A Framework for Global Illumination in Animated Environments[1]

Jeffry Nimeroff
Department of Computer and Information Science
University of Pennsylvania

Julie Dorsey
Department of Architecture
Massachusetts Institute of Technology

Holly Rushmeier
Computing and Applied Mathematics Laboratory
National Institute of Standards and Technology

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 from our implementation demonstrate the efficient generation of multiple tours through a complex space, and a tour of an environment in which objects move.

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 [3] and in computer graphics [7]. 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

[1] Presented at the Sixth Eurographics Workshop on Rendering, June 1995, Dublin Ireland

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 on 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 [6, 23]. 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 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 of different approaches have been developed for attempting to build such a system.

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 non-physical 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 non-physical lighting models used and the limitations on numbers of polygons that can be used to model the environment.

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 [13, 20]. 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, walkthroughs 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 (BRDF's) that occur in real life. In a method developed by Sillion et al. [19], 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 pre-computation time, storage requirements, and time to traverse the complex scene.

Pre-Recorded Animations. To date, the most realistic animations are created by using algorithms which are capable of taking diffuse interreflection and nondiffuse reflection into account [22]. 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.

Range-Image Interpolation. Range image interpolation has been employed in flight simulators [3], and has been applied to more general graphics applications by Chen and Williams [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.

Object Space. The radiosity method, described in the previous subsection, is an object space method. 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. [21].

Another major drawback of the original radiosity method was that although it allowed walkthroughs of static environments, a costly new radiosity solution was required for each frame if any object moved. Considerable effort has been put in to develop radiosity methods that exploit temporal coherence in lighting. Baum et al. [2] 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 [5], George et al. [11], Müller and Schöffel [16] and Forsyth et al. [10] have developed progressive refinement radiosity [8] 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.

Image Space. Monte Carlo path tracing (MCPT) [15] 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. 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.

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 multi-pass progressive refinement methods [6] [18].

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 [1]. 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 multi-pass 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 is 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.

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 [14]. *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 [4]. 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 in to 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, 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 non-diffuse illumination. Radiances do not need to be stored for every object for every time.

Direct Illumination and Visibility. Relatively standard techniques for ray tracing animations 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.

Indirect Illumination. As noted 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.

Figure 1 illustrates the relationship between sampling indirect illumination in time and in space. Consider Figure 1a. 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.

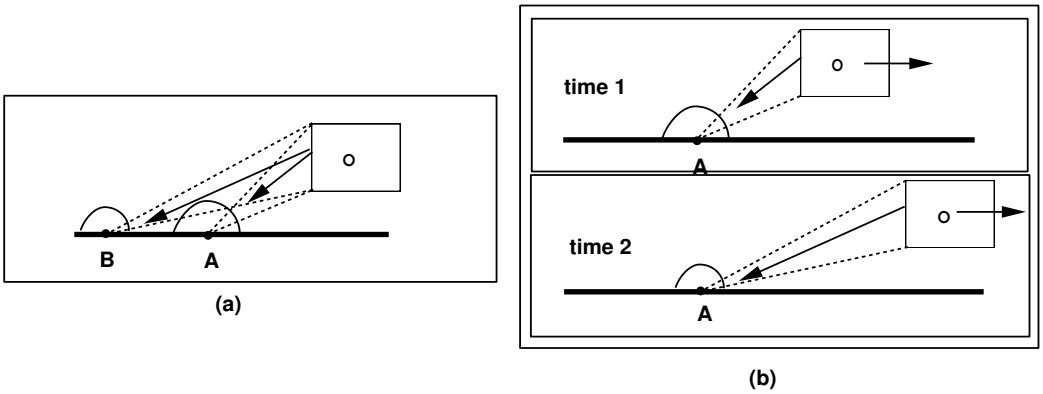

Fig. 1. Figure (a) illustrates that the indirect illumination of point A by object O is only slightly different than the indirect illumination of point B by object O. Figure (b) illustrates that the difference in indirect illumination of point A caused by the movement of object O from time 1 to time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

To build the system diagrammed in Figure 2 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

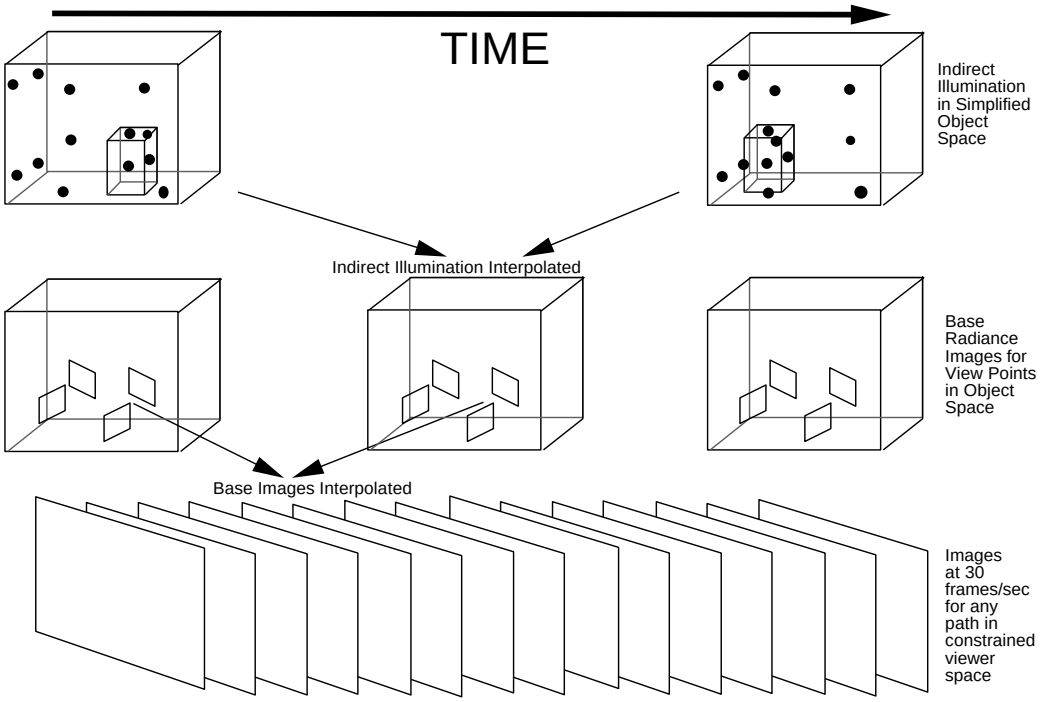

Fig. 2. Indirect illumination is sampled sparsely in time and space in a simplified, animated object space. Radiance images are computed more frequently in time for selected base views, using interpolated values from the indirect illumination solution. One image per frame is computed for any path in the view space by interpolating base images.

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

As input to our initial implementation, 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 indirect illumination solutions are calculated. The indirect illumination solutions are then used to compute the base images. We 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 that is some subset of the full environment. The view space may be 1-D, 2-D, or 3-D, allowing the user to move along a line, within a plane or through a volume respectively. In our initial implementation we consider a 2-D space, although the methodology can easily be applied to the 1-D or 3-D cases.

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 Figure 3, left). In our initial implementation, however, we consider only a hemispherical fish-eye view (see Figure 3 right), restricting the directions and fields of view available to the viewer. 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.")

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, 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 (scan-line, 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.

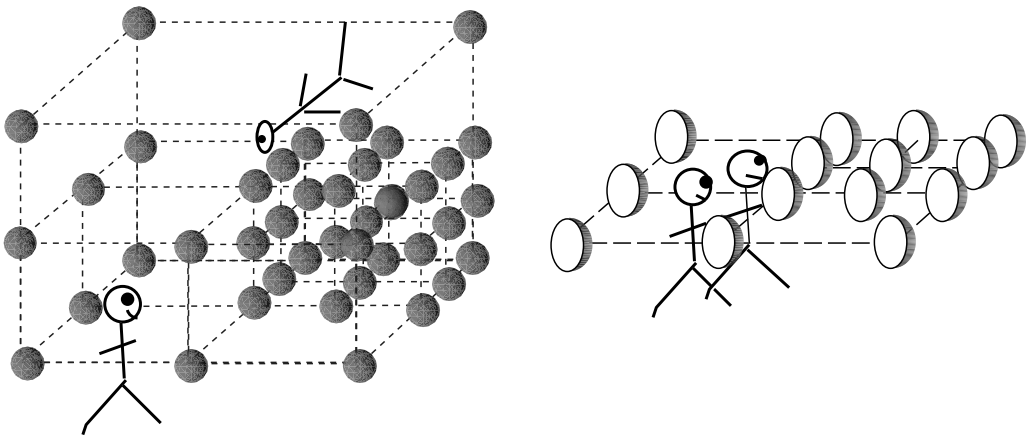

Fig. 3. A viewer has maximum freedom in a 3-D view space with full views stored (left). In the initial implementation a 2-D view space with fish-eye views is used (right).

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 the camera using a fisheye projection. Initially, this is performed for the beginning and endpoints of the animation, and a comparison of the result 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 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 hierarchical radiosity solver [14] on a simplified environment [18]. 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 [22]. 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. In this initial implementation, we use a simple criterion based on visibility from the view constraint space. For each of some number of trials, a random time is chosen. For the environment at that time, random viewpoints in the constraint 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.

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 usually used in $Radiance$.

Once all the base images are computed for the selected locations and times, animations for arbitrary paths through the view space can be generated by interpolating between the images closest in time and space. The view interpolation, which is essentially the image morphing described by Chen and Williams, is performed with the $pinterp$ function, which comes with the $Radiance$ distribution. Time interpolation is performed using the $pcomb$ function.

5 Results

To demonstrate the effectiveness of the new framework, we have applied it to two different environments – a geometrically complex architectural space and a simple "Cornell box" with two moving objects. The complex architectural environment consisted of approximately 75,000 surfaces. These surfaces included a full range of reflectance properties.

Table 1 summarizes execution times for processing the a small view space in the studio environment which we simplified to 5000 surfaces. A quality parameter of q of .7 was used and all times are for a single R4400 processor. Examples of the resulting frames are shown in Figure 8. The total time for processing the environment and generating one 30 frame animation was 3664 minutes. The power of the new framework however is the *small marginal cost to compute additional sequences.* For example, to produce two addtional walkthroughs required only 60 cpu minutes.

Building			
Algorithm	Time (minutes)	Number of Executions	Total Time (minutes)
Visibility Coherence	23	1	23
Direct Coherence	0	0	0
Indirect Illumination	220	1	221
Base Image Generation	420	8	3360
Image Interpolation	2	30	60
Time Interpolation	0	0	0
Total			3664

Table 1. Execution times for the building environment.

To demonstrate the application to an animated environment, we used a simple "Cornell box" that contained two moving cubes. The simple environment highlights the variation of indirect illumination with object movement. The two cubes are are red and blue, and have rough specular surfaces. The cubes move diagonally through a red, white and green enclosure.

Table 2 summarizes the execution times for each stage in the processing of this environment and computing a 30 frame walkthrough. For this animation, we used a quality q of .95, which is very high. The total time required was 2363.5 minutes.

For an animated environment, the additional cost per frame is higher than the static environment, since the time interpolation of base images is required. The walkthrough sequence demonstrates that the user can walk through the moving environment and observe changes in the shiny surfaces and the indirect illumination as well as the overall object motion.

Figure 9 shows images from an early, lower quality (the quality q was .7) version of the animation. Figures 9a and 9b show the indirect solutions. Figure 9c shows an intermediate solution in which both illumination and object positions have been interpolated.

Cornell Box			
Algorithm	Time (minutes)	Number of Executions	Total Time (minutes)
Visibility Coherence	1	1	1
Direct Coherence	20	1	20
Indirect Illumination	5	2	10
Base Image Generation	12	$2 \times 25 = 50$	600
Image Interpolation	1.5	$45 \times 25 = 1125$	1687.5
Time Interpolation	1	45	45
Total			2363.5

Table 2. Execution times for the Cornell box environment.

The results for these two environments demonstrate the feasibility and potential of the new framework. They do not by any means constitute a proof of robustness of the current implementation. Additional testing is required to understand the impact of the quality setting q, to determine appropriate initial discretizations in space and time to insure the adaptive procedures converge, and to measure performance for complex, animated environments.

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 error for a given animation. Second, we would like to address the notion of freedom of movement in greater detail. For instance, many tradeoffs are possible between user interactivity and the quality of the animated sequence. Third, 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. Last, there is a potential for real time walk-throughs using specialized hardware for the view and time interpolations.

References

1. J. Arvo. The irradiance jacobian for partially occluded polyhedral sources. *Computer Graphics (SIGGRAPH '94 Proceedings)*, 28(4):343–350, July 1994.
2. D. Baum, J. Wallace, and M. Cohen. The back-buffer algorithm: an extension of the radiosity method to dynamic environments. *The Visual Computer*, 2(5):298–306, 1986.
3. K. Blanton. A new approach for flight simulator visual systems. In *Simulators IV, Proceedings of the SCCS Simulators Conference*, pages 229–233, 1987.
4. K.R. Boff and J.E. Lincoln. *Engineering Data Compendium: Human Perception and Performance, Vol. 1*. Wright-Patterson Air Force Base, 1988.
5. S. E. Chen. Incremental radiosity: An extension of progressive radiosity to an interactive image synthesis system. *Computer Graphics (SIGGRAPH '90 Proceedings)*, 24(4):135–144, August 1990.
6. S. E. Chen, H. Rushmeier, G. Miller, and D. Turner. A progressive multi-pass method for global illumination. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):165–174, July 1991.
7. S. E. Chen and L. Williams. View interpolation for image synthesis. *Computer Graphics (SIGGRAPH '93 Proceedings)*, 27(4):279–288, August 1993.
8. M. Cohen, S. E. Chen, J. Wallace, and D. Greenberg. A progressive refinement approach to fast radiosity image generation. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(3):75–84, August 1988.
9. J. Dorsey, J. Arvo, and D. Greenberg. Interactive design of complex time-dependent lighting. *IEEE Computer Graphics and Applications*, 15(2):26–36, March 1995.
10. D. Forsyth, C. Yang, and K. Teo. Efficient radiosity in dynamic environments. *Fifth Eurographics Workshop on Rendering*, pages 313–324, June 1994.
11. D. George, F. Sillion, and D. Greenberg. Radiosity redistribution for dynamic environments. *IEEE Computer Graphics and Applications*, 10(4):26–34, July 1990.
12. A. Glassner. Spacetime ray tracing for animation. *IEEE Computer Graphics and Applications*, 8(2):60–70, March 1988.

13. C. Goral, K. Torrance, D. Greenberg, and B. Battaile. Modelling the interaction of light between diffuse surfaces. *Computer Graphics (SIGGRAPH '84 Proceedings)*, 18(3):212–22, July 1984.
14. P. Hanrahan, D. Salzman, and L. Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):197–206, July 1991.
15. J. Kajiya. The rendering equation. *Computer Graphics (SIGGRAPH '86 Proceedings)*, 20(4):143–150, August 1986.
16. Stefan Müller and Frank Schöffel. Fast radiosity repropagation for interactive virtual environments using a shadow-form-factor list. In *5th Annual Eurographics Workshop on Rendering*, pages 325–342, June 13–15 1994.
17. Jeffry Nimeroff, Eero Simoncelli, and Julie Dorsey. Efficient re-rendering of naturally illuminated environments. In *5th Annual Eurographics Workshop on Rendering*, pages 359–374, June 13–15 1994.
18. H. Rushmeier, C. Patterson, and A. Veerasamy. Geometric simplification for indirect illumination calculations. In *Proceedings of Graphics Interface '93*. Canadian Information Processing Society, May 1993.
19. F. Sillion, J. Arvo, S. Westin, and D. Greenberg. A global illumination solution for general reflectance distributions. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):187–196, July 1991.
20. F. Sillion and C. Puech. *Radiosity and Global Illumination*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1994.
21. B. Smits, J. Arvo, and D. Salesin. An importance-driven radiosity algorithm. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(4):273–282, July 1992.
22. G. Ward. The radiance lighting simulation and rendering system. *Computer Graphics (SIGGRAPH '94 Proceedings)*, 28(4), July 1994.
23. G. Ward, F. Rubinstein, and R. Clear. A ray tracing solution for diffuse interreflection. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4):85–92, August 1988.