Algorithms for Spherical Harmonic Lighting

Ian G. Lisle* S.-L. Tracy Huang†
School of Information Sciences & Engineering
University of Canberra

Abstract

Spherical harmonic (SH) lighting models require efficient and general libraries for evaluation of SH functions and of Wigner matrices for rotation. We introduce an efficient algebraic recurrence for evaluation of SH functions, and also implement SH rotation via Wigner matrices constructed for the real SH basis by a recurrence. Using these algorithms, we provide a freely distributable C / OpenGL implementation for SH diffuse unshadowed, shadowed and interreflected models. Our implementation allows flexible switching of scene, light probe, SH degree and lighting model at run time.

CR Categories: I.3.7 [Computing Methodologies]: Computer Graphics—Three-Dimensional Graphics and Realism; I.3.6 [Computing Methodologies]: Computer Graphics—Methodology and Techniques

Keywords: spherical harmonic lighting, Wigner matrix, rotation, recurrence

1 Introduction

Spherical harmonic (SH) functions have found increasing use in computer graphics over the last decade, in lighting, BRDF, shape recognition and other areas. Most notably, precomputed radiance transfer (PRT) methods for lighting [Kautz et al. 2002; Sloan et al. 2002] make heavy use of spherical harmonics. In outline, PRT methods work as follows [Sloan et al. 2002; Green 2003]:

1. In a preprocessing pass, expand the lighting environment L_f, scene geometry / visibility VG and (possibly) BRDF in SH series. Ray tracing is required to expand VG. SH coefficients of VG are recorded at each vertex.
2. At runtime, rotation of the scene is transferred into the SH domain by computation of Wigner matrices once per frame. The Wigner matrix is applied to rotate the lighting SH coefficients.
3. At each vertex and each frame, accumulation of a dot product of SH coefficients of VG with those of the rotated light environment L_f gives an approximation to scene lighting.

This paper is the result of building a C/OpenGL implementation of PRT methods with our 'bare hands'. As our implementation evolved, it became apparent that a significant part of the effort had to be directed not at the graphics, but at underlying mathematical issues. PRT methods are implemented in DirectX, but only an API is available: the associated papers [Kautz et al. 2002; Sloan et al. 2002] are rather dense, provide no source code and only sketch the algorithms. Green [2003] in his *Gritty Details* paper undertook to fill the gaps, providing C source for many aspects of the method. However, as a tutorial, the paper presents code for SH function evaluation that is extremely inefficient. Also, the most mathematically challenging part of PRT methods is Wigner rotation, but the discussion in [Green 2003] is incomplete, with no source code.

As a consequence, any researcher seeking to work in the area of PRT or other SH methods must first battle through some unfamiliar advanced mathematics to write a library of functions that implements SH evaluation, Wigner matrix construction, and rotation of SH coefficients. Wigner matrices are well enough known in areas such as quantum mechanics [Edmonds 1957], but transfer of these results to graphics is hampered by a number of factors.

First, physicists customarily use a complex valued SH basis which leads to formal convenience but is not well suited to the numerical demands of graphics calculations. Secondly, there is a variety of conventions for choice of phase and normalisation of SH functions (see for instance the discussion of SH conventions on the SHTools site `http://www.ipgp.jussieu.fr/~wieczor/SHTOOLS/SHTOOLS.html`). Finally, it can be difficult to disentangle mathematical and algorithmic issues from the physics of the originating field. The result of all this is that transfer of SH results from other fields is confusing and time consuming.

In this paper we present detailed algorithms for calculating SH functions and Wigner matrices, and methods for expanding functions that occur in lighting. By collecting these results here we hope to limit further reinventions of the wheel on these matters.

2 Algorithms

2.1 Spherical Harmonics

Spherical harmonic functions are eigenfunctions of the Laplace operator on the sphere S^2 [Bleecker and Csordas 1992].

SH Basis A convenient real-valued basis for SH functions is [Green 2003] for each $l = 0, 1, 2, \dots$

$$Y_{l,0}(\theta, \phi) = P_l(\cos\theta)$$

and for $m = 1, 2, \dots, l$

$$\begin{aligned} Y_{l,m}(\theta, \phi) &= \cos m\phi \, P_l^m(\cos\theta) \\ Y_{l,-m}(\theta, \phi) &= \sin m\phi \, P_l^m(\cos\theta) \end{aligned} \tag{1}$$

where P_l is a Legendre polynomial, P_l^m are the associated Legendre functions (and $P_l^0 \equiv P_l$) [Abramowitz and Stegun 1964], and (θ, ϕ) are colatitude and azimuth spherical coordinates. These spherical harmonic functions Y_{lm} are unnormalised; we denote the corresponding normalised SH functions by y_{lm}:

$$y_{lm} = K_{lm} Y_{lm}; \qquad \text{such that } \int_{S^2} y_{lm}^2 = 1 \tag{2}$$

The normalising constants K_{lm} [Green 2003, p.12] are found by a simple recurrence.

*e-mail: Ian.Lisle@canberra.edu.au
†e-mail:Tracy.Huang@canberra.edu.au

GRAPHITE 2007, Perth, Western Australia, December 1–4, 2007.
© 2007 ACM 978-1-59593-912-8/07/0012 $5.00

Algebraic SH Recurrence Because SH functions are so heavily used in computer graphics now, it is important to have an efficient evaluation procedure for them. Various high quality SH libraries implementing SH functions exist, but inconveniently use a complex SH basis

$$\hat{y}_{lm}(\theta,\phi) = N_{lm} P_l^m(\cos\theta) e^{im\phi} \tag{3}$$

For the real basis (1), a recurrence for P_l^m based on *Numerical Recipes* [Press et al. 1992] is found in e.g. the *Gritty Details* [Green 2003], although the implementation there is not particularly efficient. (For instance it is better to generate all P_l^m at once than to evaluate them one at a time.) The P_l^m recurrence is algebraic (i.e. polynomial) in the quantities $\cos\theta, \sin\theta$, so involves only multiplication and addition. The trigonometric evaluations $\cos m\phi, \sin m\phi$ apparently needed in (1) can similarly be generated by an algebraic recurrence from $\cos\phi, \sin\phi$ [Blanco et al. 1997].

However, instead of specifying a point $\mathbf{u} \in S^2$ in spherical polars (θ,ϕ), it is more convenient to give $\mathbf{u} = (x, y, z)$ as a unit vector. (For instance ray directions will already be in this form.) SH theory [Bleecker and Csordas 1992] shows that spherical harmonics Y_{lm} are homogeneous polynomials of degree l in (x, y, z). By combining recurrences eqs.(9–12) and (13–14) of Blanco et al. [1997] one obtains a purely algebraic recurrence for Y_{lm}:

BASE OF RECURRENCE:

$$Y_{0,0} = 1, \qquad Y_{1,-1} = -y, \quad Y_{1,1} = -x, \quad Y_{1,0} = z$$

EDGES: For $l = 2, 3, \ldots$

$$\begin{aligned} Y_{l,l} &= (2l-1)\big(-xY_{l-1,l-1} + yY_{l-1,-(l-1)}\big) \\ Y_{l,-l} &= (2l-1)\big(-yY_{l-1,l-1} - xY_{l-1,-(l-1)}\big) \end{aligned}$$

INTERIOR: For $l = 2, 3, \ldots$ and $m = 0, 1, 2, \ldots, l-1$

$$\begin{aligned} Y_{l,m} &= \frac{1}{l-m}\big((2l-1)\, zY_{l-1,m} - (l+m-1)\, Y_{l-2,m}\big) \\ Y_{l,-m} &= \frac{1}{l-m}\big((2l-1)\, zY_{l-1,-m} - (l+m-1)\, Y_{l-2,-m}\big) \end{aligned}$$

For $m = l-1$, invalid terms "$Y_{l-2,\pm(l-1)}$" appear in the recurrence: these terms are to be set to 0. This recurrence avoids conversion to spherical polar coordinates by computing directly in terms of (x, y, z): there are no trigonometric evaluations. Normalisation is done as a final step using the precomputed factors K_{lm} (2).

Data Structure There are $(l+1)^2$ spherical harmonics of degree up to l, and they can be stored in an array of floats, with $Y_{l,m}$ located at position $l(l+1)+m$ in the array. However, to implement the recurrence of §2.1 one would like to use the syntax `Y[l][m]` – even for m negative. Modifying a common trick for multidimensional arrays [Oliveira and Stewart 2006, §8.3], this is neatly achieved in C by defining a data type:

```
typedef struct {
   GLfloat **Y;    GLfloat *YValues;
} ShCoeffs;
```

The array of $(l+1)^2$ floats in the `YValues` field holds SH values or coefficients as per [Green 2003], while the array of $(l+1)$ pointers is initialised so that `Y[l]` points to the value Y_{l0}.

2.2 Rotations: Wigner Matrices

A remarkable feature of PRT methods is that a scene can be rotated relative to the lighting environment at run time. For a very modest computational cost one gets soft shadows that adjust dynamically to changed lighting. To perform this rotation requires computation of 'Wigner matrices' relative to the chosen SH basis. Wigner matrices correspond to the odd-dimensional irreducible representations of the rotation group $SO(3)$. If $R \in SO(3)$ and if f is a square integrable function on the sphere S^2, the left regular representation of $SO(3)$ is the action $(R \cdot f)(\mathbf{u}) = f\big(R^{-1}(\mathbf{u})\big)$. Invariance of the l-th SH subspace implies the existence of coefficients $D^l_{mn}(R)$ such that

$$R \cdot y_{lm} = \sum_{n=-l}^{l} D^l_{nm}(R)\, y_{ln} \tag{4}$$

For a given l, the entries D^l_{mn} form a $(2l+1)\times(2l+1)$ orthogonal matrix; the problem is to construct D^l_{mn} from R.

Let f have SH coefficients $\{a_{lm}\}$. Then if $R \cdot f$ has SH coefficients $\{a'_{lm}\}$, the Wigner matrix entries connect the two via

$$a'_{lm} = \sum_{n=-l}^{l} D^l_{mn}(R)\, a_{ln} \tag{5}$$

By applying this rotation to SH coefficients per frame at draw time, a scene can be rotated relative to its lighting environment.

Wigner matrix entries $D^l_{mn}(R)$ can be calculated in many ways. They depend on the choice of SH basis, so the comments in §1 about confusing basis conventions still apply. The confusion is worsened by two further factors. First, an algorithm for $D^l_{mn}(R)$ must choose a parametrisation of the rotation group $SO(3)$: via a matrix, or a quaternion, or one of the many versions of Euler angles. Secondly, there are different choices for the (l, m, n) to be used in a recurrence for $D^l_{mn}(R)$.

Recurrence for Wigner Matrices Mathematical complexity is a deterrent in calculating D^l_{mn}, and and there does not appear to be a detailed code in the public domain for carrying through the computations with respect to the basis (1). Ivanic and Ruedenberg [1996] gave a recurrence for Wigner matrix entries in the real spherical harmonic basis, while Choi et al. [1999] use a recurrence based on the complex basis (3), but the method below uses a different recurrence. Sloan, et al. [2002] describe the method in outline, but do not provide detailed formulas. Green [2003] constructs some Wigner matrices explicitly but his method is not general and cannot easily go to arbitrary SH degree.

Below we provide a detailed solution by a method that permits calculation of Wigner matrix elements $D^l_{mn}(R)$ to arbitrary degree l. The input to the algorithm is a rotation matrix $R \in SO(3)$.

Step 1. *Extract Z-Y-Z Euler angles from the rotation matrix R.* A rotation R is decomposed as $R = R_z(\gamma) R_y(\beta) R_z(\alpha)$. The Wigner matrices $D^l(R)$ can then constructed by

$$D^l(R) = D^l\big(R_z(\gamma)\big)\, D^l\big(R_y(\beta)\big)\, D^l\big(R_z(\alpha)\big) \tag{6}$$

Step 2. *Wigner matrix for Y-rotation in real SH basis.*

This is by far the most difficult step. For a Y-rotation through angle β we shall use $d^l_{mn}(\beta) = D^l_{mn}\big(R_y(\beta)\big)$ to denote Wigner matrix entries with respect to real SH basis (1). Based on the complex recurrence in [Kostelec and Rockmore 2003] we derive a real basis recurrence for $d^l_{mn}(\beta)$:

BASE:

$$d^0 = [1], \qquad d^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix}$$

EDGES: For $l = 2, 3, \ldots$

$$d^l_{l,l} = \tfrac{1}{2}\cos\beta\, d^{l-1}_{l-1,l-1} + \tfrac{1}{2}\, d^{l-1}_{-l+1,-l+1}$$
$$d^l_{-l,-l} = \tfrac{1}{2}\, d^{l-1}_{l-1,l-1} + \tfrac{1}{2}\cos\beta\, d^{l-1}_{-l+1,-l+1}$$

and for $m = 0, \ldots, l-1$

$$d^l_{m,l} = \sqrt{\tfrac{l(l-1/2)}{l^2-m^2}}\ \sin\beta\, d^{l-1}_{m,l-1}$$
$$d^l_{-m,-l} = \sqrt{\tfrac{l(l-1/2)}{l^2-m^2}}\ \sin\beta\, d^{l-1}_{-m,-l+1}$$

INTERIOR: For $l = 2, 3, \ldots$ and for $m, n = 0, \ldots, l-1$

$$d^l_{m,n} = \tfrac{l(2l-1)}{\sqrt{(l^2-m^2)(l^2-n^2)}} \left(\cos\beta\, d^{l-1}_{m,n} - \tfrac{mn}{l(l-1)}\, d^{l-1}_{-m,-n} - \tfrac{\sqrt{((l-1)^2-m^2)((l-1)^2-n^2)}}{(l-1)(2l-1)}\, d^{l-2}_{m,n}\right)$$
$$d^l_{-m,-n} = \tfrac{l(2l-1)}{\sqrt{(l^2-m^2)(l^2-n^2)}} \left(\tfrac{-mn}{l(l-1)}\, d^{l-1}_{m,n} + \cos\beta\, d^{l-1}_{-m,-n} - \tfrac{\sqrt{((l-1)^2-m^2)((l-1)^2-n^2)}}{(l-1)(2l-1)}\, d^{l-2}_{m,n}\right)$$

The elements $d^l_{-m,n}$ are zero for $m = 1, \ldots, l$, $n = 0, \ldots, l$, as are $d^l_{m,-n}$ for $m = 0, \ldots, l$, $n = 1, \ldots, l$. Some computation is also saved by using the symmetry $d^l_{n,m} = (-1)^{m-n} d^l_{m,n}$.

Step 3. *Compose with Wigner matrices for Z-rotations*

Once $d^l_{mn}(\beta)$ are known they can be pre- and post-multiplied by Z-rotation Wigners as described by Green [2003] to finally give the values $D^l_{mn}(R)$ (6).

The above method allows generation of d^l_{mn} to arbitrary degree l. The method is reasonably efficient, but further optimisation should be possible by eliminating the use of Euler angles. In the first instance PRT [Sloan et al. 2002] treats the lighting environment to be infinitely distant, so that it is the same at each point in the scene. In this case, rotation need be done to the SH lighting coefficients a_{lm} once per frame, so efficiency is not a pressing issue.

Data Structure To code the above recurrence we build on the idea of the SH data structure of §2.1 and define a data type

```
typedef struct {
  float ***D;   float **DL;   float *DValues;
} WignerMatrix;
```

If `w` is a `WignerMatrix`, then with a suitable constructor function, `w.D[l][m][n]` is the value D^l_{mn} associated with `w`. The `DValues` field is an array of floats holding the Wigner matrix entries; the `DL` field is an array of pointers to the rows of the Wigner matrices; and the `D` field is an array of pointers-to-pointers such that `D[l]` can be thought of as the l-th Wigner matrix.

3 Circularly Symmetric Function

A function f on the sphere S^2 is SH-expanded $f \sim \sum_{l,m} a_{lm} y_{lm}$ by evaluating integrals over the sphere: $a_{lm} = \int_{S^2} f y_{lm}$. A commonly arising case is where f is circularly symmetric about axis $\mathbf{n}$. In this case one can be more explicit. First we state

Theorem 3.1 (The Addition Theorem)**.** *[Baylis 1999, p.328]: Let $\{y_{lm}\}$ be normalised spherical harmonic basis functions, and let $\mathbf{u}, \mathbf{n} \in S^2$ be points on the sphere (unit vectors). Then*

$$P_l(\mathbf{u}\cdot\mathbf{n}) = \frac{4\pi}{2l+1} \sum_{m=-l}^{l} y_{lm}(\mathbf{u}) y_{lm}(\mathbf{n})$$

It is then straightforward to show the following:

Proposition 3.2. *Let f be a function on the sphere that is circularly symmetric about axis $\mathbf{n}$. Then*

(i) There is a function $h : [-1,1] \to \mathbb{R}$ such that $f(\mathbf{u}) = h(\mathbf{n}\cdot\mathbf{u})$

(ii) The SH coefficients b_{lm} of $f \sim \sum_{l,m} b_{lm} y_{lm}$ are

$$b_{lm} = 2\pi \int_{-1}^{1} h(z) P_l(z)\, dz\ y_{lm}(\mathbf{n}) \tag{7}$$

The proof of (ii) is by expanding the function h of (i) in a series of Legendre polynomials P_l then applying the Addition Theorem 3.1.

3.1 Application to Diffuse Unshadowed Lighting

The diffuse unshadowed SH lighting model developed by Ramamoorthi and Hanrahan [2001a] is a local illumination model requiring only surface normals $\mathbf{n}$ to compute shading from a lighting environment. Dot product lighting is mediated by the circularly symmetric 'geometry' function $G : S^2 \to \mathbb{R}$ defined by

$$G(\mathbf{u}) = H(\mathbf{u}\cdot\mathbf{n}) \quad \text{where } H(z) = \begin{cases} z & 0 \le z \le 1 \\ 0 & -1 \le z \le 0 \end{cases} \tag{8}$$

Its SH expansion $G \sim \sum_{l,m} b_{lm} y_{lm}$ follows immediately from Proposition 3.2 as

$$b_{lm} = \pi H_l\, y_{lm}(\mathbf{n}), \quad \text{where } H_l = 2\int_0^1 z P_l(z)\, dz \tag{9}$$

The coefficients H_l were given by Ramamoorthi and Hanrahan [2001b], who showed that $H_0 = 1$, $H_1 = \frac{2}{3}$ and that for $l \ge 3$ odd the H_l vanish. Their eq.(8) gives a formula for H_{2n}, but these but are more conveniently computed by Abramowitz and Stegun [1964, 22.13.8]

$$H_{2n} = \frac{(-1)^n \Gamma(n-\frac{1}{2})}{\Gamma(-\frac{1}{2})\Gamma(n+2)}$$

The H_{2n} can then be evaluated once and for all from the recurrence

$$H_0 = 1, \qquad H_{2n+2} = \frac{\frac{1}{2}-n}{2+n} H_{2n} \tag{10}$$

3.2 Application to Circular Light Source

Another application of Proposition 3.2 is to SH expanding a circular light window – that is a circular patch on the sphere of radius ζ radians that emits light of constant intensity. Such light sources provide good test environments and can exhibit various artifacts associated with SH lighting. Let the light function L_f be centred on vector $\mathbf{n}$ and chosen so that the irradiance from L_f is a constant 4π for all apertures ζ. Then

$$L_f(\mathbf{u}) = h(\mathbf{u}\cdot\mathbf{n}), \qquad \text{where } h(z) = \begin{cases} \frac{2}{1-\cos\zeta} & \text{if } z > \cos\zeta \\ 0 & \text{otherwise} \end{cases}$$

If the SH expansion of L_f is $L_f \sim \sum_{l,m} a_{lm} y_{lm}$, then applying Proposition 3.2 shows that

$$a_{lm} = \frac{4\pi}{1-\cos\zeta} \int_{\cos\zeta}^{1} P_l(z)\, dz\, y_{lm}(\mathbf{n})$$

Letting $f_l(z) = \frac{1}{1-z}\int_z^1 P_l(t)dt$ and applying the formulas of [Abramowitz and Stegun 1964, ch.22], we derive the recurrence

$$f_0(z) = 1, \qquad f_1(z) = \tfrac{1}{2}(1+z),$$
$$f_{l+1}(z) = \frac{1}{l+2}\big((2l+1)\, z f_l(z) - (l-1) f_{l-1}(z)\big) \qquad (11)$$

This gives the SH expansion of such a light window with very little effort – in particular without ray sampling.

4 Implementation

We have implemented three diffuse lighting models in the PRT framework: unshadowed, shadowed and interreflected transfer. Our code uses C and OpenGL, so can build on a variety of platforms. Because of the number of vertex attributes required, our implementation uses CPU for lighting calculations. (We also implemented unshadowed SH lighting in GLSL using the method of §3.1.) The code flexibly demonstrates the attributes of SH diffuse lighting models: changing lighting model, scene, light probe and SH degree at run time. As well as light probes we implement the circular light window of §3.2 with dynamically variable aperture.

4.1 Comparison

Because DirectX provides a 'black box' implementation of PRT it is difficult to find a complete implementation of PRT models in open source. Green [2003] gives the most complete details but does not provide code for SH rotation. Some SH lighting demonstrations (e.g. Dempski and Viale [2005]) do not even attempt rotation, negating one of the principal advantages of the PRT method. SH lighting is implemented in some student projects, but none has a general implementation of Wigner matrices. We make our source code publicly available (`http://ucspace.canberra.edu.au/display/SHLIGHT/`): we believe it to be the first publicly available general source for PRT SH rotation.

The diffuse unshadowed model is subject to some confusion in the literature. Ramamoorthi and Hanrahan [2001b] give the exact formula, but because Green's [2003] sample code calculated the SH coefficients by ray casting, this practice has been followed by other implementations. Such a calculation takes around 10 sec, yet the method of §3.1 can do it (exactly) in some 0.02 sec. Combined with the algebraic SH recurrence of §2.1, the calculation is fast enough to be done per vertex in a shader, as described by Ramamoorthi and Hanrahan [2001b] to SH degree 2. This method is not as efficient as rotating via a Wigner matrix, but is much simpler to code.

The algebraic recurrence of §2.1 is of interest independent of lighting calculations. It is easy to code, is much more efficient than the sample code from the *Gritty Details* [Green 2003], and is slightly faster than the method of [Blanco et al. 1997]. Benchmarking 10^6 SH evaluations to degree 6 on a 2.13 GHz Intel Core Duo 6400, the recurrence of §2.1 takes 0.234 sec, or 0.156 sec after unrolling loops. This compares favourably with `D3DXSHEvalDirection` at 0.172 sec. (The *Gritty Details* code takes over 10 sec.)

Acknowledgements

We used sample light probe images by Paul Debevec `http://www.debevec.org/probes`. Our light probe processing is based on Ravi Ramamoorthi's prefilter.c code.

References

ABRAMOWITZ, M., AND STEGUN, I. 1964. *Handbook of Mathematics Functions*. Dover Publications, Inc., New York.

BAYLIS, W. E. 1999. *Electrodynamics: A Modern Geometric Approach*. Birkhäuser, Boston.

BLANCO, M. A., FLÓREZ, M., AND BERMEJO, M. 1997. Evaluation of the rotation matrices in the basis of real spherical harmonics. *Journal of Molecular Structure (Theochem) 419*, 19–27.

BLEECKER, D., AND CSORDAS, G. 1992. *Basic Partial Differential Equations*. Van Nostrand Reinhold, New York.

CHOI, C. H., IVANIC, J., GORDON, M. S., AND RUEDENBERG, K. 1999. Rapid and stable determination of rotation matrices between spherical harmonics by direct recursion. *Journal of Chemical Physics 111*, 8825–8831.

DEMPSKI, K., AND VIALE, E. 2005. *Advanced Lighting and Materials with Shaders*. Wordware Publishing.

EDMONDS, A. R. 1957. *Angular Momentum in Quantum Mechanics*. Princeton University Press, Princeton, N.J.

GREEN, R. 2003. Spherical harmonic lighting: The gritty details. In *Game Developers' Conference*.

IVANIC, J., AND RUEDENBERG, K. 1996. Rotation matrices for real spherical harmonics. Direct determination by recursion. *Journal of Physical Chemistry 100*, 6342–6347.

KAUTZ, J., SLOAN, P.-P., AND SNYDER, J. 2002. Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In *proceedings of the 12th Eurographics Workshop on Rendering*, P. Debevec and S. Gibson, Eds., 301–308.

KOSTELEC, P. J., AND ROCKMORE, D. N. 2003. FFTs on the rotation group. Tech. Rep. 03-11-060, Santa Fe Institute.

OLIVEIRA, S., AND STEWART, D. 2006. *Writing Scientific Software: A guide to good style*. Cambridge University Press, Cambridge, UK.

PRESS, W., TEUKOLSKY, S., VETTERLING, W., AND FLANNERY, B. 1992. *Numerical Recipes in C: The art of scientific computing*, 2nd ed. Cambridge University Press, Cambridge, UK.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. On the relationship between radiance and irradiance: Determining the illumination from images of a convex Lambertian object. *Journal of the Optical Society of America A 18*, 2448–2459.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. An efficient representation for irradiance environment maps. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, New York, E. Fiume, Ed., Computer graphics Proceedings, Annual Conference Series, ACM, 497–500.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics 21*, 3, 527–536.