

# Pros and Cons of Spherical Harmonics-Based Dynamic Lighting in Indoor and Outdoor Environments

By Willem H. de Boer, January 2004

## Overview

Spherical Harmonics-based lighting is a *collection* of techniques and solutions to the common illumination problem. At the heart of every illumination problem lies the rendering equation [Kajiya86], which is a spatial function that captures all the incoming radiance (ie., lighting) at a certain point in 3-space. Some well-known solutions include radiosity-based techniques, which generally assume totally Lambertian (ie., diffuse) surfaces. The collection of ray-based techniques is another well-known solution, that greatly simplify the rendering equation (R.Eq. from hereonafter). Even ordinary lightmaps as used in Quake3 (essentially precomputed irradiance maps) form a partial (diffuse, static environments and lighting) solution to the R.Eq..

The essence of the R.Eq. is captured in a surface integral over the hemisphere enclosing a surface point in 3-space. In its most general form, it can be written as follows for a non-self-emitting surface:

$$L_r(x, v) = \int_{\Omega} f_r(x, v, w) L_i(x, w) V(x, w) (w \cdot n_x) dw$$

Where the domain of integration  $\Omega$ , is the hemisphere enclosing surface point  $x$ ;  $L_r(x, v)$  is the reflected radiance at  $x$  in direction  $v$  (for specularity).  $f_r(x, v, w)$  is the Bi-directional Reflection Distribution Function at  $x$ ,  $L_i(x, w)$  is the incoming radiance (ie, lighting) from direction  $w$ ,  $V(x, w)$  is a binary visibility term, which specifies whether or not a certain direction  $w$  is being shadowed by other surfaces, and  $n_x$  is the normal at  $x$ . Note that the R.Eq. allows for static, as well as dynamic scenes, and static as well as dynamic lighting. It are the particular solutions (ie, radiosity, path-tracing, photon mapping [Jensen01]) that put constraints on the equation so as to simplify evaluations.

Direct evaluation of the R.Eq. using numerical integration, such as Monte-Carlo quadrature, is possible. Radiosity-based techniques rewrite the R.Eq. as a linear system,  $Ax=b$ , and solve it accordingly. Classical radiosity techniques use standard elimination techniques (Gauss-Jordan) or iterative ones (Jacobi, Gauss-Seidel). Progressive refinement can drastically reduce  $O(n^2)$  space-complexity, whereas wavelet-based radiosity techniques make use of the properties of the matrix  $A$ , and transform it into a wavelet basis so as to make this matrix very sparse. Eigenvector radiosity [Ashdown01] is another technique that makes use of the symmetric property of a matrix closely related to  $A$ . Classical Whitted ray-tracing [Whitted80] evaluates  $f_r(x, v, w)$ ,  $L_i(x, w)$  and  $V(x, w)$  directly, without performing the integral, by selectively shooting rays into space, and evaluating the functions at the surface-point hit by the ray. This ofcourse, greatly simplifies the rendering equation, but its quality is very much bound by the number of rays one can trace within the time allocated. Other ray-based techniques rely on the same principle, and differ from eachother in the way rays are shot, and intersections are handled. Monte-Carlo based techniques, for example, use statistical heuristics to give a better approximation to the integral, while trying to limit the total number of rays shot.

## Introducing Spherical Harmonics

Spherical harmonics (SH from hereonafter) can be used to greatly simplify the rendering equation. Conceptually, SH form a complete and *orthonormal* basis [deBoer04] [Axler97] for functions on the unit-sphere (such as a function of incoming light  $L_i(x,v)$  which gives the radiance coming in from direction  $v$ , or the binary visibility function  $V(x,v)$ ), and allow for a function to be completely described as a collection of coefficients (floating-point, or doubles) in the SH basis. This is analogous to a Fourier series representation of functions constrained to the unit circle. ***The rendering equation can thus be rewritten as a simple dot-product, or a matrix-vector multiplication, which allows for real-time evaluation on commodity graphics hardware.*** This is the key aspect, that make spherical harmonics so attractive. Another advantage of using SH, is that it allows for ***dynamic real-time lighting of arbitrary lighting environments; not bound to point-lights, and independent of the total number of lights.*** In other words, SH allow for an arbitrary number of arbitrarily shaped lights that require no more computation time than, say, just one ordinary directional lightsource. And finally, SH allow for ***specular (in fact, any BRDF and BSDF, not limited to Phong) as well as diffuse (eg., Lambertian BRDF) lighting.*** Specular lighting (ie., lighting in which the BRDF depends on both  $v$  as well as  $w$ ) does, however, come at an extra cost (memory-wise, and real-time computational-wise), and should therefore be avoided for all but the most graphically demanding applications. One of the current key research efforts lie in addressing this last limitation ([Sloan03], [Kautz02]).

## Common SH-based illumination techniques

Several papers that utilise SH to simplify the rendering equation, have been published over the years. Among these, three techniques are best fit to be used in game technology. The remainder of this article will discuss these three techniques in more detail. Rather than straying into the technical details, this article aims to list the pros and cons of each technique. Brief descriptions of each technique are provided in each section, followed by an essay-style discussion of the advantages and disadvantages. Readers are encouraged to consult the references section at the end of this article, for pointers to more in-depth discussions of the techniques outlined.

### Efficiently represented irradiance environment maps (Erie Maps)

First introduced by Ramamoorthi and Hanrahan in [Ramamoorthi01], an irradiance environment map (A cube-map representation is used in the paper. However, the technique is independent of the particular representation) is projected [deBoer04] into the SH, and the first 9 coefficients are kept to represent the original. This is done per-frame, either analytically (for simple lightsources) or through numerical integration (which can be performed on the GPU). The paper discusses some properties of the diffuse rendering equation, and derives an analytical solution, suitable for real-time use. Then, for each vertex in the scene, its normal is transmitted, and the correct radiant exitance (ie., radiosity) is calculated in a vertex shader. This yields real-time framerates,

and has minimal overhead. Framerates of more than 60 Hz are easily achieved; the computational overhead is near negligible. It was proven in [Hanrahan01] that for diffuse illumination, only the first 9 coefficients need to be stored, while maintaining an accurate representation of the original lighting environment (ie, irradiance environment map).

*This method allows for dynamic lighting environments.* The entire lighting environment is allowed to change dynamically, and the correct 9-coefficient lighting vector is computed per frame. As mentioned earlier, SH calculations are independent of the shape and number of lightsources, so any lighting environment can be used, which all have the same computational cost: From simple pointlights, to complex skyboxes, to area lightsources.

*Dynamically deformable object geometry is allowed.* For each vertex, only the normal is needed in the computation. This means that *the lighting environment is assumed to be infinitely distant*. However, point lightsources can be approximated by having several 9-coefficient lighting vectors distributed sparsely in the scene, rather than one global one, and linearly interpolating between them to get the approximated lighting environment at a particular vertex.

*The technique assumes diffuse surfaces.* The paper assumes the BRDF in the R.Eq. to be Lambertian, and derives the analytical solution from it. n-order Phong lighting can be derived similarly (using the fact that the cosine term in the diffuse R.Eq. is a convolution, and rather than using the normal, one can use the vertex-dependent viewvector) [deBoer] [Najisire04].

*There is no extra per-vertex storage cost.* Other than storing a normal per vertex, this technique does not require any special per-vertex data.

*The solution assumes objects are convex.* Self-shadowing and interreflection are not supported. However, this technique can be combined with ambient occlusion techniques [RefXX] to approximate real-time self-shadowing.

*Global illumination effects are ignored.* The method does not support inter-object reflections, and only local lighting is taken into account.

## **Pre-computed radiance transfer (PRT)**

This solution was first proposed by Sloan et al. in [Sloan02], and spans several papers (including [Kautz02], and [Sloan03]). The key idea is to project, for each vertex, the visibility term  $V(x,v)$ , inter-object reflections and cosine term, into the SH *in an offline process*. The resulting n-coefficient vector (called a *transfer vector*) for diffuse, or n-by-n coefficient matrix (called a *transfer matrix*) for specular/glossy and diffuse, is then stored per vertex. At run-time, the lighting environment is projected into the SH each frame, using methods equivalent to the ones discussed for Erie maps. Then for diffuse lighting only, a dot-product between the resulting n-coefficient vector L, and a vertex' transfer vector yields the correct irradiance at that vertex. Or for specular and diffuse, L is multiplied by a vertex' n-by-n transfer matrix, followed by an evaluation of the resulting coefficient vector in the SH basis, to obtain the final irradiance (this accounts for evaluating the BRDF,  $fp(v,w)$ ). [Sloan02] proposes a convolution, which limits the type of BRDF to spherically symmetric ones, whereas [Kautz02] and [Kautz03] remove this limitation). The entire per-vertex process is performed in a vertex shader.

The variable  $n$  is set to 25 in the paper. Setting  $n$  to 9 will generally result in quality-loss due to the removal of high-frequency components in the self-shadowing, interreflection, and specular signals (Epic Games' latest project supports  $n=16$ ). Interactive framerates of more than 30 Hz are achieved for both diffuse, as well as specular and diffuse objects.

*The method is completely scalable.* One is entirely free to choose whether or not to support self-shadowing, and interreflections, and which type of BRDF to use. This makes the solution powerful, and versatile. See Robin Green's paper [Green03], and accompanying PS2 demo for an example.

*There is support for real-time dynamic self-shadowing and inter-object reflections ("colour bleeding").* The rendering equation reduces to a per-vertex vector dot-product, or matrix-vector multiply, which is easily evaluated in a vertex shader. We cannot overstate the fact that lighting is entirely dynamic, and self-shadows and interreflections behave as one would expect in such cases.

*An  $n$ -coefficient transfer vector, or  $n$ -by- $n$  coefficient transfer matrix must be stored per vertex.* Especially for the specular case, this can severely hamper real-time game possibilities, and affects not only storage requirements, but computation time as well. Also, due to the high-frequency nature of self-shadows and specular lighting, a reasonably dense tessellation of the geometry is required, resulting in astronomical memory requirements. However, one can choose not to support interreflections and self-shadowing, in which case, the technique reduces to Erie mapping.

However, *recent research efforts [Sloan03] have resulted in a quality-preserving compression technique, that greatly reduces storage requirements* especially for  $n$ -by- $n$  matrices. The technique uses a combination of standard statistical Principal Component Analysis, and Vector Quantisation, in an offline process. Not only does the technique reduce storage costs, it also reduces real-time computation cost by an order-of-magnitude, by making use of linearity of vectors and matrices. Framerates of 30 Hz are easily attained.

*Infinitely distant lighting is assumed.* However, the same workaround that was outlined in the previous section, applies here.

*Object geometry is not allowed to deform dynamically.* The self-shadowing and interreflection terms are calculated in a computationally expensive offline process, and therefore geometry is not allowed to deform at runtime. Research into removing this limitation has been conducted, but usable results are very sparse if not non-existent.

## **Lattice-based radiosity for dynamic lighting and scenes (SH Radiosity)**

This method [Nijasure04] utilises SH to provide an efficient representation of exitant radiance (ie., radiosity), and makes it possible to perform real-time (10 Hz) radiosity computations for dynamically lit scenes, which can themselves be locally and sparsely dynamic. While the solution is not yet suitable for use in interactive games, it does give an insight into the wide range of uses of SH in computer graphics. Note that Erie maps can be used to achieve the exact same results as this method; in fact, this technique should be regarded as an extension to Erie maps.

Rather than computing transfer vectors or transfer matrices at an object's surface points or vertices, 3-space is uniformly sampled into a set of discrete points  $\{p_i\}$

[Greger98] called a lattice, and the incoming radiance is calculated at each of these points as follows: The radiosity of all of the scene's geometry is initially set to 0, and therefore only the lights in the scene distribute radiance. For each lattice point  $p_i$ , a cubemap is constructed (which represents the irradiance environment map at  $p_i$ ), and for each face of the cubemap, the scene (without textures) is rendered as seen from  $p_i$ , looking into the direction perpendicular to the face. This cubemap is then treated as the radiance coming in at point  $p_i$  from all directions on the sphere enclosing  $p_i$  (ie., an irradiance environment map). It is projected into the SH, in exactly the same way as Erie maps, to yield a 9-coefficient vector  $I$  at  $p_i$ . Then, for each vertex  $v_i$  of the scene, the appropriate lattice cube which contains  $v_i$  is found. The incoming radiance coefficient vector at  $v_i$  is calculated by performing a weighted sum of the  $I$  vectors at the 8 corners (which is a subset of the lattice  $\{p_i\}$ ) of the cube. The radiosity at  $v_i$  is then calculated by performing the exact same calculation as Erie maps given the normal of  $v_i$ .

This can be seen as the first iteration of a progressive refinement-style radiosity engine, and successive iterations will render the exact same cubemap faces, but this time rather than the lights having non-zero radiance, the geometry will have a non-zero exitant radiance, and will therefore distribute radiance into the scene as well. In the limit of performing an infinite number of iterations, a global illumination solution is obtained. The first iteration can therefore be interpreted as accounting for local (ie., direct) illumination effects only.

*Radiosity can be computed progressively, analogous to classical progressive refinement.* This means that it is possible to navigate the scene instantly, while the scene lighting is being updated progressively.

*SH projection calculations are independent of the underlying scene geometry.* The calculations are performed at discrete points of a lattice in 3-space, rather than at the vertices of an object. The time and space complexity therefore depend on the resolution of the lattice. The eventual per-vertex cost is equal to that of Erie mapping.

*Inter-object shadowing, self-shadowing and interreflections are supported.* Inter-object shadowing and self-shadowing are consequences of the cubemap face rendering pass of each iteration. The HSR of the rendering engine (eg., the z-buffer in hardware) makes sure that only the visible surfaces are rendered. Interreflections are a result of the successive iterations of the algorithm.

*The method supports specular as well as diffuse surfaces.* Different types of BRDF are supported. However, radially symmetric BRDF's (Lambertian/Phong) are desirable, as these can be reparametrised to be 1-dimensional.

*The result is highly dependent on the resolution of the lattice.* Since the lattice is a discrete approximation of a continuous signal (ie., it samples the irradiance [a continuous signal in 3-space] at discrete locations in 3-space), all the usual signal processing rules apply. As a result, shadows may be incorrect (ie, high-frequency components are lost).

## Thoughts

Spherical harmonics can help make other solutions to the rendering equation more efficient; an example of which was outlined in the last section. Another example where SH can prove to be useful is the following. The rendering equation can be separated into several integral terms, each capturing a different aspect of the incoming radiance (ie., direct illumination, caustics, etc), as was shown in [Jensen01]. The direct illumination term is usually solved by Monte-Carlo integration (eg., path tracing). However, due to the variance of Monte-Carlo depending inversely on the number of samples taken (ie., rays shot), this may be computationally very costly. A photon map can be used to make this evaluation more efficient, by either removing the need to cast rays, or by providing a better approximation to the p.d.f., but its accuracy depends on the photon density in the scene, which should depend on the existence of high-frequency details in lighting. However, the direct illumination term may be solved by projecting the incoming radiance into the SH. A cubemap can be constructed to quickly capture this incoming radiance: Everything apart from lightsources is set to 0 radiance, equivalent to SH Radiosity. (The visibility,  $V(x,w)$ , is then implicitly captured as well). This cubemap is then projected into the SH (as in Erie maps), and the rendering equation is solved in exactly the same manner as in PRT.

## Conclusion

Spherical harmonics are very versatile mathematical objects, that have many uses including in computer graphics. A combination of direct point-lightsource illumination for deformable geometry (ie., animated characters), and SH-based lighting for distant and/or static geometry has been used in at least one other commercial game engine, with great success.

## References

- Whitted80, Whitted, Turner. An Improved Illumination Model for Shaded Display. *Communications of the ACM*, v.23 n.6, p.343-349. 1980.
- Jensen01, Jensen, Henrik W. Realistic Image Synthesis Using Photon Mapping. *A K Peters, Natick, Massachusetts*. ISBN 1-56881-147-0. 2001.
- Ashdown01, Ashdown, Ian. Eigenvector Radiosity. *Master's Thesis*, University of British Columbia, 2001.
- Kautz02, Kautz, Jan. Lehtinen, Jaakko. Matrix Radiance Transfer. *SIGGRAPH 2003 Symposium on Interactive 3D Graphics*. 2003.
- Sloan03, Sloan, Peter-Pike. Hall, Jesse. Hart, John. Snyder, John. Clustered Principal Components. *SIGGRAPH Transactions on Graphics 2003*. 2003.
- Kajiya86, Kajiya, James T. The Rendering Equation. *Proceedings of SIGGRAPH 86*, 20(4), pp. 143-150. 1986.
- de Boer04, de Boer, Willem H. Optimal Spherical Harmonics Projection. *Submitted to Journal of Games Development*. Charles-River Media, 2004.

- Axler97, Axler, Sheldon. Linear Algebra Done Right. *Springer Undergraduate Texts in Mathematics 1997*, ISBN 0-387-98258-2. 1997.
- Kautz02, Kautz, Jan. Sloan, Peter-Pike. Snyder, John. Fast, Arbitrary BRDF Shading for Low-Frequency Lighting Using Spherical Harmonics. *Eurographics Workshop on Rendering*. 2002.
- Ramamoorthi01, Ramamoorthi, Ravi. Hanrahan, Pat. An Efficient Representation of Irradiance Environment Maps. *Proceedings of SIGGRAPH 01*. 2001.
- Sloan02, Sloan, Peter-Pike. Kautz, Jan. Snyder, John. Precomputed Radiance Transfer. *Proceedings of SIGGRAPH 02*. 2002.
- Green03, Green, Robin. Spherical Harmonic Lighting: The Gritty Details. *Proceedings of Game Developers Conference 2003*. 2003.
- Hanrahan01, Ramamoorthi, Ravi. Hanrahan, Pat. On the Relationship Between Radiance and Irradiance: Determining the Illumination from Images of a Convex Lambertian Object. *J. Opt. Soc. Am. A*, Vol. 18, No. 10. Oct 2001.
- Nijasure04, Nijasure, Mangesh. Goel, Vineet. Real-Time Global Illumination on GPU.
- Greger98, Greger, Gene. Shirley, Peter. Hubbard, Philip M. Greenberg, Donald P. The Irradiance Volume. *IEEE CG&A 1998*, pp. 32-43. 1998.