Sparse Zonal Harmonic Factorization for Efficient SH Rotation

Derek Nowrouzezahrai[1,2,3], Patricio Simari[4], and Eugene Fiume[3]
[1]Université de Montréal, [2]Disney Research Zurich, [3]University of Toronto, [4]Autodesk Research

We present a sparse analytic representation for spherical functions, including those expressed in a spherical harmonic (SH) expansion, that is amenable to fast and accurate rotation on the GPU. Exploiting the fact that each band-l SH basis function can be expressed as a weighted sum of $2l+1$ rotated band-l zonal harmonic (ZH) lobes, we develop a factorization that significantly reduces this number. We investigate approaches for promoting sparsity in the change-of-basis matrix, and also introduce *lobe sharing* to reduce the total number of unique lobe directions used for an order-N expansion from N^2 to $2N-1$. Our representation does not introduce approximation error, is suitable for any type of spherical function (e.g., lighting or transfer), and requires no offline fitting procedure; only a (sparse) matrix multiplication is required to map to/from SH. We provide code for our rotation algorithms, and apply them to several real-time rendering applications.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Color, shading, shadowing, & texture*

Additional Key Words and Phrases: spherical harmonic rotation, rendering

ACM Reference Format:

Nowrouzezahrai, D., Simari, P., and Fiume, E. 2010. Sparse Zonal Harmonic Factorization for Efficient SH Rotation and Shading. ACM Trans. Graph. XX, Y, Article ZZZ (Month 2010), 8 pages.
DOI = xx.xxxx/xxxxxxx.xxxxxxx
http://doi.acm.org/xx.xxxx/xxxxxxx.xxxxxxx

1. INTRODUCTION

Spherical functions are used in several areas of computer graphics (CG) such as rendering [Sloan et al. 2002] and shape analysis [Kazhdan 2007]. In many important cases, spherical harmonics (SH) are an ideal representation for such functions: e.g., many parameterized BRDFs can be represented analytically in SH, admitting efficient, frequency-adaptive reconstructions. Shading these BRDFs with dynamic lighting in SH can also be very efficient.

An important property of SH is closure under rotation: an SH expanded function can be rotated directly from its coefficients, without requiring explicit reconstruction, rotation, and reprojection. Unfortunately, existing efficient approaches only handle a handful of low-order ($N \leq 5$) rotations per frame for real-time applications, or only support a restrictive subset of functions without introducing significant approximation error.

Contributions. We present an alternative basis for spherical functions based on rotated zonal harmonic (ZH) lobes. This basis spans the same space as SH, affords a more efficient rotation algorithm, and there is a sparse linear mapping between each space. We investigate promoting sparsity in this mapping, as well as their mathematical implications. Simple, accurate, and efficient rotation algorithms are implemented on the CPU and GPU, and benchmarked against existing techniques used in graphics. *Lobe sharing* (Section 6) reduces the number of unique lobe directions from N^2 to $2N-1$, and our algorithms are trivially parallelizable.

Our sparse, lossless representation enables a novel data-parallel optimization tailored to shader-based engines, and especially suitable for real-time relighting. Our rotation algorithms, requiring less than 40 lines of code (provided in Supplemental Material), outperform existing approaches, especially on the GPU, and we apply them to several relighting applications. Readers interested in the mathematical exposition can refer to Sections 3 to 6, whereas those more interested in the algorithm can focus on Section 7.1.

2. PREVIOUS WORK

Our goal is to derive a fast rotation algorithm with unnoticeable visual error behavior, particularly amenable to the lower order SH expansions ($N < 20$) used in CG applications. Higher-order rotation requires greater attention to numerical stability issues and thus a different compromise between speed, stability and accuracy. A recent algorithm to this end can be found in [Lessig et al. 2010].

Many signals in CG are expressed naturally in a spherical domain, and SH expansions of such functions can be appealing due to their analytic form, frequency-space properties, and identities. While SH has a long history in CG, including uses in volumetric transport [Kajiya and Von Herzen 1984] and BRDF representation [Westin et al. 1992], we focus on the recent use of SH in derivatives of Precomputed Radiance Transfer (PRT).

Sloan et al. [2002] precompute and project a linear mapping (capturing shadowing/reflection effects) of incident light to outgoing radiance into SH. At run-time, global-frame lighting is rotated using low-order SH rotation matrices computed from complex zyz-Euler recurrence formulae [Edmonds 1960]. Kautz et al. [2002] decompose the y-rotation into a zxz-rotation for local coordinate frame shading. These approaches do not map well to the GPU and quickly become the bottleneck of SH relighting approaches. An inherent no-win memory/computation trade-off exists: local-frame shading requires many per-point rotations but affords more compact transfer representation, while global-frame shading requires fewer rotations but at the cost of more storage for transfer. Our ap-

Derek Nowrouzezahrai acknowledges funding from the National Sciences and Engineering Research Council of Canada (NSERC), the Canadian Research Network for Mathematics of Information Technology and Complex Systems (MITACS), the Ontario Ministry of Research and Innovation (MRI), and the Ontario Ministry of Education and Training. {derek@iro.umontreal.ca, patricio.simari@autodesk.com, elf@dgp.toronto.edu}

DOI 10.1145/XXXXXXX.YYYYYYY
http://doi.acm.org/10.1145/XXXXXXX.YYYYYYY

Fig. 1. St. Jean Cathedral rendered on a web browser with Google's O3D API. We augment this open-source demo (with permission of Benoït Mayaux [2010]; http://www.patapom.com/O3D/Cathedral.html). with a local reflectance model and order-9 SH light rotated with our fast, accurate, and simple "signal-tailored" GPU rotation algorithm. We encourage readers to digitally zoom-in to reveal detail.

proach scales favorably to higher N, is easily implemented on the GPU, and enables efficient mixed global- and local-frame rotation.

Křivánek et al. [2006] replace the zyz-Euler approach's y-rotation with a Taylor expansion, enabling low-order local-frame shading but with added approximation error. In contrast, we perform exact SH rotation at lower computational and storage costs.

Sloan et al. [2005] fit ZH lobes to transfer functions with non-linear optimization. At run-time, transfer is rotated using fast ZH rotation (see Section 3). Their fitting performs well with transfer functions but less so for arbitrary SH vectors. Our approach is more accurate and applies to arbitrary functions without offline fitting.

In concurrent work, Lessig et al. [2010] develop a similar ZH decomposition, using Reproducing Kernel Hilbert Space (RKHS) analysis, for a new sampling theorem over the sphere that optimizes for numerical stability. This approach is amenable to wide-bandwidth signals where high numerical precision is required; we instead optimize for sparsity of the ZH decomposition, leading to very fast rotation for lower-bandwidth signals used in CG.

Rotation of spherical radial-basis functions (SRBFs) and Haar Wavelets has also been investigated in the context of PRT. Tsai and Shih [2006] use an SRBF basis that maps to ZH lobes and effectively exploit the ZH rotation rule (see Section 3.1) for fast rotation. Wang et al. [2006] precompute Haar rotation transforms for discrete rotational frames, generated using octahedral maps.

Significant work has been conducted outside CG on SH rotation. This body of work is concerned with $N > 20$ which is a non-typical use case in CG and, to the best of our knowledge, we are the first to purposefully develop a *sparse* mapping between SH and rotated ZH basis spaces for fast rotation. We refer interested readers to Lessig et al.'s [2010] review of this literature and focus our discussion and results on SH rotation for $N < 20$.

We introduce an alternative representation of spherical functions that admits high-performance, parallelizable, accurate SH rotations, even at comparably higher order than previous approaches. We provide data to compute fixed, sparse coupling matrices as well as source code (< 40 lines of code), in our Supplemental Material. We outperform existing approaches (see Section 7.1) for low-order rotation on the CPU, and a novel *signal-tailored* rotation algorithm on the GPU is able to consistently outperform existing approaches. Although the computational complexity of our approach is the same as the state-of-the-art $zxzxz$ techniques, several key differences exist: our algorithm exploits sparsity using a *single* precomputed, fixed, sparse high-dimensional transformation matrix (for a fixed order N), whereas $zxzxz$ approaches need to construct sparse z-rotation matrices algorithmically, as well as requiring two separate sparse, precomputed x-rotations. Our standard rotation approach is composed of three simple steps (dominated by arithmetic computation), allowing higher-performance at low-orders on the CPU (and GPU), and the signal-tailored GPU optimization replaces the two most expensive steps of the standard algorithm with look-ups into precomputed cubemap textures.

3. OVERVIEW AND TERMINOLOGY

We adopt notation from the real-time rendering literature (e.g., [Ren et al. 2006]) in our exposition: italics for scalars and 3D points/directions (e.g., ω), boldface italic for coefficient column vectors (e.g., $\boldsymbol{f}$), and sans serif for matrices/tensors (e.g., A_l).

Let $f(\omega)$ be a spherical function, with $\omega = (x, y, z) = (\theta, \phi) \in S^2$, and θ and ϕ are spherical (lat-long) coordinates of point (x, y, z) on the sphere's surface, S^2. Projecting f onto the real SH basis used in CG yields a coefficient vector $\boldsymbol{f} = \int_{S^2} f(\omega)\boldsymbol{y}(\omega)d\omega$, where $\boldsymbol{y}(\omega)$ is a vector of SH basis functions with

$$y_l^m(\theta,\phi) = \begin{cases} K_l^0\, P_l^0(\cos\theta), & m = 0 \\ \sqrt{2}\, K_l^m\, \cos(m\phi)\, P_l^m(\cos\theta), & m > 0 \\ \sqrt{2}\, K_l^{|m|}\, \sin(|m|\phi)\, P_l^{|m|}(\cos\theta), & m < 0 \end{cases}, \quad (1)$$

where m indexes the $(2l + 1)$ band-l basis functions, K_l^m is a normalization term, and P_l^m are associated Legendre polynomials (ALPs). Band-l basis functions are degree l polynomials in (x, y, z).

A band-limited reconstruction of f can be obtained by weighting the SH basis functions by the elements of $\boldsymbol{f}$:

$$f(\omega) \approx \tilde{f}(\omega) = \sum_{l=0}^{N-1} \sum_{m=-l}^{m=l} f_l^m\, y_l^m(\omega) = \boldsymbol{f} \cdot \boldsymbol{y}(\omega) \quad (2)$$

and order-N reconstructions use N^2 basis coefficients. Unless f is band-limited, reconstruction is approximate. At times it is convenient to use a single index $i = l(l + 1) + m$ for all basis functions.

3.1 Zonal Harmonics

Zonal harmonics are the $m = 0$ subset of SH basis functions. These functions are circularly symmetric about z and are scaled Legendre polynomials (Eq. 1, line 1). Sloan et al. [2005] show, using the Funke-Hecke convolution theorem, that a ZH function oriented about the canonical z-axis (with coefficient g_l) can be rotated to an arbitrary direction ω_d, yielding an SH function (with coefficients f_l^m), by simply scaling the SH basis functions evaluated at ω_d via

$$f_l^m = n_l^* g_l y_l^m(\omega_d) = g_l^* y_l^m(\omega_d) \text{ [Sloan et al. 2005]}, \quad (3)$$

with $n_l^* = \sqrt{4\pi/(2l + 1)}$. One of our main contributions is to show that, by carefully choosing lobe directions, band-l SH basis functions can be perfectly reconstructed with **significantly fewer** than $2l + 1$ weighted, rotated band-l ZH lobes, y_l^0 (Figure 2 and Table I). The weights and directions are fixed and pre-tabulated, not computed on-the-fly. We also reduce the number of directions required for an order-N expansion from N^2 to $2N - 1$ (Section 6).

As a consequence, we are able to develop simple and efficient algorithms that generalize the fast ZH rotation rule to *arbitrary* SH rotations, which we apply to various shading problems.

4. ZONAL HARMONIC FACTORIZATION

Put briefly, the main observation our paper brings to light is that each band-l basis function can be perfectly represented as a

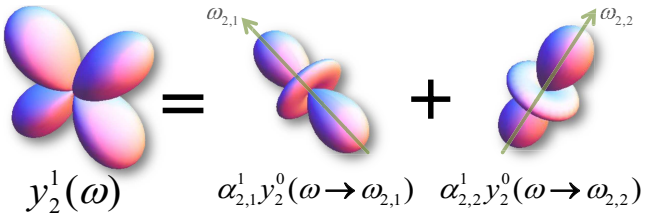

Fig. 2. Every band-l SH basis function can be exactly represented with at most, but often significantly fewer than, $2l+1$ rotated $y_l^0(\omega)$ ZH lobes. For example, $y_2^1(\omega)$ can be decomposed into a weighted sum of two $y_2^0(\omega)$ lobes rotated to $\omega_{2,1}$ and $\omega_{2,2}$. See Section 4 for notation details.

weighted sum of at most $2l+1$ rotated y_l^0 lobes. We begin by quickly illustrating this fact for the trivial cases of the $l=0$ and 1 basis functions, and follow with a general band-l formulation.

Band-0 and 1 cases. The $l=0$ function y_0^0 is constant and unaffected by rotation. Band-1 functions are scaled monomials of Cartesian coordinates: $(y_1^{-1}; y_1^0; y_1^1) = (-\nu y; \nu z; -\nu x)$ with $\nu = \sqrt{3}/(2\sqrt{2})$. An alternative interpretation of these functions is that of a double-sided cosine lobe aligned along y, z and x. The ZH lobe νz can be rotated to either the y or x axes using Equation 3 with $\omega_d = \omega_{1,-1} = (\pi/2, \pi/2) = (0,1,0)$ for y_1^{-1} and $\omega_d = \omega_{1,1} = (\pi/2, 0) = (1,0,0)$ for y_1^1. Consequently, we can express these $m \neq 0$ functions as a weighted, rotated y_1^0 ZH lobe:

$$y_1^{-1}(\omega) = \alpha_{1,-1}^{-1}\, y_1^0(\omega \to \omega_{1,-1}), \quad y_1^1(\omega) = \alpha_{1,1}^1\, y_1^0(\omega \to \omega_{1,1}),$$

where $\alpha_{l,d}^m$ is the weight of the d^{th} lobe[1] of y_l^m, with direction $\omega_{l,d}$, and $\omega \to \omega'$ denotes 3D rotation from ω to ω'. In general, we pre-compute these weights $(\alpha_{l,d}^m)$ and lobe directions $(\omega_{l,d})$ to induce sparsity and accelerate computation, as we will discuss shortly. For band-1 we have $[\alpha_{1,-1}^{-1}\ \alpha_{1,0}^{-1}\ \alpha_{1,1}^{-1}] = [-1\ 0\ 0]$ and $[\alpha_{1,-1}^1\ \alpha_{1,0}^1\ \alpha_{1,1}^1] = [0\ 0\ -1]$ which, along with $\omega_{1,-1}$ and $\omega_{1,1}$, forms the optimal solution for this band.

General case. Band-1 basis functions are monomials in (x,y,z) and only require a single rotated lobe. In general, each band-l SH basis function can be composed of a weighted sum of up to $2l+1$ rotated y_l^0 ZH lobes. Minimizing the number of lobes used in practice has significant implications, discussed in Section 5.

The $m=0$ function can always be trivially represented with

$$[\alpha_{l,-l}^0 \ \ldots\ \alpha_{l,0}^0 \ \ldots\ \alpha_{l,l}^0] = [0\ \ldots\ 1\ \ldots\ 0] \text{ and } \omega_{l,0} = (0,0). \quad (4)$$

We will show that all $m \neq 0$ basis functions can be represented as

$$y_l^m(\omega) = \textstyle\sum_{d=-l}^{l} \alpha_{l,d}^m\, y_l^0(\omega \to \omega_{l,d})\ , \quad (5)$$

where the $2l+1$ lobe directions, $\omega_{l,d}$, are shared across basis functions in the band, but a unique set of $2l+1$ weights, $\alpha_{l,d}^m$, are required for each basis function. We start by formulating the problem of solving for the unknown weights/directions as a non-linear system of equations, and then augment a simplified matrix inversion procedure to enforce sparsity in the resulting coupling matrices.

System of non-linear equations. We aim to formulate Equation 5 in a manner suitable for solving for the unknown weights and lobe directions. We apply the *SH Addition Theorem* which states that a *zonal harmonic* aligned about an arbitrary direction can be reconstructed as a weighted sum of *spherical harmonics*,

$$y_l^0(\omega \to \omega_{l,d}) = n_l^* \textstyle\sum_{m'=-l}^{l} y_l^{m'}(\omega_{l,d})\, y_l^{m'}(\omega)\ , \quad (6)$$

[1]For consistency with m-indexing, d indices start at $-l$ and end at l.

where the reconstruction weights are proportional to the SH basis functions evaluated in the lobe direction. In a sense, Equation 5 is the "dual" of the SH addition theorem in Equation 6: one states that SH basis functions can be reconstructed from weighted and rotated ZH basis functions, and the other states that a rotated ZH basis function can be reconstructed with weighted SH basis functions.

Expressing Equation 5 in matrix-vector form across all $m \in [-l, l]$, and substituting Equation 6 in for $y_l^0(\omega \to \omega_{l,d})$ yields

$$\begin{bmatrix} y_l^{-l}(\omega) \\ \vdots \\ y_l^{l}(\omega) \end{bmatrix} = \underbrace{\begin{bmatrix} \alpha_{l,-l}^{-l} & \ldots & \alpha_{l,l}^{-l} \\ \vdots & \ddots & \vdots \\ \alpha_{l,-l}^{l} & \ldots & \alpha_{l,l}^{l} \end{bmatrix}}_{\mathsf{A}_l} \begin{bmatrix} y_l^0(\omega \to \omega_{l,-l}) \\ \vdots \\ y_l^0(\omega \to \omega_{l,l}) \end{bmatrix}$$

$$= \underbrace{\mathsf{A}_l\, \mathsf{D}_l}_{\hat{\mathsf{A}}_l} \underbrace{\begin{bmatrix} y_l^{-l}(\omega_{l,-l}) & \ldots & y_l^{l}(\omega_{l,-l}) \\ \vdots & \ddots & \vdots \\ y_l^{-l}(\omega_{l,l}) & \ldots & y_l^{l}(\omega_{l,l}) \end{bmatrix}}_{\mathsf{Y}_l} \begin{bmatrix} y_l^{-l}(\omega) \\ \vdots \\ y_l^{l}(\omega) \end{bmatrix} \quad (7)$$

where D_l is a $(2l+1) \times (2l+1)$ diagonal matrix with repeated entries of n_l^*. From Equation 7 we see that $\hat{\mathsf{A}}_l\, \mathsf{Y}_l = \mathsf{I}$, where I is the identity matrix. This matrix equation defines a system of $(2l+1)$ non-linear equations in the $(2l+1)^2$ unknown α weights and $(2l+1)$ unknown (θ, ϕ) lobe direction pairs. We can reduce the total number of unknowns using Equation 4 from $(2l+1)^2 + 2(2l+1)$ to $2l(2l+3)$ and solve for them by minimizing

$$\underset{\forall d, \forall m\,:\,[\alpha_{l,d}^m, \omega_{l,d}]}{\operatorname{argmin}} \sum_{ij} \left([\mathsf{A}_l\, \mathsf{D}_l\, \mathsf{Y}_l - \mathsf{I}]_{ij} \right)^2 . \quad (8)$$

Expression 8 can be solved using a constrained non-linear solver to restrict lobe angles to $[0, 2\pi]$; however, we found an unconstrained solver sufficed with rapid convergence to 0 (see below), especially since this optimization need only be performed once:

$l=$	2	3	4	5	6	7	8	9	10
time (secs)	2	5	7	10	16	25	38	53	69

This process must be repeated for each band l and analogues of the A_l, D_l, and Y_l matrices exist for the set of *all* SH basis functions one would consider. Two of these matrices, $\hat{\mathsf{A}}$ and Y, have a block-diagonal form matching that of a standard SH rotation matrix: e.g. each block diagonal square matrix in $\hat{\mathsf{A}}$ is the appropriate $\hat{\mathsf{A}}_l$ matrix.

Linearizing the problem. In concurrent work, Lessig et al. [2010] formulate SH rotation using a new sampling theorem for the sphere based on RKHS analysis. They show that *any* non-singular choice of the $\omega_{l,d}$ is valid and results in an invertible Y_l. Thus, Equation 7 can be solved by simply picking a random set of $\omega_{l,d}$ and solving for the $\alpha_{l,d}^m$ as $\hat{\mathsf{A}}_l = [\mathsf{Y}_l]^{-1}$. While their analysis focusses on the conditioning of the system for accurate sampling, we seek instead to promote sparsity in the structure of $\hat{\mathsf{A}}_l$ for fast GPU rotation and thus require a different kind of optimization.

Interpretation as coupling coefficients. Given the set of weights and lobe directions, we can represent an *arbitrary spherical function*, with SH coefficient vector $\boldsymbol{f}$, as a weighted combination of rotated ZH functions. The band-l SH coefficient vector of the function, $\boldsymbol{f}_l = [f_l^{-l} \ldots f_l^l]^{\mathrm{T}}$, can be transformed into band-l coefficients in our **Rotated Zonal Harmonic Basis** (RZHB):

$$\hat{\boldsymbol{z}}_l = \left[\hat{\mathsf{A}}_l\right]^{\mathrm{T}} \boldsymbol{f}_l \quad \text{s.t.} \quad \sum_{i=-l}^{l} [\boldsymbol{f}_l]_i\, y_l^i(\omega) = \sum_{j=-l}^{l} [\hat{\boldsymbol{z}}_l]_j\, y_l^0(\omega \to \omega_{l,j})\ .$$

The process of representing a spherical function, with an initial SH coefficient vector $\boldsymbol{f}$, in the RZHB with a new coefficient vector $\hat{\boldsymbol{z}}$ is called **Zonal Harmonic Factorization** (ZHF). Similarly, an RZHB vector $\hat{\boldsymbol{z}}$ maps back to its corresponding SH vector using **Zonal Harmonic Expansion** (ZHE): $\hat{\boldsymbol{f}} = \mathsf{Y}^{\mathrm{T}}\,\hat{\boldsymbol{z}}$.

An important special case arises when $\hat{\mathsf{A}}_l$ is sparse. We will discuss the implications and practical benefits of this sparsity.

5. SPARSE ZH FACTORIZATION AND EXPANSION

We will discuss sparsity in both $\hat{\mathsf{A}}_l$ and Y_l, and how to enforce it, resulting in a compact mapping between SH and RZHB.

Investigating sparsity. A row in $\hat{\mathsf{A}}$ maps a single SH basis function into the RZHB; if a row is sparse, its corresponding SH basis function can be represented with *fewer* than $2l+1$ rotated ZH lobes. For example in Section 4, each band-1 SH basis function was represented with 1, as opposed to $2l+1=3$, ZH lobes by aligning the lobes along the x, y, and z axes. Promoting sparsity in $\hat{\mathsf{A}}$ is both interesting from a theoretical perspective and also admits a more efficient ZHF for rotation and shading algorithms (see Section 7).

Finding lobe directions to sparsify $\hat{\mathsf{A}}$ and Y is a continuous search problem with a candidate space whose volume grows exponentially in N. After thorough experimentation (detailed in the Supplemental Material), we were able to reduce the continuous search to a discrete search. This reduction, coupled with a genetic algorithm search, results in an effective approach for precomputing the sparse ZHF matrices. We include lobe directions (for $N \leq 8$) in the Supplemental Material.

Our empirical evidence suggests that aligning lobe directions with the zeros of SH basis functions promotes sparsity in $\hat{\mathsf{A}}$ and Y (see Supplemental Material): choosing ϕ roots of Equation 1 is straightforward, however a closed-form analytic expression for the zeros of ALPs is an open problem (although their locations' ranges can be bound [Lacroix 1984]). We numerically determine the locations of zeros in the ALPs to seed candidate θ values for the lobe directions. The empirical utility of aligning lobes along the location of SH basis function zeros can be tied to the fact that a ZH lobe rotated along such a direction is orthogonal to the SH basis function the direction zeros out (by the Funke-Hecke convolution theorem).

Obtaining sparsity. Given the insights above, we can reduce the continuous search for sparse solutions into a discrete search by only sampling θ and ϕ values (for all d) at the zeros of the band-l SH basis functions. We encode candidate angles in a vector of length $v = 2(2l+1)$, of the form $(\theta_0, \ldots, \theta_{2l}, \phi_0, \ldots, \phi_{2l})$, each element of which is selected from the Legendre and sinusoidal root candidates for bands 1 through l. Given any setting of these values, we generate an $\hat{\mathsf{A}}_l$ matrix and evaluate its sparsity by considering the fraction of zero entries that result. This is the objective function which we seek to maximize. In practice, to avoid numerical issues, we consider an entry to be zero if $|[\hat{\mathsf{A}}_l]_{i,j}| < \epsilon$, with $\epsilon = 10^{-9}$.

Exhaustive search quickly becomes intractable as the search space grows exponentially in l. Instead, we address this optimization using an evolutionary algorithm. We begin by initializing a population of individuals, assigning each entry a random value from the set of candidates. This may result in Y_l matrices so sparse that they are not invertible, resulting in a candidate fitness $-\infty$ (worse than any candidate that does lead to a valid $\hat{\mathsf{A}}_l$). We employ uniform crossover and mutation, a population size of 500, a crossover fraction of .8 and a mutation rate of v^{-1}. We use elitism of 2 and terminate after 50 stall generations [Michalewicz 1998].

Table I. Number of optimized lobes used to represent each SH function (up to $l=6$), and sparsity in $\hat{\mathsf{A}}_l$.

$m=$	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
$l=2$					2	2	1	2	2				
$l=3$				2	2	2	1	3	3	3			
$l=4$			3	4	4	5	1	5	2	4	3		
$l=5$		3	7	5	10	5	1	7	5	7	3	4	
$l=6$	9	4	8	4	9	7	1	10	6	10	5	8	5

Sparsity (as a % of total entries;higher is better)																	
$l=$	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
	64	67	62	53	49	46	42	39	38	38	37	37	36	36	36	35	35

Table I lists the number of lobes for each $l \leq 6$ SH function, as well as the *per-band* sparsity up to $l=18$. For fixed l, the $m=\pm i$ functions are rotationally symmetric by definition. We take this into account when determining candidate directions, however explicitly forcing the use of symmetric lobes *within* bands may not lead to an optimally sparse representation *across* bands in an order-N expansion. This explains the differing number of lobes for $m=\pm i$.

Signal-specific sparsity. Sloan et al. [2005] derive an *optimal linear lobe* so that *any* $l=1$ SH vector can be reconstructed with a single lobe. Analogous expressions do not exist for $l>1$, and so uniform discrete search and non-linear optimization are used to fit rotated ZH lobes to a *fixed* SH vector. Their efficiency depends on the type of signal being fit and precludes dynamic signals. Unbounded signals, e.g. environmental lighting, require many more lobes and thus more time to converge. Instead, we exploit structure in Y to pre-compute a fixed, sparse ZHF matrix that can map *any* SH vector to the RZHB on-the-fly, without any costly offline fitting.

6. SHARING LOBES ACROSS BANDS

So far we have focussed on sparse per-band ZHF matrices. For an order-N SH expansion (comprising all bands up to and including $l_{\max} = N-1$) we can simply apply the ZHF independently per band. This would require a total of N^2 ZH lobes, where each band l uses a unique set of $2l+1$ directions optimized for sparsity in $\hat{\mathsf{A}}_l$.

Apart from the theoretical implications of $\hat{\mathsf{A}}_l$'s sparsity, there are also computational benefits: the ZHF sparse matrix-vector multiplication can be hard-coded, translating to a theoretical per-band speedup proportional to the percentage of sparsity (see Table I).

For an order-N expansion the ZHE requires the evaluation of SH basis functions at these N^2 directions. When the lobe directions do not align with the zeros of SH basis functions (as will be the case for our fast SH rotation algorithm in Section 7.1), these evaluations can become costly. We have investigated a hybrid approach for reducing the *total* number of lobe directions across all bands, while maintaining sparsity in $\hat{\mathsf{A}}$. We begin with our original discrete, genetic algorithm sparsity search for band $l_{\max}$, and concurrently attempt to enforce sparsity across each $\hat{\mathsf{A}}_l$ for all $l \leq l_{\max}$. This amounts to choosing directions that induce sparsity *across all bands* while sharing lobes between bands[2].

With this approach, we maintain much of the per-band sparsity, but with only $2l_{\max}+1$ unique lobe directions instead of N^2. Figure 3 illustrates cumulative sparsity by order; specifically, an order-N expansion has sparsity measured as the number of zeros across

[2]Starting with candidates $(\theta_0, \ldots, \theta_{2l_{\max}}, \phi_0, \ldots, \phi_{2l_{\max}})$, generating $\hat{\mathsf{A}}_k$ matrices for $k \in [2, l]$, and measuring sparsity for each $\hat{\mathsf{A}}_k$ matrix.

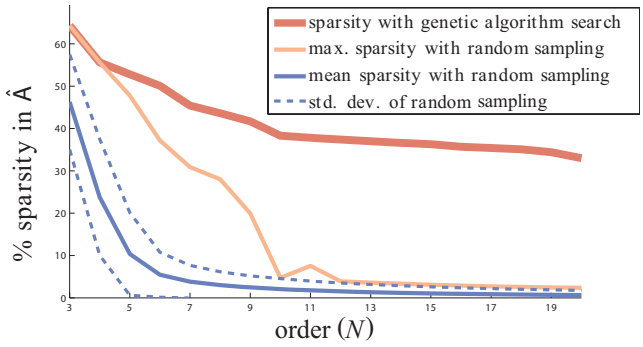

Fig. 3. Order-N sparsity of $\hat{\mathsf{A}}$ from random and evolutionary search of ALP/sinusoidal root candidates (singular Y are omitted from the statistics). For example, at $N = 8$, the mean and max sparsities from many randomly chosen lobe directions is approximately 3% and 29%, however our genetic algorithm search found a solution with a sparsity of 45%.

all $\hat{\mathsf{A}}_l$ sub-matrices, divided by the total number of elements. The average sparsity of all $\hat{\mathsf{A}}$ matrices generated from the Legendre/sinusoidal root candidates decreases sharply with N (Figure 3), yet our optimization successfully finds highly sparse solutions even for large N. Shared directions, $\boldsymbol{\Omega}$, are nested supersets across bands:

$$\boldsymbol{\Omega} = \{\ \omega_{0,0} = \omega_{1,-1} = \omega_{2,-2} = \omega_{3,-3} = \ldots = \omega_{l_{\max},-l_{\max}},\ \omega_{1,0} = \omega_{2,-1} = \omega_{3,-2} = \ldots = \omega_{l_{\max},-l_{\max}+1},\ \ldots\ldots\ldots,\ \omega_{l_{\max},l_{\max}}\ \}\ .$$

7. APPLICATIONS AND EXAMPLES

Section 7.1 details two simple, efficient, accurate and trivially parallelizable SH rotation algorithms that leverage the RZHB. We benchmark CPU and GPU implementations of our basic algorithm against existing approaches and also present and benchmark a novel GPU algorithm most suitable for rotating environmental lighting in PRT. Section 7.2 overviews shading with SH and RZHB. Section 7.3 discusses applications of our rotation to real-time rendering.

7.1 Fast and Simple SH Rotation

Given the SH coefficient vector $\boldsymbol{f}$ of a function $f(\omega)$, we will outline a simple algorithm for obtaining the SH coefficients $\boldsymbol{f}^r$ of the rotated function $f(\mathsf{R}\cdot\omega)$ using ZHF and ZHE, with $\mathsf{R}' = \mathsf{R}^{-1}$.

One of the major benefits of the RZHB representation for spherical functions is that, given $\hat{\boldsymbol{z}}$, the function can be rotated by simply rotating the lobe directions and keeping the same coefficients:

$$f(\mathsf{R}\cdot\omega) = \sum_{l=0}^{N-1}\sum_{m=-l}^{l} \hat{\boldsymbol{z}}_{l,m}\ y_l^0(\omega \to \mathsf{R}\cdot\omega_{l,m})\ . \tag{9}$$

Equation 9 is a generalization of Equation 3 from ZH to SH. For an arbitrary SH vector $\boldsymbol{f}$, the rotated SH coefficient vector $\boldsymbol{f}^r$ is

$$\boldsymbol{f}^r = [\mathsf{Y}^{\mathsf{R}}]^{\mathrm{T}}\ \hat{\mathsf{A}}^{\mathrm{T}}\ \boldsymbol{f} = [\mathsf{Y}^{\mathsf{R}}]^{\mathrm{T}}\ \hat{\boldsymbol{z}}\ , \tag{10}$$

where Y^{X} is a matrix with the same elements as Y, but with each direction ω substituted by $\mathsf{X}\cdot\omega$. Equation 10 can be derived by pre-multiplying Equation 7 by $[\boldsymbol{f}^r]^{\mathrm{T}}$ on the LHS, by $[\boldsymbol{f}_l]^{\mathrm{T}}$ on the RHS, and replacing Y_l by $\mathsf{Y}_l^{\mathsf{R}}$; or, by applying ZHE to Equation 9.

Our basic algorithm directly implements Equation 10 and, with lobe sharing, requires only a single order-N SH basis function evaluation at each rotated lobe direction. However, a common use case (e.g. in PRT) admits a significant optimization, detailed below.

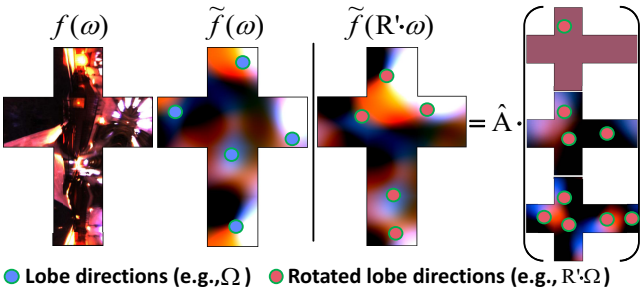

Fig. 4. With signal-tailored SH rotation, per-band SH expansions of the original function are sampled at rotated $\omega_{l,m}$ directions. By avoiding the SH basis function evaluations and a dense matrix-vector multiplication, this formulation affords a large performance improvement on the GPU.

Signal-tailored rotation. When the initial, canonically oriented SH vector $\boldsymbol{f}$ corresponds to a static function (e.g., a constant environmental light source), as opposed to a dynamically changing function (e.g., binary visibility on an animating character, or spatially-varying BRDF), a significant acceleration can be realized:

$$\begin{aligned}\boldsymbol{f}_l^r &= \mathsf{I}\ \boldsymbol{f}_l^r = \hat{\mathsf{A}}_l\ \mathsf{Y}_l\ \boldsymbol{f}_l^r = \hat{\mathsf{A}}_l\ \mathsf{Y}_l^{\mathsf{R}'}\ \boldsymbol{f}_l \\ &= \hat{\mathsf{A}}_l\ \left[\boldsymbol{f}_l\cdot\boldsymbol{y}_l(\mathsf{R}'\cdot\omega_{l,-l}),\ \ldots,\ \boldsymbol{f}_l\cdot\boldsymbol{y}_l(\mathsf{R}'\cdot\omega_{l,l})\right]^{\mathrm{T}} \\ &= \hat{\mathsf{A}}_l\ \left[\tilde{f}_l(\mathsf{R}'\cdot\omega_{0,0}),\ \ldots,\ \tilde{f}_l(\mathsf{R}'\cdot\omega_{l,l})\right]^{\mathrm{T}}\ ,\end{aligned} \tag{11}$$

where the band-l expansion $\tilde{f}_l(\omega)$ is pre-tabulated (e.g., in a cubemap) and the rotated coefficient vector $\boldsymbol{f}_l^r$ can be computed by simply sampling the canonical SH expansion at the inversely-rotated lobe directions (e.g., $\mathsf{R}'\cdot\boldsymbol{\Omega}$), avoiding all SH function evaluations! Equation 11's first line exploits the $\hat{\mathsf{A}}_l = \mathsf{Y}_l^{-1}$ property and states that SH expansion with rotated coefficients is equivalent to expansion of the unrotated coefficients evaluated in the rotated directions.

Pre-tabulated band-l expansions[3], $\tilde{f}_l(\omega)$, further increase the performance of our GPU implementation. The data-parallel nature of Equations 10 and 11 allows millions of rotations to be computed simultaneously on the GPU (see Figure 5). Moreover, signal-tailored pre-tabulation affords an additional layer of parallelism as several SH expansions can be stored and rotated concurrently.

For example, to rotate an RGB environment light, we pre-tabulate its per-band SH expansions into RGB cubemaps, and then rotate all three color signals simultaneously (at many surface points) on the GPU (see Figure 4). In summary, obtaining rotated coefficients of a band-limited function reduces to sampling its per-band reconstructions and performing ZHF.

Performance benchmark. We implemented our basic algorithm on the CPU (scalar processing) and GPU, and signal-tailored rotation on the GPU, both leveraging sparse matrix-vector multiplication and lobe sharing. We benchmark these three implementations against a CPU implementation of low-order SH rotation from the DirectX SDK, the general SH rotation framework of Lisle and Huang [2007], and optimized CPU and GPU implementations of the $zxzxz$-decomposition approach [Kautz et al. 2002] provided to us by the anonymous reviewers. In practice, we directly rotate the *optimal linear lobe* [Sloan et al. 2005] for band-1 rotation, and only compare performance for $N > 2$. A comprehensive performance comparison plot is included in our Supplemental Material, and we

[3]We use a cubemap mip-map with max resolution 6×1024^2; determining optimal per-band resolutions is an interesting problem left to future work.

Table II. # of floating point multiplications for stages of the $zxzxz$ algorithm, and our ZHF rotation algorithms.

Order (N)	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
***zxzxz* Rotation**																	
z-rotation	34	52	74	100	130	164	202	244	290	340	394	452	514	580	650	724	802
z-rotation (opt.)*	16	32	52	76	104	136	172	212	256	304	356	412	472	536	604	676	752
x^{**}-rotation	5	18	39	71	115	171	243	334	445	576	733	916	1757†	2718	3807	5032	6401
Total	112	192	300	442	620	834	1092	1400	1760	2172	2648	3188	5056†	7176	9564	12236	15208
ZHF Rotation																	
SH Basis Evaluation	14	28	47	71	100	134	173	217	266	320	379	443	512	586	665	749	838
$[\mathsf{Y}^{\mathrm{R}}]^{\mathrm{T}}$ multiplication‡	25	74	155	276	445	670	959	1320	1761	2290	2915	3644	4485	5446	6535	7760	9129
Sparse $\hat{\mathsf{A}}$ multiplication (**Signal-tailored algorithm**◇)	9	33	75	137	246	384	570	820	1118	1453	1849	2333	2851	3524	4199	5030	6014
Standard algorithm (CPU)	90	275	606	1123	1891	2930	4297	6046	8199	10783	13860	17495	21672	26550	32014	38256	45311
Standard algorithm (GPU)	34	107	230	413	691	1054	1529	2140	2879	3743	4764	5977	7336	8970	10734	12790	15143

∗ z-rotations can be optimized (compared to the routines we benchmark against) by exploiting recurrences in the (co)sine computations.

∗∗ Only one entry is shown per order for x-rotations, since the # of multiplications only varies by ± 2 between $+$ and $-$ x-rotations (for fixed N).

† We only optimize the sparsity of x-rotations up to $N < 15$, with slower loops used to (densely) compute the rotations for bands $l \geq 14$.

‡ Y^{R} is used in our (non signal-tailored) implementations; however on the GPU, its construction avoids SH basis evaluations by using tabulated functions.

◇ For GPU signal-tailored rotation, the only multiplications (apart from rotating the lobe directions) are incurred by the optimized sparse $\hat{\mathsf{A}}$ multiplication.

choose to focus our comparison against the most competitive state-of-the-art $zxzxz$ rotation approach used in CG.

Figure 5 compares performance (in rotations per second) of our CPU and GPU algorithms with optimized implementations of the $zxzxz$ approach (provided to us by the anonymous reviewers) on an Intel core i7 laptop with 6 GB of RAM and an nVidia Quadro FX 1800M with 1 GB of VRAM. Note that performance is plotted on a $\mathbf{log}_{10}$ scale. CPU performance is computed as an average of 10^6 random rotations (using the same random rotations for each technique) and GPU performance numbers include the cost of reconstructing the rotated spherical signal in a shader.

Our standard algorithm outperforms an optimized $zxzxz$ CPU implementation until a break-even point between $N = 7$ and 8 on the CPU[4]. On the GPU, signal-tailored rotation is consistently faster than all other GPU algorithms. The reviewer-provided implementation of $zxzxz$ has optimized, hard-coded sparse matrix-vector multiplications for the x-rotation routines, up to $N = 14$. We only include these optimized performance measurements (denoted using the $\diamondsuit$ symbol) in Figure 5, and our comprehensive performance plot in our Supplemental Material also includes performance numbers beyond $N = 14$ (where loops are used to rotate all bands $l \geq 14$) on the CPU and GPU. Our standard (non-signal-tailored) rotation algorithm surpasses shader compilation limits at $N = 16$ (marked with $\square$), as does the $zxzxz$ approach at $N = 18$ (see the Supplemental Material). For $N \leq 7$, our standard algorithm outperforms $zxzxz$ on the GPU before the cost of performing dense matrix-vector multiplication with $[\mathsf{Y}^{\mathrm{R}}]^{\mathrm{T}}$ begins to dominate. As noted in Table II, an additional optimization can be realized for the z-rotation of the $zxzxz$ implementation by leveraging a recurrence when computing the sines/cosines of multiples of the rotation angle. We did not implement this optimization and benchmarked directly against the code kindly provided to us by the anonymous reviewers, however including such an optimization may lead to an earlier cross-over point between our algorithm and $zxzxz$.

Performance discussion. Table II compares the number of floating point multiplication operations between the different components of our rotation algorithms, as well as the optimized $zxzxz$ rotation algorithm. The steep increase in operations at $N = 14$ for the x-rotation component of $zxzxz$ is again due to the lack of optimized sparse matrix-vector computation. Both our CPU and GPU ZHF implementations hard-code the sparse matrix-vector multiplications (with $\hat{\mathsf{A}}^{\mathrm{T}}$ and $\hat{\mathsf{A}}$, respectively) with code provided for $N = 3$ to 6 in the Supplemental Material. Furthermore, we use optimized SH basis function routines, generated programmatically, to evaluate the rotated lobe directions in our standard rotation algorithm (multiplication operations for the code generated SH basis function routines are also included in Table II). The multiplication operation count in our standard algorithm on the CPU includes the cost of evaluating SH basis functions at these rotated directions to generate the elements of Y^{R}, as well as performing the dense matrix-vector multiplication with $[\mathsf{Y}^{\mathrm{R}}]^{\mathrm{T}}$; in contrast, we use tabulated SH basis functions on the GPU and when counting multiplication operations for this GPU implementation.

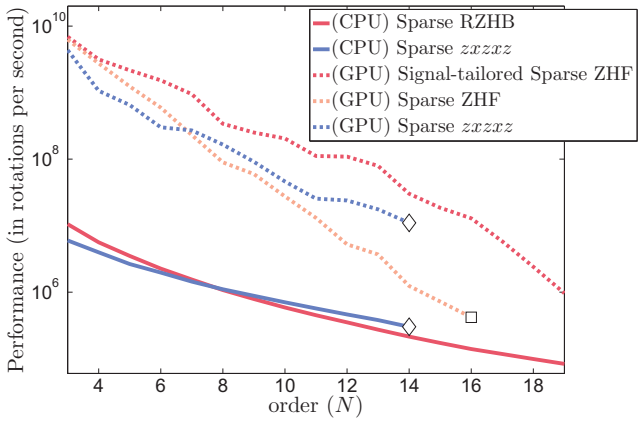

Fig. 5. Performance comparison, in rotations per second, on a $\mathbf{log}_{10}$ scale.

We note that our approaches have asymptotic complexity equal to the leading $zxzxz$ approach. We performed a thorough performance analysis to determine the major contributors to our relative performance increase over $zxzxz$. Algorithmically, $zxzxz$ performs Euler angle decomposition, followed by five sparse matrix-vector multiplies: three z-rotations and two fixed $\pm 90^\circ$ x-rotations. Hard-

[4]Our CPU implementation does not leverage SSE vectorization, however we have experimented with a naïve vectorization across 4 input SH coefficients yielding a speed up of nearly 3× (not included in performance plots).

coding functions for the sparse x-rotation matrix-vector multiplications yields a significant performance increase over algorithmic matrix composition with loops. We use these optimized x-rotation matrices up to $N = 14$ on the CPU and GPU.

Our signal-tailored rotation only requires a single, *fixed* sparse matrix-vector multiplication, which we precompute using a code generator. This is not true of our standard algorithm, which requires (the transpose of) this multiplication (also precomputed with code generation), as well as a dense multiplication by $[\mathsf{Y}^{\mathrm{R}}]^{\mathrm{T}}$. The cost of generating and performing this dense matrix multiply becomes the bottleneck on the CPU and GPU at around $N = 7$. For our standard rotation implementation on the GPU we use precomputed cubemaps for SH basis function evaluation although, from some initial experiments, the circumstances under which these lookups become beneficial compared to arithmetic evaluation of the basis functions is both unclear and inconsistent. On the other hand, our signal-tailored implementation only requires texture lookups and one static, precomputed, sparse optimized matrix-vector multiplication, yielding much simpler and more predictable execution behavior. We have also experimented with using tabulated SH basis functions on the CPU with mixed results.

We believe that investigating the trade-offs between the use of loops, tabulated data, and arithmetic operations may lead to significant performance increases, both on the CPU and GPU. Given this, while Table II gives a good sense of the scalability of the ZHF and $zxzxz$ algorithms, additional issues such as memory and cache usage, bandwidth utilization, using loops with branching versus unrolled loops, and using precomputed tables versus on-the-fly arithmetic evaluation all factor into the final observed performance.

Error. We target applications of low-order SH to CG, affording a slightly relaxed level of accuracy compared to other scientific areas. As in [Lessig et al. 2010], we ensure that our choice of $\mathbf{\Omega}$ yields a well conditioned Y, enabling stable rotation; as mentioned in Section 5, ill-conditioned Y_l are not considered in our statistics. Visually, our rotation is temporally coherent (i.e., no "wobbling"), and our results match all the approaches we have benchmarked against (to within 6 digits of accuracy using 32-bit float frame buffers).

7.2 Coupling RZHB and SH for Shading

PRT techniques compute outgoing radiance with dynamic reflectance and lighting using compact basis representations. Under certain circumstances outlined below, it is more efficient to perform this computation in local, spatially-varying coordinate frames.

Shading Overview. Direct light at point x in direction ω_o is a triple product integral of lighting L_i, visibility V, and BRDF f_r,

$$L_o(x,\omega_o) = \int_{S^2} L_i(x,\omega)\, V(x,\omega)\, f_r(x,\omega,\omega_o)\, \lfloor n\cdot\omega \rfloor d\omega$$
$$\approx \sum_{ijk} [\boldsymbol{l}]_i\, [\boldsymbol{v}]_j\, [\bar{\boldsymbol{f}}]_k \underbrace{\int_{S^2} y_i(\omega)\, y_j(\omega)\, y_k(\omega) d\omega}_{\Gamma_{ijk}}\,, \quad (12)$$

where $\boldsymbol{l}$, $\boldsymbol{v}$, and $\bar{\boldsymbol{f}}$ are the SH projection vectors of the three terms[5], and Γ is the order-3 tripling coefficient tensor. If only one term is dynamic (e.g., lighting), a double product integral can be computed,

$$L_o(x,\omega_o) = \int_{S^2} L_i(x,\omega)\, T(x,\omega,\omega_o) d\omega \approx \boldsymbol{l}\cdot\boldsymbol{t}\,, \quad (13)$$

[5]Typically, the BRDF is combined with the cosine foreshortening term.

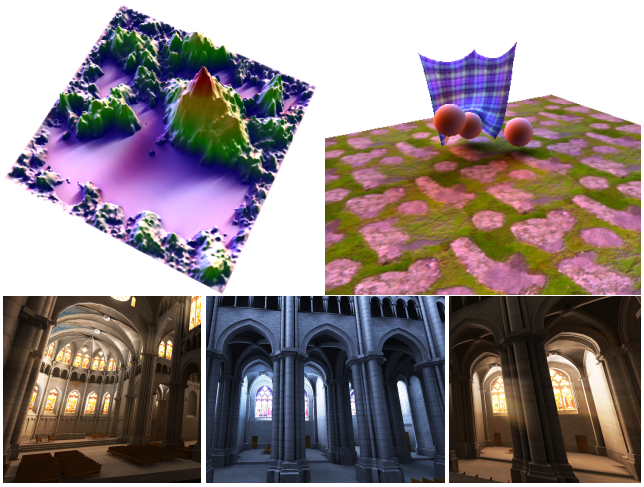

Fig. 6. Applying ZHF to several rendering applications (see Section 7.3. With permission of Benoït Mayaux [2010]; http://www.patapom.com/O3D/Cathedral.html).

where T is the visibility (and cosine) weighted BRDF term, also called the "transfer", and $\boldsymbol{t}$ is its SH projection coefficient vector.

In x's local frame, f_r is a four-dimensional function, as opposed to six-dimensional in the global frame. Furthermore, some BRDFs admit simpler analytic formulae in the local frame. In these scenarios, evaluating Equations 12 or 13 in the local frame is ideal. However, local-frame shading requires the lighting (and, in the case of dynamic geometry, visibility) to be rotated at each shading point.

Unlike previous work, we can dynamically rotate several functions into the local-coordinate frame at each pixel on the GPU.

We derive coupling matrices for when one or both functions in Equation 13 are represented in the RZHB (see Supplemental Material). Two common use cases of (LD)PRT are dynamic lighting and deformable local transfer/reflectance/visibility functions, where an efficient solution is to represent one of the functions in the RZHB and rotate into the local frame with Equations 10 or 11.

7.3 Rendering Applications

We apply ZHF and the efficient rotation algorithms to several diverse rendering scenarios, exhibiting the flexibility of our solution.

We augment an open-source St. Jean Cathedral demo (Figures 1 and 6), which executes completely in a web browser using Google's O3D javascript API, showcasing the modest computation requirements of ZHF rotation. In order to support "common denominator" hardware, O3D is restricted to Shader Model 2.0. Regardless, order-9 rotations are computed on the CPU with javascript (Equation 10), and the final shade is reconstructed using multi-pass Pixel Shader 2.0 kernels and Habel et al.'s [2008] sky model. ZHF is suitable for low-end console platforms that impose similar restrictions.

The cloth demo (Figure 6, top right) computes indirect light and shadows using [Sloan et al. 2007] and [Ren et al. 2006] for rigid objects (respectively), and a variant of [Nowrouzezahrai and Snyder 2009] for cloth geometry. World-space order-8 rotations are computed on the CPU, and then rotated into the tangent frame at every (foreground) pixel using signal-tailored rotation. On our laptop GPU, we maintain over 145 FPS in this demo. The height field demo (Figure 6, top left) computes order-9 rotations on the GPU for $256^2 = 65.5$K shade points at over 120 FPS.

Our rotation approach is compatible with existing techniques for computing dynamic SH visibility and BRDFs in a local-frame representation [Ren et al. 2006; Ramamoorthi and Hanrahan 2002].

We use 32-bit floating point operations/storage (including textures) and, other than hard-coding the sparse matrix-vector multiplication, no effort was made to "hand tune" or optimize our code.

Discussion. The performance drawbacks of current SH rotation, most notably their inability to map naturally to the GPU, imposes many limitations (sometimes indirectly) on real-time rendering applications. Ren et al. [2006] are forced to compute an SH light-product matrix every frame for every light, whereas lighting can be rotated into the local frame with our approach, where diffuse (or, e.g. Phong) lobe product matrices can be more efficiently evaluated. LDPRT [Sloan et al. 2005] is restricted to single-lobe reflectance models and approximate visibility since the non-linear lobe fitting approach is neither lossless nor well-suited for lighting functions. With ZHF, lighting can be rotated efficiently on both the CPU and GPU (or both simultaneously). Thus, more complex reflectance/visibility models can be supported without any loss in accuracy. Signal-tailored GPU rotation is of particular usefulness, mapping naturally to shader-based engines (as opposed to requiring separate GPGPU kernels and interop), and would be of immediate benefit to gaming and interactive applications. Moreover, with the integration of WebGL and HTML5 technologies, high-quality web-based rendering will seamlessly benefit from our approach.

8. CONCLUSION AND FUTURE WORK

We introduce an approach for determining lobe directions and weights such that any order-N SH expansion can be represented perfectly as a sum of static, precomputed rotated ZH lobes. We show how to promote sparsity in this mapping, yielding interesting theoretical and practical results. Signal-tailored rotation, which can be viewed as a form of spherical polynomial interpolation, is designed to leverage shader-based rendering architectures.

Our mathematical exposition is straightforward, especially compared to prior work in SH rotations, and the resulting algorithms are very easy to understand and implement. We outperform current state-of-the-art rotation algorithms, especially on the GPU.

Source code, sparsity-optimized lobe directions for $N \leq 8$, and the MATLAB code used to generate this (and higher-order) data is included in the Supplemental Material. This allows for rapid integration of our technique into existing rendering systems.

In the future, we plan to exploit ZHF for sparse data interpolation/projection, and optimal signal-specific lobe fitting (e.g., choosing directions according to the zeros of arbitrary SH expansions).

Acknowledgements

We thank Michael Kazhdan for early discussions, Andrew Willmott and EA/Maxis for generously providing planet data from Spore™ for initial experiments, Benoït Mayaux for the O3D Cathedral demo starter code, Peter-Pike Sloan for his input on SH rotation and his optimized SH basis evaluation code generator, Gaël Chaize for the Cathedral model, Paul Debevec for the environment maps, Ian Lisle and Tracy Huang for their PRT implementation, and the anonymous reviewers for their helpful comments, suggestions, and source code/executables for benchmarking *zxzxz* performance.

REFERENCES

EDMONDS, A. 1960. *Angular Momentum in Quantum Mechanics*. Princeton University Press.

HABEL, R., MUSTATA, B., AND WIMMER, M. 2008. Efficient Spherical Harmonics Lighting with the Preetham Skylight Model. In *Eurographics Short Papers*.

KAJIYA, J. T. AND VON HERZEN, B. P. 1984. Ray tracing volume densities. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '84. ACM, New York, NY, USA.

KAUTZ, J., SLOAN, P.-P., AND SNYDER, J. 2002. Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In *Proceedings of the 13th Eurographics workshop on Rendering*. EGRW '02. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.

KAZHDAN, M. 2007. An approximate and efficient method for optimal rotation alignment of 3d models. *IEEE Trans. Pattern Anal. Mach. Intell. 29*.

KŘIVÁNEK, J., KONTTINEN, J., BOUATOUCH, K., PATTANAIK, S., AND ŽÁRA, J. 2006. Fast approximation to spherical harmonics rotation. In *ACM SIGGRAPH 2006 Sketches*. ACM, New York, NY, USA.

LACROIX, N. H. J. 1984. On common zeros of Legendre's associated functions. *Mathematics of Computation 43,* 167.

LESSIG, C., DEWITT, T., AND FIUME, E. 2010. Efficient and stable rotation of finite spherical harmonics expansions. *University of Toronto - Tech. Report. http://www.dgp.toronto.edu/∼lessig/shrk/*.

LISLE, I. G. AND HUANG, S.-L. T. 2007. Algorithms for spherical harmonic lighting. In *GRAPHITE '07*. ACM, New York, NY, USA.

MAYAUX, B. 2010. Saint Jean Cathedral - O3D Web Demo. http://www.patapom.com/O3D/Cathedral.html.

MICHALEWICZ, Z. 1998. *Genetic Algorithms + Data Structures = Evolution Programs*, 3 ed. Springer.

NOWROUZEZAHRAI, D. AND SNYDER, J. 2009. Fast global illumination on dynamic height fields. *Computer Graphics Forum: Eurographics Symposium on Rendering*.

RAMAMOORTHI, R. AND HANRAHAN, P. 2002. Frequency space environment map rendering. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '02. ACM, New York, NY, USA.

REN, Z., WANG, R., SNYDER, J., ZHOU, K., LIU, X., SUN, B., SLOAN, P.-P., BAO, H., PENG, Q., AND GUO, B. 2006. Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. SIGGRAPH '06. ACM, New York, NY, USA.

SLOAN, P.-P., GOVINDARAJU, N. K., NOWROUZEZAHRAI, D., AND SNYDER, J. 2007. Image-based proxy accumulation for real-time soft global illumination. In *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*. IEEE, Washington, DC, USA.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '02. ACM, NY, USA.

SLOAN, P.-P., LUNA, B., AND SNYDER, J. 2005. Local, deformable precomputed radiance transfer. In *ACM SIGGRAPH 2005 Papers*. ACM, New York, NY, USA.

TSAI, Y.-T. AND SHIH, Z.-C. 2006. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. SIGGRAPH '06. ACM, New York, NY, USA.

WANG, R., NG, R., LUEBKE, D., AND HUMPHREYS, G. 2006. Efficient wavelet rotation for environment map rendering. In *In Eurographics Symposium on Rendering, Eurographics Association*. Springer-Verlag.

WESTIN, S. H., ARVO, J. R., AND TORRANCE, K. E. 1992. Predicting reflectance functions from complex surfaces. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '92. ACM, New York, NY, USA.

Received December 2010; accepted Month Year