November 1995

Rendering Spaces for Architectural Environments

Jeffry S. Nimeroff
University of Pennsylvania

Julie Dorsey
Massachusetts Institute of Technology

Eero Simoncelli
University of Pennsylvania

Norman I. Badler
University of Pennsylvania, badler@seas.upenn.edu

Recommended Citation

Nimeroff, J. S., Dorsey, J., Simoncelli, E., & Badler, N. I. (1995). Rendering Spaces for Architectural Environments. Retrieved from http://repository.upenn.edu/hms/56

Rendering Spaces for Architectural Environments

Abstract
We present a new framework for rendering virtual environments. This framework is proposed as a complete scene description, which embodies the space of all possible renderings, under all possible lighting scenarios of the given scene. In effect, this hypothetical rendering space includes all possible light sources as part of the geometric model. While it would be impractical to implement the general framework, this approach does allow us to look at the rendering problem in a new way. Thus, we propose new representations that are subspaces of the entire rendering space. Some of these subspaces are computationally tractable and may be carefully chosen to serve a particular application. The approach is useful both for real and virtual scenes. The framework includes methods for rendering environments which are illuminated by artificial light, natural light, or a combination of the two models.

Rendering Spaces for Architectural Environments*

Jeffry S. Nimeroff

Department of Computer and Information Science

University of Pennsylvania, Philadelphia, PA 19104-6389

Julie Dorsey

Department of Architecture

Massachusetts Institute of Technology, Cambridge MA 02139

Eero Simoncelli   Norman I. Badler

Department of Computer and Information Science

University of Pennsylvania, Philadelphia, PA 19104-6389

November 1, 1995

Abstract

We present a new framework for rendering virtual environments. This framework is proposed as a complete scene description, which embodies the space of all possible renderings, under all possible lighting scenarios of the given scene. In effect, this hypothetical rendering space includes all possible light sources as part of the geometric model. While it would be impractical to implement the general framework, this approach does allow us to look at the rendering problem in a new way. Thus, we propose new representations that are subspaces of the entire rendering space. Some of these subspaces are computationally tractable and may be carefully chosen to serve a particular application. The approach is useful both for real and virtual scenes. The framework includes methods for rendering environments which are illuminated by artificial light, natural light, or a combination of the two models.

1 Introduction

The emerging field of visualization in computer science combines calculations with computer graphics to bring a new dimension of understanding to architectural designers. Such

[*] Presented in *Presence*, the Journal of Virtual Reality and Teleoperators: Special Issue on Architecture, Volume 4, Number 3, Summer 1995

visualization techniques provide the ability to predict many aspects of a future built environment, such as the scale, degree of spatial definition, relationship to the surrounding context, and illumination. Perhaps the most important and difficult aspect to understand is the last one—the illumination. In order to qualitatively assess a design, the designer needs to see all the subtle lighting effects such as soft shadowing, indirect illumination, and color bleeding. These effects are generally only available through costly global illumination algorithms.

Although global illumination techniques [Cohen and Wallace, 1993, Kajiya, 1986, Kajiya, 1990] can produce highly realistic simulations of virtual environments, these algorithms are difficult to use for architectural design [Airey et al., 1990, Dorsey et al., 1991, Dorsey et al., 1995]. Currently, a designer begins with a geometric model, positions the lights, assigns their colors and intensity distributions, and finally computes a solution. This iterative process is repeated until the designer finds an appropriate solution; often it requires many iterations to arrive at the final design. This method is typically time-consuming and tedious. Further, in environments that are illuminated by daylight [CIE Standardization Committee, 1973, Illumination Engineering Society, 1979, Nishita and Nakamae, a central component in the design is the dynamic motion of the sun across the cycle of a day. Currently, the only method to simulate this phenomenon is to compute a global illumination solution at each time step across a typical day and compose an animation [Nishita and Nakamae, 1986, Ward, 1992]. For an environment of even moderate complexity, such an animation would be completely impractical to compute, as it would involve computing a very large number of global solutions.

In this paper we define an abstract rendering space as a reorganization of Adelson and Bergen's total optical space [Adelson and Bergen, 1991]. This rendering space exhibits subspaces that span useful configurations of artificially and naturally illuminated environments and within this framework we can describe methods that efficiently (via linear combinations of images) solve the direct problem of generating images from light source descriptions. Since we are working with the algebraic structure of these subspaces, we can also consider goal directed "inverse" problems within the same context [Baltes, 1978, Kawai et al., 1993, Schoeneman et al., 1993].

2 Plenoptic (Total Optical) Space

The plenoptic function [Adelson and Bergen, 1991] is a 7-dimensional function that captures all the information contained in the light filling a region of space. If a pinhole camera is positioned at a given point (x, y, z), it will select a particular *pencil* (set of rays passing through that point) and reveal an image. A color picture would typically sample this pencil at three wavelengths. If we limit the view, the information available at the fixed position (x, y, z) has four more coordinates: $(\theta, \phi, \lambda, t)$, the azimuths, elevations, and wavelengths of each ray of light coming into that point, possibly as a function of time. Thus, there are a total of seven coordinate axes in the plenoptic space. We propose a rendering space—a reorganization of the plenoptic function that encompasses all possible photographs of a scene, under all possible lighting conditions.

3 Rendering Space

Suppose a camera is positioned so it is looking out at a scene, with the shutter held open for an extended period of time. We will also presume that the geometry is static. One can walk through the scene, shining a laser of adjustable wavelength on various objects in the space, illuminating each object. This method of photography is very common in the advertising industry, as it allows for the synthesis of just about any light source. If one desires to view the scene as it is illuminated by a pair of light sources, one sequentially places the light at each of the two locations in space and makes a long or multiple exposure. If one wishes to see the scene lit by an area light source, one can create the effect of a rectangular source by moving the light across and up and down in a rectangular pattern.

Now suppose, instead of using a long exposure, we collected data from the scene while the light source was being shone around from all possible locations in space. Such a scenario would be impractical, for there are infinitely many possible locations in space. In fact, even a coarse sampling would be impractical. However, suppose it were possible to collect this data. (Later, we will address the practical issues by choosing useful subsets of the data). The data captured would contain all the information necessary to synthesize the scene as it would appear under any combination of light sources.

As previously discussed, it is completely impractical to capture the entire rendering space, even with coarse sampling. If, however, we carefully select a portion of the space to capture, it will still span a useful gamut of possible renderings. For example, a designer may select a rendering space in a useful way and defer many of the compositional details, particularly those of positioning and adjusting the brightness ratios of the lights, until later.

4 Linearity of Illumination Effects

If one is willing to discount physical and quantum optics and work with static scene geometries, the rendering operation is linear with respect to illumination (n-D vector of light sources, or radiances) [Busbridge, 1960, Dorsey et al., 1995, Kajiya, 1986]. For our formulation, the rendering operator R takes an illumination description $L(\hat{\mathbf{u}})$ (parameterized by illumination direction $\hat{\mathbf{u}}$) and a scene geometry G:

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

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

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

For illumination functions constructed as the combination of simpler "basis" illumination functions $L_n(\hat{\mathbf{u}})$,

$$L(\hat{\mathbf{u}}) = \sum_{n=1}^{N} \alpha_n L_n(\hat{\mathbf{u}}), \tag{3}$$

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

$$R_G(L(\hat{\mathbf{u}})) = R_G\left(\sum_{n=1}^{N} \alpha_n L_n(\hat{\mathbf{u}})\right) = \sum_{n=1}^{N} \alpha_n R_G(L_n(\hat{\mathbf{u}})). \tag{4}$$

We might imagine several different approaches for determining such a basis. For example, we could choose an N, generate a large set of possible illumination functions, and compute the N principal components of the set of generated functions. This sort of approach has been used, for example, in computer vision [Perona, 1991]. We would, however, prefer to design the basis set analytically so as to span a particular set of possible illumination functions.

5 Artificial Illumination

A rendering space representation composed of the effects of several light sources does not necessarily span the entire space for all possible pictures under all possible lighting conditions, but this representation may be chosen in such a manner that it spans a reasonable subset of possible lighting conditions. In practice, we represent the rendering space with a set of basis images that is typically neither complete nor orthogonal.

5.1 A Simple Example

We begin by considering a very simple example. Consider figures (6-8), which depict renderings from a two-dimensional rendering space. The two-dimensional space is formed by two images. The first image (figure (6)) shows the scene as it is illuminated by several lights over the bar. The second image (figure (7)) shows the scene with the same camera position as it is illuminated by several light sources on the left side of the scene. The linearity of the global illumination operator with respect to the source term allows us to obtain a large variety of renderings that are simple combinations of these two images, ranging from the extremes of no contribution from either image (all the lights are off) to full contribution of each image (all the lights are on). Figure (8) shows a new rendering consisting of this latter condition.

5.2 Coordinate Transformations and “Negative Brightness”

Suppose, again referring to figure (6-8), we had computed both of the images with the lights at one half intensity. We could still have arrived at the same set of interpolated images as in the previous example. In fact, any two linearly independent solutions within the same rendering space (any two pictures of the effects of these lights with different levels of intensities) span that same two-dimensional rendering space. We can also remove light from a scene if we ensure that the rendering space spans the light of interest and simply subtract its effects.

This two-dimensional rendering space is a vector space over the scalar field of real image brightness. The unit vectors are the two original images. This simple example allows us to vary just the intensity of the lights. Such a rendering space could also be formed by photographs of a real scene under different lighting conditions.

From experience, designers frequently know about where to place the light sources but not how the lights will combine or how bright to make them. Consequently, the task of selecting appropriate intensities for static lights is a useful subproblem of lighting design. There is no limit to the dimensionality of the vector space representation. For example, we might have a vector space that consists of 100 images. Then, each image would be a point in a 100-dimensional rendering space.

5.3 The Rendering Space Module

So far our discussion has been limited to the vector space representation of a rendering space. That is, the coefficient in front of each image has been a real number. Suppose, however, that we wish to take a weighted sum over colors. For example, if we wish to render one image[1], denoted by I_1, times the color yellow, plus another image I_2, times the color blue. We may write this sum using red, green, and blue components:

$$\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} I_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} I_2,$$

where $(1 \quad 1 \quad 0)^T$ is the $(R \quad G \quad B)$ representation for yellow and $(0 \quad 0 \quad 1)^T$ is the representation for blue.

Now the scalar field has become a vector in its own right, and it is no longer a field since the scalars form a group under multiplication. However, the vectors still form a ring. The rendering space representation, while no longer a vector space representation, still has a definite mathematical structure. We could say that it is a concatenation of three

[1]An image $I \in P^{(Rows \times Cols)}$ where $P \in \begin{pmatrix} R & G & B \end{pmatrix}^T$

vector spaces, but there is a more meaningful structure; such a representation over a ring is a module.

We may generalize our rendering space module a bit further too. Suppose we let the scalars be 3 $\times$ 3 color coordinate transformation matrices, and we let the images I_k be color images (each a set of three images). Such a space is still a module. In fact, we do not need to limit ourselves to 3 $\times$ 3 matrices. We may have a set of k images $I_1 \ldots I_k$ which each have N spectral bands (colors), and the rendering space associated with these N-banded images may be rendered for display on an M banded device. Thus, each scalar is actually an $M \times N$ spectral transformation matrix.

6 Natural Illumination

Natural illumination concerns itself with the interaction of the environment and naturally occurring illumination sources (sun). Natural illumination is an especially difficult task due to the changing intensity distribution of the sun throughout the day. The amount of time required to render complex environments makes the iterative edit-render cycle impractical and makes constructing an animation of the naturally illuminated scene time consuming, Traditionally, each frame of the animation required positioning of the sun and invoking an expensive global illumination algorithm. With the construction of an appropriate rendering basis for naturally illuminated scenes, the calls to the global illumination algorithm can be replaced by image interpolations. We show how natural illumination can be described as a linear combination of simple illumination functions.

6.1 Skylight

We start by giving a definition and analytic description of skylight. By skylight [Nishita and Nakamae, 1986, Illumination Engineering Society, 1979] we mean the illumination that emanates from the sun but is absorbed and re-radiated by the atmosphere (clouds) before illuminating the scene. The function simulates atmospheric illumination on an overcast day. It has a cosine falloff relative to a fixed point (zenith) but does not depend on the sun's position in the sky:

$$L(\hat{\mathbf{u}}) = \pounds(1 + 2(u_z))/3. \tag{5}$$

Here, u_z is the z-component of $\hat{\mathbf{u}}$ and $\pounds$ is an overall illumination constant.

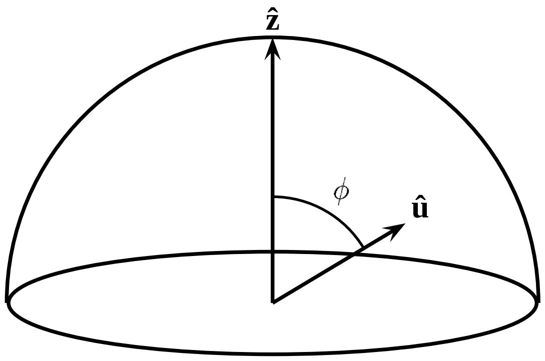

Figure 1: Skylight geometry

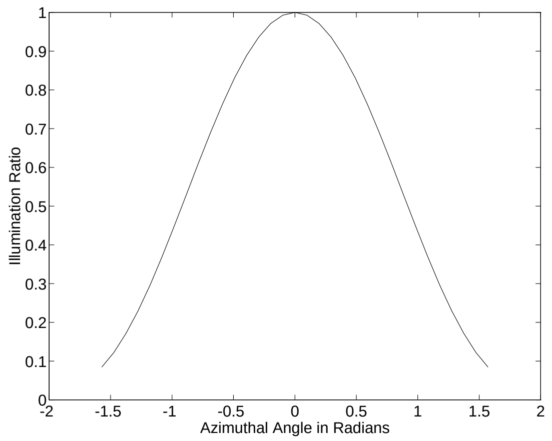

Figure 2: Skylight distribution

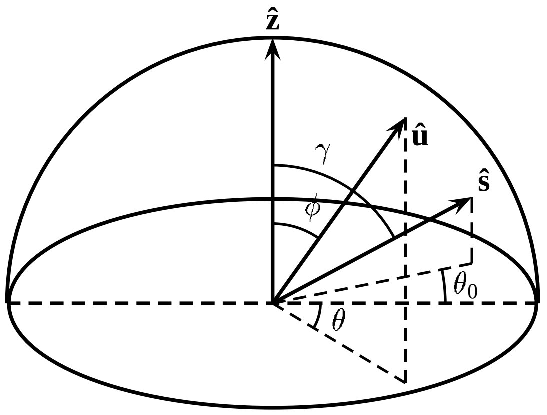

Figure 3: Direct sunlight geometry

6.2 Direct Sunlight

Direct sunlight [CIE Standardization Committee, 1973, Nishita and Nakamae, 1986] 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 as well as on the direction relative to a fixed point (zenith).

$$L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = \pounds\frac{(0.91 + 10\exp(-3\arccos(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})) + 0.45(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})^2)(1-\exp(-0.32/u_z))}{0.274(0.91 + 10\exp(-3\arccos(s_z)) + 0.45(s_z^2))} \tag{6}$$

where $\hat{\mathbf{s}}$ is the unit vector pointing to the sun and $\hat{\mathbf{u}}$ is the unit vector pointing in the sampling direction on the illuminating hemisphere.

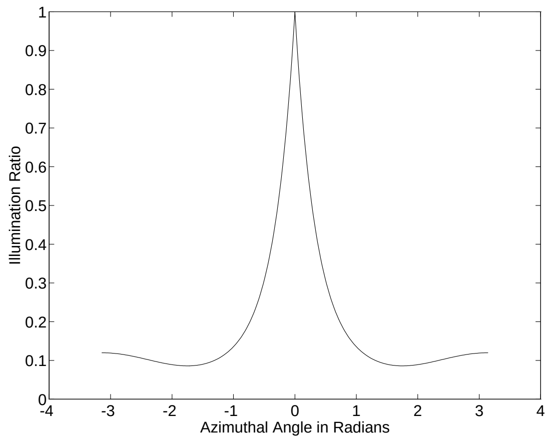

Figure 4: Direct sunlight distribution for sun at zenith

6.3 Steerable Basis Sets

A steerable function is one that can be written as a linear combination of rotated versions of itself. Freeman and Adelson [Freeman and Adelson, 1991] 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 re-rendering naturally illuminated environments

6.3.1 Steerable Functions: Directional Cosine Example

The simplest form of steerable function is a directional cosine 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 $\mathbf{R}^3$ where $(\cdot)$ indicates an inner product. We want to consider c to be a scalar function of $\hat{\mathbf{u}}$, parameterized by the direction $\hat{\mathbf{s}}$. Then we can expand the equation as:

$$\begin{aligned} c(\hat{\mathbf{u}};\hat{\mathbf{s}}) &= s_x(\hat{\mathbf{u}} \cdot \hat{\mathbf{e}}_x) + s_y(\hat{\mathbf{u}} \cdot \hat{\mathbf{e}}_y) + s_z(\hat{\mathbf{u}} \cdot \hat{\mathbf{e}}_z) \\ &= s_x c(\hat{\mathbf{u}};\hat{\mathbf{e}}_x) + s_y c(\hat{\mathbf{u}};\hat{\mathbf{e}}_y) + s_z c(\hat{\mathbf{u}};\hat{\mathbf{e}}_z), \end{aligned} \tag{7}$$

where the $\hat{\mathbf{e}}_*$ are the unit vectors corresponding to the three Euclidean coordinate axes, and s_* are the components of $\hat{\mathbf{s}}$.

Thus, we have written the directional cosine function in direction $\hat{\mathbf{s}}$ as a linear combination of the directional cosines along the three coordinate axes.

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 interpolation. The easiest way to see this is by working from equation (7). We can write $c(\hat{\mathbf{u}}; \hat{\mathbf{s}}_i)$ for each i in terms of the axis cosine functions:

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

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

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

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:

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

Given the axis cosines, we can compute *any* directional cosine function via equation (7):

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

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

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 the simplest steerable illuminant basis by simply adding unity to these:

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

The function L is written as a sum of *four* basis functions. 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:

$$L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = \begin{pmatrix} 1 \\ s_x \\ s_y \\ s_z \end{pmatrix}^T \mathbf{M}_4^{-1} \begin{pmatrix} L(\hat{\mathbf{u}};\hat{\mathbf{s}}_1) \\ L(\hat{\mathbf{u}};\hat{\mathbf{s}}_2) \\ L(\hat{\mathbf{u}};\hat{\mathbf{s}}_3) \\ L(\hat{\mathbf{u}};\hat{\mathbf{s}}_4) \end{pmatrix}, \tag{8}$$

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

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

6.4 Higher Order Basis Sets

We can generalize our choice of function by including polynomials of directional cosine functions. For example, consider the second order terms. We can easily write an equation for computing a single squared directional cosine as a linear combination of six quadratic terms:

$$(\hat{\mathbf{u}} \cdot \hat{\mathbf{s}})^2 = s_x^2(u_x^2) + 2s_x s_y(u_x u_y) + 2s_x s_z(u_x u_z) + s_y^2(u_y^2) + 2s_y s_z(u_y u_z) + s_z^2(u_z^2).$$

Again, this equation is not in steerable form but can be placed in that form by considering a set of six non-coplanar direction vectors, $\{\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, \hat{\mathbf{s}}_3, \hat{\mathbf{s}}_4, \hat{\mathbf{s}}_5, \hat{\mathbf{s}}_6\}$:

$$(\hat{\mathbf{u}} \cdot \hat{\mathbf{s}})^2 = \begin{pmatrix} s_x^2 \\ s_x s_y \\ s_x s_z \\ s_y^2 \\ s_y s_z \\ s_z^2 \end{pmatrix}^T \mathbf{M}_6^{-1} \begin{pmatrix} (\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}_1)^2 \\ (\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}_2)^2 \\ (\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}_3)^2 \\ (\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}_4)^2 \\ (\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}_5)^2 \\ (\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}_6)^2 \end{pmatrix}.$$

In general, a directional cosine raised to the Mth power will require at most $(M+1)(M+2)/2$ samples for steerable interpolation[2]. We may also (as in equation (8)), combine directional cosines of different powers into a polynomial.

Instead of polynomials, we could use a Fourier-style functional expansion in terms

[2]In fact, we may not need this many since we are not representing arbitrary homogeneous polynomials, but squares [Simoncelli, 1993].

of cos$(k\theta)$, where $\theta = \arccos(\hat{\mathbf{u}} \cdot \hat{\mathbf{s}})$. This is easily related to the expansion given above via multi-angle trigonometric identities. We will now use a second-order Fourier-style expansion to build a steerable illumination function.

6.5 Steerable Sunlight Approximation

In order to re-render naturally illuminated scenes as a set of image interpolations, a steerable approximation to the direct sunlight function above (equation (6)) must be constructed. We decompose it as follows;

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

In order to interpolate $L(\hat{\mathbf{u}};\hat{\mathbf{s}})$ from a set of illuminants $L(\hat{\mathbf{u}};\hat{\mathbf{s}}_n)$, 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 acts as an illumination scaling constant. Therefore the steerable form of $L(\hat{\mathbf{u}};\hat{\mathbf{s}})$

may be written as:

$$L(\hat{\mathbf{u}};\hat{\mathbf{s}}) = L_1(\hat{\mathbf{s}}) \cdot \sum_i (\overbrace{L_2(\hat{\mathbf{u}}) \cdot \underbrace{L_3(\hat{\mathbf{s}}_i \cdot \hat{\mathbf{u}})}_{\text{Steerable}}}^{\text{Basis lights}} \cdot \alpha_i(\hat{\mathbf{s}})) \tag{9}$$

We must construct a steerable approximation to the function L_3:

$$L_3(\hat{\mathbf{u}} \cdot \hat{\mathbf{s}}) = 0.91 + 10e^{-3\,\mathrm{arccos}(\hat{\mathbf{u}}\cdot\hat{\mathbf{s}})} + 0.45(\hat{\mathbf{u}} \cdot \hat{\mathbf{s}})^2 \tag{10}$$

The constant and the $\cos^2$ are not a problem, but the exponential is not in the desired form of a directional cosine polynomial. We can expand the entire function in a Fourier series as follows:

$$\begin{aligned} L_3(\theta) &= 2.204 + 1.925\cos(\theta) + 1.709\cos(2\theta) \\ &\quad +1.077\cos(3\theta) + 0.779\cos(4\theta) + 0.577\cos(5\theta) \\ &\quad +0.44\cos(6\theta)\cdots \end{aligned}$$

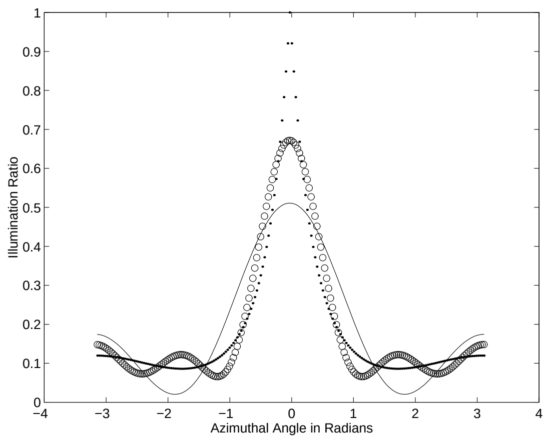

Figure 5: Approximated direct sunlight distribution for sun at zenith, 2nd order, 4th order, CIE standard

For purposes of this paper, we choose to limit our expansion to second order:

$$\begin{aligned} L_3(\theta) &= 2.204 + 1.925\cos(\theta) + 1.709\cos(2\theta) \\ &= 0.495 + 1.925\cos(\theta) + 3.418\cos^2(\theta) \end{aligned}$$

This steerable function requires a set of 10 sample directions: one for the zeroth order term, three for the first order terms, and six for the second order terms.

6.6 Implementation

Using the linearity of the rendering operator and equation (9) we can write an expression for re-rendering an image for any sun direction $\hat{\mathbf{s}}$ (figures (9-12)) from the images rendered under 10 different sun directions s_n:

$$\mathbf{I}(\hat{\mathbf{s}}) = L_1(\hat{\mathbf{s}}) \cdot \sum_n \alpha_n(\hat{\mathbf{s}}) \frac{\mathbf{I}(\hat{\mathbf{s}}_n)}{L_1(\hat{\mathbf{s}}_n)}, \tag{11}$$

where the α_n are the components of:

$$\begin{pmatrix} 1 & s_x & s_y & s_z & s_x^2 & s_x s_y & s_x s_z & s_y^2 & s_y s_z & s_z^2 \end{pmatrix} \mathbf{M}_{10}^{-1}, \tag{12}$$

the rows of $\mathbf{M}_{10}$ are in the form of the vector in equation (12) for each $\hat{\mathbf{s}}_n$[3].

Once the basis images have been computed image re-rendering is a very quick process. With the basis images in memory, the linear image combination was on the order of one second making the animation of naturally illuminated scenes very efficient.

[3]In actuality, only nine samples are needed to span the space of second order polynomials in cosine so $\mathbf{M}_{10}^{-1}$ can be replaced by $\mathbf{M}_9^{\#}$ where # is the matrix pseudo-inverse operation and $\mathbf{M}_9$ is a 9×10 matrix

6.7 “Partially Cloudy Day” Problem

As shown above, a steerable approximation for an overcast day requires a first-order expansion, while the sunny day requires a second-order expansion. To generate an image for a partially cloudy 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 $[0,1]$ telling us the brightness of the day (1=sunny, 0=cloudy). Let

$$v_0 = \left(\beta_1 \quad 0 \quad 0 \quad \beta_2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \beta_3 \right),$$

and

$$v_1(\hat{\mathbf{s}}) = \left(1 \quad s_x \quad s_y \quad s_z \quad s_x^2 \quad s_x s_y \quad s_x s_z \quad s_y^2 \quad s_y s_z \quad s_z^2 \right),$$

where $v_1(\hat{\mathbf{s}})$ is the interpolation vector for the sunny day with sun in direction $\hat{\mathbf{s}}$, and v_0 is the interpolation vector built from the cloudy day function (equation (5)). Then let

$$v(\hat{\mathbf{s}}, b) = (1-b)v_0 + b v_1(\hat{\mathbf{s}}).$$

The interpolation functions α_n (equation (11)) are the components of:

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

7 "Inverse" Problems

Along with the direct solutions to the artificial and natural illumination problems, the computer graphics community has recently started to describe and address a set of inverse problems. These problems can loosely be described as goal directed. Given a description of the desired solution, the system determines various parameters.

Schoeneman et. al. [Schoeneman et al., 1993] have used a constrained least squares approach to solve for light source intensities $(\mathbf{\Phi}^1, \ldots, \mathbf{\Phi}^n)$ that most closely a "painted" image $\mathbf{\Psi}$ provided by a user. Each function $\mathbf{\Phi}^i$ represents the effect of a single illumination source on the scene and takes the form of a set of vertex radiosities. A set of weights $(w_1, \ldots, w_n)$ is used to construct an approximated image $\hat{\mathbf{\Psi}}$,

$$\hat{\mathbf{\Psi}} = \sum_{i=1}^{n} w_i \mathbf{\Phi}^i,$$

where the objective function $\|\Psi - \hat{\Psi}\|$ is minimized. Their solution assumes that the positions of the light sources is fixed.

Kawai et. al. [Kawai et al., 1993] solve for the parameters in a radiosity solution as a constrained optimization problem. They introduce three kinds of constraints:

- *Physical Constraints:* Hard constraints between illumination parameters based on the rendering equation [Kajiya, 1986].

- *Design Goals:* User chosen constraints applied to elements in the scene.

- *Barrier Constraints:* Hard bounds on the allowable ranges of optimization variable.

Barrier constraints must be satisfied while design goals are recommendations. The constrained problem is then converted to an unconstrained form (penalty method) and the BFGS method is used to perform the search. The user provides the parameters to perform an initial rendering and then can work interactively assigning new parameters to the surface elements and then reapplying the optimization algorithm.

We have begun to solve an inverse problem based on our steerable daylight model [Nimeroff et al., 1994]. Given any image I, we can easily solve for the least-squares best choice of weights β_n for approximating that image with a set of rendered basis images, I_n. The solution is:

$$\beta = \mathbf{B}^{-1}\mathbf{c},$$

where $\vec{c}$ is the vector

$$\begin{pmatrix} I \cdot I_1 \\ I \cdot I_2 \\ \vdots \\ I \cdot I_N \end{pmatrix},$$

$\mathbf{B}$ is the matrix with components

$$B_{i,j} = I_i \cdot I_j,$$

and the operator $(\cdot)$ indicates the sum over all pixels of the pointwise product of two images. We could then construct the light source corresponding to this approximation as

$$\hat{L}_I(\hat{\mathbf{s}}) = \sum_n \beta_n L(\hat{\mathbf{s}}_n).$$

We could take this process one step further and solve for a sunlight direction $\hat{\mathbf{s}}$ such that the interpolation functions $\alpha(\hat{\mathbf{s}})$ are a good approximation to the weights β_n. This, however, requires a nonlinear optimization over $\hat{\mathbf{s}}$.

8 Conclusions and Future Work

We have demonstrated through some simple examples that rendering space representations can capture enough information from a scene to allow one to render the scene under a variety of lighting conditions. Our techniques are particularly amenable to the design of complex virtual environments, as the re-rendering times is not related to the complexity of the environment.

Several interesting research issues remain. We are currently examining environment-dependent basis representations. This avenue of research will hopefully lead to more efficient strategies for rendering the basic representations for complex environments. We are also working on a more general solution to the inverse problem. Last, while we have only begun to examine the rendering spaces from the perspective of a fixed point in space, further exploration, such as a practical implementation of rendering space representations for a moving camera, appear to be a promising area of future research.

References

[Adelson and Bergen, 1991] Adelson, E. H. and Bergen, J. R. (1991). The plenoptic function and the elements of early vision. In Landy, M. S. and Movshon, J. A., editors, *Computational Models of Visual Processing*, pages 1–20. MIT Press, Cambridge, MA.

[Airey et al., 1990] Airey, J. M., Rohlf, J. H., and Brooks, Jr., F. P. (1990). Towards image realism with interactive update rates in complex virtual building environments. In Riesenfeld, R. and Sequin, C., editors, *Computer Graphics (1990 Symposium on Interactive 3D Graphics)*, volume 24, pages 41–50.

[Baltes, 1978] Baltes, H. P., editor (1978). *Inverse Source Problems in Optics*. Springer-Verlag, New York, NY.

[Busbridge, 1960] Busbridge, I. W. (1960). *The Mathematics of Radiative Transfer*. Cambridge University Press, Bristol.

[CIE Standardization Committee, 1973] CIE Standardization Committee (1973). Standardization of luminance distribution on clear skies. CIE Publication No. 22, Commission International de L'Eclairaze, Paris.

[Cohen and Wallace, 1993] Cohen, M. F. and Wallace, J. R. (1993). *Radiosity and Realistic Image Synthesis*. Academic Press Professional, Cambridge, MA.

[Dorsey et al., 1995] Dorsey, J., Arvo, J., and Greenberg, D. (1995). Interactive design of complex time-dependent lighting. To appear in IEEE Computer Graphics and Applications.

[Dorsey et al., 1991] Dorsey, J., Sillion, F. X., and Greenberg, D. P. (1991). Design and simulation of opera lighting and projection effects. In Sederberg, T. W., editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 41–50.

[Fournier et al., 1993] Fournier, A., Gunawan, A. S., and Romanzin, C. (1993). Common illumination between real and computer generated scenes. In *Proceedings of Graphics Interface '93*, pages 254–262.

[Freeman and Adelson, 1991] Freeman, W. T. and Adelson, E. H. (1991). The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906.

[Golub and Loan, 1989] Golub, G. H. and Loan, C. F. V. (1989). *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD.

[Illumination Engineering Society, 1979] Illumination Engineering Society (1979). Recommended practice of daylighting. *Lighting Design and Application*, 9(2):45–58.

[Kajiya, 1986] Kajiya, J. T. (1986). The rendering equation. In Evans, D. C. and Athay, R. J., editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 143–150.

[Kajiya, 1990] Kajiya, J. T. (1990). Radiometry and photometry for computer graphics. In *SIGGRAPH '90 Advanced Topics in Ray Tracing Course Notes*. ACM Press.

[Kawai et al., 1993] Kawai, J. K., Painter, J. S., and Cohen, M. F. (1993). Radioptimization - goal based rendering. In Kajiya, J. T., editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 147–154.

[Muller, 1966] Muller, C. (1966). *Spherical Harmonics*. Springer-Verlag, New York, NY.

[Nimeroff et al., 1994] Nimeroff, J. S., Simoncelli, E., and Dorsey, J. (1994). Efficient re-rendering of naturally illuminated environments. In *5th Annual Eurographics Workshop on Rendering*, pages 359–374.

[Nishita and Nakamae, 1986] Nishita, T. and Nakamae, E. (1986). Continuous tone representation of three-dimensional objects illuminated by sky light. In Evans, D. C. and Athay, R. J., editors, *Computer Graphics (SIGGRAPH ’86 Proceedings)*, volume 20, pages 125–132.

[Perona, 1991] Perona, P. (1991). Deformable kernels for early vision. In *IEEE Comp. Soc. Conf. Computer Vision and Pattern Recognition*, pages 222–227, Maui.

[Poulin and Fournier, 1992] Poulin, P. and Fournier, A. (1992). Lights from highlights and shadows. In *Proceedings of the 1992 Western Computer Graphics Symposium*, pages 141–145.

[Schoeneman et al., 1993] Schoeneman, C., Dorsey, J., Smits, B., Arvo, J., and Greenberg, D. P. (1993). Painting with light. In Kajiya, J. T., editor, *Computer Graphics (SIGGRAPH ’93 Proceedings)*, volume 27, pages 143–146.

[Simoncelli, 1993] Simoncelli, E. P. (1993). *Distributed Analysis and Representation of Visual Motion*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA. Also available as MIT Media Laboratory Vision and Modeling Technical Report #209.

[Strang, 1986] Strang, G. (1986). *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley, MA.

[Ward, 1992] Ward, G. J. (1992). The radiance lighting simulation system. In *SIGGRAPH '92 Global Illumination Course Notes*. ACM Press.

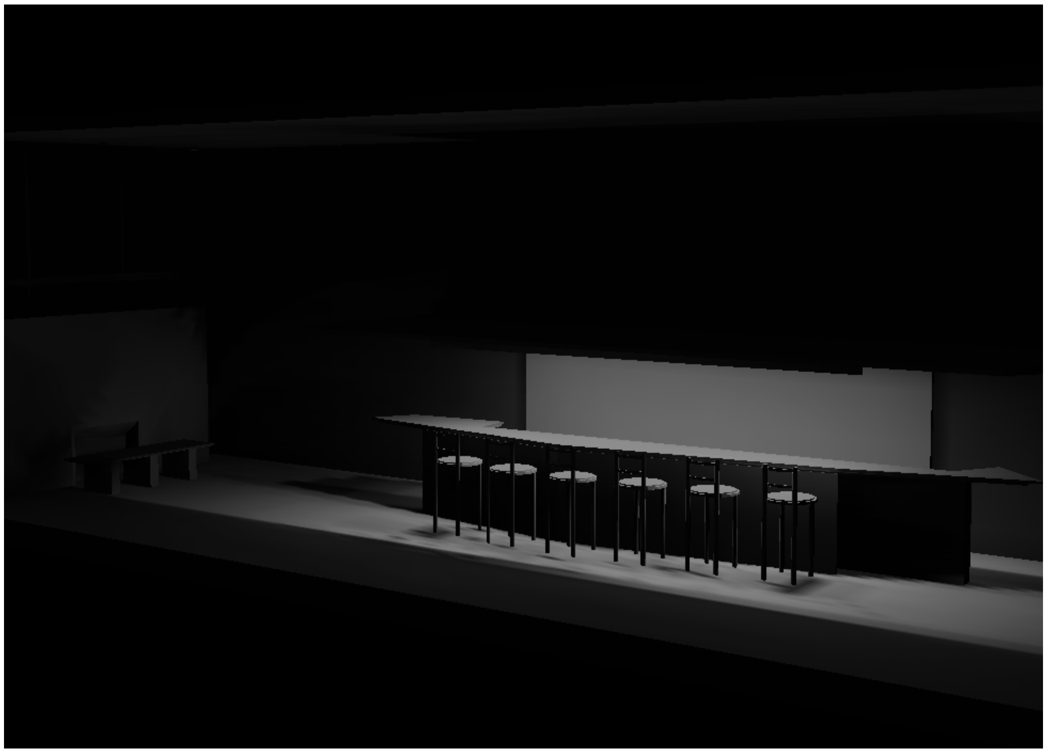

Figure 6: Bar with bar lights on

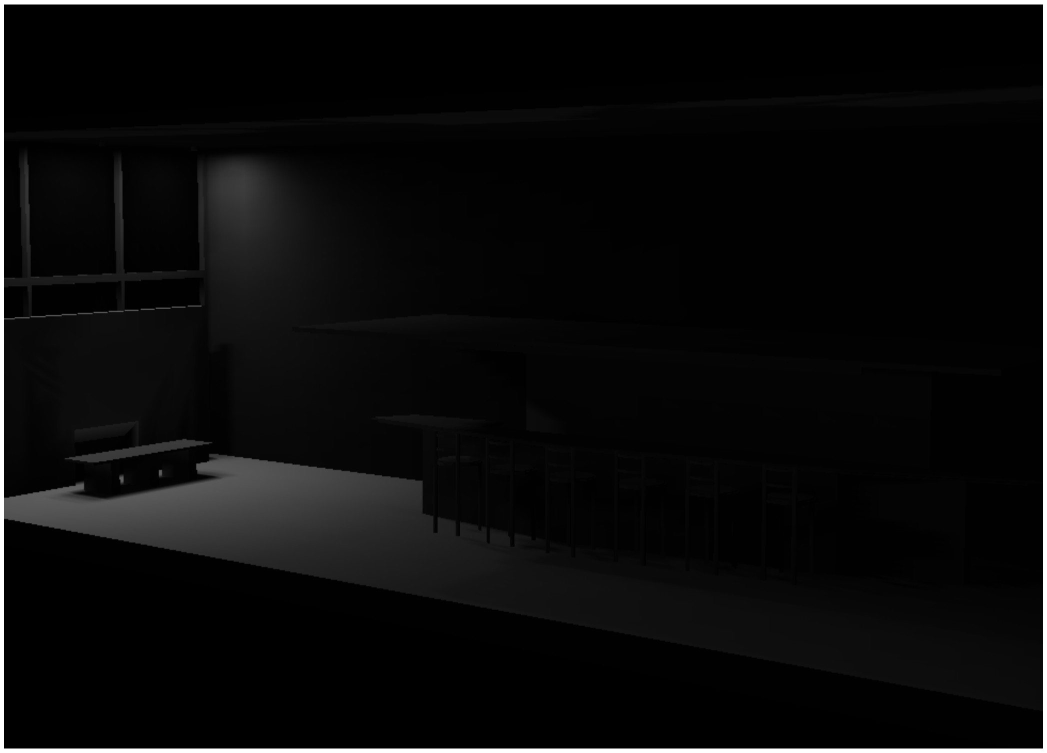

Figure 7: Bar with bank lights on

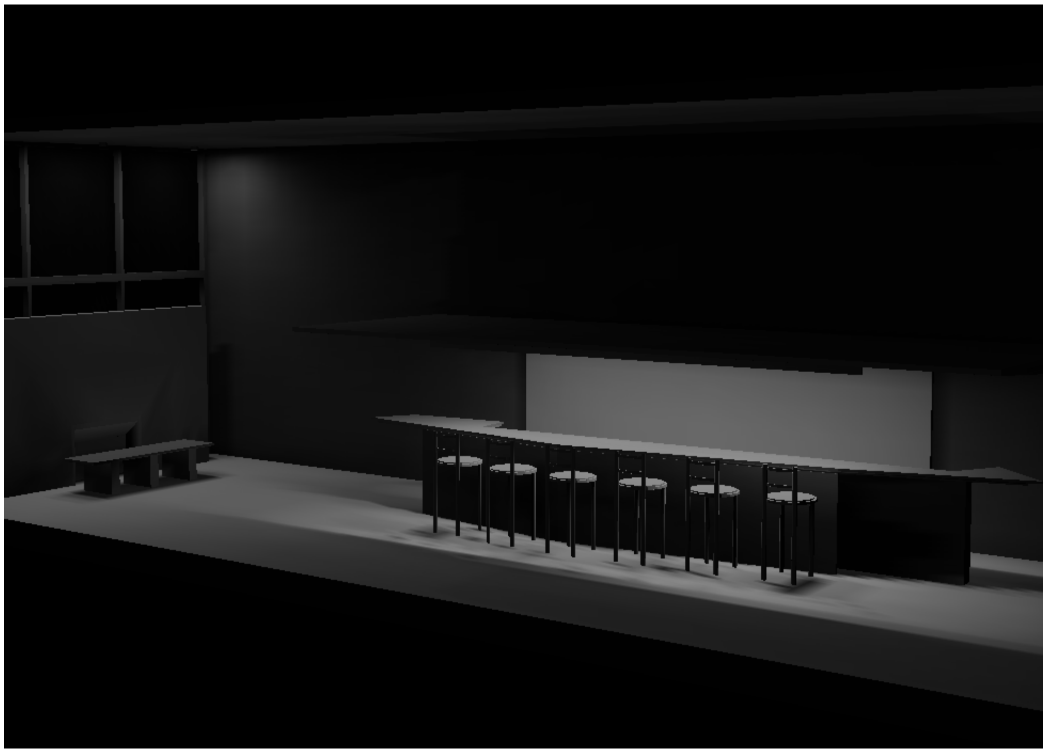

Figure 8: Bar with both sets of lights on

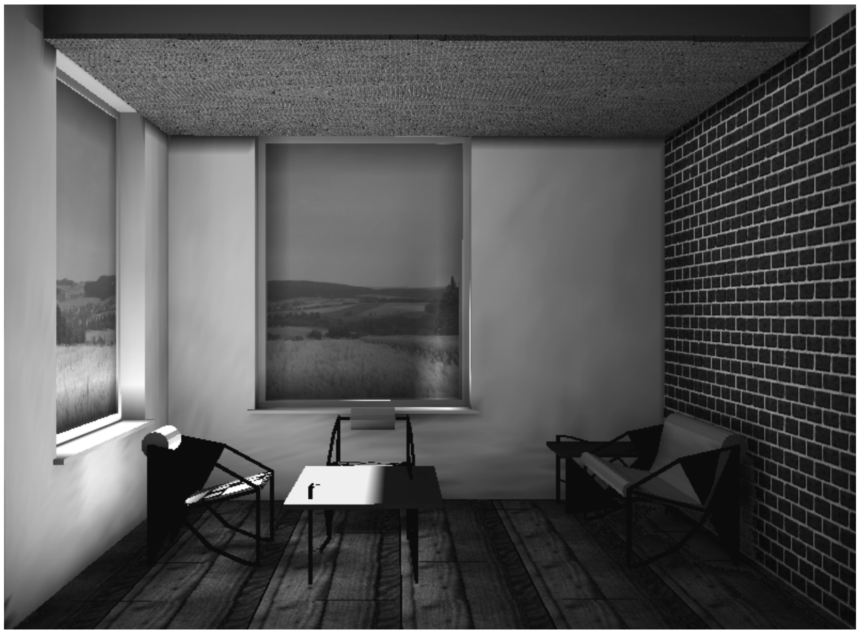

Figure 9: Room with sun at position 1

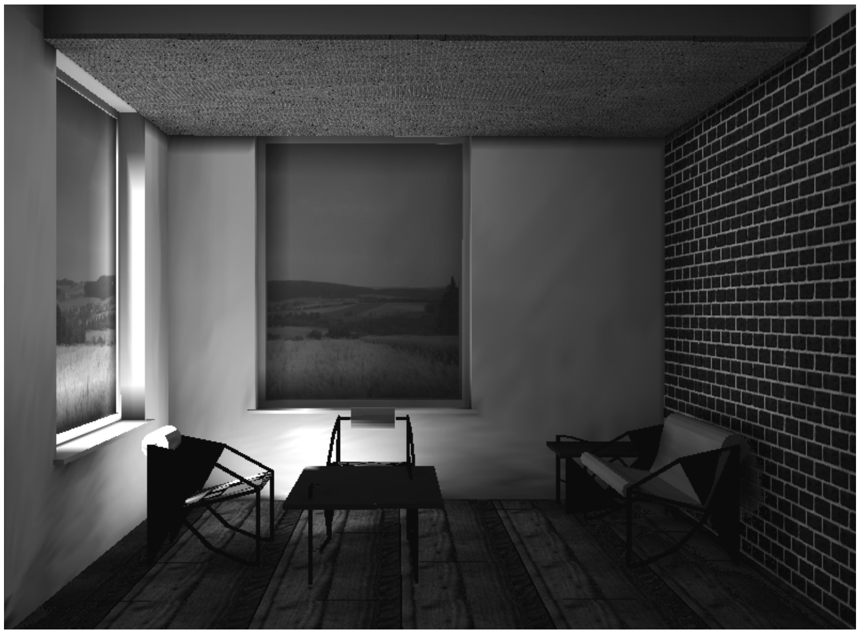

Figure 10: Room with sun at position 2

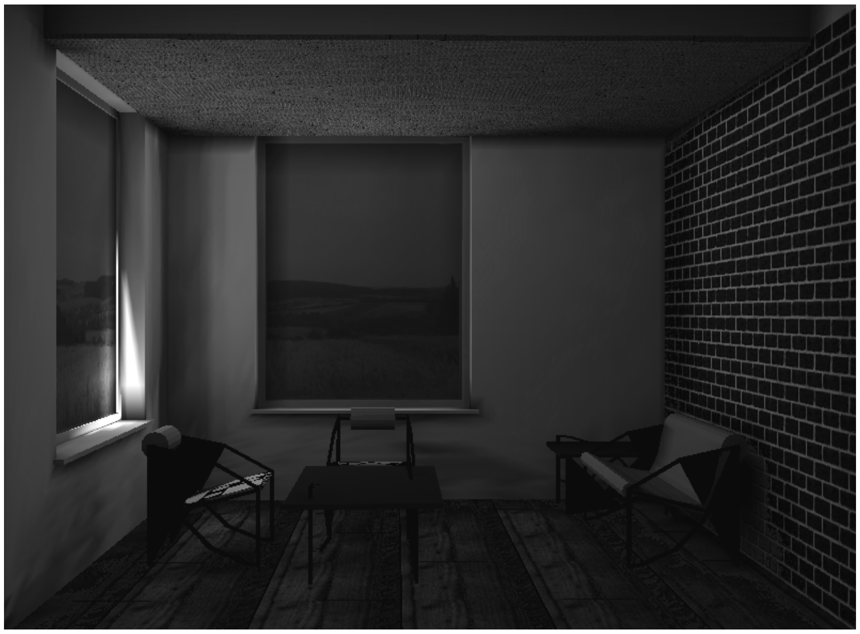

Figure 11: Room with sun at position 3

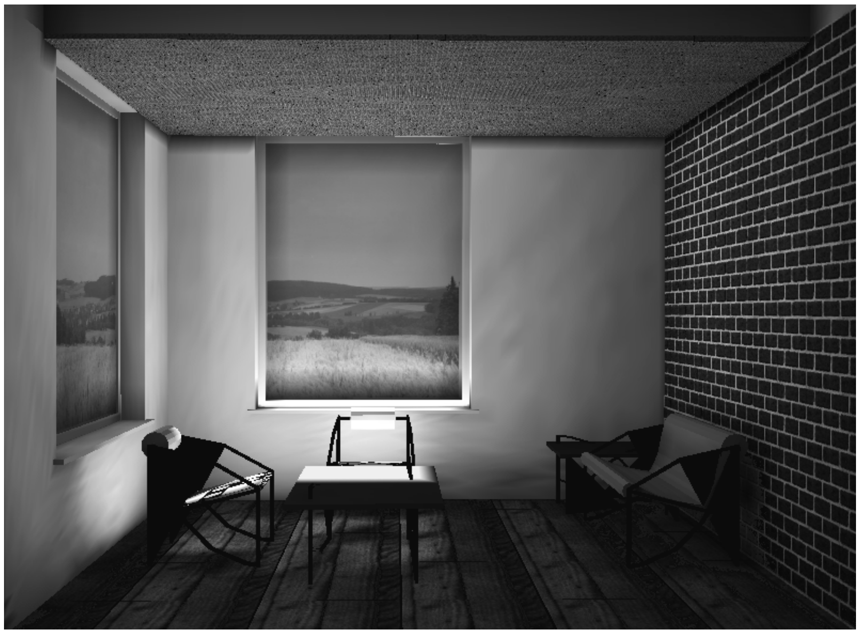
Figure 12: Room with sun at position 4

9 Information

Authors:	Jeffry Nimeroff,
	Eero Simoncelli,
	Julie Dorsey,
	Norman I. Badler
Email:	jnimerof@graphics.cis.upenn.edu,
Lmail:	University of Pennsylvania
	Department of Computer and Information Science
	200 South 33rd St.
	Philadelphia, PA 19104
Telephone:	(215)898-8745
	(215)898-1976
Fax:	(215)898-0587