Efficient Re-rendering of Naturally Illuminated Environments*

Jeffry S. Nimeroff[1] *Eero Simoncelli*[1] *Julie Dorsey*[1,2]

[1] Department of Computer and Information Science
[2] Department of Architecture
University of Pennsylvania, Philadelphia PA 19104, USA

Abstract

We present a method for the efficient re-rendering of a scene under a directional illuminant at an arbitrary orientation. We take advantage of the linearity of the rendering operator with respect to illumination for a fixed scene and camera geometry. Re-rendering is accomplished via linear combination of a set of pre-rendered "basis" images. The theory of steerable functions provides the machinery to derive an appropriate set of basis images. We demonstrate the technique on both simple and complex scenes illuminated by an approximation to natural skylight. We show re-rendering simulations under conditions of varying sun position and cloudiness.

1 Introduction

In recent years, researchers have begun to treat the simulation of natural illumination (overcast or clear skylight [15]) with global illumination algorithms [15, 22]. A major difficulty with modeling daylight is that the direction of the sun varies continuously throughout the day. In addition, cloud conditions may also change (e.g., from clear to overcast). This paper addresses the problem of efficient dynamic re-rendering of a scene under such changes.

An example application of the technique is in the rendering of naturally illuminated architectural environments. In order to qualitatively assess a daylit interior, designers need to see subtle lighting effects such as soft shadowing, indirect illumination, and color bleeding. These effects are generally only available through costly global illumination algorithms. For instance, in order to compute an animation of the appearance of an interior space during the course of a typical day, one would need to compute a global solution at each of a large number of time steps corresponding to a sequence of sun directions. Furthermore, varying the sky conditions (e.g. cloudiness) would require an entirely new animation with new global solutions.

Rather than computing a global solution for each time step, we describe a technique for computing a time-independent basis—a small number of global solutions—that

* Presented at the 5th Eurographics Workshop on Rendering, June 1994, Darmstadt Germany

suffice to simulate the entire animation. The theory of steerable functions provides the means to construct the basis images (one per solution) needed for rendering via linear combination.

2 Linearity of Rendering

For our formulation, the rendering operator R takes an illumination description $L(\hat{\mathbf{u}})$ (the illuminant intensity as a function of illumination direction $\hat{\mathbf{u}}$), a scene and camera geometry G, and yields an image I:

$$R(L(\hat{\mathbf{u}}), G) \rightarrow I \quad . \tag{1}$$

Since the geometry G is fixed, an individual rendering operator can be thought of as existing for each chosen geometry. We denote this operator as:

$$R_G(L(\hat{\mathbf{u}})) \rightarrow I \quad . \tag{2}$$

If one is willing to discount physical and quantum optics and work with static scene geometries, the rendering operation is linear with respect to the illumination [11, 8, 4] operator. More specifically, rendering obeys the rules of *superposition*:

- the image resulting from an additive combination of two illuminants is just the *sum* of the images resulting from each of the illuminants independently,
- multiplying the intensity of the illumination sources by a factor α results in a rendered image that is multiplied by the same factor.

What is the advantage of this linearity? Consider an illuminant constructed as the combination of simpler "basis" illumination functions $L_i(\hat{\mathbf{u}})$,

$$L(\hat{\mathbf{u}}) = \sum_{i=1}^{N} \alpha_i L_i(\hat{\mathbf{u}}) \quad . \tag{3}$$

Linearity allows us to write the desired rendered image as a linear combination of images rendered under each of the basis functions:

$$R_G(L(\hat{\mathbf{u}})) = R_G\left(\sum_{i=1}^{N} \alpha_i L_i(\hat{\mathbf{u}})\right) = \sum_{i=1}^{N} \alpha_i R_G(L_i(\hat{\mathbf{u}})) \quad . \tag{4}$$

The $R_G(L_i(\hat{\mathbf{u}}))$ constitute a set of images rendered under each of the basis illuminants $L_i(\hat{\mathbf{u}})$. The equation states that we may compute an image for any linear combination of the basis illuminants via a linear combination of these pre-rendered images.

3 Choosing the Illuminant Basis

In order to make use of this property, we must select an appropriate *basis set* of illumination functions $L_i(\hat{\mathbf{u}})$. Some desirable properties for this set of functions are as follows:

1. The basis functions should be general enough to form any light source we desire (via linear combination).
2. The number of basis functions, N, should be small, since this corresponds to the number of basis images we must actually render.
3. For each illumination function that can be represented, we should also be able to compute (via linear combination) the same function rotated by any three-dimensional rotation. In other words, the (linear) space of illumination functions that we represent should be *rotation-invariant*.

The first two of these properties have conflicting requirements: the smaller the number of basis functions, the less flexibility we have in the variety of representable illuminants. This is true of any sort of function expansion, such as Taylor or Fourier series approximations.

The last property is a more explicit and restricted version of the first property and is not necessary for linear combinations. Nevertheless, it is extremely useful in the case of sunlight illuminants, since the direction of the sun varies along a path on a sphere. This property places a strong constraint on the functions known as "steerability" [9]. A steerable function is one that can be written as a linear combination of rotated versions of itself. Freeman and Adelson [9] have developed a theory of such functions and demonstrated their use in two-dimensional image processing problems. For the purposes of the present paper, we will develop a set of steerable functions in three dimensions and demonstrate their use as illumination basis functions for the linear re-rendering problem.

3.1 Steerable Functions: Directional Cosine Example

We begin by illustrating the idea of steerability using the simplest form of steerable function. Let $c(\hat{\mathbf{u}};\hat{\mathbf{s}})$ be defined as:

$$c(\hat{\mathbf{u}};\hat{\mathbf{s}}) = \hat{\mathbf{u}} \cdot \hat{\mathbf{s}} \ ,$$

where $\hat{\mathbf{s}}$ and $\hat{\mathbf{u}}$ are unit vectors in $\mathbf{R}^3$ and $(\cdot)$ indicates an inner product. We consider $c()$ as a function on the sphere parameterized by the unit vector $\hat{\mathbf{u}}$, with $\hat{\mathbf{s}}$ a fixed direction vector. Clearly, the function computes the cosine of the angle between a point on the sphere and the fixed direction $\hat{\mathbf{s}}$. We therefore refer to this as a *directional cosine* function.

We can expand the equation as:

$$\begin{aligned} c(\hat{\mathbf{u}};\hat{\mathbf{s}}) &= s_x(\hat{\mathbf{u}} \cdot \hat{\mathbf{e}}_x) + s_y(\hat{\mathbf{u}} \cdot \hat{\mathbf{e}}_y) + s_z(\hat{\mathbf{u}} \cdot \hat{\mathbf{e}}_z) \\ &= s_x c(\hat{\mathbf{u}};\hat{\mathbf{e}}_x) + s_y c(\hat{\mathbf{u}};\hat{\mathbf{e}}_y) + s_z c(\hat{\mathbf{u}};\hat{\mathbf{e}}_z) \ , \end{aligned} \tag{5}$$

where the $\hat{\mathbf{e}}_*$ are the unit vectors corresponding to the three Euclidean coordinate axes, and s_* are the components of $\hat{\mathbf{s}}$. We have written the directional cosine function in direction $\hat{\mathbf{s}}$ as *a linear combination of directional cosines* along the three coordinate axes. That is, we can compute *any* rotated version of this function via linear combinations of a *fixed set* of rotated copies. We say that the set of three functions, $c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_\cdot)$, constitute a *steerable basis set*.

Of course, there is nothing particularly special about the coordinate axis directions $\{\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y, \hat{\mathbf{e}}_z\}$. We could choose *any* three non-coplanar directions, $\{\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, \hat{\mathbf{s}}_3\}$, and still perform this computation. The easiest way to see this is by working from (5). We can write $c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_i)$ for each i in terms of the axis cosine functions:

$$c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_i) = s_{i,x} c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_x) + s_{i,y} c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_y) + s_{i,z} c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_z) \quad .$$

This is a set of three linear equations, which we can write in matrix form as:

$$\begin{pmatrix} c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_1) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_2) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_3) \end{pmatrix} = \mathbf{M}_3 \begin{pmatrix} c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_x) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_y) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_z) \end{pmatrix} \quad ,$$

where $\mathbf{M}_3$ is a 3×3 matrix containing the vectors $\hat{\mathbf{s}}_i$ as its rows.

If the three vectors $\hat{\mathbf{s}}_i$ are non-coplanar, they span $\mathbf{R}^3$, and we can invert the matrix $\mathbf{M}_3$ to solve for the axis cosines:

$$\begin{pmatrix} c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_x) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_y) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_z) \end{pmatrix} = \mathbf{M}_3^{-1} \begin{pmatrix} c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_1) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_2) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_3) \end{pmatrix} \quad .$$

Given the axis cosines, we can compute *any* directional cosine function via (5):

$$c(\hat{\mathbf{u}}; \hat{\mathbf{s}}) = \hat{\mathbf{s}}^T \mathbf{M}_3^{-1} \begin{pmatrix} c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_1) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_2) \\ c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_3) \end{pmatrix} \quad .$$

Thus, we have an expression for computing the directional cosine in any direction $\hat{\mathbf{s}}$ from the cosines in three fixed but arbitrary non-coplanar directions, $\hat{\mathbf{s}}_i$.

3.2 Directional Cosine Polynomials

The steerable basis we have derived is not yet suitable as an illuminant basis, since the functions take on both positive and negative values. We can form a simple steerable illuminant basis by simply adding unity to these:

$$\begin{aligned} L(\hat{\mathbf{u}}; \hat{\mathbf{s}}) &= 1 + c(\hat{\mathbf{u}}; \hat{\mathbf{s}}) \\ &= 1 + s_x c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_x) + s_y c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_y) + s_z c(\hat{\mathbf{u}}; \hat{\mathbf{e}}_z) \quad . \end{aligned}$$

The function L is written as a sum of *four* terms. This equation is still not in a steerable form, but we can write it steerably using the same trick as before. We choose four

arbitrary non-coplanar unit vectors $\{\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, \hat{\mathbf{s}}_3, \hat{\mathbf{s}}_4\}$, write four linear equations for L in each of these directions, and invert this to get:

$$L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = \begin{pmatrix} 1 \\ s_x \\ s_y \\ s_z \end{pmatrix}^T \mathbf{M}_4^{-1} \begin{pmatrix} L(\hat{\mathbf{u}};\hat{\mathbf{s}}_1) \\ L(\hat{\mathbf{u}};\hat{\mathbf{s}}_2) \\ L(\hat{\mathbf{u}};\hat{\mathbf{s}}_3) \\ L(\hat{\mathbf{u}};\hat{\mathbf{s}}_4) \end{pmatrix} \quad , \tag{6}$$

where $\mathbf{M}_4$ is now the matrix:

$$\mathbf{M}_4 = \begin{pmatrix} 1 & s_{1,x} & s_{1,y} & s_{1,z} \\ 1 & s_{2,x} & s_{2,y} & s_{2,z} \\ 1 & s_{3,x} & s_{3,y} & s_{3,z} \\ 1 & s_{4,x} & s_{4,y} & s_{4,z} \end{pmatrix} \quad .$$

The directional cosine functions described thus far are quite broad in their extent: the magnitude of each function is significant over a large fraction of the sphere. When used for purposes of scene illumination, these functions may be too diffuse. How can we build narrower illuminants? One simple generalization can be achieved by considering polynomials of directional cosine functions.

Consider a general second order polynomial of the directional cosine function defined in (5):

$$\begin{aligned} f(\hat{\mathbf{u}};\hat{\mathbf{s}}) &= a_0 + a_1(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}}) + a_2(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})^2 \\ &= a_0 \\ &\quad + a_1[s_x u_x + s_y u_y + s_z u_z] \\ &\quad + a_2[s_x^2(u_x^2) + 2s_x s_y(u_x u_y) + 2s_x s_z(u_x u_z) \\ &\qquad + s_y^2(u_y^2) + 2s_y s_z(u_y u_z) + s_z^2(u_z^2)] \quad . \end{aligned} \tag{7}$$

As in the previous section, this equation may be placed in steerable form by considering a set of 9 direction vectors $\{\hat{\mathbf{s}}_i | i \in [1,9]\}$:

$$f(\hat{\mathbf{u}};\hat{\mathbf{s}}) = \begin{pmatrix} 1 \\ s_x \\ s_y \\ s_z \\ s_x^2 \\ s_x s_y \\ s_x s_z \\ s_y^2 \\ s_y s_z \\ s_z^2 \end{pmatrix}^T \mathbf{M}_9^{\#} \begin{pmatrix} f(\hat{\mathbf{u}};\hat{\mathbf{s}}_1) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_2) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_3) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_4) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_5) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_6) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_7) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_8) \\ f(\hat{\mathbf{u}};\hat{\mathbf{s}}_9) \end{pmatrix} \quad ,$$

where the rows of the matrix M_9 contain the combinations of the $\hat{\mathbf{s}}_i$ found in each of the terms of (7). The symbol # indicates a least-squares pseudo-inverse operation. This is used in place of a standard matrix inverse: the matrix M_9 is not of full rank because the functions $f(\hat{\mathbf{u}};\hat{\mathbf{s}}_i)$ do not span the full space of second order polynomials. Nevertheless, the interpolation of (7) is exact.

The function $f(\hat{\mathbf{u}};\hat{\mathbf{s}})$ may be made narrower (through proper choice of the a_k) than the first order polynomial of the previous section. But note that we have paid a price for this: we now require nine basis functions for steerability as opposed to four. This is a fundamental tradeoff in steerable basis sets: *a narrower steerable function requires a larger number of basis functions*. [3]

4 Bases for Natural Illumination

In this section, we discuss the design of illuminant basis sets for approximating sunlight. We consider overcast skylight and clear skylight separately and then discuss general skylight.

4.1 Overcast Skylight

Overcast skylight [6, 15] is the illumination that emanates from the sun but is completely absorbed and re-radiated by the atmosphere (clouds) before illuminating the scene. It is a function that describes an overcast/cloudy day. The function has a cosine falloff relative to a fixed point (zenith) but does not depend on the sun's position in the sky:

$$L(\hat{\mathbf{u}}) = L_z(1 + 2(u_z))/3 \ , \tag{8}$$

where u_z is the z-component of $\hat{\mathbf{u}}$, and L_z is an overall illumination constant. Since the orientation of this function never changes, there is no need to construct a steerable basis for it.[4]

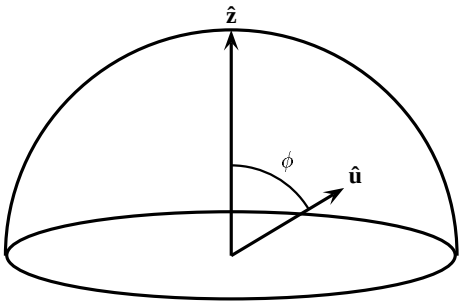

Fig. 1. Overcast skylight geometry

[3] In two dimensions, this is simply a restatement of the Nyquist criterion for sampling of bandlimited signals [20].

[4] We could construct a basis of size 4 using (6).

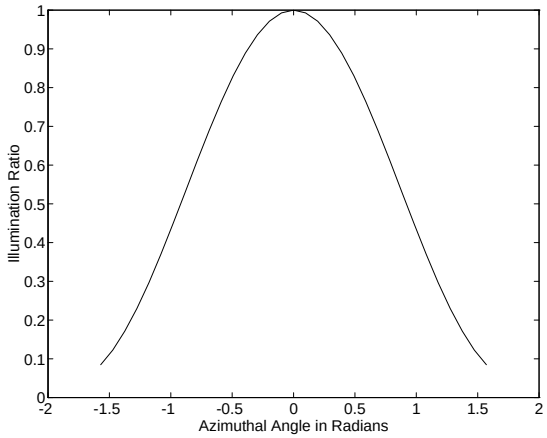

Fig. 2. Overcast skylight distribution

4.2 Clear Skylight

Clear skylight [1, 15] describes the distribution of the sun's energy on a clear day. It is very complex and is dependent on the sun's position in the sky and on the direction relative to a fixed point (zenith):

$$L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = L_z \frac{(0.91 + 10\exp(-3\arccos(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})) + 0.45(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})^2)(1-\exp(-0.32/u_z))}{0.274(0.91 + 10\exp(-3\arccos(s_z)) + 0.45(s_z^2))}\,, \tag{9}$$

where $\hat{\mathbf{s}}$ is the unit vector pointing to the sun.

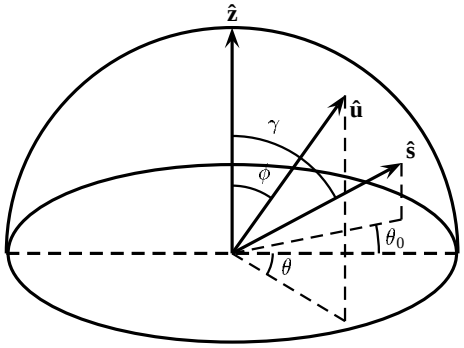

Fig. 3. Clear skylight geometry

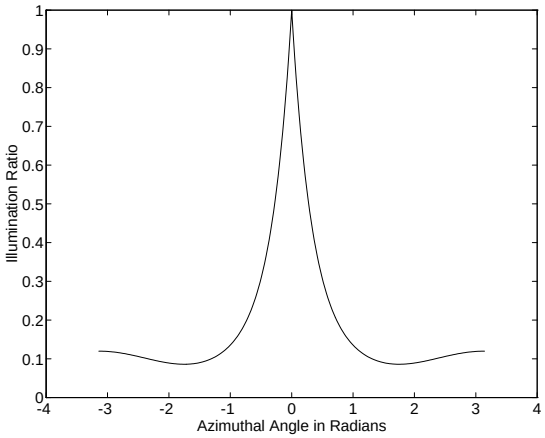

Fig. 4. Clear skylight distribution for sun at zenith

4.3 Steerable Sunlight Approximation

In order to re-render naturally illuminated scenes via linear combinations of rendered images, a steerable approximation to the clear skylight function in (9) must be constructed. We rewrite the equation as:

$$\begin{aligned} L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = & \left[\frac{L_z}{0.274(0.91+10\exp(-3\arccos(s_z))+0.45(s_z^2))}\right] \\ & \cdot \left[1-\exp(-0.32/u_z)\right] \\ & \cdot \left[0.91+10\exp(-3\arccos(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}}))+0.45(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})^2\right] \\ = & \; L_1(\hat{\mathbf{s}})\cdot L_2(\hat{\mathbf{u}})\cdot L_3(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}}) \quad . \end{aligned}$$

In order to compute $L(\hat{\mathbf{u}};\hat{\mathbf{s}})$ from a set of illuminants $L(\hat{\mathbf{u}};\hat{\mathbf{s}}_i)$, note that only L_3 must be steerable. L_2 is a function of $\hat{\mathbf{u}}$ only and does not affect the steerability. L_1 is a function of $\hat{\mathbf{s}}$ only and acts as an illumination scaling constant. Therefore the steerable form of $L(\hat{\mathbf{u}};\hat{\mathbf{s}})$ may be written as:

$$L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = L_1(\hat{\mathbf{s}})\cdot\sum_i(\alpha_i(\hat{\mathbf{s}})\cdot\overbrace{L_2(\hat{\mathbf{u}})\cdot\underbrace{L_3(\hat{\mathbf{s}}_i\cdot\hat{\mathbf{u}})}_{\text{Steerable}}}^{\text{Basis lights}}) \quad . \tag{10}$$

We must construct a steerable approximation to the function L_3. For purposes of this paper, we use a second order approximation:

$$\begin{aligned} L_3(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}}) &= 0.91+10\exp(-3\arccos(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}}))+0.45(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})^2 \\ &\approx 0.5012+1.9116(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})+3.3911(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})^2 \quad . \end{aligned} \tag{11}$$

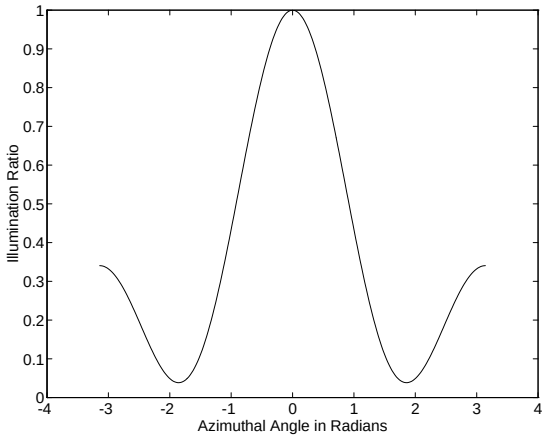

Fig. 5. Approximated clear skylight distribution at zenith

This steerable function requires a basis set of size ten. We note again that higher order polynomials would provide better approximations at the expense of a larger basis set requiring additional executions of the costly global illumination algorithm.

Using the linearity of the rendering operator and (10) we can write an expression for re-rendering an image for any sun direction $\hat{\mathbf{s}}$ from the images rendered under 9 different sun directions s_i:

$$R_G(L(\hat{\mathbf{u}};\hat{\mathbf{s}})) = L_1(\hat{\mathbf{s}}) \cdot \sum_i \alpha_i(\hat{\mathbf{s}}) \frac{R_G(L_2(\hat{\mathbf{u}}) \cdot L_3(\hat{\mathbf{s}}_i \cdot \hat{\mathbf{u}}))}{L_1(\hat{\mathbf{s}}_i)} \quad , \tag{12}$$

where the α_i are the components of:

$$\begin{pmatrix} 1 & s_x & s_y & s_z & s_x^2 & s_x s_y & s_x s_z & s_y^2 & s_y s_z & s_z^2 \end{pmatrix} \mathbf{M}_9^{\#} \quad . \tag{13}$$

4.4 General Skylight

As demonstrated above, a steerable approximation for an overcast day requires a first-order expansion, while the clear day requires a second-order expansion. To generate an image for a partially overcast day, we have to gradually phase in the second order terms. This can be done by linearly interpolating between the two sets of coefficients.

Let b be a number in $[0, 1]$ indicating the brightness of the day (1=sunny, 0=cloudy). Let

$$v_0 = \begin{pmatrix} \beta_0 & 0 & 0 & \beta_1 & 0 & 0 & 0 & 0 & 0 & \beta_2 \end{pmatrix} \quad ,$$

and

$$v_1(\hat{\mathbf{s}}) = \begin{pmatrix} 1 & s_x & s_y & s_z & s_x^2 & s_x s_y & s_x s_z & s_y^2 & s_y s_z & s_z^2 \end{pmatrix} \quad ,$$

where $v_1(\hat{\mathbf{s}})$ is the interpolation vector for the clear skylight function with sun in direction $\hat{\mathbf{s}}$, and v_0 is the interpolation vector that gives a least-squares best fit to the overcast skylight function (8). Then let

$$v(\hat{\mathbf{s}}, b) = (1 - b)v_0 + bv_1(\hat{\mathbf{s}}).$$

The interpolation functions α_i in (12) are the components of:

$$v(\hat{\mathbf{s}}, b)\mathbf{M}^{\#}.$$

5 Implementation and Results

Figure 6 provides an overview of our approach. There are two main components to the system: the pre-processing stage, which entails computing the basis images using a renderer, and the post-processing phase, which involves re-rendering the scene at a specific time of day under desired sky conditions (e.g. overcast or clear).

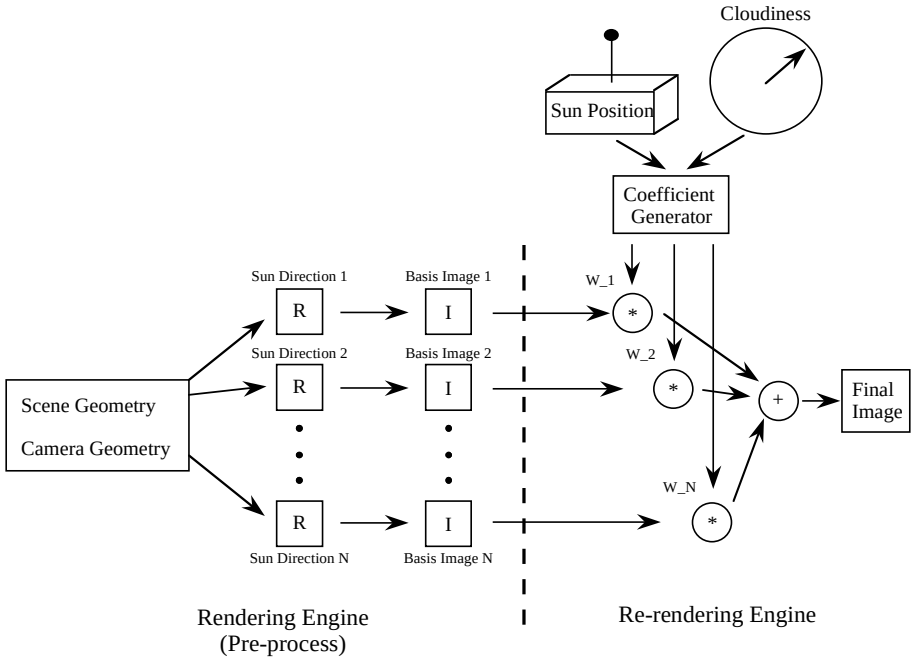

Fig. 6. Implementation architecture

Once the basis images have been computed, the scene can be quickly re-rendered based on a new sun position and desired sky conditions. This is accomplished by performing a linear combination of the basis images, given the coefficients for the sun direction and sky definition. Thus, a designer can see a variety of lighting configurations

throughout a typical day or observe an animation of these effects in the time required to combine the basis images together. That is, no calls to the costly global illumination operator are required for this interaction.

We used the illuminant approximation of (11) to generate our set of nine basis images for a sample architectural scene.

The set of $\hat{\mathbf{s}}_i$ used were

$$\begin{pmatrix} \hat{\mathbf{s}}_1 \\ \hat{\mathbf{s}}_2 \\ \hat{\mathbf{s}}_3 \\ \hat{\mathbf{s}}_4 \\ \hat{\mathbf{s}}_5 \\ \hat{\mathbf{s}}_6 \\ \hat{\mathbf{s}}_7 \\ \hat{\mathbf{s}}_8 \\ \hat{\mathbf{s}}_9 \end{pmatrix} = \begin{pmatrix} (\ 0.0000, & 0.0000, & 1.0000\ ) \\ (\ 0.9102, & 0.0000, & 0.4142\ ) \\ (\ 0.0000, & 0.9102, & 0.4142\ ) \\ (\ -0.9102, & 0.0000, & 0.4142\ ) \\ (\ 0.0000, & -0.9102, & 0.4142\ ) \\ (\ 0.6436, & 0.6436, & -0.4142\ ) \\ (\ -0.6436, & 0.6436, & -0.4142\ ) \\ (\ -0.6436, & -0.6436, & -0.4142\ ) \\ (\ 0.6436, & -0.6436, & -0.4142\ ) \end{pmatrix}.$$

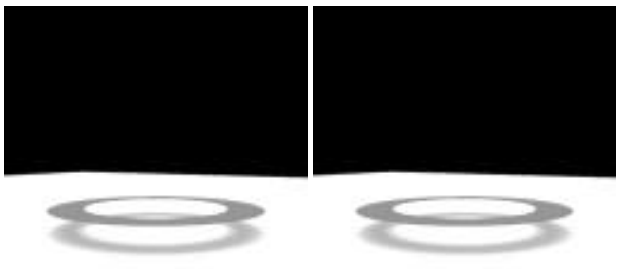

Fig. 7. a) Rendered image; b) re-rendered image (linear combination of nine basis images)

For each sun position $\hat{\mathbf{s}}_i$, an image was rendered using a multi-pass global illumination algorithm. The renderer combines a progressive radiosity pass to simulate diffuse lighting effects and a stochastic ray-tracing pass to simulate specular effects. The rendering was performed on a 50 MHz SGI Crimson VGXT with 64 megabytes of RAM. Each 640 by 486 image took approximately one hour to complete with a nine hour total rendering/setup time. Once the initial nine hour rendering phase is performed, re-rendering time becomes negligible (approximately one second per image, not including I/O).

It is important to note that our rendering times are representative of a global illumination (progressive radiosity) simulation running to convergence. However, our re-rendering implementation does not require this. The basis images can be computed with virtually any rendering algorithm—affording the possibility to trade quality for time. For example, during the early design stages of a building design, one could use a local illumination model or a very coarse global illumination solution to minimize the time required to compute the basis images. Later, when greater precision is necessary,

Fig. 8. Image rendered with CIE standard

Fig. 9. Image rendered with a steerable second order approximation

Fig. 10. Image rendered with a steerable fourth order approximation

more accurate images could be used. Also, we could use a larger number of basis images to get a more exact description of the daylight distribution function.

Figure 7 shows two images. The first is an image of a simple scene rendered with our sunlight model. The second has the sun at the same location as the first, but is a linear combination of the basis images for the scene. The RMS error calculated on these two images is negligable (> 0.008).

Figures 8-10 shows three images. The first is an image of a simple scene rendered with the CIE standard clear day function. The second image uses our second order approximation to the CIE standard and the third uses a fourth order approximation. As

Fig. 11. Sun from 7:00am through 11:45am in 15 minute increments. Each of these 20 images is computed via linear combination of the 9 basis images.

we increase the order of our approximation we can more closely match the CIE standard function. In the limit, our approximation is exact.

Color plates 1-4 (see color section) show four images. The first two are basis images (actually rendered). The third is a sunny-day image (computed via linear combination) with the sun position between that of the two basis images. The fourth is a cloudy-day image also computed via linear combination.

Figure 11 shows a sequence of twenty images corresponding to a sequence of sun positions during the course of a day. This is the prototypical example for showing the benefit of our approach. Once the small number of basis images are computed, obtaining a rendering of the environment with a new sun position and sky description is reduced

to a fast image combination step. Thus, the process is very interactive and alleviates many of the problems involved with constructing animations.

6 Discussion

We have presented a method for the fast re-rendering of naturally illuminated scenes using linear combinations of pre-rendered images. The method requires a set of "basis" illumination functions. In general, this set is unconstrained, but we have imposed a constraint of steerability in order to guarantee that the set of representable illuminants is rotationally invariant. This constraint is particularly useful when modeling natural illumination functions.

Our method allows for an interactive edit-render cycle by eliminating calls to the renderer once the basis images have been computed. This provides the user with the ability to animate naturally illuminated scenes in a fraction of the time that it would require with conventional methods. The limitation of the method is that efficiency requires broad extended illuminants. Narrower illuminants require a larger basis set and consequently more pre-rendering computation.

Several interesting research directions remain. First, for most environments, constructing the approximation of sunlight over the entire sphere is not necessary. Interiors are rarely exposed to sunlight from all directions as they often contain a limited number of openings. Additional work is necessary to quantify the relationship between basis size, basis quality, and spatial coverage. Second, we would like to take advantage of common computation between the different renderings. Recomputing the geometric relationships between objects is unnecessary in a static environment. Third, the approach can extended to use the vertex radiosities as a basis set. This would allow images to be created on the fly for any camera position. Although certain aspects of our approach require further investigation, our preliminary results indicate that this method offers a promising solution to daylight simulation.

References

1. CIE Techinical Committee 4.2. Standardization of luminance distribution on clear skies. CIE Publication No. 22, Commission International de L'Eclairaze, Paris, 1973.
2. John M. Airey, John H. Rohlf, and Frederick P. Brooks, Jr. Towards image realism with interactive update rates in complex virtual building environments. In *Computer Graphics (1990 Symposium on Interactive 3D Graphics)*, volume 24, pages 41–50, March 1990.
3. Chris Buckalew and Donald Fussell. Illumination networks: Fast realistic rendering with general reflectance functions. In *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 89–98, July 1989.
4. I. W. Busbridge. *The Mathematics of Radiative Transfer*. Cambridge University Press, Bristol, 1960.
5. Shenchang Eric Chen. Incremental radiosity: An extension of progressive radiosity to an interactive image synthesis system. In *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 135–144, August 1990.
6. IES Daylighting Committee. Recommended practice of daylighting. *Lighting Design and Application*, 9(2):45–58, 1979.

7. Julie Dorsey. *Computer Graphics Techniques for Opera Lighting Design and Simulation* . PhD thesis, Cornell University, Program of Computer Graphics, Ithaca, NY, January 1993.
8. Julie Dorsey, James Arvo, and Donald Greenberg. Interactive design of complex time-dependent lighting. Submitted for publication, 1994.
9. William T. Freeman and Edward H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 13(9):891–906, September 1991.
10. Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, 1989.
11. James T. Kajiya. The rendering equation. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 143–150, August 1986.
12. James T. Kajiya. Radiometry and photometry for computer graphics. In *SIGGRAPH '90 Advanced Topics in Ray Tracing Course Notes* . ACM Press, August 1990.
13. Claus Muller. *Spherical Harmonics*. Springer-Verlag, New York, NY, 1966.
14. Jeffry S. Nimeroff, Eero Simoncelli, Julie Dorsey, and Norman I. Badler. Rendering spaces for architectural environments. Submitted to Presence, the Journal of Virtual Reality and Teleoperators, April 1994.
15. Tomoyuki Nishita and Eihachiro Nakamae. Continuous tone representation of three-dimensional objects illuminated by sky light. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 125–132, August 1986.
16. P Perona. Deformable kernels for early vision. In *IEEE Comp. Soc. Conf. Computer Vision and Pattern Recognition*, pages 222–227, Maui, 1991.
17. Chris Schoeneman, Julie Dorsey, Brian Smits, James Arvo, and Donald Greenberg. Painting with light. In *Computer Graphics (SIGGRAPH '93 Proceedings)* , volume 27, pages 143–146, August 1993.
18. Carlo H. Séquin and Eliot K. Smyrl. Parameterized ray tracing. In *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 307–314, July 1989.
19. Eero P. Simoncelli. *Distributed Analysis and Representation of Visual Motion* . PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA, January 1993. Also available as MIT Media Laboratory Vision and Modeling Technical Report #209.
20. Eero P. Simoncelli, William T. Freeman, Edward H. Adelson, and David J. Heeger. Shiftable multi-scale transforms. *IEEE Trans. Information Theory* , 38(2):587–607, March 1992. Special Issue on Wavelets.
21. Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley, MA, 1986.
22. Gregory J. Ward. The radiance lighting simulation system. In *Computer Graphics (SIGGRAPH '94 Proceedings)* , volume 28, July 1994.

This article was processed using the LaTeX macro package with LMAMULT style