The Irradiance Volume

Gene Greger [*], Peter Shirley [†], Philip M. Hubbard [‡], Donald P. Greenberg
Program of Computer Graphics
580 Rhodes Hall
Cornell University
Ithaca, NY 14853

Abstract

This paper presents a volumetric representation for the global illumination within a space based on the radiometric quantity irradiance. We call this representation the irradiance volume. Although irradiance is traditionally computed only for surfaces, we extend its definition to all points and directions in space. The irradiance volume supports the reconstruction of believable approximations to the illumination in situations that overwhelm traditional global illumination algorithms. A theoretical basis for the irradiance volume is discussed and the methods and issues involved with building the volume are described. The irradiance volume method shows good performance in several practical situations.

keywords: computer graphics, global illumination, realistic image synthesis, light field

1 Introduction

One of the major goals in the field of computer graphics is realistic image synthesis. To this end, illumination methods have evolved from simple local shading models to physically-based *global illumination* algorithms. Local illumination methods consider only the light energy transfer between an emitter and a surface (direct lighting) while global methods account for light energy interactions between all surfaces in an environment, considering both direct and indirect lighting. Even though the realistic effects that global illumination algorithms provide are frequently desirable, the computational expense of these methods is often too great for many applications. Dynamic environments and scenes containing a very large number of surfaces often pose problems for many global illumination methods.

A different approach to calculating the global illumination of objects is presented in this paper. Instead of striving for accuracy at the expense of performance, the goal is rephrased to be a *reasonable approximation with high performance*. This places global illumination effects within reach of many applications in which visual appearance is more important than absolute numerical accuracy.

Consider a set of surfaces enclosing a space (e.g., the walls of a room). The visual appearance of a diffuse surface within the space can be computed from the reflectance of the surface and the radiometric quantity *irradiance* at the surface. As Section 2 will explain, calculating the irradiance analytically requires a cosine-weighted integral of another radiometric quantity, *radiance*, for all incoming directions. For complex environments, this is a resource intensive computation that overwhelms many applications.

We extend the concept of irradiance from surfaces into the enclosure. This extension is called the *irradiance volume*. The irradiance volume represents a volumetric approximation of the irradiance function. The volume is built in an environment as a pre-process, and can then be used by an application to quickly

[*] Current address: STEP Tools Incorporated, 1223 Peoples Avenue, Troy, NY 12180; gene@steptools.com, http://www.graphics.cornell.edu/~gene/.

[†] Current address: Department of Computer Science, 3190 Merrill Engineering Building, University of Utah, Salt Lake City, UT 84112; http://www.cs.utah.edu/~shirley/.

[‡] Current address: Department of Computer Science, Campus Box 1045, Washington University, One Brookings Drive, St. Louis, MO 63130-4899; http://www.cs.wustl.edu/~pmh/.

approximate the irradiance at locations within the environment. The irradiance volume is similar in spirit to the *illumination solid* of Cuttle [1], although he uses photometric rather than radiometric quantities.

By considering light transport through space instead of between surfaces, our method yields high performance in several important situations for which traditional global-illumination algorithms are too slow. These include "semi-dynamic" environments and environments with a very large number of small geometric features. Due to the ease with which the irradiance volume can be represented in a data structure, querying the irradiance takes approximately constant time independent of the complexity of the environment in which the irradiance volume is built.

We expect our method to be used primarily as a replacement for the "ambient" term in current lighting models. The ambient term is really a global constant approximation to irradiance due to indirect lighting. The irradiance volume can provide a local approximation to indirect irradiance that is potentially much more visually compelling than the traditional ambient term.

Our approach has many similarities to the irradiance caching of Ward [2]. However, irradiance caching at surfaces only works for smooth static surfaces, while our method caches irradiances within a volume and is intended for environments where surface locations can be dynamic, or for environments with very rapid surface irregularities such as displacement mapped surfaces.

Although using the irradiance volume to approximate indirect irradiance implies that direct lighting will be excluded from the computations, we present the method in the context of providing both direct and indirect lighting to simplify the discussion.

2 Radiance, Irradiance and the Irradiance Volume

The fundamental radiometric quantity is *radiance*, which describes the density of light energy flowing through a given point in a given direction. This quantity determines the luminance (a directional intensive quantity analogous to the intuitive quantity "brightness") of a point when viewed from a certain direction. This concept is shown in Figure 1, where a point $\mathbf{x}$ is shown in the center of a room with four constant-colored walls. At all points on the left wall, and for all outgoing directions, the radiance is 4.0. At point $\mathbf{x}$, the radiance is thus 4.0 for all directions that come from the left wall. The radiances at $\mathbf{x}$ for many directions are shown in Figure 1(b). Note that the radiances are shown with arrows pointing in the direction from which the light comes; although this convention seems somewhat backwards, it will prove convenient when dealing with irradiance.

The radiance in all directions at $\mathbf{x}$ makes a directional function called the *radiance distribution function* [3], which is shown as a radial plot in Figure 1(c). This figure emphasizes the well-known fact that the radiance distribution function is not necessarily continuous, even in very simple environments. Since there is a radiance distribution function at every point in space, radiance is a five-dimensional quantity defined over all points and all directions[1].

The radiance of a surface is determined entirely by its reflective properties and the radiance incident upon it. The incident radiance defined on the hemisphere of incoming directions is called the *field-radiance function* [3]. The field radiance for a point on a very small surface is shown as a radial function in Figure 2(a). Because the surface is very small, the radiance in the room is not significantly different from the empty room of Figure 1.

If the surface is diffuse then its radiance is conveniently defined in terms of the surface's *irradiance*, H. Irradiance is defined to be

$$H = \int L_f(\omega) \cos\theta \; d\omega, \tag{1}$$

where $L_f(\omega)$ is the field radiance incident from direction ω, and θ is the angle between ω and the surface normal vector. With this definition, the radiance of the diffuse surface, $L_{surface}$, is

$$L_{surface} = \frac{\rho H}{\pi}, \tag{2}$$

where ρ is the reflectance of the surface. This radiance is constant for all outgoing directions.

[1] Three dimensions are spatial, and two are directional, analogous to latitude and longitude over the spherical set of directions.

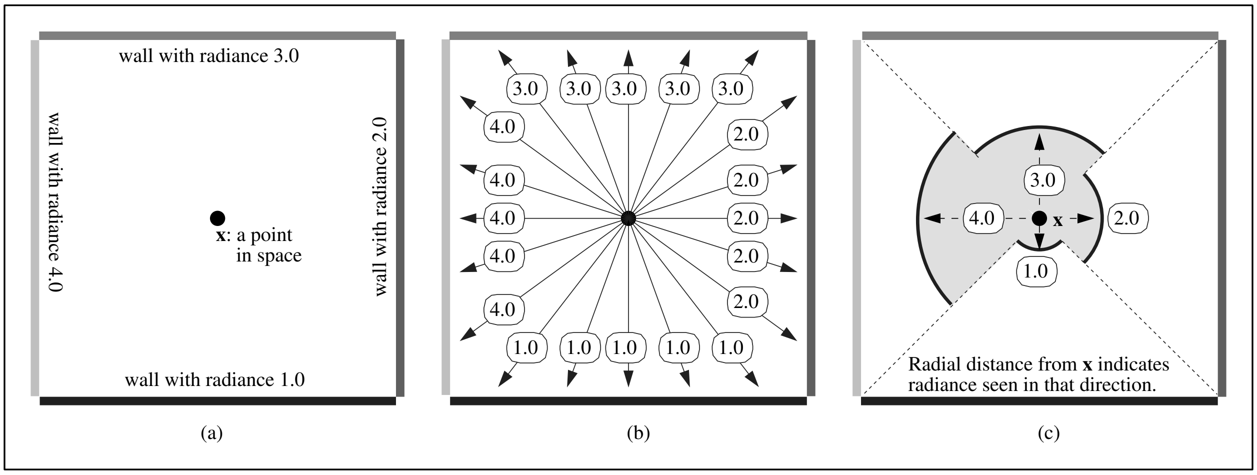

Figure 1: *Building a radial plot of radiance as seen from a point in space.*

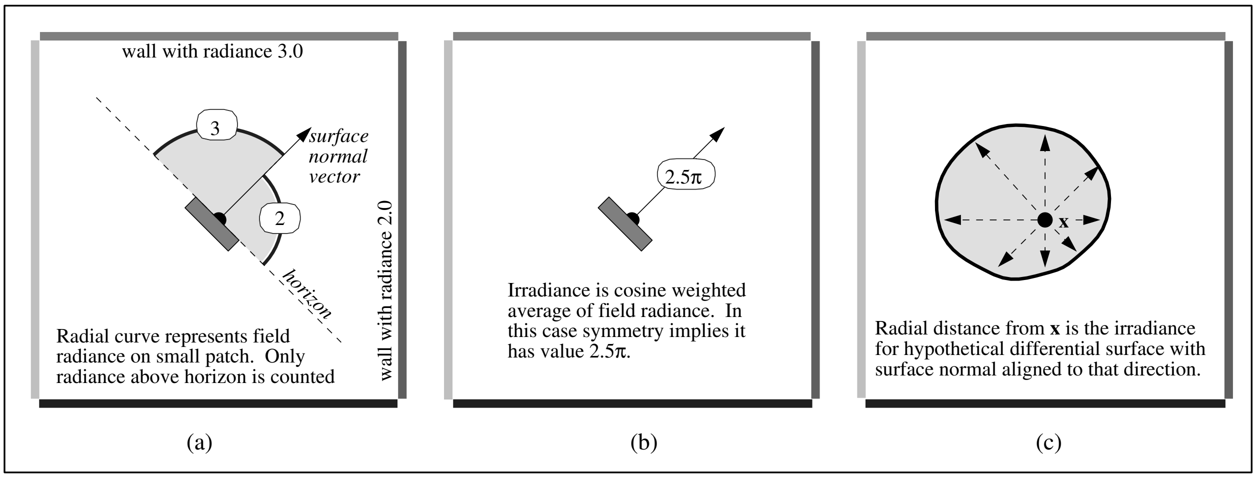

Figure 2: *Building a radial plot of irradiance for irradiance for all potential surfaces at a point in space.*

If we rotate the small patch shown in Figure 2(b) by a small amount counter-clockwise, then the field-radiance will become slightly larger on average. Thus, the irradiance will increase slightly, but will not change radically because of the smoothness of the averaging process in Equation 1. If we compute the irradiance for all possible orientations of the small patch, then we can plot these irradiances as a radial function, as shown in Figure 2(c). We call this the *irradiance distribution function* at the point. Note that this function is continuous over direction, even though the underlying field radiance is discontinuous.

An irradiance distribution function can be computed at every point in space, yielding a five-dimensional function. This function represents the irradiance of a hypothetical differential surface with any orientation within a space. We represent this five-dimensional irradiance function with the notation $H(\mathbf{x}, \omega)$. $\mathbf{x}$ corresponds to the three spatial dimensions, and ω to the two directional dimensions.

Evaluating $H(\mathbf{x}, \omega)$ efficiently is the key to the fast rendering of diffuse surfaces. Since evaluating $H(\mathbf{x}, \omega)$ would require explicit visibility and radiometric calculations for all surfaces in the environment, explicit evaluation of $H(\mathbf{x}, \omega)$ is not a practical option for complex environments. For high efficiency, we must approximate $H(\mathbf{x}, \omega)$.

Suppose we have a volumetric approximation to $H(\mathbf{x}, \omega)$, called $H_v(\mathbf{x}, \omega)$, within the space enclosed by an environment. If we have H_v in a form we can evaluate quickly, we can avoid the costly explicit evaluation of H. This approximation is what we call the *irradiance volume*, and the key assumption is that we can

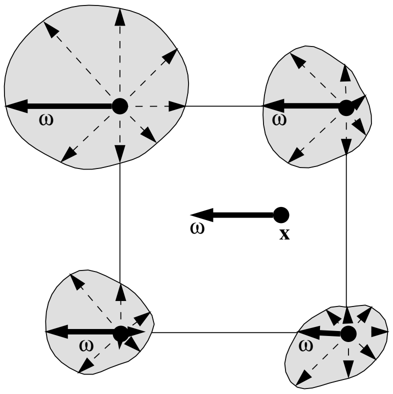

Figure 3: *The irradiance for normal direction ω at point $\mathbf{x}$ is interpolated from the values stored at the vertices of the surrounding cell.*

approximate H accurately enough so that the visual artifacts of using H_v are not severe.

The simplest way to approximate a function is to evaluate it at a finite set of points and use interpolation techniques to evaluate the function between these known points. This approach is practical only if the function is reasonably smooth. In the case of $H(\mathbf{x},\omega)$, $H(\mathbf{x},\cdot)$ (the function of direction while holding position constant) is continuous, while $H(\cdot,\omega)$ is not necessarily continuous. The latter statement should make us hesitate to approximate $H(\mathbf{x},\omega)$ using interpolation. However, we gain confidence by the fact that researchers have had great success interpolating irradiance on surfaces even though discontinuities exist at some surface locations.

In practice, irradiance on surfaces is almost continuous except at transition zones between umbra and penumbra, and transition zones between penumbra and unshadowed regions [3]. Since the relatively smooth regions between these visibility events are large in practice, approximating the spatial variation of irradiance using interpolation is practical for most surface locations, as has been shown in many working radiosity systems. The same facts are true in the volume, as shown by Lischinski [4] who computed discontinuity regions in space and projected them onto surfaces.

Thus, we can store the approximation to $H(\mathbf{x},\omega)$ as point samples, and use interpolation methods to approximate the irradiance at locations between those samples. We then use this reconstructed irradiance to compute the radiance of a surface. Figure 3 illustrates this process in two dimensions. The crucial issue of where to put sample points and the methods used to obtain an approximation of the irradiance distribution function at these points are discussed in Section 3.

3 Implementing the Irradiance Volume

3.1 Sampling Strategy

To build the irradiance volume we need to be able to sample the radiance at any point in any direction. For environments with an explicit geometric representation, such as one composed of polygons, acquiring the radiance can be easily done with ray-casting techniques provided a global-illumination solution exists. An irradiance volume can also be built in environments which do not contain explicit geometry, such as ones used by picture-based methods [5]. The type of source environment does not matter, provided that radiance can be freely point-sampled.

In deciding which points and directions are sampled, and how to build these samples into a data structure, we gain some inspiration from previous work on building radiometric data structures. Patmore [6] and Greene [7] utilized regular grids to represent their radiometric volumes. Since they also were approximating quantities much less well-behaved than irradiance (i.e., visibility or radiance), we should expect similar data structures to work at least as well for irradiance.

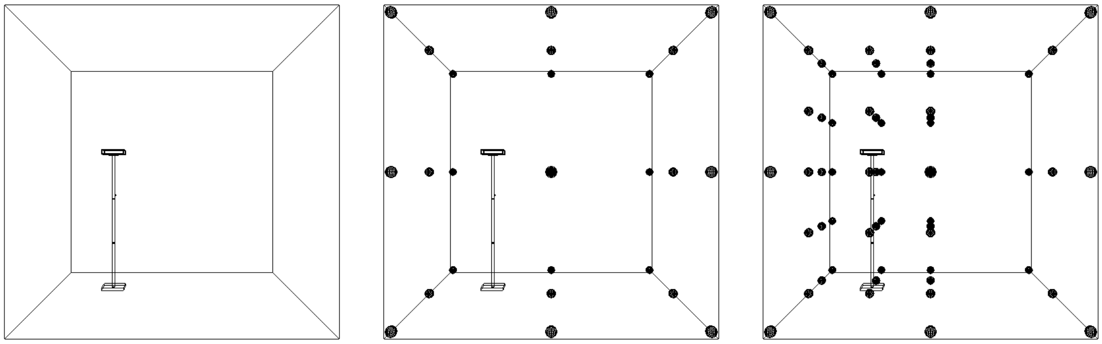

Figure 4: *A bilevel grid being built in a room with a lamp. The center panel shows the first-level grid. In the right panel, second-level grids have been built in the the cells containing the lamp. Each sphere represents a spatial sample location.*

An immediate question is whether to build the irradiance volume with uniformly spaced samples or adaptive samples. Since the irradiance distribution function at a point is always continuous, we uniformly sample in the directional dimensions. For the spatial dimensions, we chose a bilevel grid approach for adaptive sampling. We do this in preference to a deep hierarchy because our primary concern is speed rather than extreme accuracy. The grid is computed as follows:

```
1. Subdivide the bounding box of the scene into a regular grid.

2. For any grid cells that contain geometry, subdivide into a finer grid.

3. At each vertex in the grid, calculate the irradiance distribution function.
```

Subdividing within cells containing geometry alleviates the most likely source of serious error in irradiance interpolation, since most discontinuities are caused by interruptions of light flow by surfaces. Using a bilevel grid also allows low density sampling in open areas, while providing a higher sampling resolution in regions with geometry. Ultimately, a finite-element-style mesh based on volume discontinuities is needed to eliminate interpolation errors completely, but good results can be attained in practice with surprisingly coarse subdivisions.

Figure 4 shows an example of a bilevel grid. First, a $3\times3\times3$ first-level regular grid is built within the room. Next, every grid cell containing geometry is subdivided into a second-level $3\times3\times3$ cell. Note that the regular grid has been placed slightly within the outer walls of the environment so that the grid vertices at the edges of the volume do not intersect pieces of geometry.

3.2 Approximating the Irradiance Distribution Function At a Point

The irradiance distribution function at a point is approximated using the following procedure:

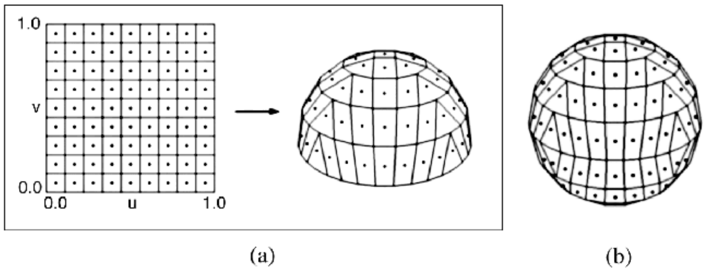

Figure 5: *(a) shows the hemispherical mapping of a grid. (b) shows a sphere composed from two hemispheres.*

```
1. Given a directional sampling resolution, choose a set of directions over the
   sphere of directions in which to sample radiance.

2. In each direction chosen, determine and store the radiance.

3. Using the stored radiance, compute the irradiance distribution function.
```

These steps are performed at each grid vertex to form the irradiance volume. Each step is discussed in detail in the following sections.

3.2.1 Computing Sampling Directions

When gathering the radiance at a point $\mathbf{x}$, we point sample the radiance in a set of directions over the spherical set of directions $S^2(\theta, \phi)$. This results in a piecewise constant approximation of the radiance distribution, each sample defining a region of constant radiance over S^2.

Given that we have chosen to sample uniformly over the two directional dimensions, we would like to pick a good method for selecting a set of directions for a specified sampling resolution. In addition to covering S^2 as uniformly as possible, we would like the regions surrounding the sample points to have equal area in order to simplify the computation of the irradiance distribution function later on.

Shirley *et al.* [8] provide a method of mapping a unit square to a hemisphere while preserving the relative area of regions on the square. By using this procedure, we are afforded the mathematical convenience of being able to uniformly subdivide to a given resolution on a square. This mapping procedure expresses better behavior than latitude-longitude mapping because the mapped areas have better aspect ratios, reducing distortion.

Figure 5(a) shows a 9×9 grid on a unit square and its hemispherical mapping. The dot in the center of each grid square represents a direction in which the radiance will be obtained from a point at the center of the sphere. Each grid cell shows the size of the region that the sample represents. The mapping is performed twice to form the upper and lower hemispheres of the sphere shown in Figure 5(b). As the resolution of the sampling grid increases, the approximation of the radiance distribution becomes more accurate.

3.2.2 Sampling Radiance

Given a point in an environment and a direction, the radiance, $L(\mathbf{x}, \omega)$, is computed by casting a ray from $\mathbf{x}$ in the direction ω and determining the first surface hit. If that surface is diffuse, the radiance is queried from that surface. Otherwise, the ray is specularly reflected off the surface and the next intersection is found. This process is continued until the ray intersects a diffuse surface or meets some other stopping criteria, such as the attenuation of the ray being below a specified tolerance. If a diffuse surface is ultimately hit, the remaining specular coefficient after attenuation from intersections with specular surfaces is multiplied with

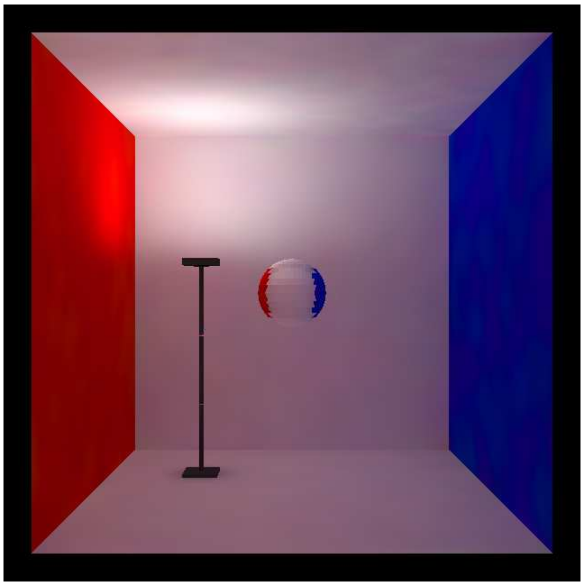

Figure 6: *A radiance sphere, representing radiance samples taken from the center of the room. Note that the front wall of the room is not shown for display purposes.*

the diffuse color to give the radiance; otherwise, a constant quantity analogous to an ambient component is returned. To reduce the number of ray-object intersection tests performed when ray-casting, a uniform spatial-subdivision grid is computed in the environment before the volume is built.

At each grid vertex in the volume, the radiance is sampled in a pre-computed set of directions, $\{\omega_1, \ldots, \omega_n\}$, obtained by using the method discussed in the previous section. Each radiance value is stored in an array and is used to compute the irradiance distribution at that point once the sampling is finished.

Figure 6 shows a diffuse environment. Radiance samples were taken from a point in the center of the room and the results displayed as a *radiance sphere*. The radiance sphere is a useful representation of directional radiance data at a point. It can be thought of as a magnification of an infinitesimal sphere centered at the point where radiance samples were taken from. The color of a point on the surface of the sphere corresponds to the radiance seen in that direction from the center. Each polygon comprising the sphere corresponds to the region, or "bin", of constant radiance defined by a radiance sample taken through its center.

The number of radiance samples needed to ensure a good approximation of the irradiance distribution function at a point depends heavily on the environment. A scene with many objects may require a higher sampling rate so that "important" features (such as bright surfaces) aren't missed.

3.2.3 Computing Irradiance

After the radiance values are gathered at a point, they are used to compute the irradiance for each bin on the sphere. For convenience, we compute the irradiance at the same resolution and in the same directions as the radiance. Since irradiance is a smoother function than radiance, we can assume that a higher sampling rate is not needed. The optimal rate for sampling is an open question; in practice, fewer irradiance samples may actually be needed.

The irradiance in a given direction is computed for a hypothetical surface whose normal vector points in that direction. Given a directional set of radiance samples $\{\omega_1, \ldots, \omega_n\}$, taken at a point, $\mathbf{x}$, in space, the irradiance in direction ω is calculated using the following formula:

$$H(\omega) \approx \sum_{i=1}^{N} L(\omega_i) \max(0, \omega_i \cdot \omega) \Delta\omega_i, \tag{3}$$

where L is the radiance seen in direction ω_i from point $\mathbf{x}$, $\Delta\omega_i$ is the solid angle of the bin associated with

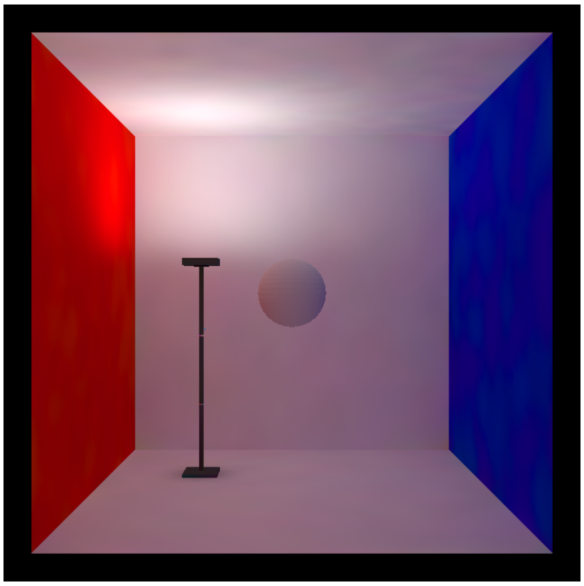

Figure 7: *The irradiance distribution at a point in the center of the room, displayed as an irradiance sphere.*

ω_i, and N is the number of bins on the sphere. This formula is a different form of Equation 1; instead of analytically integrating the radiance over a hemisphere to find the irradiance, simple quadrature is used instead to produce an approximation.

If the sphere of directions is subdivided into regions of equal area, then $\Delta\omega_i \approx \frac{4\pi}{N}$ [2] and Equation 3 becomes

$$H(\omega) \approx \frac{4\pi}{N} \sum_{i=1}^{N} L(\omega_i) \max(0, \omega_i \cdot \omega). \tag{4}$$

This equation is evaluated for a number of directions at $\mathbf{x}$ to obtain a set of directional irradiance samples. For each direction, ω, the irradiance is computed by summing the cosine-weighted contribution of each radiance sample on the hemisphere oriented in that direction. The approximate irradiances form an approximation of the irradiance distribution function at $\mathbf{x}$.

The irradiance distribution function can be displayed in a manner analogous to the radiance sphere as an *irradiance sphere*. Figure 7 shows an irradiance sphere corresponding to the radiance values shown in figure 6.

Figure 8 presents a completed irradiance volume. An irradiance sphere is displayed at each grid vertex, representing the irradiance data stored at that point in the volume.

3.3 Querying the Irradiance Volume

The irradiance volume is represented as three distinct data structures: *samples*, *cells*, and *grids*. Samples contain directional irradiance values corresponding to a particular point in the environment. A cell represents a box[3] in space bounded by eight samples, one at each of its corners. A grid is a three-dimensional array of cells. Each cell may also contain another grid, forming a bilevel structure.

Given a point, $\mathbf{x}$, and a direction, ω, querying the irradiance from the volume requires the following steps:

[2]This formula is only exact if none of the bins which lie on the hemisphere centered around ω_i span the edge of the hemisphere, since only the portions of the bins on the hemisphere should contribute to the irradiance.

[3]The term box is commonly used as shorthand for the three-dimensional analog of the rectangle, which is more precisely known as the "rectangular parallelepiped".

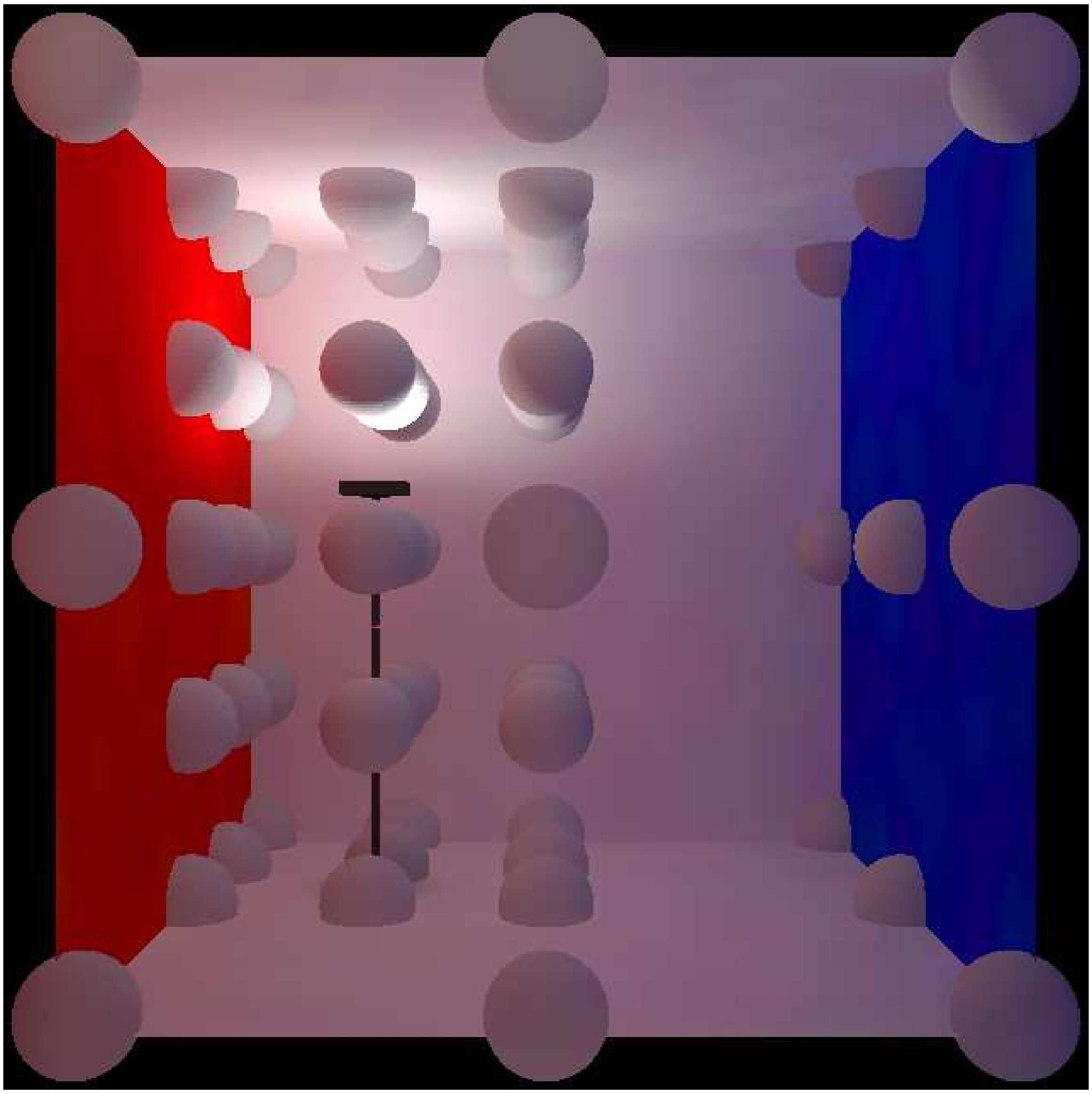

Figure 8: *A completed irradiance volume.*

```
1. Calculate which cell in the first-level grid contains x.

2. If the cell contains a grid, calculate which cell in the second-level grid
   contains x.

3. Find which data value in the cell's samples corresponds to ω.

4. Given the position of x within the cell and the eight values from the surrounding
   samples, interpolate to get the irradiance.
```

Finding which cell contains a point is a simple operation, since the bounding box of the volume is known. Finding the data values corresponding to a direction, ω, involves inverting the hemispherical mapping described earlier. Every spatial sample is stored as a pair of two-dimensional arrays. Each array represents the irradiance computed over a hemisphere, with each array element corresponding to a direction in (u, v) space (Figure 5). A simple check is first performed on ω to determine which hemisphere contains it. Then, an inverse mapping of the method used in Section 3.2.1 is used to find the which bin on the unit square the direction lies in. After the eight values are obtained, trilinear interpolation is used to get the irradiance at a point $\mathbf{x}$.

4 Applications

4.1 Rendering Dynamic Objects in Semi-Dynamic Environments

Our definition of a *semi-dynamic* environment is one in which most surfaces are stationary, and the few dynamic objects are small relative to the scale of the environment. For example, positioning a piece of furniture in an architectural application would fall into this category. An irradiance volume is well suited for semi-dynamic environments and can be used to approximate the global-illumination of dynamic objects. Such a system starts with a preprocessing stage that builds an irradiance volume from the static objects, as described in Section 3. Once the volume is built, it is queried to acquire the illumination of non-static surfaces.

This section presents an example implementation of an interactive semi-dynamic system and discusses several issues related to using the irradiance volume method in this context. Our implementation builds an irradiance volume in a environment rendered from a radiosity solution and allows a user to move an object, illuminated from the volume, around the environment. The model used for this example, an office scene, is pictured in Figure 9. The dynamic object positioned under user control is a grey polygonal model of a rabbit.

Figures 10 and 11 each show a sequence of frames of the rabbit in motion. In Figure 10 the rabbit moves from an open area of the office to a position under the desk, becoming darker as it enters the desk's shadow. Figure 11 shows color-bleeding effects as part of the rabbit takes on a yellow tinge as it approaches a divider.

4.1.1 Performance

Building the Volume

The irradiance volume used for the office model consists of a 7^3 (343) sample first-level grid and 72 $3 \times 3 \times 3$ second-level grids, equaling a total 2287 samples. Each sample has a directional resolution of 2×17^2 (578) angular bins. The volume took approximately 24 minutes[4] to compute after the initial radiosity computation and used about 18 megabytes of main memory for storage. Figure 12 shows the structure of the completed volume.

[4]All timings in this section were made on a Hewlett Packard 9000-715/100 system with a PA-RISC 7100LC processor (100 Mhz, 121 MIPS, SPECfp92 138.3).

Figure 9: *A view of the office model.*

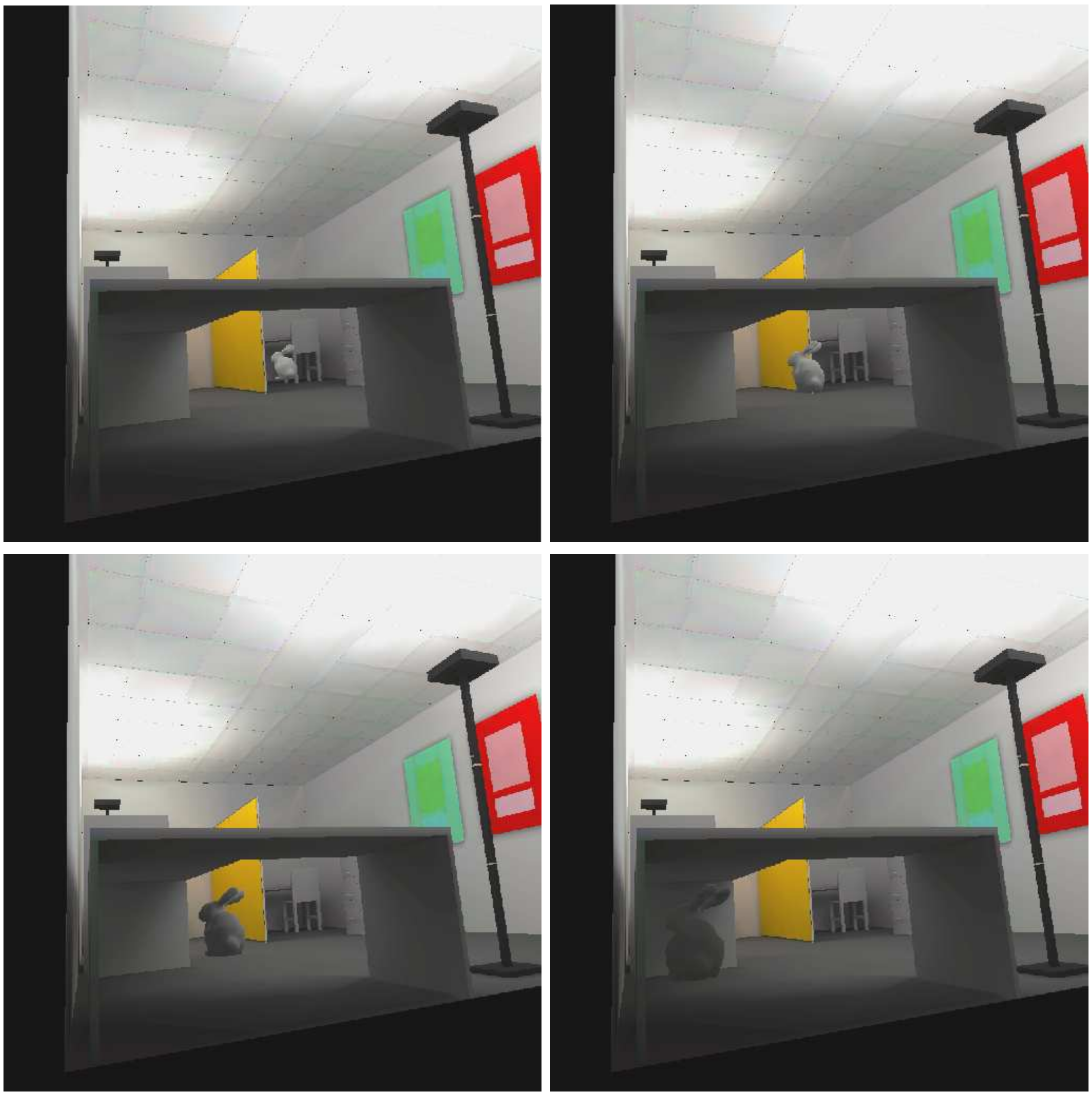

Figure 10: *Rabbit moving under desk.*

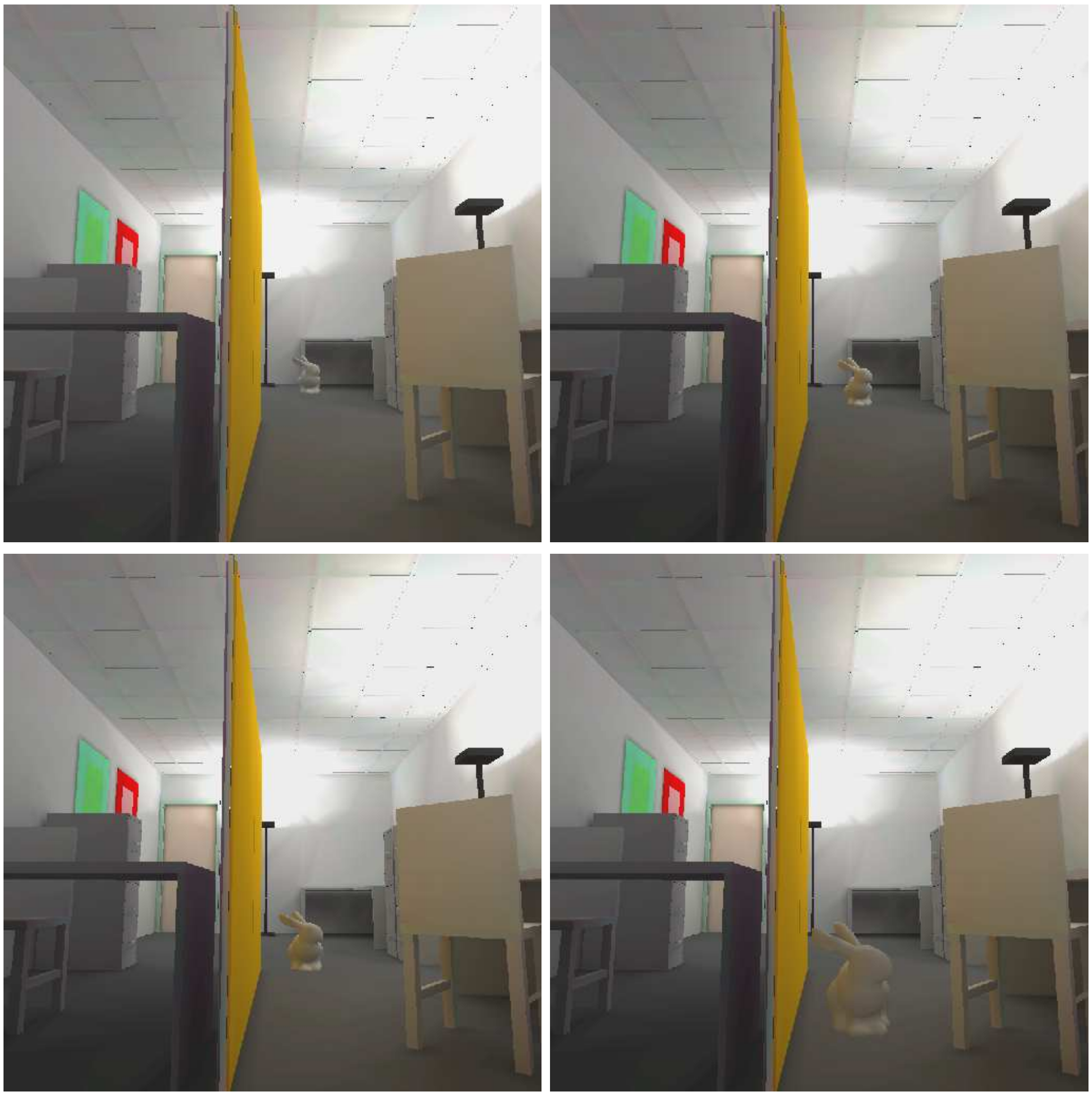

Figure 11: *Rabbit moving by divider.*

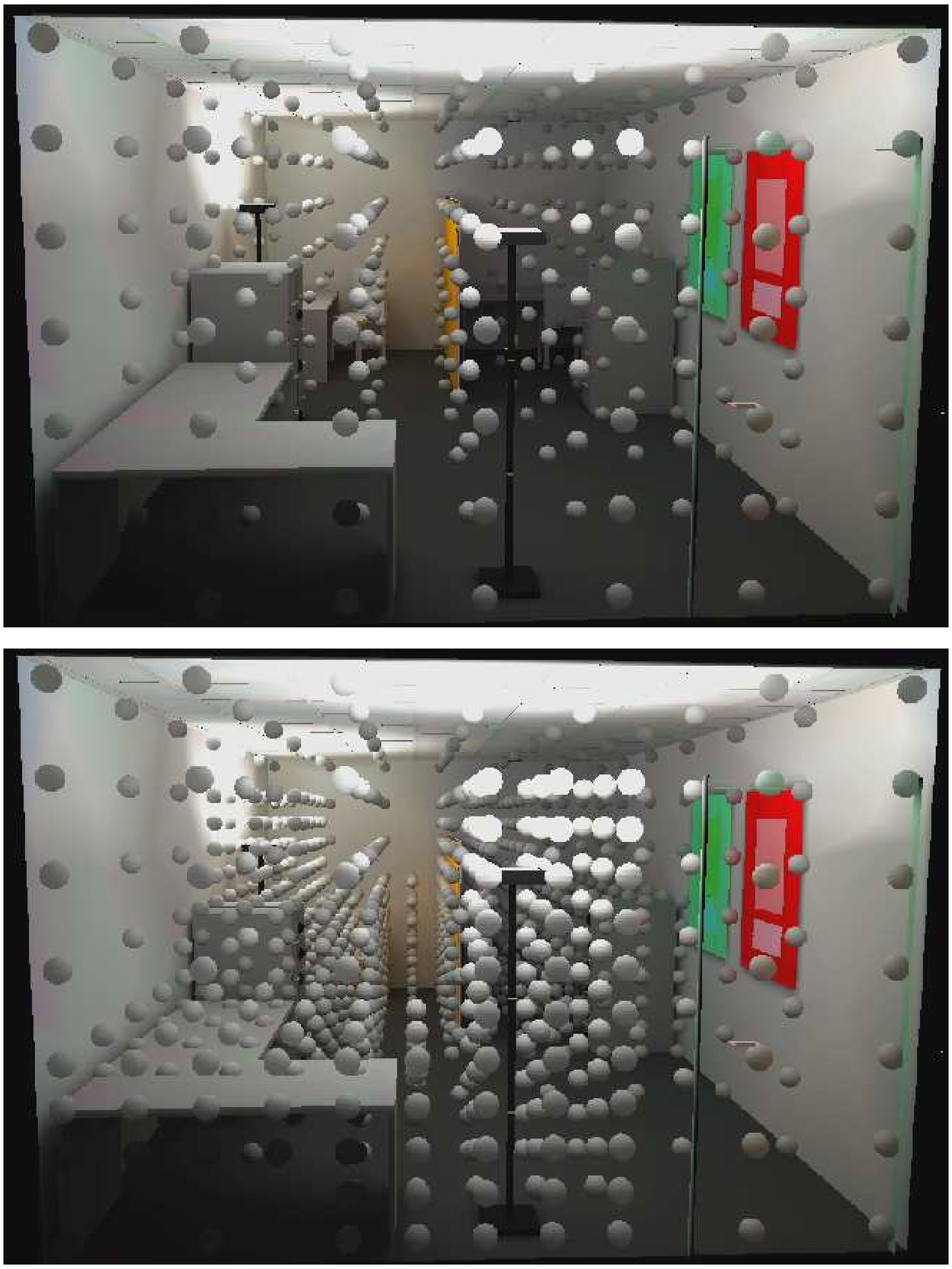

Figure 12: *The top image shows the first-level volume grid, and the bottom image the entire bilevel grid.*

Querying the Volume

Ignoring system factors such as memory paging, querying the volume is a near-constant time operation. Algorithmically, queries will differ at most by a "find which cell contains a point" operation, if the level of hierarchy of the queried cells are not the same (Section 3.3). Finding the correct cell is a relatively inexpensive procedure, resulting in only a small time difference between first- and second-level queries.

Because queries to the volume are well-bounded[5], the use of the volume is well suited to time-critical applications. In addition, the volume supports[6] levels of detail (LOD's) for dynamic objects. While the user is moving an object, the system can draw the lower resolution LOD, and then draw the higher resolution LOD when stationary. The addition of a scheduling algorithm [9], could enable truly time-critical rendering.

Interacting with the Application

In a semi-dynamic application implementation of the kind presented here, three main computational procedures account for the large majority of time needed to compute a display frame. The geometry of the static objects are displayed (in this case the office), the volume is queried to obtain the illumination of the dynamic object or objects (the rabbit), and the dynamic geometry is displayed.

The office model contains 43946 polygons and the rabbit contains 1162 polygons with 601 shared vertices, resulting in 45108 displayed polygons and 601 volume queries per frame. Using high-end graphics hardware[7], we were able to achieve near real-time interactive rates, averaging approximately 5.3 frames per second.

4.1.2 Comparison with a Full Global-Illumination Solution

Several comparisons were made between a rabbit illuminated from the volume and a rabbit rendered using a standard radiosity method. To obtain a global-illumination solution, the polygonal rabbit model was added to the office environment and rendered using radiosity. A visual comparison[8] was then made on images acquired from the two methods (Figure 13). As shown in figure 13, the volume method provides a believable approximation to the analytical solution[9].

4.1.3 Display Issues

Self-Occlusion of Dynamic Objects

Significant inaccuracies can result in the shading of object from the volume if it has large areas of self-occlusion. As discussed in the previous section, for a given direction ω, the volume approximates the irradiance based upon the gathered radiance over the entire hemisphere centered around ω. When the irradiance is queried at a point on an object whose hemispherical "view" of the environment is partially blocked by the object itself, the approximation obtained from the volume becomes less accurate. The amount of potential inaccuracy caused by self-occlusion depends on how much of the hemisphere above a point on the object is subtended by the object itself.

---

[5]The lower bound on the time it takes to illuminate an object from the volume can be expressed as the *number of queries* $\times$ *time it takes to perform first-level query*. Similarly, the upper bound can be described as *number of queries* $\times$ *time it takes to perform a second-level query*.

[6]Some rendering algorithms attempt to take advantage of object coherence between frames, and run into problems if the object changes. The irradiance volume is meant to be used to completely re-shade an object every frame. This means that dynamic objects can be deleted, added, or modified freely.

[7]An Evans and Sutherland Freedom 3000 series with 14 display processors.

[8]There is not a standard metric for quantifying visual differences between images because there is no accurate model of human vision. Direct visual comparison is currently the most practical method to evaluate images.

[9]Note that the polygonal facets comprising the rabbits shown in the figure are much more apparent than in previous images. In the previous figures, smoother shading was obtained by averaging the normals at vertices shared by several polygons. The radiosity method used to make a comparison did not have the capability to interpolate the illumination between polygons. Because of this, the rabbit from the volume was displayed faceted for comparison purposes. Note that the facets tend to maximize visual error; if both rabbits were displayed with interpolated shading, any differences would likely be less noticeable.

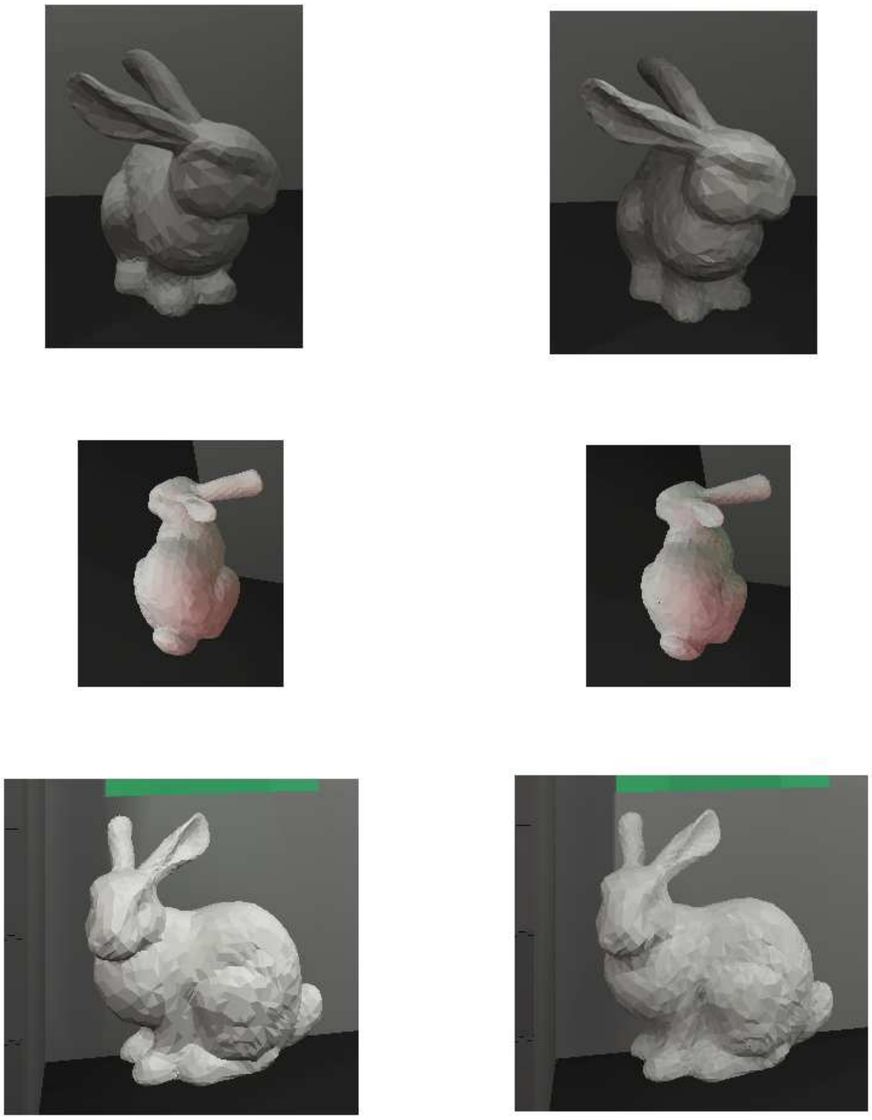

Figure 13: *The images on the left were produced using the volume, and the images on the right were obtained from a radiosity solution. The top row shows the rabbits partly under a desk, the middle row shows them by the wall paintings, and the bottom row shows them in the center of the office.*

Shadows

In our implementation, dynamic objects do not cast any shadows on the environment. Calculating shadows from an object due to indirect illumination is a difficult problem and except for very simple cases, cannot be performed quickly enough for use in interactive applications. Several high-end graphic systems, such as a Silicon Graphics RealityEngine system, support direct illumination in hardware and can compute shadows at real-time rates.

4.2 Rendering Objects with High Complexity

Rushmeier *et al.* [10] discuss the use of *geometric simplification* to accelerate global-illumination renderings of complex environments. In their method, clusters of surfaces are replaced with optically similar boxes before performing the global-illumination calculation. After the global-illumination solution is obtained, it is used in rendering the original geometry. The irradiance volume can be used in an analogous manner. After a global-illumination solution is obtained with simplified objects, an irradiance volume is built[10]. The simplified geometry is then replaced with the original complex surfaces, which are shaded using the volume. This has the advantage over Rushmeier's method by using fast queries from the volume rather than an expensive gather operation requiring tens or hundreds of traced rays.

We replaced one of the flat walls in the model with a polygonal model of a cinder block wall. An irradiance volume was created from a global solution of the room with the flat wall (Figure 9). After the volume was built, each polygon of the cinder block wall was read into memory one at a time and rendered from the volume. The wall consists of approximately 612,000 polygons; not counting disk-access time, it took about 31 seconds to shade the polygons, entailing approximately 19,000 volume queries per second. Figure 14 shows a view of of the wall.

Rendering Procedural Models

Many rendering algorithms produce complex procedural models while executing. The best known example is the software developed by Pixar [11]. In Pixar's software, geometric primitives are processed independently without reference to others, allowing the rendering of large scenes containing numerous highly-detailed complex objects. During rendering, each surface is composed with optional surface maps[11], and diced into "micropolygons" which are then shaded and scan-converted. Because each primitive is rendered separately, only local illumination techniques are used to shade the micropolygons.

An irradiance volume could be used to approximate the global illumination on each micropolygon as it is created. This method would require an initial rough rendering of a scene using a global-illumination algorithm to get the approximate light flow in the environment. An irradiance volume would then be built from this rough rendering and used to shade the micropolygons created during the final rendering.

5 Future Applications

5.1 Illuminating objects in picture-based environments

Picture-based methods use sets of pictures of an environment, either rendered or taken through photographic means, to compose a scene. Currently, picture-based methods are only used to render static environments. However, Chen [5] notes that interactive rendered objects can be composited onto a reconstructed view using layering, alpha-blending, or z-buffer[12] techniques. As these methods become more popular, picture-based applications containing non-static objects will likely increase in number.

Irradiance volume samples are taken at the same locations, or "nodes", as the radiance samples used to form the picture-based environment. Actual views of an environment only exist at these nodes; views

[10]Note that the volume need not encompass the entire environment, but could be built only around a region of interest.

[11]Such as texture, bump, and displacement maps.

[12]If the pictures composing an environment are obtained through rendering, depth information can be recorded for each pixel. This is, of course, impractical for photographically captured images.

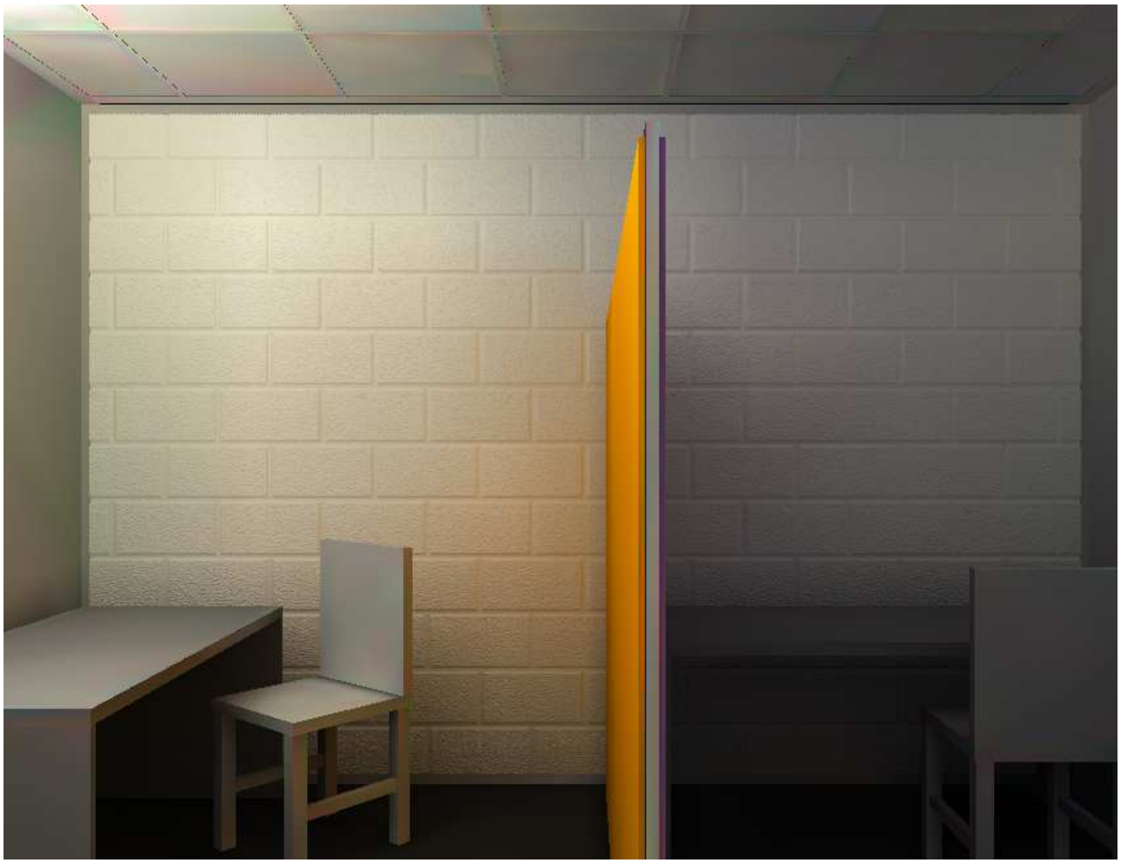

Figure 14: *A polygonal model of a cinderblock wall, illuminated from the volume.*

at locations other than these nodes are either interpolated from the nodes or are ignored[13], depending on the application. If an application does support interpolation between nodes, additional volume samples can be constructed between nodes from an interpolated image panorama. However, using this process only makes sense if the interpolated panoramas result in volume samples that provide better accuracy than simply linearly interpolating between volume samples at the nodes (Figure 15).

The method used to acquire the radiance, $L(\mathbf{x}, \omega)$, is application-dependent, depending on such factors as the way the image data is stored and the shape of the panorama at a node[14]. Once the radiance is sampled, the irradiance is computed using the method described in Section 3.2.3. Querying the irradiance from the volume remains the same.

5.2 Illuminating Non-Diffuse Objects

The irradiance volume can be extended to simulate non-diffuse effects by storing higher order moments than irradiance [3]. This allows the simulation of such phenomena as Phong-like glossy reflections, as long as the specularity is not too high[15].

[13] Some applications constrain the position of the viewer to the node locations.

[14] In order to get a true radiance distribution at a point, data must be gathered over the entire sphere of directions. In practice, cylindrical panoramas are often used instead to avoid problems inherent to spheres, such as the distortion and difficulty of stitching images together at the poles. When sampling radiance from a cylindrical panorama, a default value needs to be assigned to the samples that are taken in directions which point out the top or bottom of the cylinder.

[15] One of the main assumptions in the creation of the irradiance volume is that we can interpolate between samples to get a good approximation of the irradiance. As we move away from irradiance towards increasingly specular functions, this assumption becomes less valid.

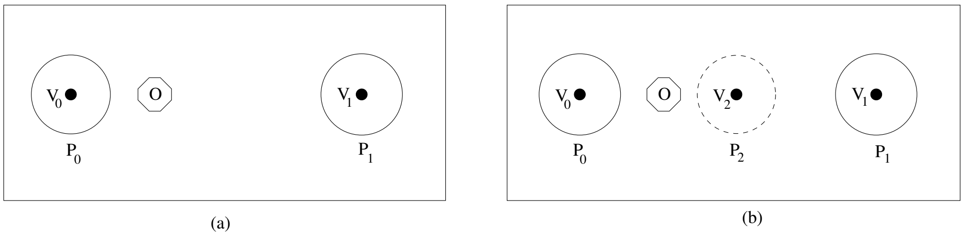

Figure 15: *P_0 and P_1 represent panoramic views samples. P_2 is a sample view interpolated from P_0 and P_1. V_0, V_1, and V_2 are irradiance volume samples computed from their respective panoramas. In (a), the illumination of object O is linearly interpolated from V_0 and V_1. In (b), the illumination is linearly interpolated from V_0 and V_2. If adding V_2 increases the accuracy of the illumination of O, the (application-dependent) node interpolation method can be used to calculate volume samples at non-node locations.*

To build a volume with higher-order moments, we use the following formula[16] instead of Equation 4:

$$H^n(\omega) \approx \left(\frac{n+1}{2}\right) \frac{4\pi}{N} \sum_{i=1}^{N} L(\omega_i) \max(0, \omega_i \cdot \omega)^n, \tag{5}$$

where n is the order of the moment. Note that if $n = 1$, the above equation is equivalent to Equation 4, the formula we use to compute irradiance.

As discussed in Section 3.2.3, the irradiance in a particular direction ω is a cosine-weighted average of radiance samples on the hemisphere oriented around ω. As n increases, it has the effect of weighting the integration of the radiance further towards ω.

When a higher-moment volume is used in an application, changes must be made to the volume query function. Since we are no longer storing and using irradiance to shade objects, the illumination on a surface is no longer view-independent. Queries to the volume must take into account the position of the viewer. Instead of querying in the direction of the normal of the surface, the view vector is mirrored about the normal to obtain the query vector in the direction of specular reflection.

As n increases, the frequency of the illumination function also rises. As it becomes more specular, higher directional and spatial sampling rates are needed to maintain a good approximation. In addition, interpolated shading across a surface (such as Gouraud shading) becomes less accurate. Objects which query the volume to get the illumination at their vertices may need to be composed of finer meshes for good results.

6 Conclusion

This paper presented an exploratory study on the feasibility and effectiveness of sampling and storing irradiance in a spatial volume. Constructing a volumetric approximation of the irradiance throughout a space enabled the approximate irradiance at any point and direction within that space to be quickly queried. This allowed the reconstruction of believable approximations to the illumination in situations which overwhelm traditional global illumination algorithms, such as semi-dynamic environments.

Acknowledgments

Jim Arvo supplied several ideas, including using higher moments than irradiance. Dani Lischinski provided code used to make radiosity solutions. Ian Ashdown provided pointers to valuable references. The rabbit models are courtesy of Stanford University and the University of Washington. This work was supported by the NSF Science and Technology Center for Computer Graphics and Scientific Visualization (ASC-8920219). Much of the work of this project was conducted on workstations generously donated by the Hewlett-Packard Corporation.

[16] The notation H^n is used here to indicate the nth moment of H.

References

[1] C. Cuttle. Lighting patterns and the flow of light. *Lighting Research and Technology*, 18(6), 1971.

[2] Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 85–92, August 1988.

[3] James Arvo. *Analytic Methods for Simulated Light Transport*. PhD thesis, Yale University, December 1995. This text is available from http://csvax.cs.caltech.edu/ arvo/papers.html.

[4] Daniel Lischinski, Filippo Tampieri, and Donald P. Greenberg. Combining hierarchical radiosity and discontinuity meshing. In *Computer Graphics Proceedings, Annual Conference Series, 1993*, pages 199–208, 1993.

[5] Shenchang Eric Chen. Quicktime VR - an image-based approach to virtual environment navigation. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, Computer Graphics Proceedings, Annual Conference Series, pages 29–38, July 1995. ACM Siggraph '95 Conference Proceedings.

[6] Christopher Patmore. Illumination of dense foliage models. In Michael F. Cohen, Claude Puech, and Francois Sillion, editors, *Fourth Eurographics Workshop on Rendering*, pages 63–72. Eurographics, June 1993. held in Paris, France, 14–16 June 1993.

[7] Ned Greene. Efficient approximation of skylight using depth projections. *Photorealistic Volume Modeling and Rendering Techniques*, 1993. ACM Siggraph '91 Course Notes 27.

[8] Peter Shirley and Kenneth Chiu. Notes on adaptive quadrature on the hemisphere. Technical Report 411, Department of Computer Science, Indiana University, July 1994. This text is available from http://www.cs.utah.edu/ shirley/papers.html.

[9] Thomas A. Funkhouser and Carlo H. Sequin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. *Computer Graphics*, pages 247–254, August 1993. ACM Siggraph '93 Conference Proceedings.

[10] Holly Rushmeier, Charles Patterson, and Aravindan Veerasamy. Geometric simplification for indirect illumination calculations. In *Proceedings of Graphics Interface '93*, pages 227–236, Toronto, Ontario, Canada, May 1993. Canadian Information Processing Society.

[11] Robert L. Cook, Loren Carpenter, and Edwin Catmull. The reyes image rendering archtecture. *Computer Graphics*, 21(4):95–102, July 1987. ACM Siggraph '87 Conference Proceedings.