EUROGRAPHICS '99
Tutorial T1
Advanced Radiosity:
Complex Scenes and Glossy Reflections

Marc Stamminger, MPI Informatik, Saarbrücken, Germany
Daniel Wexler, Pacific Data Images, USA
Wolfram Kresse, Fraunhofer Institute for Computer Graphics, Darmstadt, Germany
Nicolas Holzschuch, ISA research team, INRIA-Lorraine, Nancy, France
Per H. Christensen, Square USA, USA

Abstract

A lot of research towards global illumination has been focussed on the radiosity method. Nevertheless, it is still a rather academic topic which finds very slowly its way into commercial products. The scope of this tutorial is to describe recent developments in radiosity research that might narrow the gap with commercial applications. The first part of the tutorial course will be given by a pioneer in commercial computer graphics, who will set the stage for the demands of commercial rendering products and assess why radiosity has not been used until now.

Daniel Wexler

Pacific Data Images
3101 Park Blvd., Palo Alto, CA 94306, USA
wexler@pdi.com

1 Global Illumination at PDI

Global illumination techniques for complex scenes have advanced to the point where they are just becoming practical for use on feature film production. The film Antz had an average scene complexity measuring in the tens of millions of primitives, and our next feature Shrek may average over a hundred million primitives. We must render tens of thousands of test frames each week and final frames must include depth of field, motion blur, volumetric effects and must render in under eight hours. These requirements alone would daunt even the newest global illumination techniques, but this talk will show why, despite advances in algorithms and computing power, the production industry may forever eschew physically correct lighting algorithms. The aesthetic requirements of an animator often require violating physical laws and success is often achieved by providing a tool which is more controlable and understandable for artists, not researchers.

Marc Stamminger

Max-Planck-Institute for Computer Science
Im Stadtwald, 66123 Saarbrücken, Germany
stamminger@mpi-sb.mpg.de

2 Radiosity

This section will briefly review the basics of global illumination computations with finite elements.[1] The goal is to describe the common basis for the following lectures, which will then pick out interesting aspects and focus to them.

2.1 Radiometry

The following physical quantities are used to describe light physically. A detailled description can be found in [3].

Flux is the power emitted by or arriving at a surface in the form of light. It has unit Watts $[W]$.

Radiosity is the flux per unit area leaving a surface. Its unit is Watts per square meter $[W/m^2]$.

Irradiance is flux per unit area arriving at a surface. It has the same unit as radiosity.

Radiance is the flux per projected unit area and solid angle arriving at or emitted by a surface point. The unit is Watts per square meter and steradian $[W/m^2 sr]$.

Intensity is flux per solid angle. It is typically used to describe point light sources. Its unit is $[W/sr]$.

The properties of these photometric quantities, their derivation, physical interpretation and their limitations will not be described in more detail here. A broad number of publications treats these issues in depth. Good ones, suitable for the context of computer graphics, can be found in [41, 3, 23].

2.2 The Rendering Equation

The rendering equation [28] is a simple formulation of the global illumination problem as an integral equation. It describes the equilibrium of light exchange in a scene, neglecting atmospheric and wave optics effects like interference. The informal description of the rendering equation is simple: the radiance of a surface point y in some direction ω is the self emitted radiance of y in ω plus the incident light at y reflected into direction ω. To compute the reflection, the incident light at y is integrated over the incident hemisphere Ω and weighted by the BRDF of y:

$$L(y,\omega) = L^e(y,\omega) + \int_\Omega f_r(\omega, y, \omega') L(\mathrm{ray}(y,\omega'), -\omega') n(y) \circ d\omega' \tag{1}$$

The function $n(x)$ is the surface normal in x, $\mathtt{ray}(y,\omega')$ returns the point visible from y in direction ω' and $x \circ y$ is the scalar product of x and y, if it is positive, and zero otherwise. Note that in the

[1] More correctly, the approach is a boundary element approach. In the context of global illumination, however, the name 'finite element methods' has been established.

original work of Kajiya, a different form of the rendering equation with other quantities, namely intensity instead of radiance, was used. For consistency reasons and since the above notation has been established as a standard, the radiance formulation is here. A detailled description of the rendering equation in the form of Equ. 1 can be found for example in [41].

The radiosity equation is a special case of the above rendering equation. It assumes that all surfaces in the scene are diffuse. As a result, the 4D radiance functions L and L^e can be replaced by the corresponding 2D radiosity functions B and E and the 6-dimensional general BRDF can be substituted by the 2D reflectance $\rho(x)$:

$$B(x) = E(x) + \rho(x) \int_S G(x,y) B(y)\, dy. \tag{2}$$

Since the integral is no longer parameterized over the directional domain, but over the scene S, a geometric term G is introduced to account for this:

$$G(x,y) = \frac{(n(x) \circ (y-x))\,(n(y) \circ (x-y))}{||x-y||^4} V(x,y). \tag{3}$$

$V(x,y)$ is the visibility function, which is 1 if and only if x and y are mutually visible. Since the integration is over all surface points as potential senders, the visible ones have to be filtered out by V. The BRDF f_r gets a triple of points (x,y,z) and defines the reflection at y for light coming from point x and reflection towards point z.

2.3 Classical Radiosity

In the first publication on radiosity [20], the radiosity distribution $B(x)$ is approximated by partitioning all surfaces into a finite set of patches $\{P_i\}$. $B(x)$ is assumed to be constant over each of these elements, the constant radiosity value of patch i is described by its average radiosity value B_i. Also the emission $E(x)$ of patch i is approximated by a constant value E_i.

The radiosity problem can be solved approximately using a finite element approach with the above discretization. Each patch plays the role of a finite element. A Galerkin ansatz leads to a linear system of equations in the values B_i [41, 10]:

$$B_i = E_i + \rho_i \sum_j F_{ij} B_j. \tag{4}$$

The values F_{ij} are called *form factors* and can be computed as:

$$F_{ij} = \frac{1}{A_i} \int_{P_i} \int_{P_j} G(x,y)\, dx\, dy, \tag{5}$$

Equ. 4 can be solved directly by $B = (1-\rho F)^{-1} E$. Inverting the matrix requires the time $\mathcal{O}(n^3)$ (or $\mathcal{O}(n^{2.807})$ [35]), which increases the complexity of the algorithm from quadratic to cubic. Using iterative methods that directly unroll the Neuman-Series in Equ. 4, approximate solutions can be found in the time $\mathcal{O}(n^2)$ using Jacobi- or Gauss-Seidel iteration. The form factor matrix F is in practice so well-behaved that these methods converge sufficiently after a few iterations.

One big advantage of the radiosity method is its view-independence. Once a solution has been computed, different views of it can be rendered quickly, even achieving interactive framerates if appropriate hardware is available. A smoothing rendering step is always neccessary to cover up the division into patches (see figure 1).

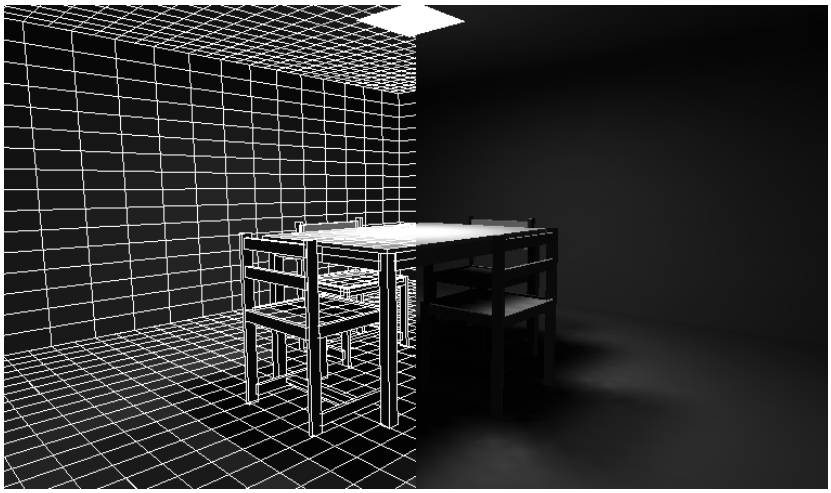

Figure 1: Uniform mesh and solution created by classical radiosity.

2.4 Progressive Refinement

As an alternative to gathering, the *shooting* scheme is introduced by Cohen et al. along with the *Progressive Refinement* method [9]. In this method, one patch (usually the currently brightest), is selected. The energy of this patch is then distributed into the scene, resulting in an increase of the radiosity of the illuminated objects, which then, in turn, can become shooters in later iterations.

Form factors are determined exploiting graphics hardware. For each shooter the scene is drawn to a half cube, which is centered at the shooter's midpoint and covers the frontfacing half-space of the sender. Each pixel on the hemicube corresponds to some form factor. A form factor approximation for a receiver is obtained by summing up all form factors for the pixels the receiver covers. The computations are very fast, but due to the fixed resolution of the hemi-cube the result is prone to aliasing artifacts.

2.5 Hierarchical Radiosity and Clustering

In [24] a hierarchical approach for solving the radiosity problem was presented. The algorithm does not use an a priori created fine mesh of the scene, but subdivides the patches during the progress of the computation. As a result, the created mesh is adaptive, i.e. ideally the patches are finely subdivided, where radiosity changes most, and only a coarse subdivision is created in approximately constant regions.

The idea of the algorithm is rather simple, and all following methods are based on the same principle. To compute the light transport from P to Q, a so called *refiner* or *oracle* estimates the error introduced by linking P and Q at the current level, i.e. by transporting the radiosity from P to Q via a single form factor. If this estimated error is below some user definable error threshold, the form factor is computed, stored in a so called *link*, and the light is transported via that link.

Otherwise, if the error will probably be too large, sender or receiver are refined (for example the larger one), and the light transport is computed recursively between the smaller children. If the receiver is refined, the representation error is decreased. After a subdivision of the sender, light is transported via four form factors, which are more accurate than only one. Fig. 2 shows the result of a hierarchical

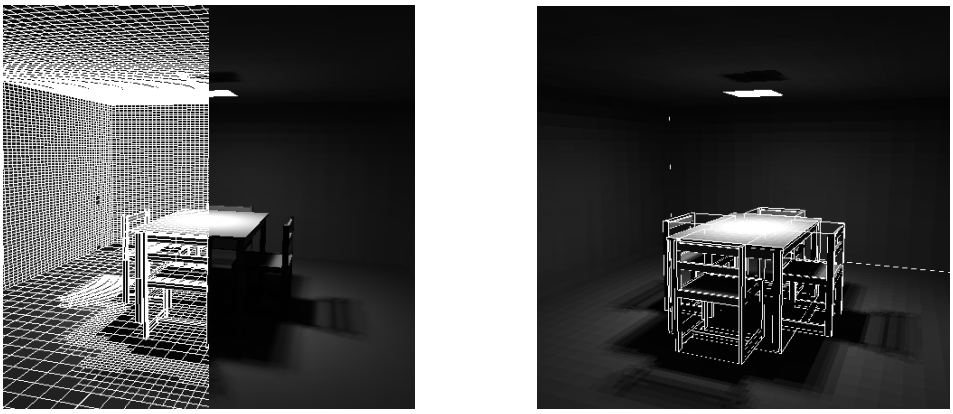

Figure 2: Left: Mesh and solution created by hierarchical radiosity. Right: Clustering of the same scene.

radiosity computation for a standard typical scene.

The above approach allows us to compute the light transport between two patches P and Q in a hierarchical way. In the original algorithm, one light transport step through the scene is computed by applying the above hierarchical transport scheme to all pairs of primitives. For complex scenes with many small objects, this "initial linking", however, dominates computation time again.

As a solution to the initial linking problem, clustering algorithms have been developed, which do not only create a hierarchy below the primitives, but also build a hierarchy upon them [44, 42, 18]. The computation time can be improved enormously, if not only patches but also clusters can interact. Therefore, a method is needed to compute form factors between patches and clusters. Note that in the presence of non-convex objects (as for instance clusters), self form factors become necessary. These describe the amount of light which is emitted by an object and then hits the same object again, before it can leave it.

If a cluster hierarchy has been built upon the primitives, the hierarchical radiosity method described above, which only works hierarchically below the primitives, can be extended upwards. The algorithm uses one initial link only, which transports light from the root cluster to itself. If this link is considered to be too coarse, which is almost always the case, the root cluster is subdivided, and the interactions between all pairs of children are considered. This way, for instance the light transport between two distant chairs, each one in a cluster, can be computed with one single form factor instead of considering each pair of patches of the chairs. The complexity of the computation becomes $\mathcal{O}(n)$ in the best case and is thus independent of the initial scene complexity k.

Wolfram Kresse

Fraunhofer Institute for Computer Graphics (IGD)
Department Visualization and Virtual Reality
Rundeturmstr. 6, 64283 Darmstadt, Germany
wkresse@igd.fhg.de

3 Making Radiosity Usable

3.1 Introduction

The development of the first radiosity algorithms for realistic lighting simulation happened fifteen years ago, but still radiosity systems are not in everyday's use and are just starting to emerge in commercial products. There is a growing awareness of this issue in several industries which would directly benefit from advanced lighting simulations. Applications for high-quality image synthesis include simulation for the lighting, building and public works industries, architectural and product design, entertainment industry, virtual studios and networked/distributed visualisation systems used, for instance, for virtual shopping. The use of this technology is currently limited mainly by problems in treating realistic models, the (automatical) preparation of CAD scenes, and the complex parameter selection of the algorithm itself.

This talk will concentrate on the various issues a radiosity system must address to be usable and to get accepted by a non-expert user. It will be structured as follows:

Preprocessing issues Importing data (geometrial as well as photometrical) from other industrial processes is very difficult because of the specific constraints placed by radiosity modules. Normal consistency and correct handling of special geometries (e.g., non-planar polygons) must be assured.

Complex data sets A usable radiosity algorithm must be able to process scenes of arbitrary size. The limited amount of available memory is an important issue. Intelligent clustering methods capable of automatically creating a clustering hierarchy are essential.

User requirements Refinement criteria are often not flexible enough to sufficiently handle various scene types or requirements of different user types. To make a radiosity system usable, it has to be capable of automatically adapting to different user- and scene requirements, and spend the computational effort in places of high importance for a specified user type.

Lighting simulation control The core of the radiosity algorithm is controlled by a huge number of parameters which require distinct knowledge of every aspect and principle of the underlying algorithm. The number of parameters has to be reduced, and their application must be automated as good as possible.

Dynamic environments In the design phase, a user of a radiosity system needs very fast updates of the solution after any scene modifications (e.g., adding/deleting/moving objects, changing light source emission characteristics, changing a surface's reflectance properties). A naive approach would force a complete re-evaluation of the global lighting situation for each modification, causing unacceptable computational costs.

The goal of the European Union LTR Project ARCADE is to provide the bridge between state-of-the-art research technology and usable systems which can be integrated into commercial products. To achieve this goal each of the shortcomings of existing systems are being attacked.

3.2 Application Requirements and Pre-Processing

One of the key reasons for the failure of radiosity systems to have won wide take-up, and probably the first hurdle encountered by any potential user, is their unfriendliness towards commonly-available data. Raw 3D CAD data will generally be unsuitable for direct use radiosity systems, since such systems have specific requirements concerning the input data.

In the first step it must be provided that the data is processed so as to remove/resolve anything which the internal system might have difficulty with. These can be things like:

- concave polygons
- polygons with holes
- self-intersecting polygons
- non-planar polygons
- non-polygonal input (NURBS, ...)

Non-conformant input geometry has to be converted to radiosity-conformant input geometry, triangulation has to be performed if the radiosity system does not support non-planar geometry.

In addition to *geometrical* issues, problems can arise in the context of light source definitions. A usable radiosity system should provide different methods of specification (e.g., photometric/radiometric) of such data. In addition, luminaire descriptions from standard industrial formats (IESNA, CIE, ...) must be supported.

After processing the data, several other problems have to be taken care of in preprocessing steps, such as:

- inconsistent normal orientation
- holes and missing adjacency information in tesselated objects
- mesh reduction for pre-meshed and possibly ill-triangulated geometries.

For larger data sets it is imperative that all these steps are performed automatically.

3.3 Data Complexity

Several issues concerning complex data sets have to be addressed:

- The construction of a suitable cluster hierarchy. Early clustering algorithms were suitable rather for research purposes, than for everyday's use. Both the construction of the cluster hierarchy, as well as the underlying data structures used need to be optimized to reduce artifacts or unexpected behavior [25]. The hierarchy construction must be performed automatically.

- Meshing data structures. The data structures used for subdivided polygons must be robust and as slim as possible, to ensure the ability to handle very complex scenes with limited memory.

- Hierarchical treatment of natural light sources. Apart from artificial lighting, the incorporation of skylight and sunlight, in a manner which conforms to emerging standards [13], is crucial. Natural lighting due to sunlight and skylight should be accurately and efficiently simulated within the context of a hierarchical radiosity algorithm with clustering [12].

3.4 Application Scenarios

The term "lighting simulation" has many different aspects and characteristics. One quickly realizes that those expectations are highly dependent on the requirements of the user, and even on the *type* of the user itself. A radiosity system can compute physical illumination properties and visualize them e.g. as still images or with real-time rendering. These computations can sometimes be done quickly, but they can also often require considerable computation time. In general the result will become more exact the more effort is spent on the simulation. The visualization of the illumination distribution can be done, for example, within an interactive lighting or architectural design system, or the result can be computed in a preprocessing step, visualizing it afterwards.

3.4.1 Definition of user groups

From these aspects one can identify certain user groups, each with different needs:

- VR/Real-time: A VR user is mainly interested in a result which must not be too complex so that it can still be represented in real-time. He is not interested in precise physical values of the used light sources and will be pleased if the visible artifacts are reduced to a minimum. He definitely wants to take advantage of the linear representation of the light distribution using Gouraud shading. He might also require a certain interaction with the scene objects and materials.

- Lighting Engineers: The lighting engineer does not really care about nice images, he is instead interested in highly accurate and reliable simulations of direct and indirect lighting situations for different physical light sources.

- Animation: Animation sequences depend on a high visual accuracy and do not allow for any kind of artifacts, either static or varying over time (e.g. temporal aliasing). Since the whole sequence will be constructed beforehand, no interaction is necessary during the computation phase.

- CAD User: CAD people might be viewed as a mixture of these three types. While not opting for a perfect visual representation, they do require a certain correctness and quality; interaction and quick updates are essential.

One important requirement that all groups have in common is memory. A usable radiosity system has to take great care of the memory consumption, otherwise it will be unsuitable for large and complex scenes.

Together with a compilation of typical problems of current radiosity systems the requirements of the user groups can then be compared and prioritized. The result is an overview of different goals with respect to the relevance to each user group (3—very important, 2—nice to have, 1—don't care).

	VR	Lighting Engineers	Animation	CAD User
accuracy (physical)	1	3	2	1
accuracy (visual)	2	1	3	2
visual artifacts	2	1	3	2
memory issue	3	3	3	3
low mesh complexity	3	1	1	2
interaction	1 or* 3	1	1	3
dynamics	1 or* 3	1	3	2
linear representation	3	1	3	2

*depending on simple walkthrough or interactive environment

Note the difference between *physical* and *visual* accuracy: A physically precise simulation does not necessarily have to produce a high quality visual representation. If only the illumination values are required there will be no need to add additional effort to reduce visual artifacts. On the other hand, a visual result does not necessarily have to be physically correct to be good. In extreme situations one will even allow "illegal" shading situations just to create the desired optical effect (e.g. in the entertainment industry [1]).

3.4.2 Definition of scene types

Similar to the user groups, different scene types exist, each posing different requirements on the refinement criterion and its parameters, the involved lighting algorithms, and probably also the construction of the clustering hierarchy:

architectural scenes: mainly large, planar faces (walls, floor) which necessitate a fine subdivision.

car interiors: many curved surfaces, often already tesselated into tiny polygons.

interior scenes: several artificial light sources, objects closely grouped together.

exterior scenes: daylight simulation, standalone objects often distributed within a plane.

entertainment: any imaginable type of scene.

3.4.3 Basis functions for sender and receiver

The VR group will definitely make use of hardware supported Gouraud shading. This suggests using linear basis functions on the receiver. The only artifact compared to higher orders are Mach bands, which can be avoided by appropriate (e.g., gradient based) refinement criteria.

For the sender part, it appears that a linear representation here does not influence the visual result since the sender radiosities are not directly represented in the result and thus cannot cause any visible artefacts; they are important only for the energy transfer criterion.

Higher order basis functions therefore are not well suited for a practical radiosity system:

- Computational and storage complexity will increase rapidly with every additional order [52].
- Gouraud shading used for fast display is limited to linear interpolation.
- The human visual system is incapable of (or at least very insensitive to) perceiving breaches (i.e. discontinuities) in the second derivative or higher of intensity functions.

We therefore choose the constant sender and the linear receiver scenario. Even the animation user, where visual quality is very important, can be fully satisfied with this constant sender/linear receiver combination because of the properties of the human visual system. As long as the mesh is sufficiently subdivided so that Mach bands disappear, the perceived visual quality of a smooth function will not increase with an interpolation order higher than linear.

3.5 Control and Automatic Steering

3.5.1 Parameters

The radiosity system itself involves a number of independent modules (e.g., meshing, refinement, visibility, visualization, ...). Each module has its own list of independent parameters, which are regularly empirically defined, fail to relate to any quantity with which the user is familiar and can therefore only be applied by advanced and experienced users. A poor choice of parameters usually results in glaring image artifacts or inefficient radiosity solutions. The number and complexity of parameters therefore has to be reduced, and parameter selection should be automated as much as possible.

3.5.2 Refinement

Historically, refinement oracles have been based on the estimation of maximum energy transfers [24]. However, such a simple criterion will either give poor results or over-refinement in many places.

Starting from the constant radiosity assumption, Lischinski et al. [31] find that instead of using just the amount of energy being transported (i.e., the resulting intensity) for the refinement criterion one should rather try to integrate the assumption of piecewise constant functions directly: bounds for the radiosity function are developed to decide if the function is constant, or if it is diverging too much from a constant function, in which case the refinement criterion will force a subdivision.

Such approaches can be extended further, using linear functions to represent the radiosity on the receiver. One approach might be to extend Lischinski's idea of bounding the radiosity function with constant bounds, to linear bounds [34].

Another possibility is to combine Lischinski's constant error bounds with a gradient based criterion [29, 39]. Vertex radiosities can be used, to help avoid additional sources of error, when extrapolating midpoint radiosities to patch vertices.

The refinment oracle should therefore be based at least on the following informations:

- a bound on the radiosity on the sender area (deviation from the assumption "constant")
- a bound on the energy transported between sender and receiver
- and a bound on the variation of radiosity on the receiver (deviation from the assumption "linear")

3.5.3 Error types

It is important to notice that there are fundamentally different kinds of errors, within a radiosity simulation.

First, there is the *error in the energy transfer* between a sending and a receiving patch. The amount of transferred energy is computed using the form factor between the two patches involved. However, since it would be too expensive (and until recently even impossible [38]) to compute an exact form factor, estimates are used. Other sources where errors can be introduced exist, for example the radiosity distribution on the sender, which is assumed to be constant for the form factor computation, or incorrect visibility determination for partially occluded situations - especially for area-to-area visibility tests it is not sufficient to provide a simple percentage value: depending on where a point is located on the receiver the amount of the sender that is visible will vary. Therefore the visibility has to be described by a bound.

Furthermore, an energy bound does not provide sufficient information about the *perceived appearance* of the result [19]. A certain mathematical error in certain places of a radiosity configuration doesn't necessarily mean that the human visual system will perceive this error as an artefact, a breach in the smoothness of an illuminated scene. On the other hand, although the mathematical error might be small in other places, the human eye will find that "something is wrong"; that artefacts exist which are not sufficiently described by energy transfer error estimation.

This brings in the second type of error, the *error concerning the visual quality.* What the user sees of any illuminated scene is the light distribution on the objects, the radiosity function on the receivers.

While the energy transfer bounds give the right radiosity *value* at the receiver, bounding the receiver function gives the right *shape* to the function across the extent of the receiver. The general value of the function is defined by the amount of energy exchanged between the objects, while the visual appearance of the light distribution across objects is defined by the shape of the radiosity function, and how well it has been adapted during the simulation.

3.5.4 Refinement tools

It becomes apparent that different applications, or different phases of a single application, probably require different refinement strategies.

The "refinement criterion" therefore has to be seen as a collection of tools based on low-level information about the radiosity transfer and the estimated appearance to the human visual system, rather than a single subdivision function returning "yes" or "no".

Possible low-level informations that have to be provided are:

- form factor estimate
- form factor extrema
- form factor gradient
- cluster bounds
- radiosity extrema
- radiosity residual
- visibility bounds
- error bounds
- perceptual control

- quantization technique

3.6 Dynamic Update

In their initial form, radiosity algorithms suffered from a serious drawback: when the geometry or the material properties of objects in the scene were changed, a new, time-consuming, radiosity solution needed to be recomputed.

In addition, a user will probably work in two steps: first, using fast, iterative solutions, and finally doing a converged solution. Solutions for dynamic radiosity update are required for the iterative phase, and might be extended for a converged solution.

In order to make radiosity usable in interactive, dynamic applications, appropriate data-structures and algorithms have to be designed to achieve efficient update of shadows [40, 15]. The memory requirements of such approaches are high, but can be improved upon [48, 16].

When the system deems it impossible to achieve interactive radiosity update rates using such techniques, approximate but more efficient realtime feedback methods can be employed [5, 37, 46].

These shadows will be potentially inaccurate in the first stage, but will then be replaced by progressively more accurate computations as the update process proceeds, until finally the correct radiosity solution is displayed.

The combination of different approaches and the modification of meshes at different levels in the hierarchy may result in flickering and other temporal aliasing artifacts, which have to be addressed.

3.7 ARCADE references

ARCADE website: `http://www-imagis.imag.fr/ARCADE/ARCADE.html`

ARCADE Partners:

- FRAUNHOFER-IGD: German applied research, leader in VR and interactive real-time applications
- iMAGIS/GRAVIR (France): French academic research (Universities UJF/INPG, CNRS, INRIA).
- LIGHTWORK DESIGN LTD: British SME specialising in computer graphics.

Nicolas Holzschuch

ISA research team, INRIA-Lorraine
615 rue du Jardin Botanique, BP 101
54602 Villers-ls-Nancy CEDEX, France
Nicolas.Holzschuch@inria.fr

4 Complex Scenes and Radiosity

4.1 Introduction

In this section, we show how to deal with complex scenes, and how hierarchical radiosity algorithms can be adapted in order to cope with them. By complex scene, we mean a scene with a large number of input primitives, say in the order of magnitude of a million input primitives.

Such scenes generally come from interesting fields, and produce beautiful pictures (see fig. 3). Unfortunately, they also are quite difficult to handle for radiosity algorithms.

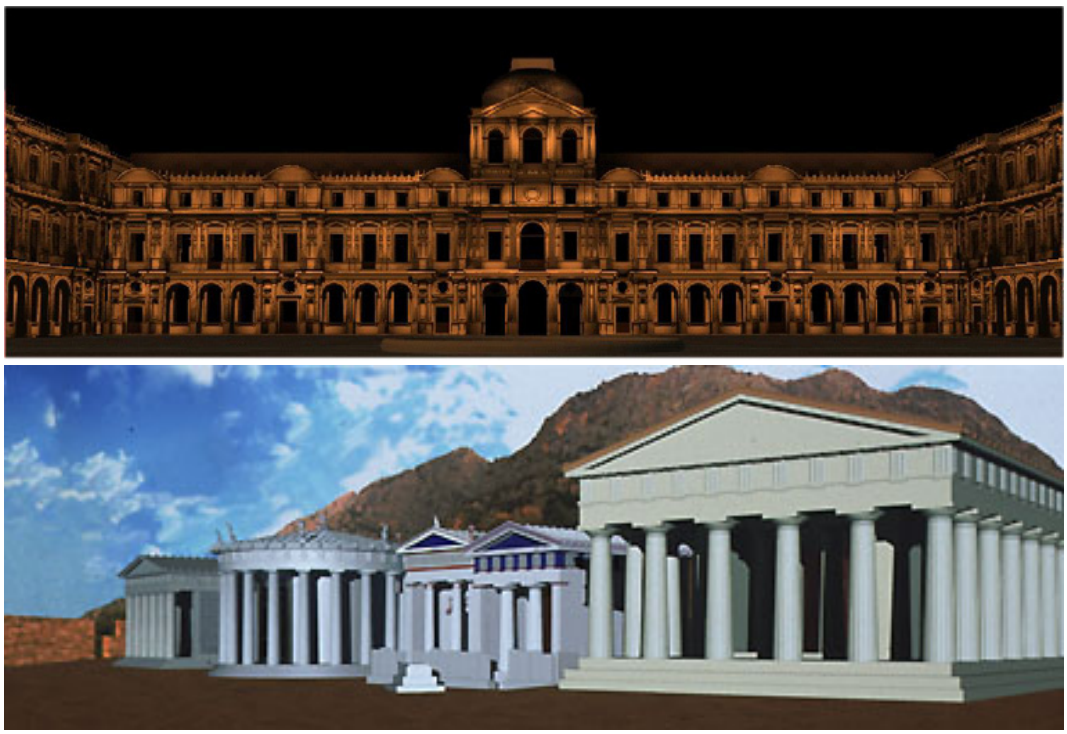

Figure 3: Complex scenes in illumination simulations

4.2 The Memory Problem

The first problem that occurs with complex scenes is avoiding to use all the memory available in the computer. As soon as the computer starts using virtual memory (by swapping), the computation speed

is drastically reduced. Unfortunately, a scene with a million input polygons tend to use a huge amount of memory. Anything that can be done to reduce the memory used by the program is therefore a good thing.

4.2.1 Why linear can still be quadratic

As we have seen in the second lecture, hierarchical radiosity has linear complexity with respect to the total number of patches generated, which makes it a good candidate for dealing with complex scenes. However, hierarchical radiosity still has quadratic complexity with respect to the total number of input primitives in the scene, since it starts with building interactions between the input primitives.[2] In a scene with a million input primitives, anything quadratic in memory is out of the question. Which is why we have to use some alternative techniques such as clustering (see second lecture) or getting rid of link storage.

4.3 Getting rid of link storage

Another method to reduce the memory requirement for the hierarchical radiosity algorithm is to eliminate link storage [48]: when we consider the interaction between two polygons, we build the links between patches as we use them. Once we are finished with this interaction, we delete all these links and move on to the next interaction.

4.3.1 Shooting or Gathering?

Getting rid of links storage obviously makes more sense if we are shooting than if we are gathering. In gathering, the links computed between two polygons are reused for each interaction, whereas in shooting, the links computed will only be reused when the shooting polygon receives enough further energy to be at the top of the list of shooting polygons. Furthermore, with shooting, the unshot energy distribution is likely to be different every time the polygon shoots, and hence will not require the same links. Hence shooting also makes more sense without link storage.

4.3.2 Choice of a wavelet basis

An important choice when designing a hierarchical radiosity algorithm is the choice of the wavelet basis to use (Haar, linear, quadratic wavelets). Previous experiments [52] have shown that linear and quadratic wavelets achieve better results than Haar wavelets, but with the downside of a (much) bigger memory cost. If we get rid of link storage, most of this memory cost disappears. As a consequence, linear and quadratic wavelets can become more interesting.

4.4 Reducing the complexity of the scene

Possibly one of the easiest ways to deal with a complex scene is to reduce its complexity. That is to say, reduce the total number of primitives. One way to reduce the total number of primitives is to work with more abstract primitives, such as cylinders and spheres, instead of just planar polygons: tesselating a cylinder into planar polygons implies replacing a single primitive by more than twenty.

[2] What is worse, each of these interactions does require a visibility test.

Similarly, most scenes coming from architecture models have complex planar polygons, that are usually converted to triangles, inducing a large number of triangles (see fig. 4). A method has been developed that allows hierarchical radiosity to work with the initial polygons, whatever their shapes [11].

Figure 4: Reducing the number of primitives in the scene

4.5 Parallelisation

If all the previous methods, combined together, aren't sufficient to do a radiosity computation on your computer, the best thing to do is probably to switch to a more powerful computer. The hierarchical radiosity method lends itself well to parallelisation, either on a cluster of workstations [17] or on a shared-memory system [6].

4.6 Visibility Methods

4.6.1 Importance of visibility methods

According to an old study [27], visibility queries take more than 70 % of total computation time in hierarchical radiosity computations. This proportion is still valid on recent, state-of-the-art, hierarchical radiosity softwares

Visibility queries are, therefore, the most important point in radiosity computations. Anything that can reduce the time spent doing visibility queries will reduce the total time for radiosity computations.

4.6.2 Clustering

One of the positive side effects of clustering is that it also extends to visibility. If a cluster is blocking visibility between two objects we can estimate the proportion of light blocked by this cluster, using heuristics based on surface repartition and surface orientation. We also estimate the lack of precision on this visibility estimate, and we can decide to refine the cluster if this imprecision appears too large.

4.6.3 Scene partitioning

Another method for reducing the costs of visibility computations is to use spatial partitioning based on the scene specificities [50]. If we are inside a building with many rooms and doors, then if a room is invisible, obviously all of its content is also invisible.

4.6.4 Hardware-based visibility methods

Finally, we can use the graphics hardware present in most graphics workstations (Z-buffer, mostly) to do some of the job in our visibility queries. By nature, graphics hardware gives results with only finite precision. However, we can use some heuristics [26] to overcome this limitation, thereby ensuring fast visibility queries.

4.7 Displaying the results

If the input scene had roughly a million input polygons, we can expect the final result to have tens of millions of patches. Such a large number of patches is difficult to display in a smooth manner on current graphics hardwares. To ease this problem, techniques have been developed to display results in a quick and efficient manner.

4.7.1 Culling and impostors

Culling methods are used to avoid displaying some parts of the scene, because they aren't visible, or not visible enough to have a significative impact on the visual result. They include the obvious (back-face, visibility pyramid) and more sophisticated methods, using hierarchical occlusion maps [54], portal detection and cell visibility [36].

Impostors are used to replace a subset of the scene with something simpler, but without noticeable difference. They include levels of details, replacing a planar polygon that has been refined into many small patches by a single polygon with a texture to represent the patches [32], replacing a non-planar set of objects by a combination of texture-map and depth map [14] and replacing what's visible through a portal by a warped texture [36]

4.7.2 Non-frame-based rendering

Finally, you can render complex scenes in an interactive manner if you don't render the entire scene for the entire picture, but update only a subset of the pixels for every frame: frameless rendering [33, 53] or RenderCache [51].

Per H. Christensen

c/o Square USA
55 Merchant Street, suite 3100, Honolulu, HI 96813, USA
per@squareusa.com

5 Hierarchical Techniques for Glossy Global Illumination

5.1 Abstract

This part of the course notes gives a summary of my Ph.D. research done at the University of Washington. Finite element methods used for simulation of diffuse global illumination (radiosity) can be extended to also handle directional (glossy) reflection. We focus on glossy reflections since diffuse reflection is a simple special case and specular reflections can be incorporated [43]. So if we can solve the glossy global illumination problem efficiently, the general global illumination problem can also be solved efficiently.

5.2 Overview

These notes first describe how hierarchical representations (wavelets and clustering) reduce the number of light interactions needed in a global illumination simulation. Then they describe how importance-driven refinement focuses the computations where they contribute most to the quality of the rendered image. Finally, these methods are put together in an algorithm for importance-driven hierarchical glossy global illumination, and a conclusion is given.

5.3 Hierarchical representation

Recall that in the finite element method, the solution is found by transporting light between basis functions. We want as few transports as possible to reduce computation time.

Previous work includes hierarchical radiosity [24] and wavelet radiosity [21]. We extended this to glossy global illumination.

5.3.1 Wavelet representation

Wavelets form a convenient hierarchical basis. They represent a function as coarse overall shape along with detail at finer and finer resolution. The simplest example is the Haar basis, but there are many other wavelet bases. More complicated wavelets have the advantage that there are fewer significant transports, but they also have the disadvantage that each transport becomes more expensive to compute.

Radiance is defined on the domain $S \times \Omega+$: all surface points in the scene and the hemisphere above each point. The surfaces are split into patches that each can be parameterized as a unit square. The hemisphere is transformed to the unit square by a gnomonic projection followed by a radial "stretch". With these two transformations, the domain of radiance is then the four-dimensional hypercube $[0, 1]^4$. Four-dimensional wavelets are formed in this domain as tensor products of univariate wavelets.

Radiance is transported between these basis functions. Initially, it is only tranported between the coarsest basis functions. Later, as the refinement progresses, radiance is also transported between detail basis functions.

5.3.2 Clustering

Many surface patches are required to represent realistic scenes. To speed up the computation, we want to reduce the complexity from $O(p^2)$ to $O(p)$ (where p is the number of surface patches).

We base this reduction on the observation that when an object is far away, the actual geometry is irrelevant — we are only interested in the light that comes from it. So we can group surface patches into clusters to simplify light transports.

Previous work includes the n-body problem (computing the gravitational pull between n stars in a galaxy) [2, 4, 22] and clustering for diffuse scenes [44].

Without clustering, we have to transport light between all pairs of patches, which gives $O(p^2)$ transports.

Our clustering method represents the light from a cluster as coming from just one point. Similarly, all light to a cluster is assumed to be in the direction of the cluster center.

For refinement of the solution, an upper bound on the transport error is given (how wrong the cluster approximation is for light transport). If the error is too high, light is transported between subclusters or patches instead.

5.4 Importance

We would like to know where the solution has to be computed accurately, and where a coarse solution is sufficient. Importance can give the answer to these questions. Using importance, we only need to compute the solution to high accuracy in a fraction of the scene.

Importance is transported like light, but emitted from the eye. In areas receiving much importance, the light contributes much to the image and it should therefore be computed to high accuracy.

Previous work includes importance in nuclear physics [30] and importance for diffuse global illumination [45].

The advantage of using importance in a diffuse scene is that the radiosity is refined if it is in an important part of the scene. But even better, radiance is refined only if it is in an important part of the scene *and in an important direction.*

Importance is also extended to clustering — both transport and refinement.

5.5 Algorithm

The algorithm for importance-driven hierarchical global illumination looks like this:

```
Compute coarse solution
repeat
        Refine based on solution
        Compute better solution
until sufficient accurate
Render solution
```

5.6 Rendering

A simple rendering method is to evaluate the solution at points corresponding to each pixel. Better images result from doing a so-called final gathering: light is transported to points corresponding to each pixel. This improves the visual quality (the meshing artifacts disappear), but also increases rendering time considerably.

5.7 Conclusion

In these course notes, we introduced the following hierarchical techniques for glossy global illumination:

- wavelets,
- clustering,
- importance

For more detailed information, see [8].

Marc Stamminger

Max-Planck-Institute for Computer Science
Im Stadtwald, 66123 Saarbrücken, Germany
stamminger@mpi-sb.mpg.de

6 Illumination Samples

In this section an algorithm will be presented with the goal of extending an HR implementation to also handle non-diffuse surfaces. The key to the algorithm is how the directional information of the incident light is maintained [47].

6.1 Incident Light Representation

In HR the incident light is computed in small portions. Since only diffuse surfaces are considered, these incident irradiance portions are immediately added to the irradiance field of the receiving object, so that the directional information is lost.

To maintain directional information, in the illumination samples approach each incident light portion is stored with the receiving object in a list of *illumination samples.* An illumination sample is a pair of an incident direction and an irradiance value. Each object has a list of illumination samples. Light propagation is computed as in HR, but incident light portions are not added together in a single irradiance value. Instead, for each transported light portion an illumination sample is created that describes this light portion. This sample is then added to the object's list of illumination samples. As direction of the sample the vector to the sender's center is used.

The idea is depicted in Figure 5. The scene shows for a point x the arriving links computed for a simple 2D box scene. The point receives light via three generations of patches (left three images). For each link, during propagation computation, an illumination sample is added to the receiver. These sample describe the illumination at x (right).

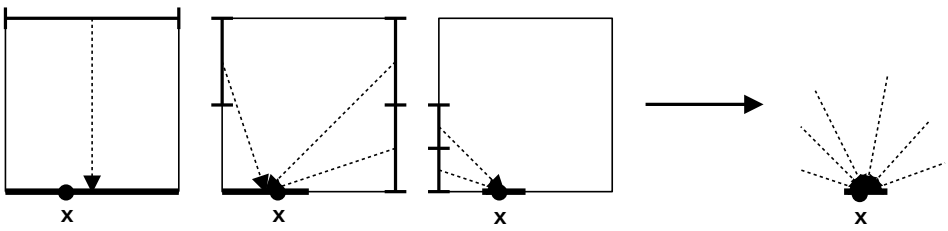

Figure 5: Links arriving at a some point x for a 2D box scene. For each link an illumination sample is added to the patches. These samples are then used as illumination description at the patch (right).

The scene hierarchy resulting from this example is shown in Figure 6. The left image shows the part of the scene containing point x. The illumination samples created at the nodes on the path to the patch containing x are inserted as dotted arrows. In the right image the samples have been pushed down this path, resulting in a description of the entire illumination at the leaf node.

This way, the directional information of the incident light is stored in an efficient manner. No information about the range of incidence is stored, only its main direction. This is an approximation, but

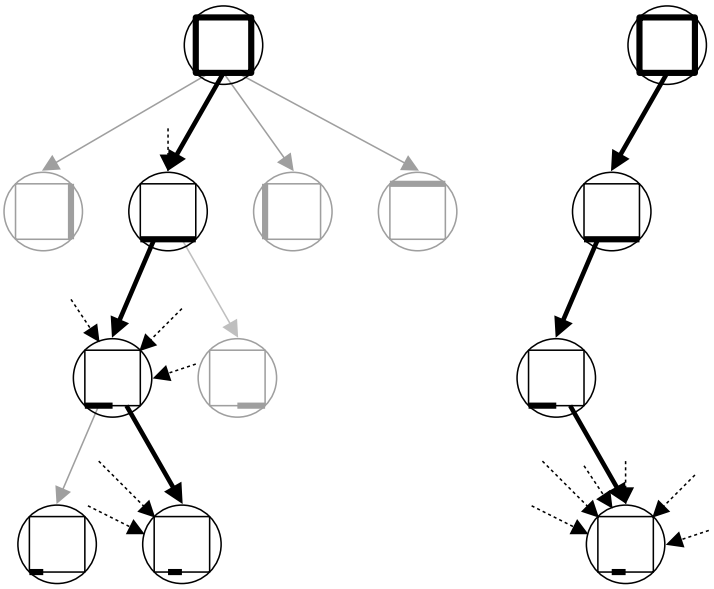

Figure 6: Illumination samples created for the situation in Figure 5 (left). After pushing the samples down the hierarchy, the entire illumination at the leaves is known (right).

due to the smoothing nature of most BRDFs this is deemed sufficient.

6.2 Exitant Light Representation

The exitant light of a non-diffuse surface is stored in a similar manner as in [7, 42]. The object is approximated by a point source with a two-dimensional directional exitant intensity distribution.

6.3 Reflection Computation

The illumination sample algorithm is very similar to the HR method. The main difference is that in the gather step the irradiance portions are not summed directly at the receiver, but added to the receiver's illumination sample list.

After this gather step, the illumination of the scene is known in the form of illumination samples, distributed over the entire scene hierarchy. Next this illumination has to be reflected. This is done as in HR in a push/pull step, performed in a traversal of the scene hierarchy. In the push phase the illumination gathered at inner nodes is pushed down to the leaves, so that every leaf knows about its complete illumination. In the illumination context this means that every inner node simply copies its illumination samples down to all the leaves beyond (cp. Figure 6).

Instead of copying all samples down the hierarchy, the push can be performed virtually during the traversal. In a list of current illumination samples all samples of the current branch are gathered. This can be done very efficiently, if the list of current samples is organized as "list of lists". If a node is traversed downwards, its samples are added to the current list, if the node is traversed upwards again, its samples at the end of the current list are removed again.

When a leaf is reached during traversal, all illumination samples contributing to its illumination are in

the current list. Next, this incident light has to be reflected to obtain the new exitant light of the leaf:

$$\begin{aligned} L(x,\omega) &= \int_\Omega f_r(\omega', x, \omega) L^{\text{in}}(x, \omega') n(x) \circ d\omega' \\ &= \int_\Omega f_r(\omega', x, \omega) \sum_{i=1}^{n} I_i \delta(\omega' - \omega_i) \circ d\omega' \\ &= \sum_{i=1}^{n} f_r(\omega_i, x, \omega) I_i n(x) \circ \omega_i \end{aligned} \tag{6}$$

Each incident light impulse (I_i, ω_i) results in a BRDF response which is the BRDF at x with fix incident direction ω_i times the irradiance I_i. The sum of all these responses is the total reflection. So the reflection in some direction ω can be computed exactly by n BRDF evaluations. Using the `DirDistr` library, a representation of the reflection is then computed.

If we have the exitant intensity distributions at the leaves, we still have to perform a pull step to also update the exitant light for the inner nodes. This is done by averaging the exitant light distributions of the children bottom up.

7 Three Point Clustering

As described previously, a finite element radiance algorithm needs to store the directional distribution of the simulated light. Depending on the formulation of the rendering equation—directional or 3-point—different function bases have been used for this purpose. In this section an approach will be shown that is based on the three point formulation of the rendering equation [49].

The radiance distribution is computed within a line space hierarchy and stored together with the links. This representation has several advantages. The hierarchy fits very well with hierarchical radiosity, it is very efficient, because links and radiance can be stored within one hierarchy, and the one representation can be used as incident as well as as exitant representation.

7.1 The Line Space Hierarchy

The final function we are looking for describes the radiance between two mutually visible front-facing surface points, which we will call *lines* in the following. Since radiance does not change along a ray, it is constant along such a line. All these lines span a space, which is in another context referred to as the *line space* [15]:

$$\mathcal{L} := \{(x,y) : x, y \in S, V(x,y) = \text{true} \wedge (y-x) \circ n(x) > 0 \wedge (x-y) \circ n(y) > 0\} \tag{7}$$

The radiance function in the scene is defined on this line space: $L : \mathcal{L} \to \mathcal{C}$, where $\mathcal{C}$ is the chosen color space, e.g. RGB. Note that the lines in this space are oriented, i.e. (x,y) and (y,x) are different lines.

The goal of a hierarchical algorithm is the computation of an approximation to the radiance function L. In three point clustering this is achieved by building a hierarchy on $\mathcal{L}$ directly and computing the approximation within this hierarchy. It is interesting to notice that during a standard hierarchical clustering computation such a hierarchy of the line space is implicitly used: if the transport between a sending and a receiving object is refined, one or both objects are subdivided implying a partition of the line space between sender and receiver.

This is depicted in Figure 7, starting with a pair of interacting patches P and Q. The node $(P \rightarrow Q)$ describes all unoccluded lines from P to Q. Subdividing the sender P into two children results in two new nodes $(P_0 \rightarrow Q)$ and $(P_1 \rightarrow Q)$, which partition the line space $(P \rightarrow Q)$ into two children. In the next step, each of these children $(P_i \rightarrow Q)$ is subdivided again into children $(P_i \rightarrow Q_j)$ and so forth.

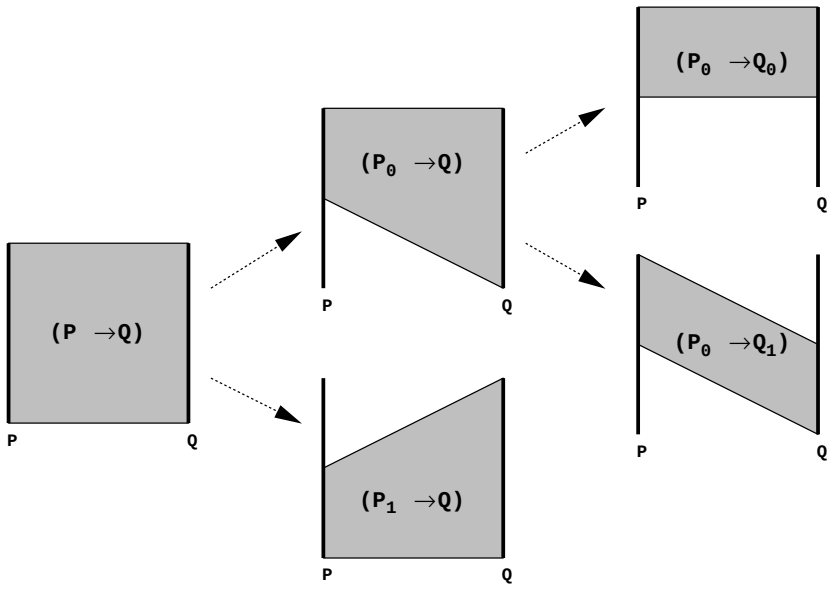

Figure 7: Building a hierarchy on the line space from an object P to an object Q. Starting with all lines from P to Q $(P \rightarrow Q)$ first the sender P is subdivided, resulting in new nodes (P_0, Q) and (P_1, Q). These nodes in turn can be subdivided by splitting the receiver Q and so forth.

So the scene hierarchy can also be used to construct a hierarchy on the line space. Each node of the line space hierarchy is a pair of nodes of the scene hierarchy, but not every pair of scene nodes is also a line space node.

7.2 Radiance Representation

The problem of a radiance computation in comparison to a radiosity computation is that some way is needed to store the directional light information of incident and exitant light. In three point clustering this is achieved by storing the radiance within a line space hierarchy. For efficiency reasons, the link data—form factors and the visibility information—is also stored within the same hierarchy. From another point of view this means that the radiance is stored together with the links. In the following we will refer to this whole structure of link and radiance data as the *link hierarchy*.

As radiance does not change along an unoccluded ray, this representation can either be interpreted as exitant light from the sender or as incident light at the receiver. As a result we do not need different representations for both and transportation of light becomes trivial.

7.3 Computation of Light Propagation

This representation fits very well with the HR method. The light propagation computation is performed by traversing the link hierarchy. If the oracle decides to transport light via some link, this light is not added to the receiver directly as in HR, because the directional information would be lost then. Instead, the light is stored together with the link, so later the light's origin is still known.

7.4 Reflection Computation

If the light incident at some scene node is needed for reflection computations, it can be obtained easily from the link hierarchy. Figure 8 shows the links established in a simple flatland box scene, as already used in the previous section. In order to get the illumination at patch P, a traversal of the link hierarchy is performed, thereby only traversing nodes, that have P as receiver. All these nodes form a subtree within the link hierarchy. All leaves of this subtree describe the complete illumination of P. If the link hierarchy is adaptive in the sense that all leaves carry about the same energy, the illumination at P obtained this way is also adaptive. So from dark directional regions around P only a few links will arrive, whereas from bright regions many links and thus a detailled directional information will be provided.

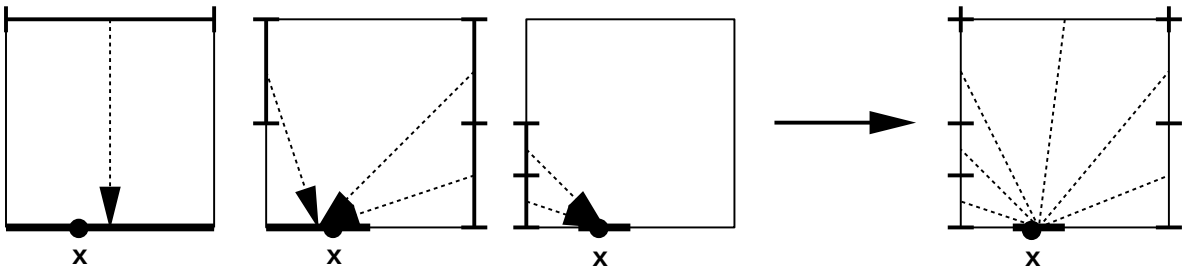

Figure 8: Links established in a simple box scene towards a point x on three different levels (left to center right). The illumination at x can be obtained by putting them all together (right)

Knowing the incident light field for a particular patch, the glossy reflection can be computed. In three point clustering, this means to set the radiance values in the line space hierarchy, that own the current patch as sender, to a new iteration value. By this, the incident light at the corresponding receivers is immediately changed, resulting in a Gauss-Seidel-like iteration scheme.

References

[1] Anthony A. Apodaca. Photosurrealism. In *Proc. 9th Eurographics Workshop on Rendering*, 1998.

[2] Andrew W. Appel. An efficient program for many-body simulation. *SIAM Journal on Scientific and Statistical Computing*, 1(6):85–103, January 1985.

[3] Ian Ashdown. Photometry and radiometry – a tour guide for computer graphics enthusiasts. Technical report, Ledalite Architectural Products, Inc., 1996.

[4] Josh Barnes and Piet Hut. A hierarchical $o(n \log n)$ force-calculation algorithm. *Nature*, 4(324):446–449, December 1986.

[5] J. F. Blinn. Me and my (fake) shadow. *IEEE Computer Graphics and Applications*, 9(1):82–86, 1988.

[6] Xavier Cavin, Laurent Alonso, and Jean-Claude Paul. Parallel wavelet radiosity. In *Proceedings of the Second Eurographics Workshop on Parallel Graphics and Visualisation*, pages 61–75, Rennes, France, September 1998. Eurographics.

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

[8] Per Henrik Christensen. *Hierarchical Techniques for Glossy Global Illumination.* PhD thesis, University of Washington, 1995.

[9] Michael Cohen, Shenchang E. Chen, John R. Wallace, and Donald P. Greenberg. A progressive refinement approach to fast radiosity image generation. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4):75–84, August 1988.

[10] Michael F. Cohen and John R Wallace. *Radiosity and Realistic Image Synthesis.* Academic Press, 1993.

[11] Franois Cuny, Laurent Alonso, Christophe Winkler, and Nicolas Holzschuch. Radiosit base d'ondelettes sur des mailles quelconques. *Revue Internationale de CFAO et d'Informatique Graphique*, 1999.

[12] Katja Daubert, Hartmut Schirmacher, Francois Sillion, and George Drettakis. Hierarchical lighting simulation for outdoor scenes. In Julie Dorsey and Phillip Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 229–238, New York, NY, 1997. Springer Wien. ISBN 3-211-83001-4.

[13] Commission Internationale de L'Eclairage. Spatial distribution year = 1994 of daylight - luminance distributions of various reference skies. Technical Report CIE 110-1994, ISBN 3 900 734 52 6.

[14] Paul E. Debevec. *Modeling and Rendering Architecture from Photographs.* PhD thesis, University of California at Berkeley, Computer Science Division, Berkeley CA, 1996.

[15] George Drettakis and François Sillion. Interactive update of global illumination using A line-space hierarchy. In *SIGGRAPH 97 Conference Proceedings*, pages 57–64. ACM SIGGRAPH, 1997.

[16] A. Pomi F. Schoeffel. Reducing memory requirements for interactive radiosity using movement prediction. In *Proc. Eurographics Workshop on Rendering '99*, pages 233–242, 1999.

[17] Thomas A. Funkhouser. Coarse-Grained Parallelism for Hierarchical Radiosity Using Group Iterative Methods. In *Computer Graphics Proceedings, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings)*, pages 343–352, 1996.

[18] S. Gibson and R. J. Hubbold. Efficient hierarchical refinement and clustering for radiosity in complex environements. *Computer Graphics Forum*, 15(5):297–310, dec 1996.

[19] Simon Gibson and R. J. Hubbold. Perceptually driven radiosity. *Computer Graphics Forum*, 16(2):119–128, June 1997.

[20] C. M. Goral, K. E. Torrance, and D. P. Greenberg. Modeling the interaction of light between diffuse surfaces. *Computer Graphics (SIGGRAPH '84 Proceedings)*, 18(3):212–222, July 1984.

[21] Steven J. Gortler, Peter Schröder, Michael Cohen, and Pat M. Hanrahan. Wavelet radiosity. *Computer Graphics (SIGGRAPH '93 Proceedings)*, 27:221–230, August 1993.

[22] Leslie F. Greengard. *The Rapid Evaluation of Potential Fields in Particle Systems.* PhD thesis, Yale University, 1987.

[23] Pat Hanrahan. *Radiosity and Realistic Image Synthesis*, chapter Rendering Concepts. Academic Press, 1993.

[24] Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):197–206, 1991.

[25] J.-M. Hasenfratz, C. Damez, F. Sillion, and G. Drettakis. A practical analysis of clustering strategies for hierarchical radiosity. In *Proc. EUROGRAPHICS '99*, 1999. to appear.

[26] Nicolas Holzschuch and Laurent Alonso. Using graphics hardware to speed-up your visibility queries. Submitted to the *Journal of Graphics Tools*, 1999. available from http://www.loria.fr/publications/1999/99-R-030/99-R-030.ps.

[27] Nicolas Holzschuch, Francois Sillion, and George Drettakis. An Efficient Progressive Refinement Strategy for Hierarchical Radiosity. In *Fifth Eurographics Workshop on Rendering*, pages 343–357, Darmstadt, Germany, June 1994.

[28] James T. Kajiya. The rendering equation. *Computer Graphics (SIGGRAPH '86 Proceedings)*, 20(4):143–150, August 1986.

[29] Wolfram Kresse. Effizientes und anwendbares hierarchisches radiosity unter besonderer bercksichtigung der visibilittsbetrachtung. Master's thesis, Darmstadt University of Technology, October 1997.

[30] Jeffery Lewins. Importance, the adjoint function: The physical basis of variational and perturbation theory in transport and diffusion problems, 1965.

[31] Dani Lischinski, Brian Smits, and Donald P. Greenberg. Bounds and Error Estimates for Radiosity. In *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings)*, pages 67–74, 1994.

[32] Karol Myszkowski and Tosiyasu L. Kunii. Texture Mapping as an Alternative for Meshing During Walkthrough Animation. In *Fifth Eurographics Workshop on Rendering*, pages 375–388, Darmstadt, Germany, June 1994.

[33] Steven Parker, William Martin, Peter-Pike Sloan, Peter Shirley, Brian Smits, and Chuck Hansen. Interactive ray tracing. In *Interactive 3D*, 1999.

[34] Sumanta N. Pattanaik and Kadi Bouatouch. Linear Radiosity with Estimation Error. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 170–185, New York, NY, 1995. Springer-Verlag.

[35] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flennery. *Numerical Recipes in C*. Cambridge University Press, 1988.

[36] Matthew M. Rafferty, Daniel G. Aliaga, Voicu Popescu, and Anselmo A. Lastr. Images for accelerating architectural walkthroughs. *IEEE Computer Graphics and Applications*, 18(6):38–45, November/December 1998.

[37] Frank Schoeffel and Michael Meixner. Realtime shadow feedback for interactive radiosity scenes using shaded multipoints. In *Proceedings of the ACM Sympoaium on Virtual Reality Software and Technology (VRST '98)*, November 1998.

[38] Peter Schroder and Pat Hanrahan. On the Form Factor Between Two Polygons. In *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pages 163–164, 1993.

[39] Adrian Seccia. A perceptual refinement oracle for hierarchical radiosity. Master's thesis, University of Cape Town, April 1998.

[40] E. S. Shaw. Hierarchical Radiosity for Dynamic Environments. M.Sc. thesis, Ithaca, NY, August 1994.

[41] François Sillion. Clustering and volume scattering for hierarchical radiosity calculations. In *Photorealistic Rendering Technique (Proceedings Fifth Eurographics Workshop on Rendering)*, pages 105–120, Darmstadt, June 1994. Springer.

[42] François Sillion, George Drettakis, and Cyril Soler. A clustering algorithm for radiance calculation in general environments. In *Rendering Techniques '95 (Proceedings of Sixth Eurographics Workshop on Rendering)*, pages 196–205. Springer, August 1995.

[43] Francois X. Sillion, James R. Arvo, Stephen H. Westin, and Donald P. Greenberg. A global illumination solution for general reflectance distributions. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 187–196, July 1991.

[44] Brian Smits, James Arvo, and Donald Greenberg. A clustering algorithm for radiosity in complex environments. *Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 435–442, July 1994.

[45] Brian Smits, James Arvo, and David Salesin. An importance driven radiosity algorithm. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):273–282, July 1992.

[46] Cyril Soler and Francois X. Sillion. Fast calculation of soft shadow textures using convolution. In *Computer Graphics (ACM SIGGRAPH '98 Proceedings)*, pages 321–332, 1998.

[47] Marc Stamminger, Annette Scheel, Xavier Granier, Frederic Perez-Cazorla, George Drettakis, and François Sillion. Efficient glossy illumination with interactive viewing. In *Graphics Interface '99*, pages 50–57, 1999.

[48] Marc Stamminger, Hartmut Schirmacher, Philipp Slusallek, and Hans-Peter Seidel. Getting rid of links in hierarchical radiosity. *Computer Graphics Forum (EUROGRAPHICS '98 Proceedings)*, 17(3), September 1998.

[49] Marc Stamminger, Philipp Slusallek, and Hans-Peter Seidel. Three point clustering for radiance computations. In *Rendering Techniques (Proc. Eurographics Workshop on Rendering '98)*, 1998.

[50] Seth Teller and Pat Hanrahan. Global Visibility Algorithms for Illumination Computations. In *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pages 239–246, 1993.

[51] Bruce Walter, George Drettakis, and Steven Parker. Interactive rendering using the render cache. In *10th Eurographics Workshop on Rendering*, June 1999.

[52] Andrew Willmott and Paul Heckbert. An empirical comparison of progressive and wavelet radiosity. In Julie Dorsey and Phillip Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 175–186, New York, NY, 1997. Springer Wien. ISBN 3-211-83001-4.

[53] Ellen Scher Zagier. Defining and refining frameless rendering. Technical Report TR97-008, Department of Computer Science, University of North Carolina, July 1997.

[54] Hansong Zhang, Dinesh Manocha, Thomas Hudson, and Kenneth E. Hoff III. Visibility culling using hierarchical occlusion maps. In *Computer Graphics (ACM SIGGRAPH '97 Proceedings)*, volume 31, pages 49–56, August 1997.