An Efficient Representation for Irradiance Environment Maps

Ravi Ramamoorthi Pat Hanrahan

Stanford University *

Abstract

We consider the rendering of diffuse objects under distant illumination, as specified by an environment map. Using an analytic expression for the irradiance in terms of spherical harmonic coefficients of the lighting, we show that one needs to compute and use only 9 coefficients, corresponding to the lowest-frequency modes of the illumination, in order to achieve average errors of only 1%. In other words, the irradiance is insensitive to high frequencies in the lighting, and is well approximated using only 9 parameters. In fact, we show that the irradiance can be procedurally represented simply as a quadratic polynomial in the cartesian components of the surface normal, and give explicit formulae. These observations lead to a simple and efficient procedural rendering algorithm amenable to hardware implementation, a prefiltering method up to three orders of magnitude faster than previous techniques, and new representations for lighting design and image-based rendering.

CR Categories: I.3.7 [*Computer Graphics*]: Three-Dimensional Graphics and Realism—Environment Maps

Keywords: Environment Maps, Rendering Hardware, Signal Processing, Irradiance, Radiance, Illumination, Lambertian Reflectance, Prefiltering, Spherical Harmonics

1 Introduction

Lighting in most real scenes is complex, coming from a variety of sources including area lights and large continuous lighting distributions like skylight. But current graphics hardware only supports point or directional light sources. One reason is the lack of simple procedural formulas for general lighting distributions. Instead, an integration over the upper hemisphere must be done for each pixel.

One approach to using general lighting distributions is the method of environment maps. Environment maps are representations of the incident illumination at a point. Blinn and Newell [3] used them to efficiently find reflections of distant objects. Miller and Hoffman [14], and Greene [8] *prefiltered* environment maps, precomputing separate reflection maps for the diffuse and specular components of the BRDF. Cabral et al. [5] handled general BRDFs by using a 2D set of prerendered images. Prefiltering is generally an offline, computationally expensive process. After prefiltering, rendering can usually be performed at interactive rates with graphics hardware using texture-mapping.

This paper focuses on the Lambertian component of the BRDF. We use the term *irradiance environment map* for a diffuse reflection map indexed by the surface normal, since each pixel simply stores the irradiance for a particular orientation of the surface. For applications like games, irradiance maps are often stored directly on the surface, instead of as a function of the normal vector, and are called *light maps*. Irradiance environment maps can also be extended to spatially varying illumination by computing an *irradiance volume*, as done by Greger et al. [9]. Many of the same ideas can be applied to speeding up global illumination algorithms. The slowly varying nature of irradiance has led to Ward and Heckbert [18] proposing interpolation using irradiance gradients, while the idea of storing irradiance as a function of surface orientation in *orientation lightmaps* has been proposed by Wilkie et al. [19].

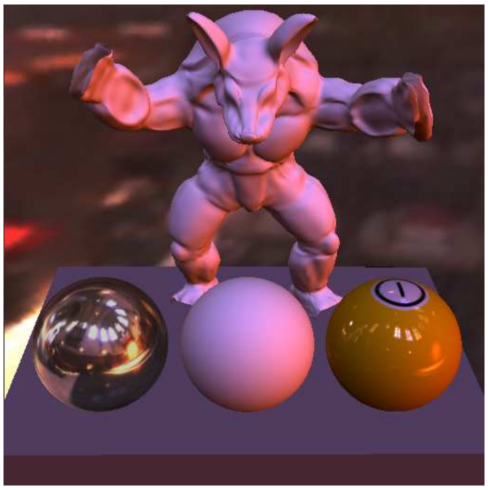

Figure 1: *The diffuse shading on all the objects is computed procedurally in real-time using our method. The middle sphere, armadillo, and table are white diffuse reflectors. The colors come from the environment—owing to a variety of colored sources, including blue stained-glass windows. Our method can also be combined with standard texture mapping—used to modulate the albedo of the pool-ball on the right—and reflection mapping—used for specular highlights on the pool-ball, and for the mirror sphere on the left. The environment is a light probe of the Grace Cathedral. Tone mapping is used to convey high dynamic range for the background and the mirror sphere; the remaining objects are shaded using a linear scale.*

The key to our approach is the rapid computation of an analytic approximation to the irradiance environment map. For rendering, we demonstrate a simple procedural algorithm that runs at interactive frame rates, and is amenable to hardware implementation. The procedural approach is preferable to texture-mapping in some applications. Since irradiance varies slowly with orientation, it need only be computed per-vertex and interpolated across triangles. Further, we require only a single texturing pass to render textured objects with irradiance environment maps, since the irradiance is computed procedurally. On the other hand, the standard approach requires a separate texture for the irradiance, and needs *multitexturing* support or multiple texturing passes. In other applications, where per-fragment texture-mapping is relatively inexpensive, our method can be used to very efficiently compute the irradiance environment map texture. Our novel representation also suggests new approaches to lighting design and image-based rendering.

2 Background

Empirically, it is well known that the reflected intensity from a diffuse surface varies slowly as a function of surface orientation. This qualitative observation has been used to justify representing irradiance environment maps at low resolutions [14], and in efficiently computing the shading hierarchically [11, 12]. Our goal is to use an analytic quantitative formula for the irradiance which formalizes these observations, and allows for principled approximations.

Let L denote the distant lighting distribution. As is common

* (ravir,hanrahan)@graphics.stanford.edu

with environment map algorithms, we neglect the effects of cast shadows and near-field illumination. The irradiance E is then a function of the surface normal $\mathbf{n}$ only and is given by an integral over the upper hemisphere $\Omega(\mathbf{n})$.

$$E(\mathbf{n}) = \int_{\Omega(\mathbf{n})} L(\omega)(\mathbf{n}\cdot\omega)\, d\omega \tag{1}$$

Note that $\mathbf{n}$ and ω are unit direction vectors, so E and L can be parameterized by a direction (θ, ϕ) on the unit sphere.

We must scale E by the surface albedo ρ, which may be dependent on position $\mathbf{p}$ and be described by a texture, to find the radiosity B, which corresponds directly to the image intensity.

$$B(\mathbf{p},\mathbf{n}) = \rho(\mathbf{p})E(\mathbf{n}) \tag{2}$$

Our main concern will be approximating E. We [16] have been able to derive an analytic formula for the irradiance. Similar results have been obtained independently by Basri and Jacobs [2] in simultaneous work on face recognition. Our original motivation was the study of an inverse rendering problem—estimating the lighting from observations of a Lambertian surface, i.e. from the irradiance. In this paper, we will apply the formulae to a forward rendering problem—rendering diffuse objects with environment maps.

Our formulae are in terms of spherical harmonic [4, 13, 17] coefficients. Spherical harmonics Y_{lm}, with $l \geq 0$ and $-l \leq m \leq l$, are the analogue on the sphere to the Fourier basis on the line or circle. The first 9 spherical harmonics (with $l \leq 2$) are simply constant ($l = 0$), linear ($l = 1$), and quadratic ($l = 2$) polynomials of the cartesian components (x, y, z). and are given numerically by

$$\begin{aligned}
(x, y, z) &= (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta) \\
Y_{00}(\theta,\phi) &= 0.282095 \\
(Y_{11}; Y_{10}; Y_{1-1})\,(\theta,\phi) &= 0.488603\,(x; z; y) \\
(Y_{21}; Y_{2-1}; Y_{2-2})\,(\theta,\phi) &= 1.092548\,(xz; yz; xy) \\
Y_{20}(\theta,\phi) &= 0.315392\left(3z^2 - 1\right) \\
Y_{22}(\theta,\phi) &= 0.546274\left(x^2 - y^2\right)
\end{aligned} \tag{3}$$

Note that these basis functions are closely related to the spherical polynomials used by Arvo [1] in his irradiance tensor formulation.

$E(\theta, \phi)$ and $L(\theta, \phi)$ can be represented by the coefficients—E_{lm} and L_{lm}—in their spherical harmonic expansion.

$$\begin{aligned}
L(\theta,\phi) &= \sum_{l,m} L_{lm} Y_{lm}(\theta,\phi) \\
E(\theta,\phi) &= \sum_{l,m} E_{lm} Y_{lm}(\theta,\phi)
\end{aligned} \tag{4}$$

We also define $A = (\mathbf{n}\cdot\omega)$ with coefficients A_l. Since A has no azimuthal dependence, $m = 0$ and we use only the l index.

$$A(\theta) = \max\left[\cos\theta, 0\right] = \sum_l A_l Y_{l0}(\theta)$$

With these definitions one can show [16] that

$$E_{lm} = \sqrt{\frac{4\pi}{2l+1}} A_l L_{lm} \tag{5}$$

It will be convenient to define a new variable $\hat{A}_l$ by

$$\hat{A}_l = \sqrt{\frac{4\pi}{2l+1}} A_l \tag{6}$$

For rendering, it will be convenient to expand out the irradiance.

$$E(\theta,\phi) = \sum_{l,m} \hat{A}_l L_{lm} Y_{lm}(\theta,\phi) \tag{7}$$

An analytic formula for A_l can be derived [16]. It can be shown that $\hat{A}_l$ vanishes for odd values of $l > 1$, and even terms fall off very rapidly as $l^{-5/2}$. The analytic formulae are given by

$$\begin{aligned}
l = 1 \quad & \hat{A}_1 = \frac{2\pi}{3} \\
l > 1, odd \quad & \hat{A}_l = 0 \\
l \;\; even \quad & \hat{A}_l = 2\pi \frac{(-1)^{\frac{l}{2}-1}}{(l+2)(l-1)} \left[\frac{l!}{2^l (\frac{l}{2}!)^2}\right]
\end{aligned} \tag{8}$$

Numerically, the first few terms are

$$\begin{gathered}
\hat{A}_0 = 3.141593 \quad \hat{A}_1 = 2.094395 \quad \hat{A}_2 = 0.785398 \\
\hat{A}_3 = 0 \quad \hat{A}_4 = -0.130900 \quad \hat{A}_5 = 0 \quad \hat{A}_6 = 0.049087
\end{gathered} \tag{9}$$

Approximation: For rendering, the key observation is that $\hat{A}_l$ decays so fast that we need consider only low-frequency lighting coefficients, of order $l \leq 2$. Equivalently, **the irradiance is well approximated by only 9 parameters**—1 for $l = 0, m = 0$, 3 for $l = 1, -1 \leq m \leq 1$, and 5 for $l = 2, -2 \leq m \leq 2$. By working in frequency-space, we exploit the low-frequency character of $A = (\mathbf{n}\cdot\omega)$, using a few coefficients instead of a full hemispherical integral. The simple form of the first 9 spherical harmonics, given in equation 3, makes implementation straightforward.

3 Algorithms and Results

In this section, we discuss three applications of this result. First, we show how to rapidly prefilter the lighting distribution, computing the coefficients L_{lm}. Next, we develop a simple real-time procedural shader for rendering that takes these coefficients as inputs. Finally, we discuss other applications of our representation.

3.1 Prefiltering

For a given environment map, we first find the 9 lighting coefficients, L_{lm} for $l \leq 2$, by integrating against the spherical harmonic basis functions. Each color channel is treated separately, so the coefficients can be thought of as RGB values.

$$L_{lm} = \int_{\theta=0}^{\pi}\int_{\phi=0}^{2\pi} L(\theta,\phi) Y_{lm}(\theta,\phi)\sin\theta\, d\theta d\phi \tag{10}$$

The expressions for the Y_{lm} are found in equation 3. The integrals are simply sums of the pixels in the environment map L, weighted by the functions Y_{lm}. The integrals can also be viewed as moments of the lighting, or as inner-products of the functions L and Y_{lm}.

Since we compute 9 numbers, the prefiltering step takes O(9S) time, where S is the size (total number of pixels) of the environment map. By comparison, the standard method of computing an irradiance environment map texture takes $O(T \cdot S)$ time, where T is the number of texels in the irradiance environment map. Our method will therefore be approximately $T/9$ times faster[1]. Even if a conventional irradiance environment map is computed at a very low resolution of 64×64, corresponding to $T = 4096$, our method will be nearly 500 times faster.

We have implemented prefiltering as a preprocessing step for a given environment map. Values of L_{lm} for a few light probes are tabulated in figure 2. The computation time for a 300x300 environment map was less than a second. This indicates that our approach might be able to handle scenes with dynamic lighting in the future. By contrast, the standard method of performing a hemispherical integral for each pixel to compute the irradiance environment map took approximately two hours. In fact, if an explicit representation of the irradiance environment map texture is required, we believe

[1] It may be possible to use a hierarchical integration scheme, as demonstrated by Kautz et al. [11] for Phong BRDFs, to speed up both our method and the conventional approach. Hardware acceleration may also be possible.

	Grace Cathedral			Eucalyptus Grove			St. Peters Basilica		
L_{00}	.79	.44	.54	.38	.43	.45	.36	.26	.23
L_{1-1}	.39	.35	.60	.29	.36	.41	.18	.14	.13
L_{10}	-.34	-.18	-.27	.04	.03	.01	-.02	-.01	-.00
L_{11}	-.29	-.06	.01	-.10	-.10	-.09	.03	.02	.01
L_{2-2}	-.11	-.05	-.12	-.06	-.06	-.04	.02	.01	.00
L_{2-1}	-.26	-.22	-.47	.01	-.01	-.05	-.05	-.03	-.01
L_{20}	-.16	-.09	-.15	-.09	-.13	-.15	-.09	-.08	-.07
L_{21}	.56	.21	.14	-.06	-.05	-.04	.01	.00	.00
L_{22}	.21	-.05	-.30	.02	-.00	-.05	-.08	-.06	.00

Figure 2: *RGB values of lighting coefficients for a few environments. These may be used directly for rendering, and for checking the correctness of implementations. Note that L_{1-1}, corresponding to the linear moment along the y-axis or vertical direction, is relatively large and positive for all environments. This is because the upper hemisphere is significantly brighter than the lower hemisphere, owing to skylight or ceiling lamps. For the Eucalyptus grove, where the lighting is almost symmetric about the y-axis, the other moments are relatively small. Therefore, for that environment, the most significant lighting coefficients are the constant (ambient) term L_{00}, and L_{1-1}.*

the best way of computing it is to first compute the 9 coefficients L_{lm} using our method, and then use these to very rapidly generate the irradiance environment map using the rendering method described below.

It is important to know what errors result from our 9 parameter approximation. The maximum error for any pixel, as a fraction of the total intensity of the illumination, is 9% and corresponds to the maximum error in the order 2 approximation of $A(\theta)$. Furthermore, the average error over all surface orientations can be shown to be under 3% for any physical input lighting distribution [2]. For the environment maps used in our examples, corresponding to complex natural illumination, the results are somewhat better than the worst-case bounds—the average error is under 1%, and the maximum pixel error is under 5%. Finally, figure 3 provides a visual comparison of the quality of our results with standard prefiltering, showing that our method produces a perceptually accurate answer.

3.2 Rendering

For rendering, we can find the irradiance using equation 7. Since we are only considering $l \leq 2$, the irradiance is simply a quadratic polynomial of the coordinates of the (normalized) surface normal. Hence, with $\mathbf{n}^t = (x\ y\ z\ 1)$, we can write

$$E(\mathbf{n}) = \mathbf{n}^t M \mathbf{n} \tag{11}$$

M is a symmetric 4x4 matrix. Each color has an independent matrix M. Equation 11 is particularly useful for rendering, since we require only a matrix-vector multiplication and a dot-product to compute E. The matrix M is obtained by expanding equation 7:

$$M = \begin{pmatrix} c_1 L_{22} & c_1 L_{2-2} & c_1 L_{21} & c_2 L_{11} \\ c_1 L_{2-2} & -c_1 L_{22} & c_1 L_{2-1} & c_2 L_{1-1} \\ c_1 L_{21} & c_1 L_{2-1} & c_3 L_{20} & c_2 L_{10} \\ c_2 L_{11} & c_2 L_{1-1} & c_2 L_{10} & c_4 L_{00} - c_5 L_{20} \end{pmatrix}$$
$$c_1 = 0.429043 \quad c_2 = 0.511664$$
$$c_3 = 0.743125 \quad c_4 = 0.886227 \quad c_5 = 0.247708 \tag{12}$$

The entries of M depend[2] on the 9 lighting coefficients L_{lm} and the expressions for the spherical harmonics. The constants come from the numerical values of $\hat{A}_l$ given in equation 9, and the spherical harmonic normalizations given in equation 3.

On systems not optimized for matrix and vector operations, it may be more efficient to explicitly write out equation 7 for the irradiance as a sum of terms, i.e. expand equation 12:

$$\begin{aligned} E(\mathbf{n}) &= c_1 L_{22}\left(x^2 - y^2\right) + c_3 L_{20} z^2 + c_4 L_{00} - c_5 L_{20} \\ &+ 2c_1\left(L_{2-2}xy + L_{21}xz + L_{2-1}yz\right) \\ &+ 2c_2\left(L_{11}x + L_{1-1}y + L_{10}z\right) \end{aligned} \tag{13}$$

We implemented equations 11 and 13 as procedural shaders in the Stanford real-time programmable shading system [15]. We used the ability of that system to perform computations per-vertex. Since E varies slowly, this is adequate and the shading is insensitive to how finely the surfaces are tessellated. The irradiance computations may be performed in software or compiled to vertex programming hardware, if available. The simple forms of equations 11 and 13 indicate that a per-fragment method could also be implemented in programmable hardware.

We were able to achieve real-time frame rates on PCs and SGIs. As shown in the accompanying video—available on the SIGGRAPH 2001 conference proceedings videotape—we can interactively rotate objects and move our viewpoint, with the irradiance being procedurally recomputed at every frame. We can also rotate the lighting by applying the inverse rotation to the normal $\mathbf{n}$. Images rendered using our method look identical to those obtained by texture-mapping after precomputing irradiance environment maps.

[2]A symmetric 4x4 matrix has 10 degrees of freedom. One additional degree is removed since $\mathbf{n}$ lies on the unit sphere.

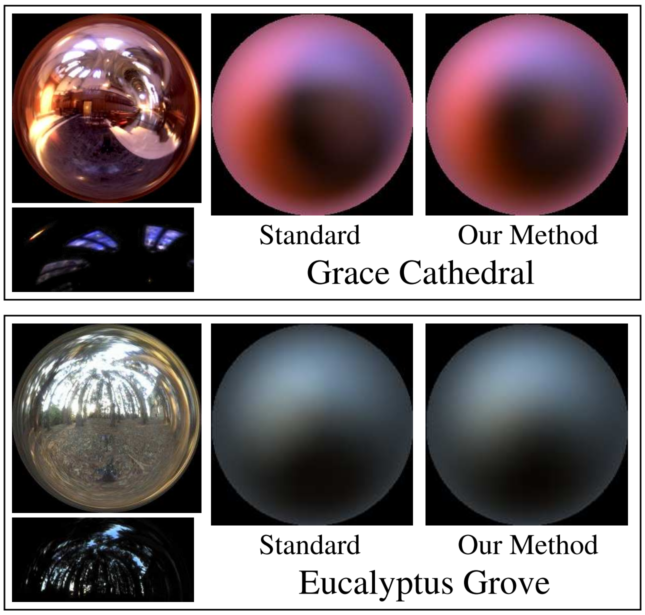

Figure 3: *A comparison of irradiance maps from our method to those from standard prefiltering. The irradiance map resolutions are 256x256. For each light probe, the left image is a tone-mapped version of the environment. Below that, we show the brightest parts of the environment on a linear scale. Both environments have bright bluish lights—from stained-glass windows, and the sky respectively—which are not apparent in the tone-mapped images. This accounts for the bluish portions of the irradiance maps. It can be seen that our method produces a result very close to the correct answer. Note that our rendering algorithm does not actually use irradiance maps; we computed them here solely for the purposes of the quality comparison. The coordinate mapping in the images is such that the center of the image is straight forward ($\theta = 0$, the north pole or +Z), the circumference of the image is straight backwards ($\theta = \pi$, the south pole or -Z), and θ varies uniformly in the radial direction from 0 to π. The azimuthal angle ϕ corresponds to the image polar angle.*

3.3 Representation

Conceptually, the final image is composed of a sum of spherical harmonic basis functions, scaled by the lighting coefficients L_{lm}. These 3D irradiance basis functions depend on the surface normal and are defined over the entire object, making it possible to generate an image from any viewpoint. We may also manually adjust the 9 lighting coefficients L_{lm} to directly control appearance, as shown in figure 4. The lighting coefficients can often be assigned intuitive meanings. For instance, L_{1-1} is the moment about the vertical or y-axis, and measures the extent to which the upper hemisphere is brighter than the lower hemisphere. As can be seen from figure 2, L_{1-1} is usually large and positive, since most scenes are lit from above. By making this value negative, we could give the appearance of the object being lit from below.

Our representation may also be useful in the future for *image-based rendering with varying illumination*. Hallinan [10] and Epstein et al. [7] have observed empirically that, for a given view, images of a matte object under variable lighting lie in a low-

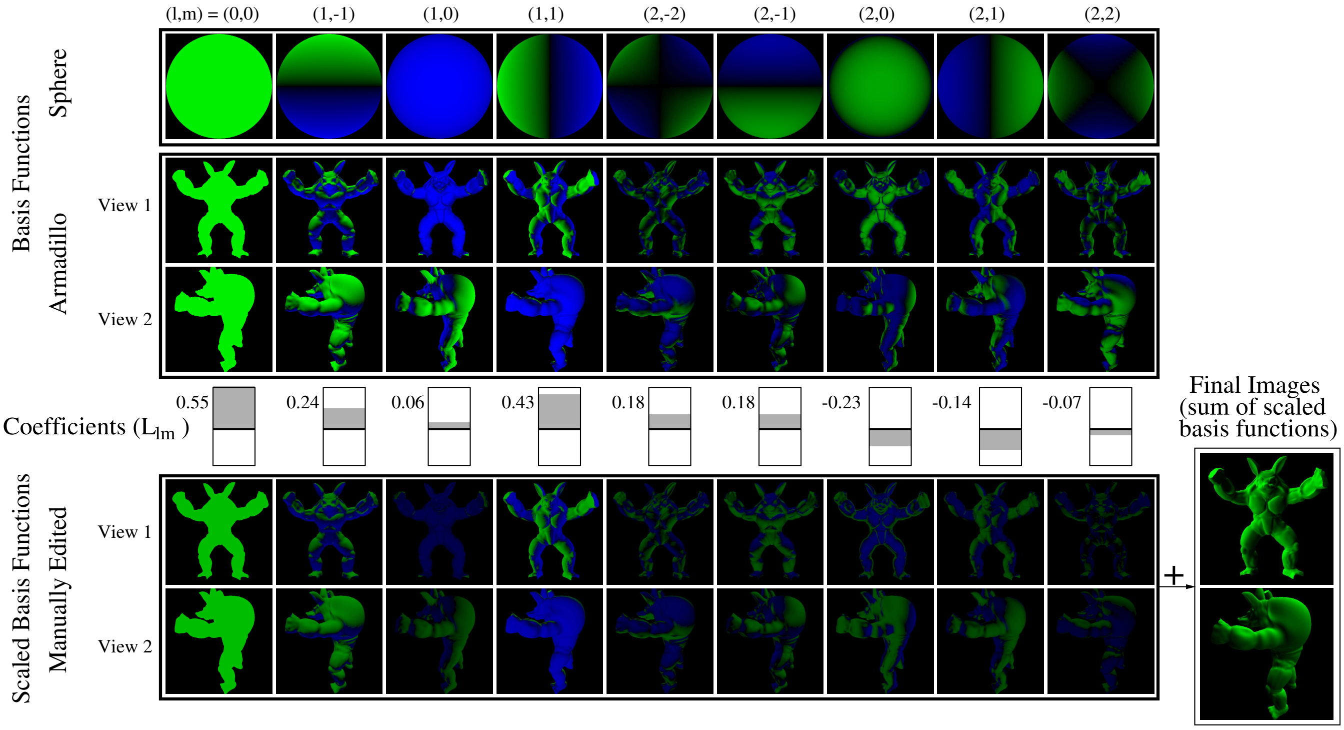

Figure 4: *Illustration of our new representation, and applications to controlling appearance. The basis functions have both positive values, shown in green, and negative values, shown in blue. Topmost, we show the spherical harmonic basis functions on a sphere—note that these are actual images, not the coordinate mappings of figure 3—and the armadillo. The basis functions are defined over the entire object surface; we show only two views. The rightmost 5 functions are dimmer since they have the highest frequency* $(l = 2)$ *and contribute the least. Conceptually, the basis functions are then scaled by the lighting coefficients* L_{lm} *and added to produce renderings.* L_{lm} *are actually RGB values; for simplicity, we show the coefficients for only one color (green). The coefficients* L_{lm} *may be adjusted manually to manipulate appearance. This editing can be fairly intuitive—for instance, we make* L_{11} *large and positive to darken the right side (with respect to us) and left arm of the armadillo image, since the basis function* $(1, 1)$ *is negative in that region.*

dimensional subspace. Our theory explains this observation, and indicates that a 9D subspace suffices. Basri and Jacobs [2] have obtained similar theoretical results. To synthesize images of a diffuse object under arbitrary illumination, we therefore need only the 9 basis functions, which could be computed from a small number of photographs. Such an approach would significantly speed up both acquisition and rendering in a method such as Debevec et al. [6].

4 Conclusions and Future Work

We have described a novel analytic representation for environment maps used to render diffuse objects, and have given explicit formulae for implementation. Our approach allows us to use an arbitrary illumination distribution for the diffuse component of the BRDF, instead of the limitation of current graphics hardware to point or directional sources. We simply specify or compute the first 9 moments of the lighting. Even where more conventional texture-mapping methods are desired, our approach allows us to very efficiently compute irradiance environment map textures. In the future, we wish to develop similar frequency-space methods for the specular BRDF component, and more general non-Lambertian BRDFs. We would also like to further explore the applications to lighting design and image-based rendering discussed above.

Acknowledgements: We thank Kekoa Proudfoot and Bill Mark for explaining the details of the Stanford Real-Time Programmable Shading system, and helping us implement the algorithms within its framework. Thanks also to all those who read early drafts of the paper—Bill Mark, Kekoa Proudfoot, Sean Anderson, David Koller, Ron Dror, and the anonymous reviewers. Finally, we thank Venkat Krishnamurthy for providing the armadillo model and Paul Debevec for allowing us to use his light probes (available from http://www.debevec.org). This work was supported in part by a Hodgson-Reed Stanford graduate fellowship and NSF ITR grant #0085864 "Interacting with the Visual World."

References

[1] J. Arvo. Applications of irradiance tensors to the simulation of non-lambertian phenomena. In *SIGGRAPH 95*, pages 335–342, 1995.

[2] R. Basri and D. Jacobs. Lambertian reflectance and linear subspaces. In *International Conference on Computer Vision*, 2001.

[3] J. F. Blinn and M. E. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19:542–546, 1976.

[4] B. Cabral, N. Max, and R. Springmeyer. Bidirectional reflection functions from surface bump maps. In *SIGGRAPH 87*, pages 273–281, 1987.

[5] B. Cabral, M. Olano, and P. Nemec. Reflection space image based rendering. In *SIGGRAPH 99*, pages 165–170, 1999.

[6] P. Debevec, T. Hawkins, C. Tchou, H.P. Duiker, W. Sarokin, and M. Sagar. Acquiring the reflectance field of a human face. In *SIGGRAPH 00*, pages 145–156.

[7] R. Epstein, P.W. Hallinan, and A. Yuille. 5 plus or minus 2 eigenimages suffice: An empirical investigation of low-dimensional lighting models. In *IEEE Workshop on Physics-Based Modeling in Computer Vision*, pages 108–116, 1995.

[8] N. Greene. Environment mapping and other applications of world projections. *IEEE Computer Graphics & Applications*, 6(11):21–29, 1986.

[9] G. Greger, P. Shirley, P. Hubbard, and D. Greenberg. The irradiance volume. *IEEE Computer Graphics & Applications*, 18(2):32–43, 1998.

[10] P.W. Hallinan. A low-dimensional representation of human faces for arbitrary lighting conditions. In *CVPR 94*, pages 995–999, 1994.

[11] J. Kautz, P. Vázquez, W. Heidrich, and H.P. Seidel. A unified approach to prefiltered environment maps. In *EuroGraphics Rendering Workshop 00*, pages 185–196, 2000.

[12] P. Lalonde and A. Fournier. Filtered local shading in the wavelet domain. In *EGRW 97*, pages 163–174, 1997.

[13] T.M. MacRobert. *Spherical harmonics; an elementary treatise on harmonic functions, with applications*. Dover Publications, 1948.

[14] G. Miller and C. Hoffman. Illumination and reflection maps: Simulated objects in simulated and real environments. *SIGGRAPH 84 Advanced Computer Graphics Animation seminar notes*, 1984.

[15] K. Proudfoot, W. Mark, S. Tzvetkov, and P. Hanrahan. A real-time procedural shading system for programmable graphics hardware. In *SIGGRAPH 01*, 2001.

[16] R. Ramamoorthi and P. Hanrahan. On the relationship between radiance and irradiance: Determining the illumination from images of a convex lambertian object. *To appear, Journal of the Optical Society of America A*, 2001.

[17] F. X. Sillion, J. Arvo, S. H. Westin, and D. Greenberg. A global illumination solution for general reflectance distributions. In *SIGGRAPH 91*, pages 187–196.

[18] G. Ward and P. Heckbert. Irradiance gradients. In *EGRW 92*, pages 85–98, 1992.

[19] A. Wilkie, R. Tobler, and W. Purgathofer. Orientation lightmaps for photon radiosity in complex environments. In *CGI 00*, pages 279–286, 2000.