix}$
TO FINAL COMBINER
$\begin{pmatrix}1\\1\\0\end{pmatrix}$
GL_CONSTANTCOLOR0_NV
A
$\begin{pmatrix}\rho_R\mathbf{E}_R\cdot\mathbf{n}\\ \rho_G\mathbf{E}_G\cdot\mathbf{n}\\ 0\end{pmatrix}$
GL_SPARE0_NV
B
$\begin{pmatrix}0\\0\\0\end{pmatrix}$
GL_ZERO
C
GL_E_TIMES_F_NV
D
$\begin{pmatrix}0\\0\\\rho_B\end{pmatrix}$
GL_CONSTANTCOLOR1_NV
E
$\begin{pmatrix}\mathbf{E}_B\cdot\mathbf{n}\\ \mathbf{E}_B\cdot\mathbf{n}\\ \mathbf{E}_B\cdot\mathbf{n}\end{pmatrix}$
GL_SPARE1_NV
F
1
GL_CONSTANTCOLOR0_NV
G
OUTPUT:
$\mathrm{RGB} = \mathrm{A}\circ\mathrm{B} + (1-\mathrm{A})\circ\mathrm{C} + \mathrm{D} = \mathrm{A}\circ\mathrm{B} + (1-\mathrm{A})\circ\mathrm{C} + \mathrm{E}\circ\mathrm{F} = \begin{pmatrix}\rho_R\mathbf{E}_R\cdot\mathbf{n}\\ \rho_G\mathbf{E}_G\cdot\mathbf{n}\\ \rho_B\mathbf{E}_B\cdot\mathbf{n}\end{pmatrix}$
$ALPHA = G = 1$
(∘ is the componentwise product)

(a) General combiners setup

(b) Final combiner setup

Figure 3.4: OpenGL register combiners setup for computing vertex radiosity from a linear vector irradiance map

hardware by selecting glShadeModel(GL_SMOOTH) and glTexParameteri(GL_TEXTURE_MAG_FILTER, GL_LINEAR). Figure 3.4 illustrates the OpenGL register combiners setup for computing vertex radiosity from a linear vector irradiance map.

Chapter 4

Implementation and Results

An experimental software library and a radiosity renderer application supporting the hierarchical higher order face cluster radiosity algorithm described in this report has been implemented and tested on Linux, Silicon Graphics IRIX and Windows NT machines. The software supports combinations of constant, linear, bilinear, quadratic, and cubic bases for representing radiosity and vector irradiance functions. We have implemented both a gathering solver based on the Jacobi iteration and the linkless shooting solver discussed in this report.

The preliminary results presented here were obtained on a Dell Inspiron 8100 laptop with a Pentium III 1.13GHz and 512 MB RAM running Linux (kernel 2.4). We plan to expand this section in the updated version of this report.

As in Willmott [Wil00], a multiresolution model is stored using a face cluster table, a triangle table (with tree 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 reflectance. 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.

Using our 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. The typical combinations we select are:

- a constant basis for radiosity and a linear basis for vector irradiance (3 coefficients); this combination requires 60 bytes for an internal node and 168 bytes for a leaf node;
- a linear basis for radiosity and a quadratic basis for for vector irradiance (6 coefficients); this combination requires 168 bytes for an internal node and 384 bytes for a leaf node.

In the example presented here, global illumination is computed for a scene containing a highly tessellated object (the Cyberware Venus head, 100K triangles), positioned near three flat colored walls and illuminated by an area light source (see figure 4.1). Preprocessing time takes 23 s, and the memory required for the geometric model is about 24MB. The preprocessing time can be amortized over multiple renderings. Moreover, since the solution

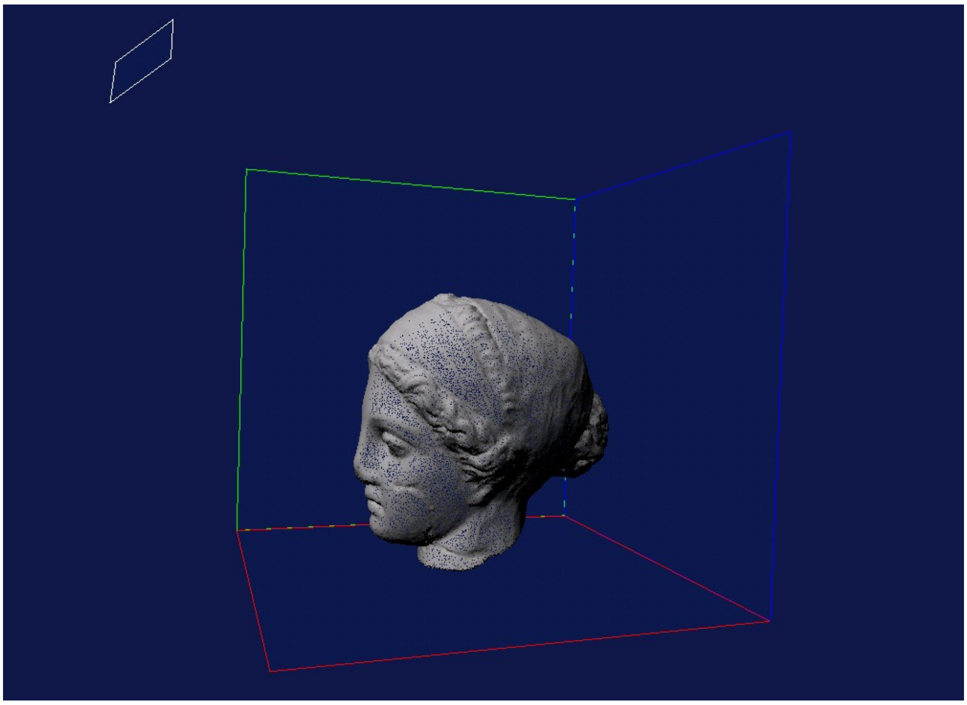


Figure 4.1: Test scene setup, rendered in wireframe to show initial model tessellation.

hierarchy is separate from the model hierarchy, multiple instances of the same model may be referenced in the same scene.

Figure 4.2 presents two solutions computed using constant bases for both irradiance vectors and radiosity, which corresponds to the original face cluster radiosity algorithm. Both images where produced with four shooting iterations. The left image has a link error theshold of 0.001 times the power of the emitter, while the left image has a link error threshold ten times smaller. The rendering time for the left image was 8 s, and the number of leaf elements in the solution is 2403. The higher quality rendering took 120 s and produced 16676 leaf elements. Storage costs for the solution hierarchy range from 165KB to 1.54Mb. While the number of elements is sensibly smaller than the the number of input polygons, fine illumination effects are clearly visible. Blocking effects are however clearly visible even in the higher quality image.

Figure 4.3 presents a solution computed using constant bases for the radiosity, but linear

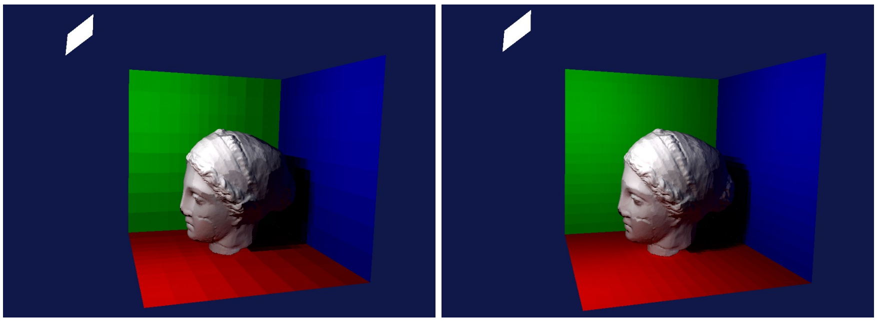


Figure 4.2: Renderings of the venus with constant radiosity basis and constant irradiance basis, using two different link errors.

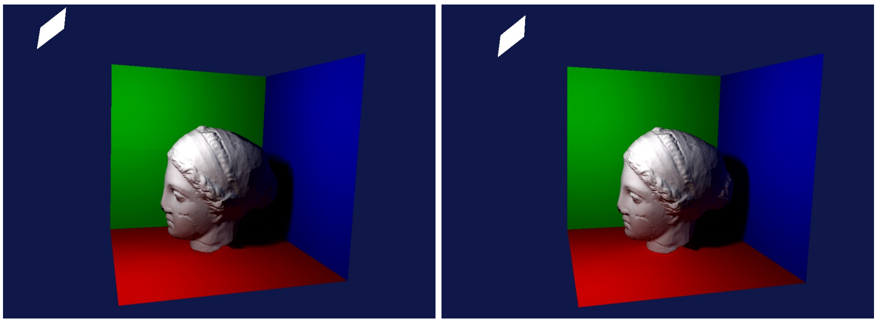


Figure 4.3: Rendering of the venus with constant radiosity basis and linear irradiance basis, using two different link errors.

bases for irradiance vectors, using similar renderer settings. The left rendering took 8 s and produced 2370 leaf elements, while the right rendering took 122 s and produced 16654 leaf elements. Storage costs for the solution hierarchy range from 545KB to 3.8Mb. Rendering times are similar to the previous ones, since they are dominated by visibility computations, that use the same cubature rules. However, both solutions are clearly smoother than the higher quality solution using constant bases.

Chapter 5

Conclusions and Future Work

We have presented an algorithm for simulating diffuse interreflection in scenes composed of highly tessellated objects. The 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 in areas where global illumination has a good approximation using the selected irradiance basis. The OpenGL register combiners extension can be used to render illuminated models directly from the vector irradiance map, exploiting hardware acceleration for computing vertex radiosity on commodity graphics boards.

Our current work is concentrating on improving the implementation of the prototype renderer and on evaluating the effect of the various accuracy parameters on rendering quality and speed. We are also planning to incorporate normal bounds information to speed-up visibility testing and improve the quality of the projected areas estimation. As for the original face cluster radiosity method, a detailed analysis of the error introduced by face clustering, and in particular by the clipping of vector irradiance during the push phase, is not available. This is an important area for future work.

We are also planning to to improve the rendering of the illuminated geometry by adding specular reflections as a visual effecty at rendering time. Since a smooth irradiance map is available, we expect that a good range of specular BRDFs would be rendered with acceptable visual results.

Acknowledgments

The authors would like to thank Cyberware for making the Venus model available to the research community.

This research is partially supported by the DIVERCITY project (EU-IST-13365), funded under the European IST programme (Information Society Technologies). We also acknowledge the contribution of Sardinian regional authorities.

Bibliography

[ACP^{+}01] 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):169–201, July 2001.

[Air90] John M. Airey. *Increasing Update Rates in the Building Walkthrough System with Automatic Model-Space Subdivision and Potentially Visible Set Calculations*. PhD thesis, University of North Carolina, July 1990.

[BG01] 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):115–134, January 2001.

[BW96a] Philippe Bekaert and Yves Willems. Error control for radiosity. In Xavier Pueyo and Peter Schröder, editors, *Eurographics Rendering Workshop 1996*, pages 153–164, New York City, NY, June 1996. Eurographics, Springer Wien. ISBN 3-211-82883-4.

[BW96b] Phillipe Bekaert and Yves D. Willems. HIRAD: A hierarchical higher order radiosity code. In Werner Purgathofer, editor, *12th Spring Conference on Computer Graphics*, pages 213–227. Comenius University, Bratislava, Slovakia, June 1996. ISBN 80-223-1032-8.

[CAH00] F. Cuny, L. Alonso, and N. Holzschuch. A novel approach makes higher order wavelets really efficient for radiosity. In M. Gross and F. R. A. Hopgood, editors, *Computer Graphics Forum (Eurographics 2000)*, volume 19(3), pages 99–108, 2000.

[Che90] Shenchang Eric Chen. Incremental radiosity: An extension of progressive radiosity to an interactive image synthesis system. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, pages 135–144, August 1990.

[CLSS97] Per H. Christensen, Dani Lischinski, Eric J. Stollnitz, and David H. Salesin. Clustering for glossy global illumination. *ACM Transactions on Graphics*, 16(1):3–33, January 1997. ISSN 0730-0301.

[DB00] Reynald Dumont and Kadi Bouatouch. Combining hierarchical radiosity with LODs. In *Proc. IASTED International Conference on Graphics and Imaging*, November 2000.

[DBG99] Reynald Dumont, Kadi Bouatouch, and Philippe Gosselin. A progressive algorithm for three point transport. In David Duke, Sabine Coquillart, and Toby Howard, editors, *Computer Graphics Forum*, volume 18(1), pages 41–56. Eurographics Association, 1999.

[FTSK96] Thomas Funkhouser, Seth Teller, Carlo Sequin, and Delnaz Khorramabadi. The UC berkeley system for interactive visualization of large architectural models. *Presence, the Journal of Virtual Reality and Teleoperators*, 5(1):13–44, 1996.

[GD99] Xavier Granier and George Drettakis. Controlling memory consumption of hierarchical radiosity with clustering. In I. Scott MacKenzie and James Stewart, editors, *Proceedings of the Conference on Graphics Interface (GI-99*, pages 58–65, Toronto, Ontario, June 2–4 1999. CIPS.

[GH96] S. Gibson and R. J. Hubbold. Efficient hierarchical refinement and clustering for radiosity in complex environments. *Computer Graphics Forum*, 15(5):297–310, 1996. ISSN 0167-7055.

[GSHG98] Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. The irradiance volume. *IEEE Computer Graphics and Applications*, 18(2):32–43, March/April 1998.

[GWH01] Michael Garland, Andrew Willmott, and Paul S. Heckbert. Hierarchical face clustering on polygonal surfaces. In *ACM Symposium on Interactive 3D Graphics*, pages 49–58, 2001.

[HDSD99] 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)*, pages C–221–C–232, September 1999.

[MBMD98] D. Meneveaux, K. Boauatouch, E. Maisel, and R. Delmont. A new partitioning method for architectural environments. *Journal of Visualization and Computer Animation*, 9(4):195–213, October/November 1998.

[MSF00] Gordon Müller, Stephan Schäfer, and W. Dieter Fellner. Automatic creation of object hierarchies for radiosity clustering. In David Duke, Sabine Coquillart, and Toby Howard, editors, *Computer Graphics Forum*, volume 19(4), pages 213–221. Eurographics Association, 2000.

[RPV93] Holly E. Rushmeier, Charles Patterson, and Aravindan Veerasamy. Geometric simplification for indirect illumination calculations. In *Proc. Graphics Interface '93*, pages 227–236, Toronto, Ontario, May 1993. Canadian Inf. Proc. Soc.

[SAG94] Brian Smits, James Arvo, and Donald Greenberg. A clustering algorithm for radiosity in complex environments. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 435–442. ACM Press, 1994.

[SD95] Francois Sillion and George 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)*, pages 145–152, 1995.

[SDS95] Francois Sillion, George Drettakis, and Cyril Soler. A Clustering Algorithm for Radiance Calculation in General Environments. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 196–205, New York, NY, 1995. Springer-Verlag.

[Sil95] Francois X. Sillion. A Unified Hierarchical Algorithm for Global Illumination with Scattering Volumes and Object Clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):240–254, September 1995.

[SSSS98] M. Stamminger, H. Schirmacher, Ph. Slusallek, and H.-P. Seidel. Getting rid of links in hierarchical radiosity. In David Duke, Sabine Coquillart, and Toby Howard, editors, *Computer Graphics Forum*, volume 17(3), pages 165–174. Eurographics Association, 1998.

[TFFH94] Seth Teller, Celeste Fowler, Thomas Funkhouser, and Pat Hanrahan. Partitioning and ordering large radiosity computations. In Andrew Glassner, editor, *Proceedings of SIGGRAPH ’94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 443–450. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[WH97] Andrew J. Willmott and Paul S. Heckbert. An empirical comparison of progressive and wavelet radiosity. In Julie Dorsey and Philipp Slusallek, editors, *Eurographics Rendering Workshop 1997*, pages 175–186, New York City, NY, June 1997. Eurographics, Springer Wien. ISBN 3-211-83001-4.

[WHG99] Andrew J. Willmott, Paul S. Heckbert, and Michael Garland. Face cluster radiosity. In Dani Lischinski and Greg Ward Larson, editors, *Rendering Techniques ’99*, Eurographics, pages 293–304. Springer-Verlag Wien New York, 1999.

[Wil00] Andrew J. Willmott. *Hierarchical Radiosity with Multiresolution Meshes*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, November 2000. Available from http://reports-archive.adm.cs.cmu.edu/anon/2000/abstracts/00-166.html.

[Zat93] Harold R. Zatz. Galerkin radiosity: A higher order solution method for global illumination. In James T. Kajiya, editor, *SIGGRAPH 93 Conference Proceedings*, Computer Graphics Proceedings, Annual Conference Series, pages 213–220. ACM SIGGRAPH, ACM Press, August 1993.