Implementation and Analysis of an Image-Based Global Illumination Framework for Animated Environments

Jeffry Nimeroff, *Member, IEEE Computer Society*, Julie Dorsey, and Holly Rushmeier

Abstract—We describe a new framework for efficiently computing and storing global illumination effects for complex, animated environments. The new framework allows the rapid generation of sequences representing any arbitrary path in a "view space" within an environment in which both the viewer and objects move. The global illumination is stored as time sequences of range-images at base locations that span the view space. We present algorithms for determining locations for these base images, and the time steps required to adequately capture the effects of object motion. We also present algorithms for computing the global illumination in the base images that exploit spatial and temporal coherence by considering direct and indirect illumination separately. We discuss an initial implementation using the new framework. Results and analysis of our implementation demonstrate the effectiveness of the individual phases of the approach; we conclude with an application of the complete framework to a complex environment that includes object motion.

Index Terms—Animation, global illumination, image-based rendering, radiosity, ray tracing, walk-throughs.

1 Introduction

The ultimate goal of global illumination algorithms for computer image generation is to allow users to interact with accurately rendered, animated, geometrically complex environments. While many useful methods have been proposed for computing global illumination, the generation of physically accurate images of animated, complex scenes still requires an inordinate number of CPU hours on state of the art computer hardware. Since accurate, detailed images must be precomputed, very little user interaction with complex scenes is allowed.

In this paper, we present a range-image based approach to computing and storing the results of global illumination for an animated, complex environment. Range-image based systems have been used previously in flight simulators [4] and in computer graphics [9]. Range-images store the distance to the visible object for each pixel, as well as the radiance. While previous research has demonstrated the potential of range-image systems to allow a user to tour a complex scene at interactive rates, the problem of efficiently rendering animated, globally illuminated environments within the context of such a system has not been considered. In this paper, we present a new framework that addresses this problem.

The contributions of this paper are several. We build upon previous work by considering how animated environments (i.e., environments in which objects as well as the user can move) can be represented as time sequences of range-images. We explore how to select a set of base views for the range-images as well as the time steps required to capture the effects of object motion. Further, we consider how global illumination can be efficiently computed to generate each of the range-images. Previous global illumination methods have successfully exploited spatial coherence by separating the calculation of direct and indirect illumination [8], [32]. We build on this idea and exploit temporal coherence as well by separating the calculation of temporal variations in direct and indirect illumination. These innovations form the basis of a new framework that allows the rapid generation of views along arbitrary paths within a "view space," which is a subspace of the full environment. We present results and an analysis from a preliminary implementation that demonstrate the potential of the framework to expand the level of interaction possible with complex, accurately rendered, animated environments.

2 Background

The approach we present builds on work in two areas—the traversal of complex, realistically shaded synthetic environments, and the calculation of global illumination in animated environments.

2.1 Traversal of Complex, Realistic Environments

The ultimate visualization system for interacting with synthetic environments would render perfectly accurate images in real time, with no restrictions on user movement or the movement of objects in the environment. A number

- *J. Nimeroff is with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104. E-mail: jnimerof@graphics.cis.upenn.edu.*
- *J. Dorsey is with the Department of Architecture and Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. E-mail: dorsey@graphics.lcs.mit.edu.*
- *H. Rushmeier is with IBM T.J. Watson Research Center, Hawthorne, NY 10532. E-mail: holly@watson.ibm.com.*

For information on obtaining reprints of this article, please send e-mail to: transvcg@computer.org, and reference IEEECS Log Number V96028.

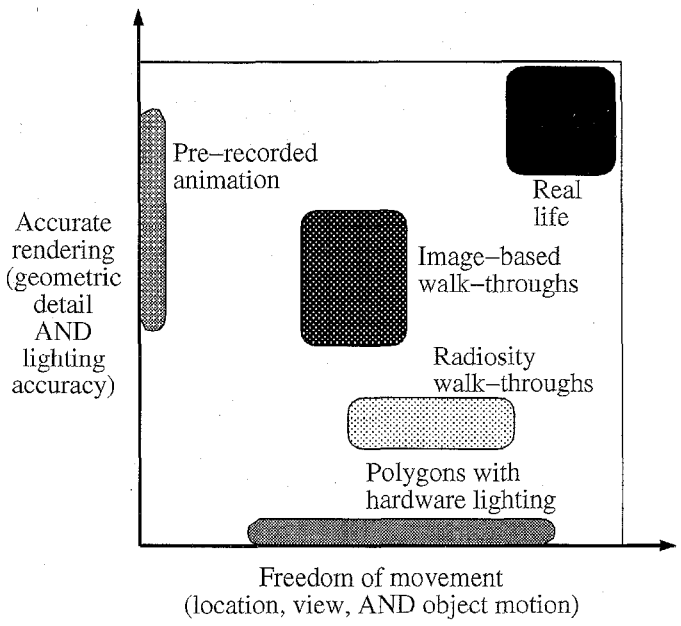


Fig. 1. Systems for interacting with complex, animated environments can be classified according to the freedom of movement they allow, and the accuracy with which the environment is rendered.

of different approaches have been developed for attempting to build such a system. Fig. 1 shows a general classification of methods for real time interaction with synthetic environments.

The space of visualization systems could be modeled as a higher dimensional space. However, for simplicity we show two basic dimensions—freedom of movement and rendering accuracy. By freedom of movement, we mean freedom of the viewer to move through the scene, and for objects to move, either on their own or as moved by the user. By rendering accuracy, we mean both the accuracy of the geometric detail in which objects are represented and the accuracy of the quantity of visible light computed for each object. While the axes for both movement and accuracy can each be extended indefinitely, even in the "real life," which we wish to simulate, there are limits. Depending on the application, our freedom of movement is limited by factors such as gravity and obstructions. Our perception of the geometry and visible light leaving any object is also limited by the spatial frequencies we can resolve and the dynamic ranges our eyes can accommodate.

Fig. 1 diagrams four basic approaches to traversing complex, realistic environments—polygons with hardware lighting, radiosity methods, prerecorded animations, and image based walk-throughs. There are many variations of each approach, so each is shown covering a range of freedom of movement and accuracy.

2.1.1 Polygons with Hardware Lighting

One approach is to use hardware lighting effects to render sequences of images with heuristic, *local* illumination models. In this scheme, the illumination of a surface depends only on its own characteristics and that of the parallel or nonphysical point (i.e., no $1/r^2$ drop off) light sources. While hundreds of thousands, or millions of polygons can be rendered per second, for complex scenes this means that individual objects must have simplified geometries to achieve real time speeds. Hardware rendering effects can be very useful for giving a sense of traversing a space for some applications. However, they are far from realistic because of the nonphysical lighting models used and the limitations on numbers of polygons that can be used to model the environment.

2.1.2 Radiosity

In contrast, radiosity techniques explicitly model the physical interreflection of light in a scene to compute the radiance L, energy per unit time, projected area and solid angle, leaving each object [14], [27]. This representation aids in the spatial perception of a scene. A radiosity solution is a set of radiance values at locations distributed over surfaces in an environment. The results of such a solution are view-independent. Given a solution, walk-throughs can be performed by converting the radiances to *RGB* values, which can be used in place of hardware lighting. Thus, radiosity approaches represent an improvement in both the ability to interact with a scene as well as in the accuracy of the illumination effects.

The primary limitation of the radiosity method as originally introduced, however, was that it was restricted to ideal diffuse reflectors—that is to surfaces for which L is independent of the direction of view (θ, ϕ) from the surface normal. Since the human visual system is very good at detecting and interpreting highlights that result from the directional variation of L, this representation is restrictive. Extensions of the radiosity method have been developed to account for the full range of bidirectional reflectance distribution functions (BRDFs) that occur in real life. In a method developed by Sillion et al. [26], a directional radiance function $L(\theta, \phi)$ is computed for sample points (x, y, z) on surfaces in the environment, rather than simply a radiance value L. Such a method, however, substantially increases the precomputation time, storage requirements, and time to traverse the complex scene.

2.1.3 Prerecorded Animations

To date, the most realistic animations are created by using algorithms that are capable of taking diffuse interreflection and nondiffuse reflection into account [31]. These algorithms are often termed photorealistic. While these algorithms are capable of depicting very accurate illumination effects, it is at the cost of interactivity. Creating an animated sequence with these algorithms is very time consuming when compared to the algorithms discussed above. Further, animated image sequences may only be generated if the viewer paths and object motions are specified a priori. Once a sequence is computed, the user is restricted to viewing a fixed set of frames as they were computed. A small amount of freedom of movement can be allowed by recording a network or tree of paths for the viewer to tour.

2.1.4 Range-Image Interpolation

Range-image interpolation has been employed in flight simulators [4], and has been applied to more general graphics applications by Chen and Williams [9], [7]. In this approach, the three-dimensional scene is replaced by a set of images for which the view point, and the radiances and ranges (i.e., the distance to nearest visible object) for each

pixel are stored. As a user traverses the environment appropriate views are synthesized by morphing the base range-images. Chen and Williams focus on how to perform this morphing, and suggest how the images can be obtained—e.g., examples cited include physical image capture and using images from a radiosity solution. Any global illumination method could be used to generate the base images.

A major advantage of the range-image approach is that storing and traversing the scene are only weakly dependent on object space complexity. A scene with hundreds of thousands of polygons can be represented by range-images which, after data compression, are at most a couple of orders of magnitude larger than range-images representing a few simple cubes. In general, range-image interpolation sacrifices some of the freedom of movement possible in a radiosity walk-through for increased accuracy in geometric detail and lighting accuracy. Note, however, that unlike prerecorded animations, the user can move freely in a subspace of the environment rather than moving only along predefined paths. The number of base images required to represent an environment for a given subspace of possible views is an open research problem.

2.2 Global Illumination of Animated Environments

We wish to tour complex environments in which objects move. Systems that produce animations with simple local shading can easily exploit image coherence. If local shading only is considered, new frames can be produced rapidly by only changing the portions of the image in which there are moving objects. For images generated using complete global illumination solutions, the problem is more difficult. When one object moves, it changes the illumination of all of the other objects in the scene to some extent.

There are essentially three possible ways to compute global illumination: object space, image space, and hybrid object/image space methods. In general, relatively little work has been done to develop efficient methods for animated environments.

2.2.1 Object Space

The radiosity method, described in the previous subsection, is an object space method. As shown in Fig. 2a, radiance distributions $L(x, y, z, \theta, \phi)$ are computed for objects without reference to the images in which the object will appear. The object space approach has the advantage that no interreflection calculation has to be performed as images are finally computed. It has the disadvantage that many radiances have to be computed that never appear in any images. The exception to this is the importance driven radiosity algorithm proposed by Smits et al. [28].

Another major drawback of the original radiosity method was that although it allowed walk-throughs of static environments, a costly new radiosity solution was required for each frame if any object moved. Considerable effort has been put in to developing radiosity methods that exploit temporal coherence in lighting. Baum et al. [3] developed a method for identifying geometric form factors that would not need to be recomputed for a full matrix radiosity solution as an object moved through a prescribed path. Chen [6], George et al. [13], Müller and Schöffel [19],

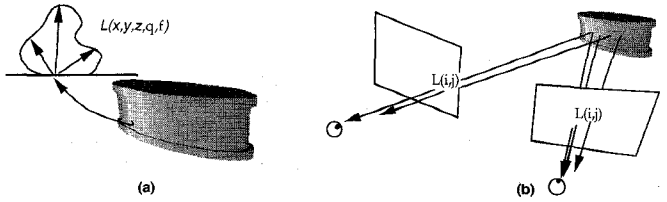


Fig. 2. In object space methods (a), radiance L is computed for each point and in each direction for each object. In image space and hybrid methods (b), radiance L is computed only for the objects which are visible in the image, and only for the points and directions which are visible.

Forsyth et al. [11], and Shaw [25] have developed progressive refinement radiosity solutions that are updated incrementally for temporal changes in object locations. Each method essentially starts with the solution for the previous time step, "undoes" the effect of the object that has moved, and then computes the effect of the object in its new position. Since most objects have a limited spatial effect, such incremental approaches converge quickly. Generally, the shadows are recomputed, and interreflected light is only propagated to a small extent before it falls below the acceptable threshold of "unshot radiosity." These methods effectively exploit the temporal coherence in the radiosity solution.

In all of the temporal radiosity methods, full illumination, rather than just direct or indirect, is computed. None of the methods have examined whether the temporal sampling rate for new radiosity solutions can be different from the frame generation rate.

2.2.2 Image Space

Monte Carlo path tracing (MCPT) [17] is an image space method. In MCPT, stochastic methods are used to compute the radiance $L(i, j)$ which will be seen through a pixel at location (i, j) on the screen, as shown in Fig. 2b. MCPT has the advantage that if an object in the environment doesn't appear in a particular image, its radiance does not have to be computed. This is an advantage for environments that have many more objects than the image representing it has pixels. MCPT has the disadvantage that it does not exploit spatial coherence—each pixel is computed independently. In general, this failure to exploit spatial coherence has kept MCPT from becoming a widely used technique. No work has been done to accelerate MCPT for animated sequences.

2.2.3 Hybrid Object/Image Space

Hybrid methods for global illumination combine the advantages of object and image approaches. Detailed radiance is only computed for objects which appear in the image. Spatial coherence is exploited by calculating multiple reflections in object space. Examples of hybrid methods are the *Radiance* system and the multipass progressive refinement methods [8], [23], [31].

Specifically, in hybrid methods, visibility and direct illumination calculations, for which the level of detail is limited by the pixel resolution, are computed in image space. Indirect illumination is computed in object space. In general, illumination is a continuous field, with sharp discontinuities occurring only when point light sources are instantaneously obscured [2]. Since indirect illumination is the

result of multiple reflections, giving rise to many extended "secondary" light sources, indirect illumination is generally a smoother function than direct illumination. As a result, indirect illumination can be sampled relatively sparsely in space, and intermediate values found by interpolation. In the *Radiance* system, indirect illumination is saved as a set of "cached" values in object space. In the multipass progressive refinement method, indirect illumination is found by a radiosity solution for a crude discretization of the environment.

To the authors' knowledge, no work has been published on efficiently updating hybrid object/image space global illumination solutions for moving objects.

3 A New Framework

In this section, we describe a new framework for computing global illumination for a system for traversing complex animated scenes. This framework is based on the techniques outlined in the previous section that are most able to achieve real-life freedom of movement and rendering accuracy. The new framework

1) is a range-image based system, and
2) exploits coherence by computing direct and indirect illumination separately in both space and in time.

3.1 An Image-Based System

As mentioned in Section 2, range-image based systems have the advantage of very efficiently representing geometrically complex environments. This advantage over polygon based systems is compounded when we wish to represent a photometrically accurate version of the environment. This advantage can be seen by referring again to Fig. 2.

As shown in Fig. 2a, for a photometrically accurate radiosity solution, we must compute a representation of the directional radiance distribution $L(x, y, z, \theta, \phi)$ for all points on every object. As shown in Fig. 2b, in a range-image system, we need only to compute and store the radiance for the small number of pixels in which an object is visible.

The advantage of the image-based approach over radiosity for photometrically accurate scenes extends even further when allowable error in the solution is considered. For any global illumination solution, the computation time can be drastically reduced by allowing some known error level in the results, rather than attempting to compute results to machine precision [16]. *But what is an allowable error level?* The allowable error depends on viewer perception, not on the characteristics of the object. In a radiosity solution, a high degree of accuracy is required, because the view of the object is unknown. The "worst case" must be assumed. Perceptual error metrics are inherently image based, since the accuracy required for a particular radiance depends on the radiance distribution in the visual field [5], [24]. In an image-based system, much better estimates can be made of allowable error in the radiance solution, and the solution can be computed much more efficiently.

In our new framework then, the global illumination solution will be computed in the form of a set of base range-images, which will be interpolated to produce a frame at each time step. A user will then be able to move freely within the space spanned by the base range-images.

3.2 Exploiting Temporal Coherence in Global Illumination

As discussed earlier, several researchers have proposed radiosity solutions that efficiently exploit the temporal coherence in global illumination variations as objects move. In these approaches, the location and radiance distribution is stored for each object. As the observer moves through the scene, objects are projected into the view with the appropriate radiance. The advantage of this approach is that the global illumination for each time step has been incrementally computed by exploiting object space coherence. That is, a completely new global illumination solution is not needed for every frame. The disadvantages are the precomputation time and storage space required as the number of objects becomes very large.

In a temporally varying range-image based system (Fig. 3), we move through time by interpolating between images in a time series for each base view. In this case, we are producing frames that represent a full global illumination solution—taking into account both diffuse and nondiffuse illumination. Radiances do not need to be stored for every object for every time.

3.2.1 Direct Illumination and Visibility

Relatively standard techniques for ray tracing animations

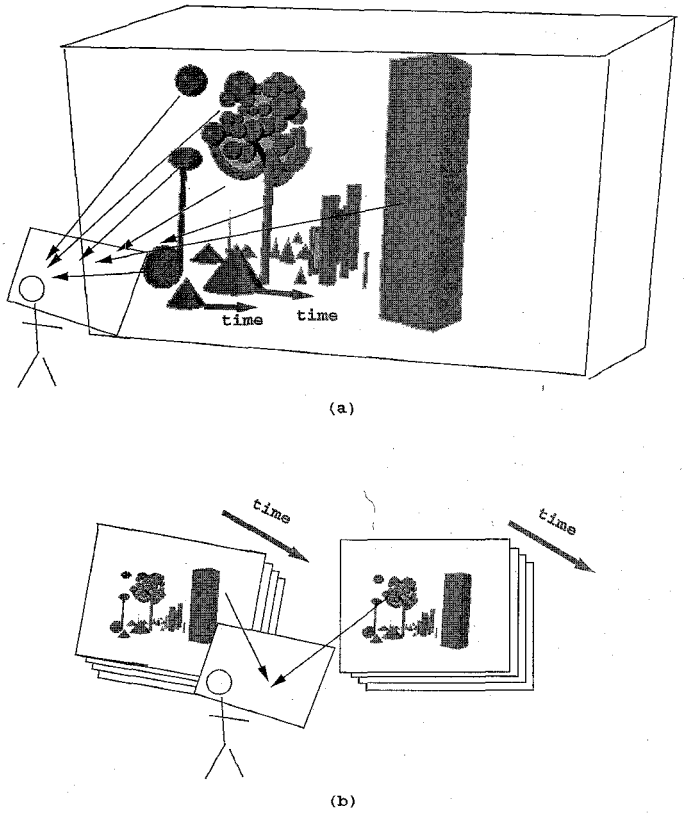


Fig. 3. The effects of object motion are stored differently in object space and image space systems. In an object space approach (a), motion is displayed by reprojecting objects in their new positions as time increases and either recomputing or looking up the radiances for those objects at that time. In an image space approach (b), motion is displayed by interpolating the radiances in a time series of base range-images, and morphing the range-images to produce the appropriate view.

can be used to exploit the coherence of direct visibility and shadowing in the base range-images. Rather than computing images for every 1/30 sec, the time steps for these base images can be determined by detecting the amount of geometry or shadowing change over longer lengths of time.

Even with this reduced number of base-time images, how can we avoid a pixel by pixel recalculation of the global illumination for each time step? As in the temporal radiosity methods, we seek to exploit the temporal coherence in the full global solution to reduce the calculations.

3.2.2 Indirect Illumination

As noted in Section 2, hybrid approaches exploit spatial coherence by computing indirect illumination effects using relatively sparse spacing in object space. We can use the same sparse sampling of object space to exploit temporal coherence.

Fig. 4. illustrates the relationship between sampling indirect illumination in time and in space. Consider Fig. 4a. For a static environment, we can sample indirect illumination at points A and B and interpolate between them, because the effect of object O on diffuse or near-diffuse reflection varies continuously between A and B. The amount of light from O that is reflected from A is slightly higher than the amount reflected from B because O subtends a larger solid angle from A, and because the angle of O to the surface normal is slightly lower at A.

If object O is moving, indirect illumination at point A at times **time 1** and **time 2** varies in the same manner that the indirect illumination varied with position between A and B in the static case. At **time 2**, the light A reflects from O is a bit less because the solid angle subtended by O is smaller, and the angle of O to the surface normal has increased.

Because the changes in indirect illumination resulting from object motion are equivalent to the changes in indirect illumination as a function of distance in the static case, we can sparsely sample indirect illumination in time as well as in space.

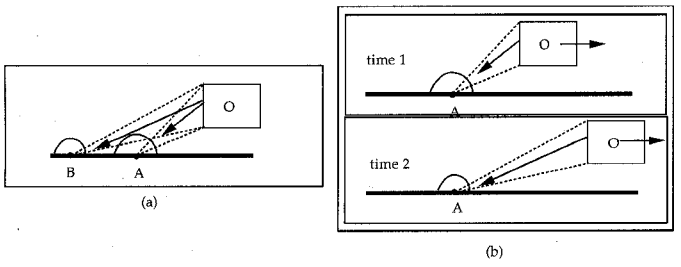


Fig. 4. (a) illustrates that the indirect illumination of point A by object O is only slightly different than the indirect illumination of point B by object O. (b) illustrates that the difference in indirect illumination of point A caused by the movement of object O from time 1 to time 2 is the same as the difference in indirect illumination between A and B in the static case.

Our approach then is to compute indirect illumination for sparsely separated points for very large time steps. Interpolation between these solutions will then be used to compute the indirect illumination for the time series of images at each base view point.

3.3 The Whole System

The overall framework of our approach is shown in Fig. 5. The indirect illumination is sampled sparsely in time and space in object space (top row). The indirect illumination solution is then interpolated and used as the basis to produce the full global illumination for each of the base images (middle row). Finally, for any path in the view space, images are generated for each frame by interpolating the base images (bottom row).

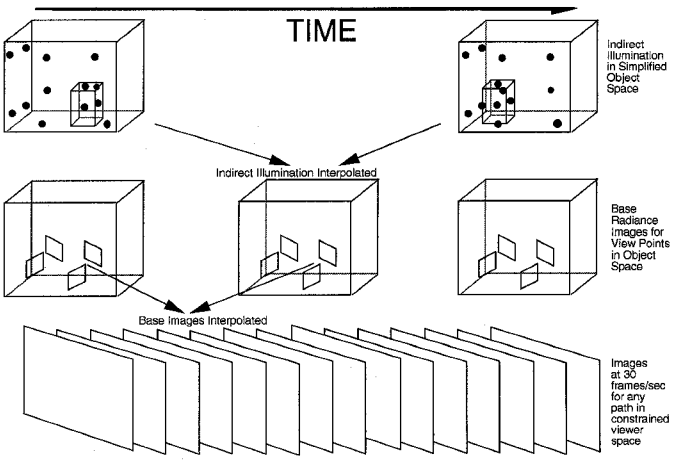


Fig. 5. Indirect illumination is sampled sparsely in time and space in a simplified, animated object space. Radiance images are computed more frequently in time for selected base views, using interpolated values from the indirect illumination solution. One image per frame is computed for any path in the view space by interpolating base images.

To build the system diagrammed in Fig. 5, we need the following:

1) Rules for selecting the base view positions so that the entire space a user may wish to tour is spanned,
2) Rules for selecting the times steps at which base views are computed so that motions are adequately represented, and
3) Rules for selecting the points and times for which indirect illumination is to be computed.

We also need to choose a particular method for computing the indirect illumination, the base images, and for efficiently performing the various interpolations. In the next section, we describe an initial implementation of the system we have just outlined.

4 Implementation

The overall structure of our initial implementation is shown in Fig. 6. As input, we have the environment description, the specification of objects and a *view space*. The view space defines the portion of the environment that can be toured. Locations for the base views are found by adaptive subdivision. The initial time discretization is estimated by accounting for direct lighting effects. The time discretization for each view is then further refined. In parallel to the establishment of base views, the sparsely sampled (geometrically simplified) indirect illumination solutions are calculated. The indirect illumination solutions are then interpolated and used to compute the base images. We

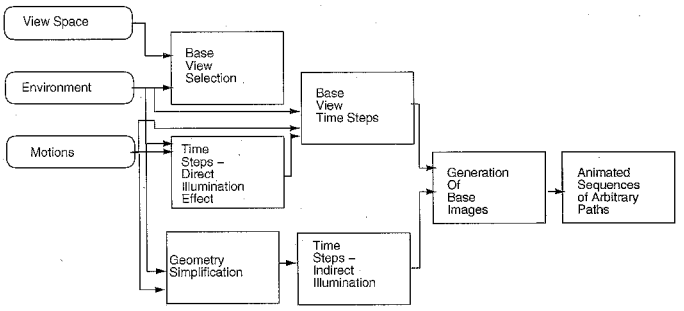


Fig. 6. Overview of initial implementation.

now consider each of the elements of the implementation in more detail.

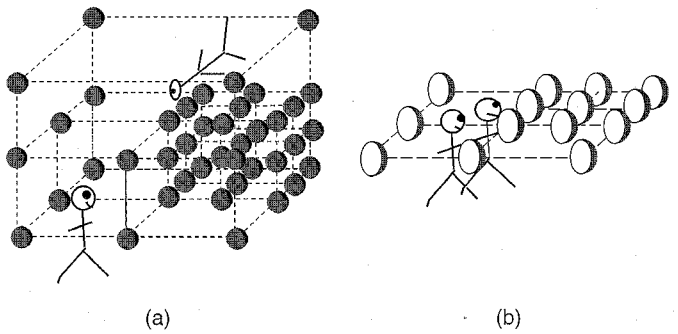
4.1 Selecting Base Views

The view space S is a set of view locations (i.e., eye points) that is some subset of the full environment. The view space may be 1D, 2D, or 3D, allowing the user to move along a line, within a plane or through a volume, respectively. In our initial implementation, we consider 1D and 2D spaces, although the methodology can easily be applied to the 3D case.

Ideally, to allow the user to have a free range of movement within S, the view for the entire sphere of directions should be stored, (see Fig. 7a). In our initial implementation, however, we consider only hemispherical fish-eye (i.e., 180 degree field) views (see Fig. 7b). To extend our method, we would simply need to define a scheme for sampling the sphere of directions nearly uniformly (i.e., avoiding concentrations of samples at the "poles"). Two hemispherical cameras with opposite view directions can also be used.

The base view locations in S are determined by adaptive subdivision. The subdivision level is controlled by a preselected quality parameter q, which ranges from zero to one. A value of q equal to zero will result in no subdivision returning the boundary elements of the space as the base locations; a value of q equal to one will result in a very dense population of view locations.

The subdivision begins by defining synthetic cameras at the corners of the space. A P by Q resolution *id-image* is formed for each camera. In an id-image, the pixel values are unique numerical identifiers for the objects visible at the pixel. Depending on the geometry of the objects, the id-image could be formed by a variety of techniques (scanline, ray tracing, etc.) In our implementation, we use the SGI display hardware to form the image, by assigning a unique 32-bit color to each polygon in place of its physical color. The number of pixels N that have the same id for all of the cameras is counted. The "fitness" f of this set of locations is computed as N/PQ. If f is less than the quality parameter q, the space is subdivided, and the test is applied recursively.



(a) (b)

Fig. 7. A viewer has maximum freedom in a 3D view space with full views stored (a). In the initial implementation, a 2D view space with fish-eye views is used (b).

4.2 Time Steps for Base Views

The frequency of time sampling for each base view is determined first by checking the general variability in direct illumination, and then by refining the time sequence according to a visibility test for each view point.

To check the general variability in direct illumination, we place a camera at the position of each light source and project all objects in the scene onto a camera using fish-eye projections. Initially, this is performed for the beginning and ending times of the animation, and a comparison of the resulting id-images is used to compute a value of f for the two points in time. The time interval is subdivided recursively until the value of f for all pairs of successive times exceeds q for each of the light sources.

The approach implicitly assumes point light sources. These points may be the centers of mass of clusters of small light sources, or may be sample points chosen on area light sources. Note that no illumination calculations are being performed with this process, we are simply estimating the time frequency for which highlights and shadows will need to be updated.

Next, for each of the base views, we wish to refine the time sequence further by using visibility tests for each of the viewpoints. Each view point could have its own time sequence—i.e., there may be relatively little motion in some views relative to others. In our initial implementation, just one master time sequence is generated. To do the base view time refinement, the procedure is the same as just described, but the list of base view locations is used in place of the light source locations.

4.3 Indirect Illumination

To compute indirect illumination, we use a wavelet radiosity solver [16], [15] on a simplified environment [23]. The time steps used for recomputing the indirect illumination are found by recursive subdivision. Alternatively, a static set of points could have been chosen to compute indirect illumination with the *Radiance* package. The points could have been chosen by doing a set of "overture" calculations for random views within the view space. The advantage of the radiosity preprocess, however, is that the resulting values are associated with objects. For large moving objects, both the value of the illumination and the position at which that illumination is valid can be interpolated in time.

A number of methods could be used to simplify the full mega-object environment to a relatively small number of polygons for the radiosity solution. We have tried two methods. The first uses a simple criterion based on visibility from within the view space. For each of some number of trials, a random time is chosen. For the environment at that time, random viewpoints in the view space are chosen. For each view, a large number of rays are shot through

a wide viewing angle. A record is kept of how many times each object is hit in this process. Based on the number of hits, an object may be ignored (a very small number of hits), simplified (somewhat larger number of hits), or restricted in subdivision (large number of hits) in the radiosity solution. The second method uses an area/volume limit criteria and prunes surfaces comparing their size to the average size of the environment. Also incorporated in this method is a tesselation criteria that allows us to simplify geometric primitives that pass the area/volume criteria into a simplified polygonal mesh. Here, we place an artificial limit on the minimum size into which a surface may be decomposed. This bounds the growth of the environment—precluding individual surfaces from being subdivided into patches that are either smaller than a certain size, or limiting the number of subdivision levels for a particular surface.

The times for environment simplification are given in Table 1. The scene decomposition times are given in terms of seconds, rather than the hours required for image generation. Note that with the first method, the number of rays required to explore the environment is very small compared to the number of rays needed to accurately compute illumination.

In the indirect solution, the objects move and the radiosities change. However, once the simplified geometry has been defined, the list structure storing the indirect illumination does not change. This makes it easy to perform an object space "fitness" test. In the indirect case, f is defined as the number of vertices in the solution that have changed less than some small percentage ϵ. Once again, the time sequence is subdivided recursively based on a comparison of f and q. Note that the list of times for which each indirect illumination solution is computed is determined completely apart from the times for which each new base view is computed.

4.4 Computing and Using the Base Images

To compute the base images, we use *Radiance*, substituting the indirect illumination values we computed with the radiosity solver, interpolated for the appropriate time, for the cached values normally used.

Once all the base images are computed for the selected locations and times, frame sequences for arbitrary paths through the view space can be generated by interpolating between the images closest in time and space. The view interpolation is performed with the *pinterp* function, which comes with the *Radiance* distribution. *Pinterp* treats the array of pixels in each range image as a set of points in three dimensional space, and forms a new image by projecting these points into the screen coordinates of the new view direction. This is essentially the image morphing for view interpolation described by Chen and Williams [9]. Time interpolation is performed by linearly interpolating between images pixel by pixel using the *pcomb* function supplied with *Radiance*.

5 Results and Analysis

In this section, we critically evaluate the time and quality tradeoffs involved in using the framework. We take the approach of evaluating individual phases of the framework for a small number of environments to get some initial insights into the tradeoffs involved. Specifically, we consider the following components: geometric simplification and indirect illumination, temporal interpolation of indirect illumination, base view selection, and motion interpolation. We conclude with the application of the complete framework to a complex environment that includes object motion.

Since our framework is an image-based approach, we use image space comparisons to determine the quality of different solutions. Solutions computed with the *Radiance* system with high quality settings are used as the reference standards. Since high quality settings are rarely used, we calibrate the quality of our solutions by comparing their accuracy to *Radiance* images using standard settings. These settings have been determined empirically over the lifetime of the program and often lead to the best time/quality ratio in the solutions that are generated. Although it is possible to compute full Monte Carlo reference images, these images are prohibitively expensive to compute.

The measurement of image quality is still an open research issue [24]. Despite its inadequacy on many counts, pixel by pixel RMS error is the only universally accepted measure for evaluating images. Here, we use a relative luminance RMS error (luminance RMS/reference solution average luminance) as a quality metric.

5.1 Indirection Illumination

An important premise on which we have based the framework is that indirect illumination is generally a smoother function than direct illumination and can, therefore, be sampled more sparsely in space. Also, as we observed in Section 3.2, we can use the same coarse sampling of object space to exploit temporal coherence.

We begin the study by analyzing our methods of geometric simplification for indirect illumination calculation and temporal indirect illumination interpolation. *Radiance's* mechanism for incorporating precomputed indirect irradiance values in the form of an ambient file allows us to easily test our methods of approximating the indirect illumination.

5.1.1 Geometric Simplification

Geometric simplification can yield a significant reduction in the computation of indirect illumination with minimal loss in image fidelity. The problem is knowing how much simplification can be performed before a noticeable error has been introduced.

We used a model of a studio environment that contains a large number of surfaces and a set of varied surface BRDFs to test the effectiveness of our geometric simplification modules. The model includes approximately 70,000 surfaces and has many windows that admit sunlight. A very challenging view has been selected in which indirect illumination from the sunlight reflected off the vertical surfaces is very important.

5.1.1.1 Testing Procedure. We simplified the studio model using object pruning and decomposition. The user-defined quality parameter was used to calculate the minimum area/volume requirement for primitives and

the tesselation detail level specifier was then used to decompose those primitives that met the area/volume requirement (see Section 4.3). For this analysis, we simplified the studio environment using low, medium, and high quality settings. We then ran the wavelet radiosity solver on each of the environments to generate the indirect illumination solutions. These solutions were converted to *Radiance* ambient files and then used to generate the final solution images. We compared these three images to an image generated by *Radiance* using the high quality settings; we used a relative luminance RMS metric, the ratio of image LRMS to the average luminance of the *Radiance* HQ solution.

5.1.1.2 Observations. The results (Table 1 and Fig. 8) of this analysis are very promising. The low and medium quality indirect solutions are comparable to the *Radiance* typical solution and take a similar amount of time to compute. The advantage of our approach, over the *Radiance* software, is that the indirect component of illumination does not need to be recomputed or updated for each view. Furthermore, in our framework the indirect component is computed at particular surfaces, making possible temporal reconstructions that are otherwise intractable.

5.1.2 Temporal Interpolation of Indirect Illumination

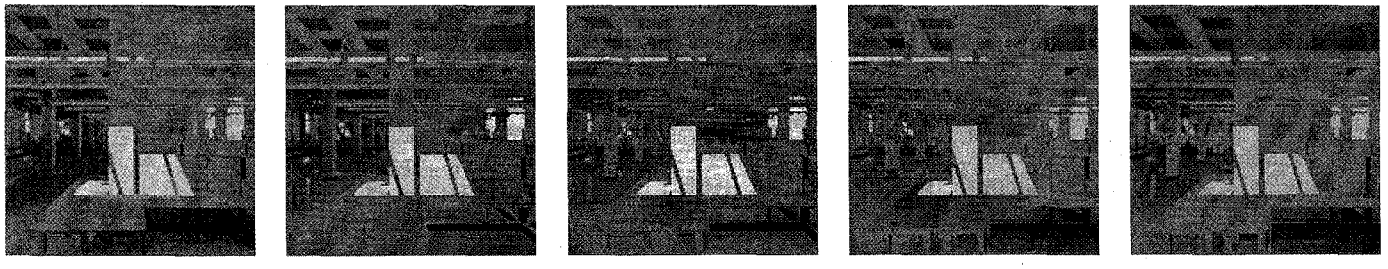
To examine our approach to modeling the temporal variations in indirect illumination, we elected to work with the Cornell box environment. We chose this simple model as a means to help isolate the interpolation error from the geometric simplification error presented in the previous section. We modified this model to include time-based motion paths for the enclosed boxes.

5.1.2.1 Testing Procedure. We decomposed the environment into temporal segments in which the indirect illumination on the surfaces did not vary beyond the user-specified quality. For the environment and the chosen time interval, we placed the objects where they should be at the ends of the time interval and then generated an indirect irradiance solution for both times using the wavelet radiosity solver. We then computed the average of the object space pointwise irradiance RMS over the irradiance average and compared it to the quality parameter. This computation weights the magnitude of the irradiance RMS by the average irradiance of the two solutions to give a sense of the relative magnitude of the RMS. If the particular time interval is deemed to be "incoherent," the interval is subdivided and the same procedure is applied recursively to the two subintervals. The decomposition procedure terminates with a set of times representing the endpoints of coherent indirect illumination intervals from which the intermediate indirect solutions can be computed via interpolation. Interpolation is performed by placing the objects where they should be at the appropriate time and linearly interpolating pointwise indirect irradiance values for the surfaces.

We decomposed the box environment using three different quality parameters, which yielded three time lists. For each element in the time lists, an indirect illumination solution was stored. We then chose an intermediate time and generated four indirect illumination solutions. Three were

TABLE 1
GEOMETRIC SIMPLIFICATION STATISTICS FOR THE STUDIO

Geometric Simplification–Studio				
Scene Decomposition				
Quality	Initial Surfaces	Pruned Surfaces	Final Surfaces	Time (secs)
Low–.3	68,718	54,710	14,008	32.33
Medium–.6	68,718	37,120	31,598	38.41
High–.9	68,718	2,616	66,102	34.95
Analysis				
Image	Computation Time (hours)		LRMS Error/Rad. HQ Avg. Lum.	
Radiance HQ	42.201		0.000/5.468 = 0.000	
Radiance Typical	23.719		1.967/5.468 = 0.360	
WR Low Quality	18.240		2.081/5.468 = 0.381	
WR Medium Quality	26.867		1.858/5.468 = 0.340	
WR High Quality	35.960		1.853/5.468 = 0.339	



(a) (b) (c) (d) (e)

Fig. 8. Comparison of studio images generated using geometrically simplified indirect illumination solutions: (a) radiance HQ, (b) radiance typical, (c) wavelet radiosity low, (d) wavelet radiosity medium, (e) wavelet radiosity high.

linearly interpolated from the nearest indirect solutions in the different quality time lists. One was generated by the wavelet radiosity solver for the chosen time. We then converted these indirect solutions to *Radiance* ambient files and generated multipass images. These four images were compared to a *Radiance* HQ image using the relative LRMS described earlier.

5.1.2.2 Observations. The results for this simple environment (Table 2 and Fig. 9) show that our WR method for computing indirect illumination at several quality levels outperforms the typical *Radiance* solution, while requiring less computation time. While the accuracy increases with the quality of the reconstructions, the dependence is not very strong; the error curve is approximately linear with errors of .147, .138, and .119 for quality levels of .3, .6, and .9. Since we sample the indirect illumination in both space and time, we can interpolate the indirect illumination at any point within a time sequence, without making new calculations. Although *Radiance* can approximate a view-independent indirect illumination solution with a set of overture calculations, it does not associate the irradiance values to points on surfaces. For environments in which objects move, there is no easy way to interpolate between different *Radiance* indirect illumination solutions, which means that the costly indirect illumination computation must be performed whenever any object motion occurs. This is intractable for animations in which there are objects moving in every frame.

5.2 Image Generation

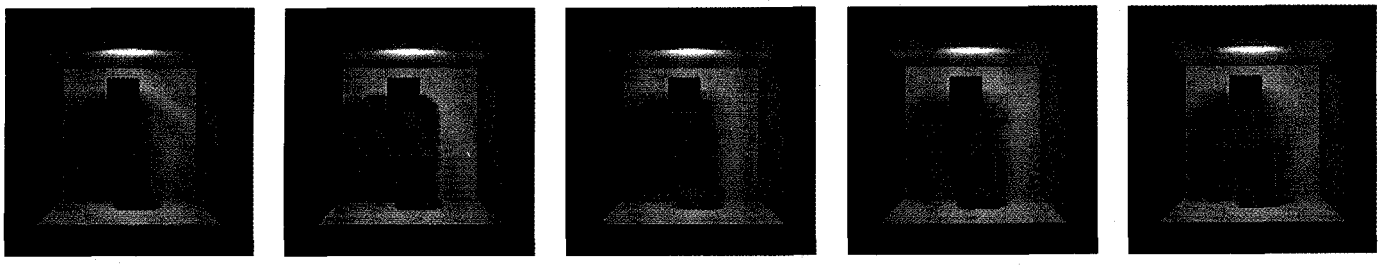
The analysis in the previous section related approximations in the object space phases of the framework to error in the final images. We now consider those phases that operate in image space; these affect direct illumination, visibility, and view coherence.

5.2.1 Base View Selection–View Reconstruction

In order to test the error incurred by performing quality-driven view space decomposition and view reconstruction, we return to the studio environment. The large number of surfaces challenges the coherence testing procedure, and the view-dependent illumination effects test the reconstruction procedure.

TABLE 2
TEMPORAL INDIRECT ILLUMINATION INTERPOLATION STATISTICS FOR THE CORNELL BOX ENVIRONMENT

Temporal Indirect Illumination Interpolation–Cornell Box			
Image	Phase	Comp. Time–30 Frame Seq. (mins)	LRMS Error/Rad. HQ Avg. Lum. (per frame)
Radiance HQ	Rendering	1,179.00	0.000
Radiance Typical	Rendering	190.80	0.020/0.109 = 0.183
WR Low Quality	Scene Decomposition	0.04	
	Indirect Generation	0.73	
	Indirect Interpolation	0.94	
	Rendering	88.20	
	Total	89.91	0.016/0.109 = 0.147
WR Medium Quality	Scene Decomposition	0.14	
	Indirect Generation	1.03	
	Indirect Interpolation	0.94	
	Rendering	93.60	
	Total	95.71	0.015/0.109 = 0.138
WR High Quality	Scene Decomposition	0.31	
	Indirect Generation	1.69	
	Indirect Interpolation	0.94	
	Rendering	111.60	
	Total	114.54	0.013/0.109 = 0.119



(a) (b) (c) (d) (e)

Fig. 9. Comparison of Cornell box images generated with temporally interpolated indirect components: (a) radiance HQ, (b) radiance typical, (c) wavelet radiosity low, (d) wavelet radiosity medium, (e) wavelet radiosity high.

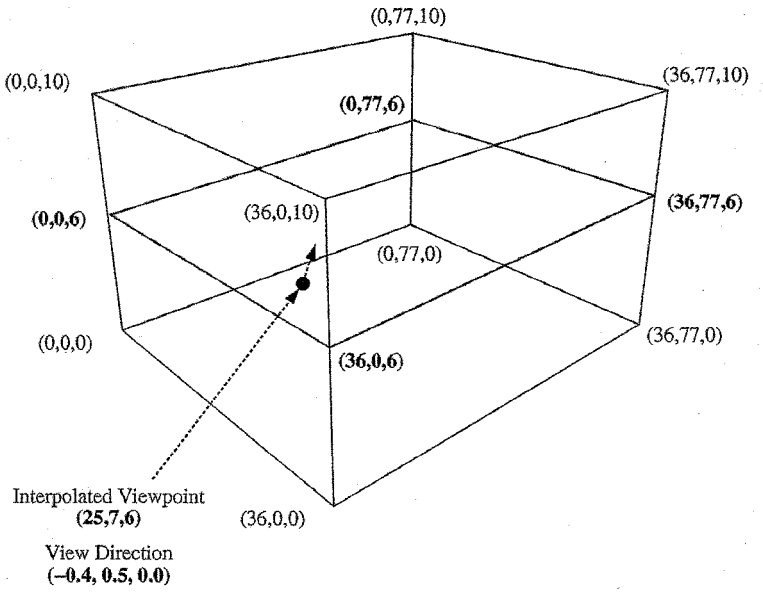
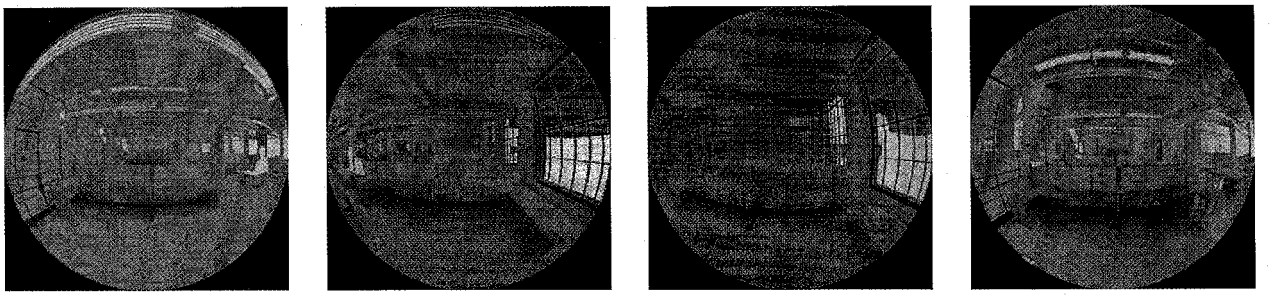


Fig. 10. Studio dimensions–2D view space.

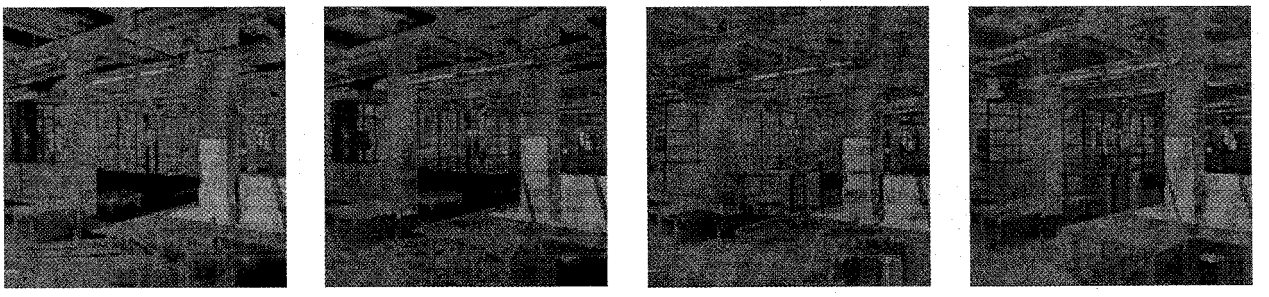
5.2.1.1 Testing Procedure. In order to test the effectiveness of the view space decomposition and interpolation modules, we have chosen a two-dimensional view space within the studio environment in which we will move freely. Fig. 10 provides a diagram of the view space. Figs. 11-14 show sample base views and different quality reconstructions using four fill algorithms available with the *Radiance pinterp* function. In the "no fill" algorithm, each pixel in each base image is projected onto at most one pixel in the new image, and some pixels in the new image are left with a default background color. In "foreground fill," pixels in the base images are treated as polygons, and may be projected onto multiple pixels in the new image. In "background fill," each base image is projected to at most one new image pixel, and unfilled pixels are assigned the values of surrounding filled pixels. In "foreground/background," the foreground fill technique is used, with the background technique used to fill any remaining holes.

For simplicity in analyzing the effect of quality in a single image, we considered a simplified one-dimensional subset of the view space as shown in Fig. 15. The spatial decomposition module was executed with three different



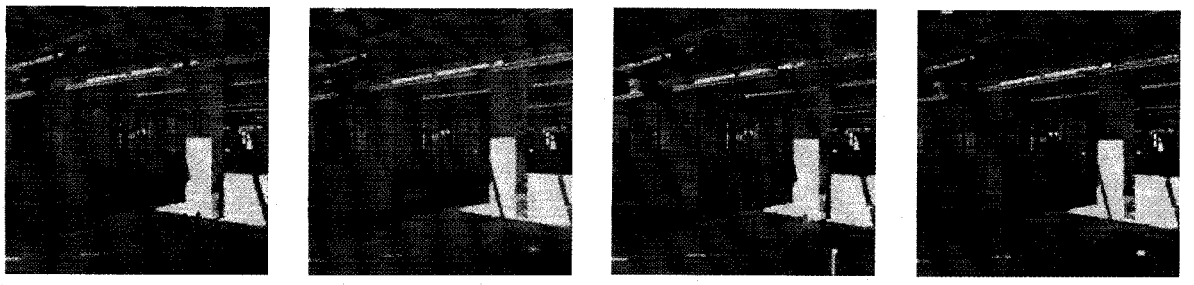
(a) (b) (c) (d)

Fig. 11. Four medium quality base images from the two-dimensional studio view space.



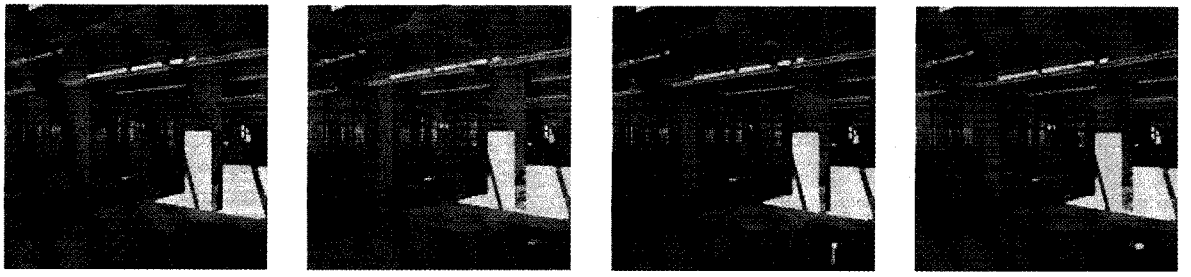
(a) (b) (c) (d)

Fig. 12. Four pixel-based view reconstruction fill algorithms (low quality spatial decomposition): (a) no fill, (b) foreground fill, (c) background fill, (d) foreground and background fill.



(a) (b) (c) (d)

Fig. 13. Four pixel-based view reconstruction fill algorithms (medium quality spatial decomposition): (a) no fill, (b) foreground fill, (c) background fill, (d) foreground and background fill.



(a) (b) (c) (d)

Fig. 14. Four pixel-based view reconstruction fill algorithms (high quality spatial decomposition): (a) no fill, (b) foreground fill, (c) background fill, (d) foreground and background fill.

quality levels; it returned three pairs of base locations (corresponding to the respective quality levels) to be used for the different quality reconstructions. For each pair of base locations, 180° range-images were generated in *Radiance* for use as the base images. We then chose a viewpoint and field of view to reconstruct that fell in the middle of the pairs of base locations. Using the three pairs of base images, the view interpolation module, *pinterp*, was used to reconstruct the different quality images of the environment from the chosen view. *Radiance* was used to generate a reference image for the chosen view and field of view. The analysis was completed by calculating the relative LRMS error between the various images.

5.2.1.2 Observations. The raw statistics and images (Table 3 and Fig. 16) for the pixel-based view reconstruction are especially interesting. One source of error is due to the fact that *pinterp* produces slightly different results depending on the ordering of the input files and a tolerance factor. We ordered the files by linear distance from the interpolation point under the assumption that the images from closer base viewpoints would be more valid than those from greater distances. Since the method uses reprojection of pixels from the base images as its means of reconstruction, a high initial resolution is required to remove any quantization effects and resampling errors. This effect is shown in the row labeled "Resampled Radiance," which

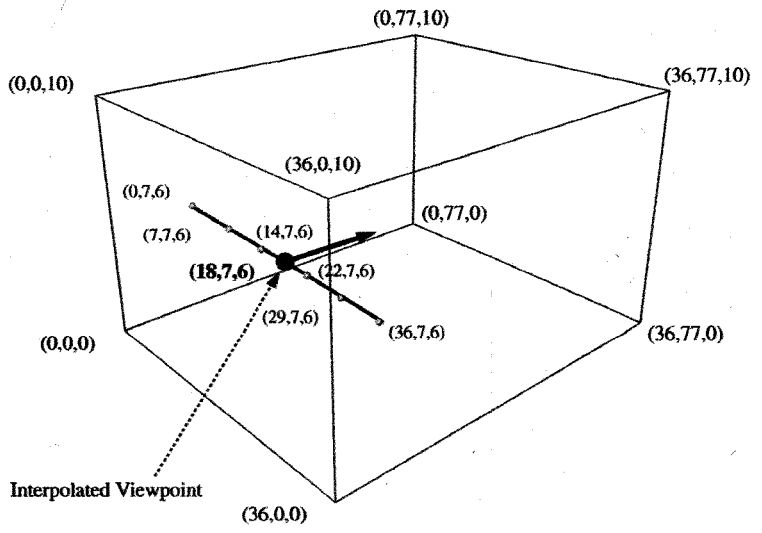


Fig. 15. Studio dimensions–1D view subspace.

specifically measures the error involved in downsampling from a spherical image at the chosen viewpoint to the desired field-of-view image at the same point. This error is very large and suggests that a more sophisticated approach is needed; future work will address this shortcoming. It is important to recognize the existence of this error when viewing the statistics.

In addition, to ensure good results, the setting for q for this step has to be high ($\geq .9$), as the results are very sensitive to having the correct objects visible. Although q was high, we were able to get reasonable results by interpolating views that are seven meters apart.

The reconstruction process itself works well for views that flank the chosen view. However, the error increases dramatically as we start to move our base images away from the chosen viewpoint. Mutually occluded information in the base images that must be "filled" in the reconstructed image can also cause severe discrepancies in the compared images and a large RMS error. Fortunately, the areas that are most problematic are somewhat isolated. Perhaps rather than having a spatially denser set of complete base images, a few partial images for areas of the view that change rapidly with location could be used. These partial samples could benefit other image-based approaches as well, including our own motion interpolation method.

5.2.2 Motion Interpolation

Interpolating motion from images is a difficult problem. Pixel-based interpolation methods are inadequate for the reconstruction of object motion unless a large set of images is generated. To keep our initial implementation as simple as possible, we used pixelwise linear interpolation as a temporal reconstruction algorithm. This, of course, means that our temporal decomposition approach oversamples in an attempt to remove undesirable visual artifacts.

In order to test this phase of the framework, we chose to return to the Cornell box model. The simplicity of this environment allows us to easily isolate the error introduced by motion interpolation from other errors that can be introduced by the framework.

5.2.2.1 Testing Procedure. We analyzed motion sampling and reconstruction from the point of view of a single, fixed view. For the chosen view, we decomposed time into coherent segments using our visible object coherence and direct illumination coherence algorithms. At the ends of the chosen time interval, if the percentage of pixels that "sees" the same

TABLE 3
VIEW RECONSTRUCTION STATISTICS FOR THE STUDIO ENVIRONMENT

View Reconstruction–Studio		
Scene Decomposition		
Quality	Number of Views	Computation Time (secs)
Low Quality–.3	2	10.77
Medium Quality–.6	3	34.12
High Quality–.9	5	116.05
Base View Synthesis		
Image		Computation Time (mins)
180° camera		728.7
Base 1–Low Quality (.3)		621.8
Base 2–Low Quality (.3)		570.0
Base 1–Medium Quality (.6)		725.6
Base 2–Medium Quality (.6)		647.2
Base 1–High Quality (.9)		707.2
Base 2–High Quality (.9)		592.3
Synthesis		
Image	Computation Time (mins)	LRMS Error/Rad. HQ Lum. Avg.
Radiance HQ	45.7	0.000
Low Quality (.3)	0.048	8.75/13.72 = 0.638
Medium Quality (.6)	0.061	8.57/13.72 = 0.625
High Quality (.9)	0.046	1.61/13.72 = 0.117
Resampled Radiance	0.035	1.64/13.72 = 0.120

(a) (b) (c) (d) (e)

Fig. 16. Comparison of studio images generated via view reconstruction: (a) radiance HQ, (b) low quality reconstruction, (c) medium quality reconstruction, (d) high quality reconstruction, (e) resampled radiance.

object is greater than the user-specified quality parameter, the interval is deemed coherent with respect to object visibility. A similar test is performed from the point of view of the light sources to test direct illumination coherence. If the interval passes both tests, it is coherent. Subdivision occurs for any interval that fails either test until a list of coherent time intervals is created.

We decomposed the box environment using three different quality levels, which yielded three time lists. For each of the time lists, we generated the set of temporal base images. We then compared a fully rendered set of twelve *Radiance* images against the different quality base image sets; intermediate frames were filled in by linear reconstruction. This simple test allowed us to isolate the error introduced by motion reconstruction alone.

5.2.2.2 Observations. The results (Table 4, Fig. 17, and Fig. 18) from this phase indicate that we can achieve

TABLE 4
MOTION INTERPOLATION STATISTICS FOR THE CORNELL BOX ENVIRONMENT

Motion Interpolation–Cornell Box			
Scene Decomposition			
Quality	Number of Solutions	Generated Time List	Computation Time (secs)
Low Quality -.3	2	0, 48	0.76
Medium Quality -.6	3	0, 24, 48	1.93
High Quality -.9	5	0, 12, 24, 36, 48	6.64
Synthesis			
Time (secs)			
Fully Rendered	Low Quality–.3	Medium Quality–.6	High Quality–.9
15,204	2,711	3,837	6,030
Analysis			
Frame 10 - LRMS/Rendered Frame Avg. Lum.			
Fully Rendered	Low Quality–.3	Medium Quality–.6	High Quality–.9
0.000	0.118/0.139 = .849	0.104/0.139 = .748	0.080/0.139 = .576

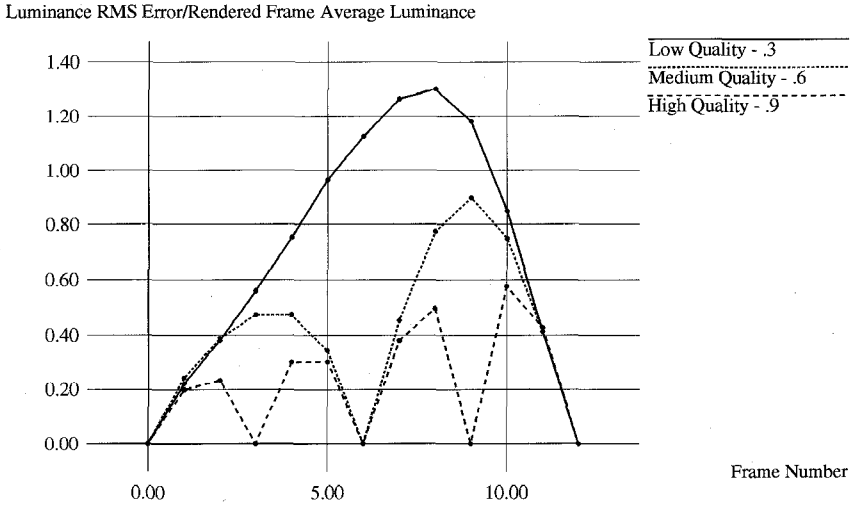


Fig. 17. Error graph for motion interpolation within the Cornell box environment.

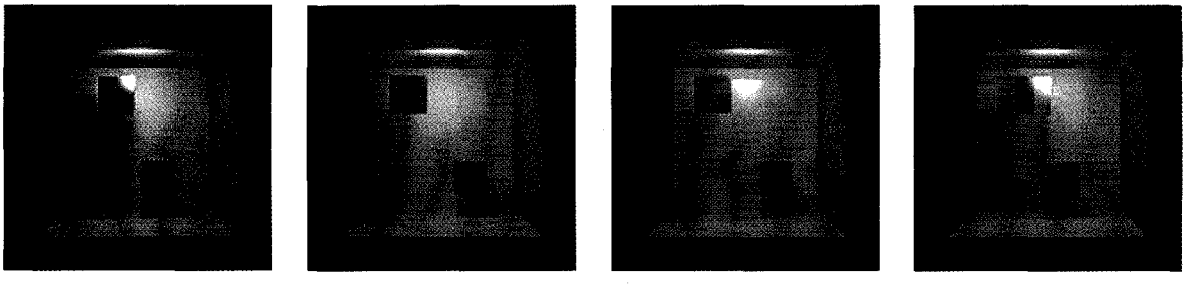


Fig. 18. Comparison of frame 10 in the motion sequence: (a) radiance HQ, (b) low quality reconstruction, (c) medium quality reconstruction, (d) high quality reconstruction.

reasonable accuracy by using a large number of temporal samples. However, the error is of a type that is especially objectionable. Pixel-flow algorithms would provide a better mechanism for this type of reconstruction, but are more complicated and require more information than is typically stored in a range-image. The pixel flow must be calculated and stored with the pixel values. We plan to address the problem of determining an adequate temporal sample set within the context of a more robust motion reconstruction algorithm in a subsequent work. As we mentioned in the previous section, it should be possible to augment any motion reconstruction algorithm to use partial samples.

5.3 Putting It All Together—The Soda Shop

In this section, we examine the performance of the framework as a whole by taking an example from the specification stage through to the walk-through stage. This analysis involves integrating all of the phases of the framework, including indirect and direct illumination, and both domains, spatial and temporal. We use a soda shop environment of moderate complexity for this analysis. The scene contains complex surface BRDFs and textures and small objects that move.

5.3.1 Testing Procedure

We began by performing a decomposition for the indirect illumination. This entailed simplifying the environment and then decomposing it as a function of time. We performed this decomposition for three different quality levels: low, medium, and high; the same quality parameters were used through the different phases of the framework. After treating indirect illumination, direct illumination and visibility were computed yielding the following sample points in space and time:

- *Low quality*: Two spatial locations with two times for each location.
- *Medium quality*: Three spatial locations with three times for each location.
- *High quality*: Five spatial locations with five times for each location.

For each of the spatial locations and each of the times, a 360° base image was generated for use in the reconstruction phase.

After the bases were constructed, a path through the environment was chosen, including the associated camera parameters. Frames were then rendered using *Radiance* with its high quality settings. We also reconstructed the path using our low, medium, and high quality bases. The individual frames were then compared using the relative LRMS error metric.

5.3.2 Observations

The results (Table 5, Fig. 19, and Fig. 20) for the walk-through are encouraging. The high quality reconstructions are comparable to the high quality *Radiance* renderings. In addition, we can successfully control the quality of the sequence and trade time for numerical accuracy with the parameters we have defined. For our results, we used the chosen quality level through all the steps in the framework. It is clear, however, that quality (as we have currently defined it) does not have the same impact on each step in the process. The framework needs to be refined to take a user-specified quality and interpret it relative to each step in the process; this will be a subject of future work.

TABLE 5
EXECUTION TIMES FOR INDIVIDUAL PHASES OF THE FRAMEWORK ON THE SODA SHOP ENVIRONMENT

Walkthrough Reconstruction–Soda Shop	
Scene Decomposition/Base Image Construction	
Quality	Time (mins)
Low Quality–.3	1,213.53
Medium Quality–.6	2,894.22
High Quality–.9	7,746.76
Walkthrough Construction–30 Frames	
Quality	Time (mins)
Fully Rendered	4,113.54
Low Quality–.3	90
Medium Quality–.6	210
High Quality–.9	480

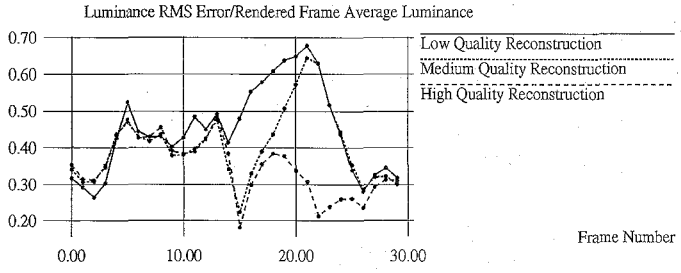


Fig. 19. Error graph for walk-through reconstruction within the soda shop environment.

6 SUMMARY AND DISCUSSION

We have presented a new framework for efficiently computing and storing global illumination effects in complex, animated environments. Our approach is a range-image based system, which exploits coherence by computing direct and indirect illumination separately in both space and time. Indirect illumination is computed for sparsely separated points for very large time steps. Interpolation between these solutions is used to compute the indirect illumination for the time series of images at each base view point. We demonstrate that the range-image approach allows the rapid generation of the views along arbitrary paths within a view space, which is a subset of the whole environment. The framework represents a major step toward the ultimate goal of allowing users to interact with accurately rendered, animated, geometrically complex environments, as it allows for a user to tour a view space rendered with full global illumination effects in which objects move.

Within the context of the framework, there are several areas that require future research. First, we would like to devise a scheme to define and quantify an acceptable quality level for each aspect of the framework. We would also like to determine an alternative approach to LRMS for measuring the quality of the solution; this would allow us to couple a notion of quality of the solution to what is going to be visible in the final image. We would also like to address the problem of accurate interpolation of object motion from images. The

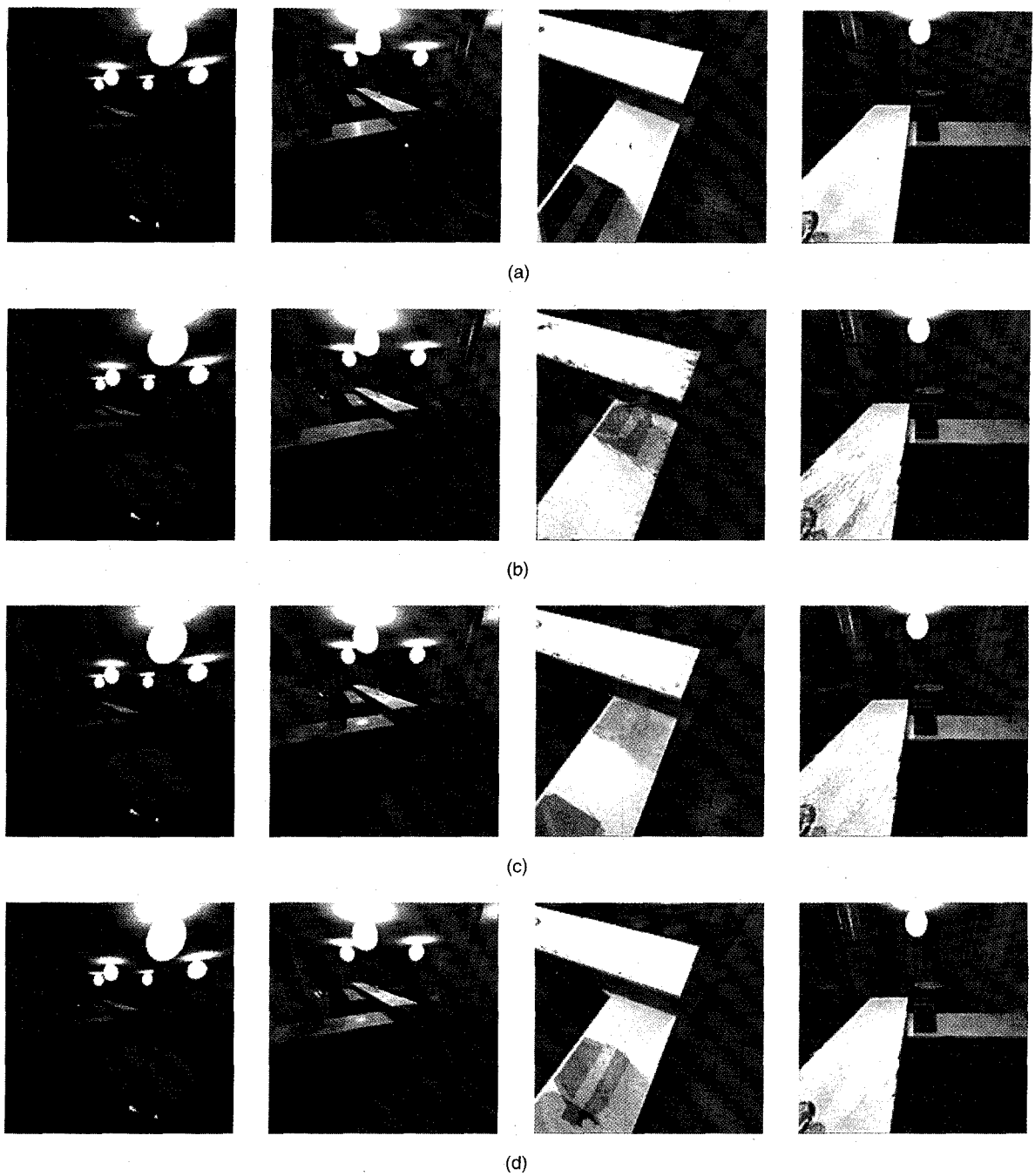


Fig. 20. Four frame comparison in the soda shop environment: (a) fully rendered, (b) low quality reconstruction, (c) medium quality reconstruction, (d) high quality reconstruction.

notion of freedom of movement in the context of the framework seems to be another potential area of interesting work. For example, many tradeoffs are possible between user interactivity and the quality of the animated sequence. The problem of automatically determining the base views for a three-dimensional environment is an open research topic, which could involve research in computational geometry. Another important problem is temporal aliasing. Methods need to be developed to avoid missing motions that may cycle in time periods shorter than the initial time samples used in the adaptive subdivision. Last, there is a potential for real time walk-throughs using specialized hardware for the view and time interpolations.

References

[1] J. Airey, J. Rohlf, and F. Brooks, Jr., "Towards Image Realism with Interactive Update Rates in Complex Virtual Building Environments," *Computer Graphics, 1990 Symp. Interactive 3D Graphics*, vol. 24, no. 2, pp. 41-50, Mar. 1990.

[2] J. Arvo, "The Irradiance Jacobian for Partially Occluded Polyhedral Sources," *Computer Graphics (SIGGRAPH '94 Proc.)*, vol. 28, no. 4, pp. 343-350, July 1994.

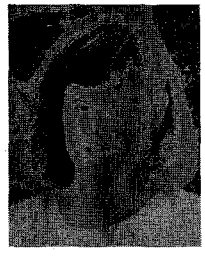
[3] D. Baum, J. Wallace, and M. Cohen, "The Back-Buffer Algorithm: An Extension of the Radiosity Method to Dynamic Environments," *The Visual Computer*, vol. 2, no. 5, pp. 298-306, 1986.

[4] K. Blanton, "A New Approach for Flight Simulator Visual Systems," *Simulators IV, Proc. SCCS Simulators Conf.*, pp. 229-233, 1987.

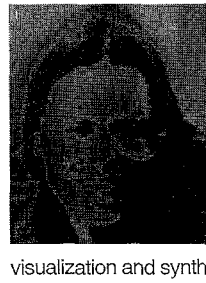
[5] K. Boff and J. Lincoln, *Engineering Data Compendium: Human Perception and Performance*, vol. 1. Wright-Patterson Air Force Base, 1988.

[6] S.E. Chen, "Incremental Radiosity: An Extension of Progressive Radiosity to an Interactive Image Synthesis System," *Computer Graphics (SIGGRAPH '90 Proc.)*, vol. 24, no. 4, pp. 135-144, Aug. 1990.
[7] S.E. Chen, "Quicktime VR—An Image-Based Approach to Virtual Environment Navigation," *Computer Graphics (SIGGRAPH '95 Proc.)*, vol. 29, no. 4, pp. 29-38, July 1995.
[8] S.E. Chen, H. Rushmeier, G. Miller, and D. Turner, "A Progressive Multi-Pass Method for Global Illumination," *Computer Graphics (SIGGRAPH '91 Proc.)*, vol. 25, no. 4, pp. 165-174, July 1991.
[9] S.E. Chen and L. Williams, "View Interpolation for Image Synthesis," *Computer Graphics (SIGGRAPH '93 Proc.)*, vol. 27, no. 4, pp. 279-288, Aug. 1993.
[10] J. Dorsey, J. Arvo, and D. Greenberg, "Interactive Design of Complex Time-Dependent Lighting," *IEEE Computer Graphics and Applications*, vol. 15, no. 2, pp. 26-36, Mar. 1995.
[11] D. Forsyth, C. Yang, and K. Teo, "Efficient Radiosity in Dynamic Environments," *Proc. Fifth Eurographics Workshop on Rendering*, pp. 313-324, June 1994.
[12] T. Funkhouser and C. Sequin, "Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments," *Computer Graphics (SIGGRAPH '93 Proc.)*, pp. 247-254, July 1993.
[13] D. George, F. Sillion, and D. Greenberg, "Radiosity Redistribution for Dynamic Environments," *IEEE Computer Graphics and Applications*, vol. 10, no. 4, pp. 26-34, July 1990.
[14] C. Goral, K. Torrance, D. Greenberg, and B. Battaile, "Modeling the Interaction of Light Between Diffuse Surfaces," *Computer Graphics (SIGGRAPH '84 Proc.)*, vol. 18, no. 3, pp. 212-222, July 1984.
[15] S. Gortler, P. Schröder, M. Cohen, and P. Hanrahan, "Wavelet Radiosity," *Computer Graphics (SIGGRAPH '93 Proc.)*, vol. 27, no. 4, pp. 221-230, Aug. 1993.
[16] P. Hanrahan, D. Salzman, and L. Aupperle, "A Rapid Hierarchical Radiosity Algorithm," *Computer Graphics (SIGGRAPH '91 Proc.)*, vol. 25, no. 4, pp. 197-206, July 1991.
[17] J. Kajiya, "The Rendering Equation," *Computer Graphics (SIGGRAPH '86 Proc.)*, vol. 20, no. 4, pp. 143-150, Aug. 1986.
[18] L. McMillan and G. Bishop, "Plenoptic Modeling: An Image-Based Rendering System," *Computer Graphics (SIGGRAPH '95 Proc.)*, vol. 29, no. 4, pp. 39-46, July 1995.
[19] S. Müller and F. Schöffel, "Fast Radiosity Repropagation for Interactive Virtual Environments Using a Shadow-Form-Factor List," *Proc. Fifth Eurographics Workshop on Rendering*, pp. 325-342, Darmstadt, Germany, June 1994.
[20] J. Nimeroff, J. Dorsey, and H. Rushmeier, "A Framework for Global Illumination in Animated Environments," *Proc. Sixth Eurographics Workshop on Rendering*, pp. 223-236, Dublin, Ireland, June 1995.
[21] C. Puech, F. Sillion, and C. Vedel, "Improving Interaction with Radiosity-Based Lighting Simulation Programs," *Computer Graphics (1990 Symp. Interactive 3D Graphics)*, pp. 51-57, Mar. 1990.
[22] M. Regan and R. Pose, "Priority Rendering with a Virtual Reality Address Recalculation Pipeline," *Computer Graphics (SIGGRAPH '94 Proc.)*, vol. 28, no. 4, pp. 155-162, July 1994.
[23] H. Rushmeier, C. Patterson, and A. Veerasamy, " Geometric Simplification for Indirect Illumination Calculations," *Proc. Graphics Interface '93*, pp. 227-236, Canadian Information Processing Society, May 1993.
[24] H. Rushmeier, G. Ward, C. Piatko, P. Sanders, and B. Rust, "Comparing Real and Synthetic Images: Some Ideas About Metrics," *Proc. Sixth Eurographics Workshop on Rendering*, pp. 213-222, Dublin, Ireland, June 1995.
[25] E.S. Shaw, "Hierarchical Radiosity for Dynamic Environments," master's thesis, Program of Computer Graphics, Cornell Univ., Aug. 1994.
[26] F. Sillion, J. Arvo, S. Westin, and D. Greenberg, "A Global Illumination Solution for General Reflectance Distributions," *Computer Graphics (SIGGRAPH '91 Proc.)*, vol. 25, no. 4, pp. 187-196, July 1991.
[27] F. Sillion and C. Puech, *Radiosity and Global Illumination*. San Francisco, Calif.: Morgan Kaufmann Publishers, 1994.
[28] B. Smits, J. Arvo, and D. Salesin, "An Importance-Driven Radiosity Algorithm," *Computer Graphics (SIGGRAPH '92 Proc.)*, vol. 26, no. 4, pp. 273-282, July 1992.
[29] S. Teller and C. Séquin, "Visibility Preprocessing for Interactive Walkthroughs," *Computer Graphics (SIGGRAPH '91 Proc.)*, vol. 25, no. 4, pp. 61-69, July 1991.
[30] G. Ward, "Real Pixels," J.R. Arvo, ed., *Graphic Gems II*. San Diego, Calif.: Academic Press, 1991.
[31] G. Ward, "The Radiance Lighting Simulation and Rendering System," *Computer Graphics (SIGGRAPH '94 Proc.)*, vol. 28, no. 4, pp. 459-472, July 1994.
[32] G. Ward, F. Rubinstein, and R. Clear, "A Ray Tracing Solution for Diffuse Interreflection," *Computer Graphics (SIGGRAPH '88 Proc.)*, vol. 22, no. 4, pp. 85-92, Aug. 1988.

Jeffry Nimeroff received the BA and MA in computer science from Boston University in 1987 and 1988, respectively, and the MSE in computer and information science from the University of Pennsylvania in 1993. He is a lecturer in the Department of Architecture and a doctoral student in the Center for Human Modeling and Simulation (Department of Computer and Information Science) at the University of Pennsylvania. His research interests include efficient algorithms for global illumination and computer animation. He is a member of the ACM, ACM SIGGRAPH, and IEEE Computer Society.



Julie Dorsey received the BS and BArch degrees in architecture (1987), and the MS (1990) and PhD (1993) degrees in computer graphics from Cornell University. She is an assistant professor in the Department of Architecture and the Laboratory for Computer Science at the Massachusetts Institute of Technology. She received the 1995 Richard Kelly Award from the Illuminating Engineering Society of North America, the 1996 Faculty Early Career Development (CAREER) Award from the National Science Foundation, and a research award from the 1996 AIA/*Architecture* Awards Program. Her research interests include image synthesis, interactive simulation, and computer-aided lighting and acoustical design.



Holly Rushmeier received the BS, MS, and PhD degrees in mechanical engineering from Cornell University in 1977, 1986, and 1988, respectively. She is a research staff member at the IBM T.J. Watson Research Center. Since receiving the PhD, she has held positions at the Georgia Institute of Technology, and at the National Institute of Standards and Technology. In 1990, she was selected as a Presidential Young Investigator. In 1996, she served as the papers chair for the ACM SIGGRAPH conference. Her research interests include data visualization and synthetic image generation.