Efficient Wavelet Rotation for Environment Map Rendering

Conference Paper · January 2006

DOI: 10.2312/EGWR/EGSR06/173-182 · Source: DBLP

4 authors, including:

Rui Wang
Linyi University
236 PUBLICATIONS **6,669** CITATIONS

SEE PROFILE

David Luebke
NVIDIA
48 PUBLICATIONS **3,692** CITATIONS

SEE PROFILE

Eurographics Symposium on Rendering (2006)
Tomas Akenine-Möller and Wolfgang Heidrich (Editors)

Efficient Wavelet Rotation for Environment Map Rendering

Rui Wang[1], Ren Ng[2], David Luebke[1], and Greg Humphreys[1]

[1]Department of Computer Science, University of Virginia
[2]Department of Computer Science, Stanford University

Abstract

Real-time shading with environment maps requires the ability to rotate the global lighting to each surface point's local coordinate frame. Although extensive previous work has studied rotation of functions represented by spherical harmonics, little work has investigated efficient rotation of wavelets. Wavelets are superior at approximating high frequency signals such as detailed high dynamic range lighting and very shiny BRDFs, but present difficulties for interactive rendering due to the lack of an analytic solution for rotation. In this paper we present an efficient computational solution for wavelet rotation using precomputed matrices. Each matrix represents the linear transformation of source wavelet bases defined in the global coordinate frame to target wavelet bases defined in sampled local frames. Since wavelets have compact support, these matrices are very sparse, enabling efficient storage and fast computation at run-time. In this paper, we focus on the application of our technique to interactive environment map rendering. We show that using these matrices allows us to evaluate the integral of dynamic lighting with dynamic BRDFs at interactive rates, incorporating efficient non-linear approximation of both illumination and reflection. Our technique improves on previous work by eliminating the need for prefiltering environment maps, and is thus significantly faster for accurate rendering of dynamic environment lighting with high frequency reflection effects.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism;

1. Introduction and Related Work

Realistic image synthesis of computer-generated scenes often requires rich lighting environments. Image-based lighting, which represents natural illumination by detailed distant environment maps [DM97], is widely used for generating convincing synthesized images. Environment maps have long been used in computer graphics to simulate realistic reflection effects [BN76, Gre86]. However, interactive rendering with complex environment maps remains challenging because it requires evaluating an expensive integral over all possible lighting directions at every surface point. Furthermore, this integral requires the lighting environment to be rotated into each surface point's local coordinate frame. Direct resampling of the global lighting in each local frame is obviously too costly in interactive settings. Therefore, fast rotation of lighting at run-time is crucial for incorporating image-based lighting to real-time applications. In the following, we provide a brief review of related work, and then present our own contributions.

Prefiltered Environment Maps. Due to the difficulty of computing the lighting integral dynamically at run-time, researchers have traditionally used *prefiltered* environment maps to simulate glossy reflections in real-time [CON99, HS99, KM00, KVHS00]. Prefiltering is an expensive process, therefore it is typically computed offline, preventing the user from dynamically changing the lighting or BRDFs on the fly. Fast prefiltering algorithms [KVHS00] have been proposed but are limited to specific classes of BRDFs and cannot be generalized.

Spherical Harmonics Basis Projection. Basis projection techniques, such as spherical harmonics (SH) [RH01, RH02, KSS02], represent lighting and transport effects as low dimensional vectors projected onto a spherical function basis, and efficiently approximate the lighting integral by computing the inner products of these vectors. Our work is related to the *spherical harmonic reflection map* (SHRM) presented by Ramamoorthi and Hanrahan [RH02]. SHRM stores the reflected radiance distribution for each normal direction on

Figure 1: *These images show a 120,000 vertex dragon model rendered using our environment map rendering algorithm. The view-dependent rendering speed is maintained at 5 fps, and we can dynamically change the lighting at 1.8 fps. The four images are rendered with a yellow plastic, skin, measured blue metallic, and measured nickel BRDF respectively, demonstrating both low and high frequency reflection effects. The renderings have been attenuated by precomputed ambient occlusion to produce shadowing effects. Our algorithm is based on efficient rotation of wavelets using precomputed matrices. The model, viewpoint, lighting environment, and surface BRDFs can all be modified interactively at run-time.*

a SH basis. Using spherical harmonics allows for rapid prefiltering of low-frequency environment maps at close to interactive rates. They have also theoretically analyzed required sampling rates using frequency space analysis.

While considerable progress has been made in the use of these techniques for interactive high-quality imagery, they do not automatically solve the rotation problem. Extensive previous work has studied SH rotation and presented both analytic and computational solutions [Edm60, CIGR99, KSS02, KKB*06]. However, as Sloan et al. [SLS05] pointed out, SH rotation remains costly for real-time evaluation at every surface point. Furthermore, in practice SH techniques are limited to approximating only low-frequency signals. For high-frequency lighting and BRDFs, an accurate approximation requires many more SH coefficients, and the rotation cost increases even more rapidly.

Gautron et al. [GKPB04] presented a novel hemispherical basis for efficient representation of hemispherical functions. Sloan et al. [SLS05] proposed an alternative basis called the *zonal harmonics* that allows for fast rotation in real-time. However, like spherical harmonics, these bases remain inefficient at representing high-frequency functions.

Wavelet Basis Projection. Wavelets [SDS96] are generally considered superior at approximating high-frequency signals, such as detailed high dynamic range (HDR) images or very specular BRDFs. Accurate rendering of high-frequency lighting effects is achievable with far fewer wavelet coefficients than with spherical harmonics. However, efficient rotation of wavelets has not been well studied, largely due to the lack of an analytic solution for wavelet rotation.

Ng et al. [NRH03] first proposed non-linear wavelet approximation of the lighting in the context of precomputed radiance transfer (PRT) [SKS02]. PRT assumes a static scene and thus avoids the rotation problem by baking the rotation into the per-vertex precomputed transport function. Interactive glossy rendering can be handled by baking a low-order BRDF basis into the transport functions [SKS02, LK03, LSSS04, WTL04]; however, these techniques are limited to low-frequency BRDFs and reflection effects. Green et al. [GKMD06] proposed an alternative basis using non-linear Gaussian functions. This technique requires prefiltering environment maps with Gaussian functions, which is not currently suitable to support fully dynamic lighting.

The triple product wavelet integral framework by Ng et al. [NRH04] handles high-frequency reflections, but dodges the rotation problem by precomputing the BRDF for a set of sampled surface orientations. This approach requires a huge amount of memory to store each BRDF. Clarberg et al. [CJAMJ05] avoid the rotation problem by precomputing rotated environment maps for a set of sampled surface orientations. Due to extensive super-sampling, these pre-rotated environment maps can take up to a couple of hours to create, disabling run-time dynamic lighting.

In this paper, we describe a computational solution to rotating spherical functions on a wavelet basis using precomputed rotation matrices. To our knowledge, this is the first attempt to solve the wavelet rotation problem. Our precomputed matrices represent the linear transformation of wavelet bases from the global coordinate system to a sampled set of local frames. Since wavelets have compact support, the resulting matrices are generally very sparse, enabling efficient storage and fast on-the-fly computation. By incorporating non-linear wavelet approximation of the lighting and BRDFs, we enable interactive environment map rendering with complex, realistic BRDFs and detailed natural lighting. Our technique improves on previous work by eliminating the need for prefiltering environment maps and produces accurate high-frequency reflection effects. Furthermore, the viewpoint, model, lighting environment, and surface BRDFs can all be modified interactively at run-time.

Although we focus on environment mapping as a driving problem, we believe that the ability to efficiently compute wavelet rotations will prove broadly applicable in interactive computer graphics and image processing. We close with a brief discussion of possible avenues for future work.

2. Rotation of Wavelet Bases

In this section we derive formulae for rotating a general spherical function basis. This results in a basis transformation matrix for each local frame. In the case of a wavelet basis, the resulting matrix is very sparse. Because our driving application is environment mapping, we make the standard assumptions that illumination is distant and direct, and we ignore self-shadowing and interreflections.

Reflection Equation We begin with the reflection equation for distant illumination:

$$B(\mathbf{n},\omega_o) = \int_{\Omega(\mathbf{n})} \widetilde{L}(\mathbf{n},\omega)\, f_r(\omega,\omega_o)\, (\omega\cdot y)\, d\omega. \qquad (1)$$

This equation describes the reflected light B as an integral over the upper hemisphere at a local frame defined by surface orientation $\mathbf{n}$. Here $\mathbf{n}$ is expressed in the global coordinate system, and incident direction ω and view direction ω_o are both expressed in the local frame. $\widetilde{L}$ is the incident lighting after rotation into the local frame; f_r is the surface BRDF; and $\omega\cdot y$ is the incident cosine term (in the local frame, the y axis corresponds to the surface normal $\mathbf{n}$).

Typically the cosine term $\omega\cdot y$ is premultiplied or "baked" into the definition of the BRDF, making Equation 1 essentially a double product integral of the lighting with the BRDF. Given known local lighting $\widetilde{L}$ and view direction ω_o, we can further apply frequency space approximations, such as spherical harmonics (SH) or wavelets, to represent both $\widetilde{L}$ and f_r as vectors $\widetilde{\mathbf{L}}$ and $\mathbf{F_r}$ over these bases. The integral then reduces to a simple dot product of the two vectors: $B = \mathbf{F_r}\cdot\widetilde{\mathbf{L}}$.

Basis Rotation Lighting $\widetilde{L}(\mathbf{n},\omega)$ in the local frame is related to the global lighting L by a rotation:

$$\widetilde{L}(\mathbf{n},\omega) = L(\mathrm{R}_{(\mathbf{n})}\cdot\omega). \qquad (2)$$

Here $\mathrm{R}_{(\mathbf{n})} = [\mathbf{s},\mathbf{n},\mathbf{t}]$ is simply a 3×3 matrix that transforms local coordinates to global coordinates, where $\mathbf{n}$ is the normal, and $\mathbf{s}$ and $\mathbf{t}$ are tangent vectors of the local frame. Since most of our models do not come with user defined tangent vectors, we use standard techniques to generate $\mathbf{s}$ and $\mathbf{t}$ from the normal $\mathbf{n}$, therefore each local frame is uniquely defined and indexed by $\mathbf{n}$. In the following, we focus on a fixed local frame (defined by $\mathbf{n}$) and a fixed view direction ω_o. In this case, we can write the rotation $\mathrm{R}_{(\mathbf{n})}$ conveniently as R, and the BRDF $f_r(\omega,\omega_o)$ as $f_r(\omega)$.

We first project the global lighting L onto an orthonormal basis set ψ_i (called the *source basis*) that is defined in the global coordinate system:

$$L(\mathrm{R}\cdot\omega) = \sum_i L_i\,\psi_i(\mathrm{R}\cdot\omega). \qquad (3)$$

Since the basis $\psi_i(\mathrm{R}\cdot\omega)$ is itself a spherical function, it can be further projected onto a (possibly different) orthonormal basis set $\varphi_j(\omega)$ (called the *target basis*) that is defined in the local coordinate frame:

$$\psi_i(\mathrm{R}\cdot\omega) = \sum_j R_{ij}\,\varphi_j(\omega). \qquad (4)$$

The coefficients R_{ij} form a matrix that we call the *basis transformation matrix*. Each R_{ij} stores the projection of source basis ψ_i onto target basis φ_j under rotation R. In other words, this matrix represents the linear transformation of source bases to target bases. Substituting Equation 4 into Equation 3 and rearranging terms, we express the local lighting over the target basis as:

$$L(\mathrm{R}\cdot\omega) = \sum_j \left(\sum_i R_{ij}\,L_i\right)\varphi_j(\omega). \qquad (5)$$

We also project the BRDF onto the target basis set:

$$f_r(\omega) = \sum_k \rho_k\,\varphi_k(\omega). \qquad (6)$$

Since local lighting and BRDF are now expressed in the same orthonormal basis, we can write their integral in a compact matrix form as:

$$B = \int L(\mathrm{R}\cdot\omega)\, f_r(\omega)\, d\omega = \sum_k \rho_k \left(\sum_i R_{ik}\,L_i\right) = \mathbf{F_r}\cdot(\mathbf{R}\times\mathbf{L}) \qquad (7)$$

where $\mathbf{F_r}$ is the column vector (ρ_k) representing the BRDF in the target basis; $\mathbf{L}$ is the column vector (L_i) representing the global lighting in the source basis; and $\mathbf{R}$ is the basis transformation matrix (R_{ik}) that relates the two bases under rotation R. Figure 2 illustrates the rotation in the wavelet basis domain vs. the corresponding rotation in the spatial domain. Intuitively, since L is expressed as a linear combination of

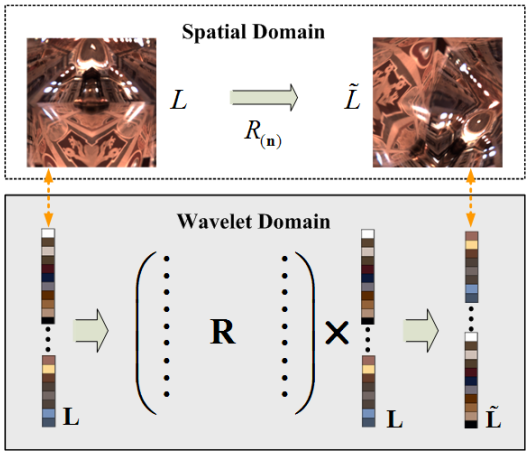

Figure 2: *Rotation of a spherical function L in the wavelet domain. L is initially represented as a column vector* **L** *in the source wavelet basis. A precomputed rotation matrix* **R**, *corresponding to the spatial domain rotation* $R_{(\mathbf{n})}$, *is then multiplied with* **L**. *This computes a new column vector* $\tilde{\mathbf{L}}$ *represented in the target wavelet basis, which directly corresponds to the rotation of L by* $R_{(\mathbf{n})}$ *in the spatial domain.*

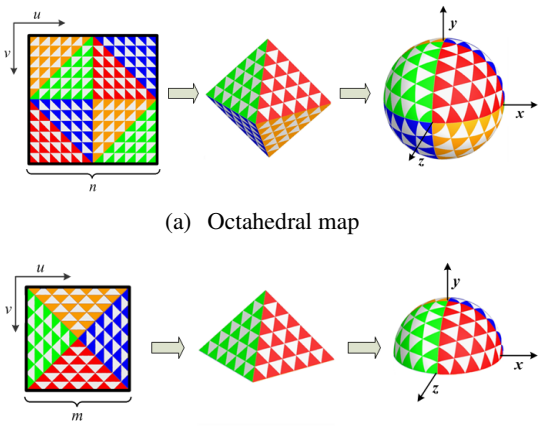

Figure 3: *(a) The octahedral map is a parametrization of the sphere onto a square domain of* $n \times n$ *resolution. It is used to represent our source lighting. (b) The hemi-octahedral map is a parametrization of the upper hemisphere onto a square domain of* $m \times m$ *resolution. It is used to represent our target lighting and BRDFs. Images courtesy of Emil Praun and Hugues Hoppe [PH03].*

the source basis functions, any transformation of L (e.g., rotation) is achievable in its basis domain by transforming the basis functions and then linearly combining the results.

Notice that the source and target bases may lie in different domains with different sampling resolutions, thus **R** may not be a square matrix. For example, the source basis usually lies in the spherical domain where the global lighting is defined, while the target basis lies in the hemispherical domain where the BRDF is defined. They may even be two entirely different basis sets.

SH Rotation vs. Wavelet Rotation SH rotation has been studied extensively [CIGR99, KSS02, KKB*06]. An order n SH approximation is composed of n bands, with each band l = 1, 2, ..., n containing $(2l-1)$ bases. Thus an order n SH approximation is composed of a total of $N = n^2$ bases. Rotation of an order n SH basis can be completely represented by SH of the same order; thus, the corresponding rotation matrix **R** is an $N \times N$ square matrix. Additionally, **R** has the property that it is partitioned into n square sub-matrices, because bases from each band project to zero on all other bands. The total number of non-zero coefficients in **R**, and hence the computational cost for SH rotation, is $O(n^3) = O(N^{\frac{3}{2}})$.

Although efficient SH rotation algorithms have been studied, the computational cost increases rapidly as n becomes large. Furthermore, a substantial number of SH coefficients are necessary to accurately approximate high-frequency functions such as high resolution lighting and BRDFs [NRH03], making SH rotation ill-suited for efficiently computing high-frequency effects.

Wavelets, on the other hand, are known to be superior at approximating high-frequency signals. However, rotation in the wavelet domain seems to have received little attention; indeed it is generally assumed that no efficient rotation procedure exists for wavelets. The main reason is that wavelets for representing spherical functions are usually parameterized in domains for which it is difficult or impossible to obtain an analytic rotation formula. If we had a perfect uniform parametrization of the sphere to a 2D domain, rotation of wavelets would reduce to translation in 2D, and an analytic solution would be possible. However, such a uniform parametrization does not exist.

This paper takes a computational approach toward this problem, and argues that a solution using precomputed rotation matrices is easy to implement and can work well in practice. One key insight is that wavelet bases have local support, which leads to a sparse rotation matrix **R**. In fact, as we will discuss later in Section 4, the total number of non-zero elements in **R** only grows linearly with the number of rows. Therefore, the computational cost for wavelet rotation is about $O(N)$, where N is the number of wavelet bases. This is asymptotically faster than spherical harmonics. In addition, using wavelets means that high-frequency lighting and BRDFs can be approximated efficiently with much fewer terms than using SH, reducing the overall cost for computing high-quality reflection effects.

3. Implementation

This section presents several implementation details. Our algorithm uses the Haar wavelet basis due to its simplicity.

Parametrization of Spherical Functions As discussed earlier, wavelet rotation matrices are tied to the parametrization methods used to represent spherical functions. Our mathematical framework does not restrict us to any particular parametrization. However, we favor a parametrization that has low overall distortion, which helps improve the sparsity of precomputed matrices. In addition, a parametrization that lies in a single square domain is more convenient for the wavelet transform. We therefore choose the *octahedral map* introduced in [PH03], which provides both properties in contrast to other parametrization methods such as longitude-latitude, parabolic, cube, or spherical map.

In particular, we use the octahedral map to parameterize our source (global) lighting, as shown in Figure 3(a). And we use a *hemi-octahedral map*, which is slightly modified from the octahedral map, to represent hemispherical functions such as the BRDF and target (local) lighting, as shown in Figure 3(b). Both maps lie in a square domain and have favorable distortion ratio and sampling uniformity. In the following, we use $N = n^2$ to denote our source lighting resolution, and $M = m^2$ to denote the target lighting resolution. Notice that hemispherical sampling of m^2 resolution is equivalent to spherical sampling of $2m^2$ resolution. Because the frequency contents after lighting rotation do not increase, a choice of $m = n$ would be a waste of storage since it results in a higher target resolution than the source resolution. We therefore choose $m = n/2$, or $M = N/4$, which reduces our target sampling rate to be one half of the equivalent source sampling rate. Although this may cause some frequency information to be lost during the rotation, it is still a better choice than $m = n$. We have experimented with various source resolutions, including $N = 32^2, 64^2$ and 128^2.

Precomputing Rotation Matrices As discussed in Section 2, each local frame is uniquely defined by its surface normal, therefore we can index the rotation using just the normal $\mathbf{n}$. Since an analytic solution for wavelet rotation is not known, we use a computational approach where we discretize the surface normal $\mathbf{n}$ and precompute one rotation matrix at a time for each sampled normal. We again use the octahedral map to tessellate the normal $\mathbf{n}$. The typical resolution we use is 32×32. A higher resolution may be necessary if we use higher resolution for source lighting.

Suppose $\mathbf{R}$ is the rotation matrix precomputed for a given orientation $\mathbf{n}$. $\mathbf{R}$ is an $M \times N$ matrix where N and M are the number of source wavelet bases and target wavelet bases respectively. The i-th column of $\mathbf{R}$ corresponds to the projection of the i-th source wavelet basis to all target bases in the local frame. We precompute one column at a time. To do so, we first construct the i-th source wavelet basis in spatial domain at the global coordinate frame. We then resample the source basis at the local frame, which gives a resampled image representing the source basis ψ_i observed by the target frame. Finally we perform a wavelet transform on the resulting image, and the wavelet coefficients directly form the i-th column of matrix $\mathbf{R}$. Figure 4 illustrates this computation.

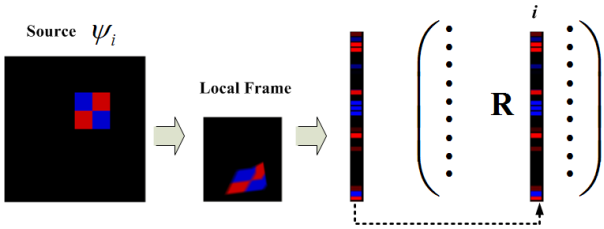

Figure 4: *This diagram shows how to precompute one column of the rotation matrix* $\mathbf{R}$. *We start by constructing the source wavelet (Haar in this example) basis* ψ_i *in the spatial domain. Blue denotes positive values and red negative. The source basis is resampled at a local frame in the target resolution, and wavelet transformed into a sparse vector, which then becomes the i-th column of* $\mathbf{R}$.

The precomputed wavelet rotation matrices are very sparse due to the compact local support of wavelet bases. Our experiments in Section 4 show that the total number of non-zero elements in the matrix only increases linearly with the number of rows M, which means the matrix is $O(N^{-1})$ sparse. For a typical $N = 128^2$ matrix, the sparsity is $0.2\% \sim 0.4\%$. In comparison, a SH rotation matrix is $O(N^{-\frac{1}{2}})$ sparse (N being the total number of SH bases), which is asymptotically worse than wavelets.

To reduce storage requirements for the precomputed matrices, we quantize each matrix element uniformly to a 16-bit integer. After quantization, we store the non-zero elements in row major order, together with its column index. We have also experimented with truncating small matrix terms. Various techniques could be used to accelerate the precomputation, by exploiting the hierarchical structure of 2D wavelets and the fact that the resampled images have compact support. We have left these improvements for future work.

Precomputing BRDFs We approximate BRDFs using wavelets as a preprocess. We first sample each BRDF in the view direction ω_o, which has been tessellated using a hemi-octahedral map. For any given view direction, we then sample the incident direction ω at the target lighting resolution M, resulting in a slice of the BRDF data for the sampled view direction ω_o, similar to [KSS02]. Finally we non-linearly approximate each slice using wavelets. This produces a sparse vector with only a few non-zero coefficients. In practice, keeping only $1 \sim 3\%$ of all wavelet terms suffices for an L^2 accuracy of $> 98\%$ for most specular BRDFs. Thus for a target frame resolution of $M = 64^2$, we only keep $64 \sim 128$ terms per slice. A drawback with non-linear ap-

proximation is that interpolating BRDF slices on the fly will be quite expensive. To avoid interpolation, we sample ω_o at quite high resolution such as 128×128. The storage requirement is still reasonable, typically $16 \sim 32$ MB per BRDF.

Environment Map Rendering To handle dynamic lighting environments, we sample the lighting at run-time onto an $n \times n$ octahedral map. The sampled lighting is projected onto wavelets and non-linearly approximated to a sparse vector containing only a fraction of coefficients. Similar to the BRDF, keeping only $1 \sim 3\%$ of all terms proves accurate enough for even very high-frequency lighting.

Our environment map rendering algorithm consists of two steps. The first step, performed every time the lighting changes, iterates through all precomputed matrices and computes the $\mathbf{R} \times \mathbf{L}$ term of Equation 7 into a buffer. This buffer essentially stores the local lighting $\widetilde{\mathbf{L}}$ (represented in wavelets) for each sampled surface normal. The second step, performed every time the viewpoint changes, then iterates through all vertices of the model and computes the vertex color as the inner product of the view-dependent BRDF and the local lighting. To do so, we use each vertex's local view direction ω_o to index into the precomputed BRDF and extract a BRDF slice $\mathbf{F_r}$. We then use the vertex's normal $\mathbf{n}$ to index into the local lighting buffers computed in the first step, and bi-linearly interpolate a local lighting $\widetilde{\mathbf{L}}$. Finally, the inner product $\mathbf{F_r} \cdot \widetilde{\mathbf{L}}$ gives vertex color B.

Notice that step one is view-independent, therefore its computational cost is primarily determined by the number of non-zero elements in the precomputed matrices and the source lighting vector. On the other hand, the rendering performance of step two primarily depends on the number of vertices of the model and the number of non-zero terms in the BRDF. The fundamental computation in both steps involves computing inner products of sparse vectors, which we currently compute entirely on the CPU. [WTL06] has described a possible solution to accelerate similar computations on the GPU. To increase the rendering realism, we modulate the resulting vertex colors with a precomputed ambient occlusion term. This gives reasonable ambient shadowing effects. To accelerate the rendering speed, we use SSE (Streaming SIMD Extensions) instructions to parallelize the computation across all three color channels: R, G and B.

4. Results and Discussion

In this section we present and discuss our results. All experiments were performed on a 2.4 GHz Intel Pentium 4 computer with 2GB memory. Both precomputation and rendering results are computed entirely on the CPU; while the algorithm seems amenable to hardware acceleration, we have left an optimized GPU implementation for future work.

Figure 1 shows a 120,000 vertex dragon model rendered with several complex BRDFs illuminated by different environment lighting. The view-dependent rendering speed is

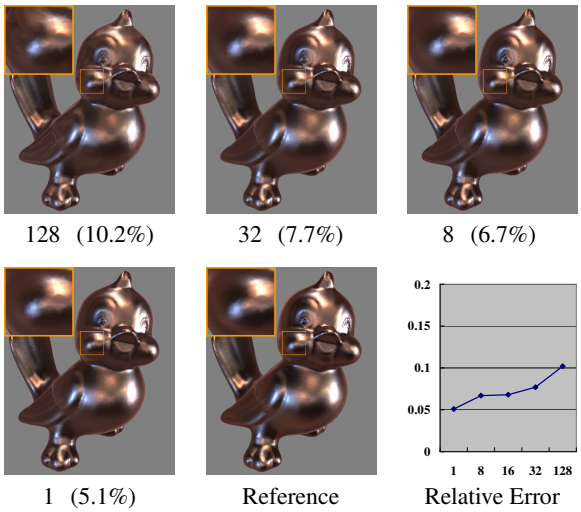

Figure 5: *Comparison of rendering quality when varying the truncation threshold used for eliminating small matrix elements. We uniformly quantize our matrix elements to 16-bit integers, and quantized values below the specified threshold will be discarded. Aggressive truncation reduces data size but also leads to loss of frequencies in the rotated lighting. For each image we compute its percent RMS error w.r.t. the reference image, and these errors are plotted in the graph. See Table 1 for details.*

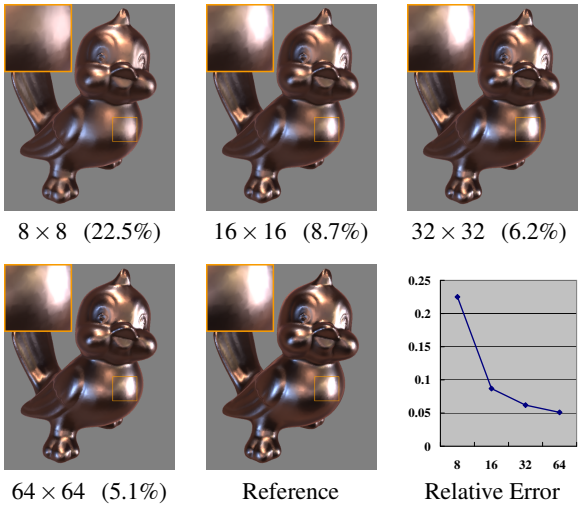

Figure 6: *Comparison of rendering quality for different normal sampling rates. Our normals are sampled on a square octahedral map, and we precompute one rotation matrix for each sampled normal. Low sampling rate reduces precomputed data size but also causes significant loss of high-frequency reflection effects due to the interpolation in the normal. For each image we compute its percent RMS error w.r.t. the reference image, and these errors are plotted in the graph. See Table 3 for details.*

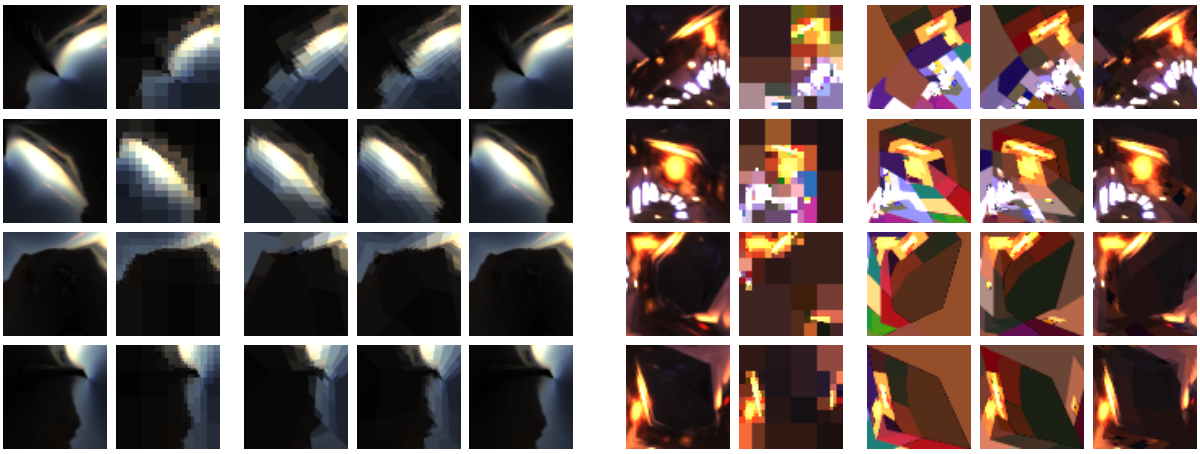

Figure 7: *These images show our rotation results for two environment maps. We used high resolution rotation matrices with $N = 128^2$ sampling rate for the source lighting, thus the rotated lighting is of 64×64 image resolution. Each row shows the result for a different randomly chosen surface normal. The first column shows the reference image generated by an expensive spatial domain algorithm; the second column is an 128-term wavelet approximation to the reference; the remaining three columns are rotation results using our proposed method, each generated by multiplying the rotation matrices with a source lighting that has been non-linearly approximated to 128, 256, and 2048 terms respectively. We typically use $128 \sim 256$ terms. The average RMS percent error is included at the bottom. The error for column 'Ref128' is directly comparable to the column '256', because our target sampling rate is one half of the equivalent source sampling rate. Notice that this error does not directly transfer to the computed radiance in the final rendering, which has much higher accuracy [NRH03].*

maintained at 5 frames per second; and we can change the lighting at 1.8 frames per second.

4.1. Accuracy

Error Analysis of Rotated Lighting Our rotation matrices are precomputed using an accurate spatial domain algorithm, therefore the matrices themselves introduce little error except for the slight inaccuracy incurred by the quantization of matrix elements and the possible truncation following that. The primary error in the rotated lighting comes from the non-linear wavelet approximation of the source lighting.

Figure 7 shows the comparison of rotated lighting with an increasing number of approximation terms for the source lighting. These rotated lightings are computed with our algorithm using precomputed matrices of $N = 128^2$ resolution. Each comparison also includes a reference image and a 128-term wavelet approximation directly applied to the reference image. The results gradually approaches the reference image as we apply more source lighting terms (2048 and beyond). The relative RMS error (averaged for all examples) of each approximation is listed at the bottom of the figure. Notice that the error in the lighting approximation does not necessarily transfer to the visual quality of the rendering, since the lighting will be integrated with the BRDF to produce the final radiance values, which will be visualized directly. In practice, the error in the computed radiance is much smaller than the error in the lighting [NRH03]. Thus in Figures 5 and 6 we directly compare errors in rendered images.

Truncation of Matrix Elements We have experimented with truncating small matrix elements to reduce storage size. Since our matrices are uniformly quantized to 16-bit integers, we set a threshold value and discard quantized values below the threshold. Table 1 shows the results of these experiments. And Figure 5 compares the rendering quality with a reference image generated offline, which also includes a graph plotting the error of computed radiance in the output image. Notice that a high truncation threshold (such as 128) reduces the matrix size by almost two thirds, but also causes rendering artifacts.

Sampling of Normal Since we discretize the normal onto an octahedral map and precompute one rotation matrix per normal, under-sampling of normals may cause serious loss of high-frequency information, manifested by the missing of specular highlights where normals are interpolated. On the other hand, the total number of the precomputed matrices will increase linearly with normal sampling rate, which will in turn impact our rendering speed upon lighting change. We typically choose a normal tessellation of 32^2 resolution, which appears qualitatively sufficient for all our renderings. Table 3 and Figure 6 shows our experiments with varying the

normal sampling resolution and the corresponding rendering results. Again, the accuracy refers to the image space RMS error of the computed radiance.

4.2. Precomputation

Table 2 shows our precomputation profiles for a number of different source lighting resolutions N. Notice that as N increases, the average number of elements per row in the precomputed matrices converges to about 63. This means the total number of elements will continue to grow linearly with N, therefore the precomputed matrix size is roughly proportional to N. In other words, the computational cost of wavelet rotation at run-time is about $O(N)$. This is asymptotically better than the $O(N^{\frac{3}{2}})$ complexity for spherical harmonics.

We also sample the BRDFs as a preprocess. Because we use wavelet rotation to compute local lighting on the fly, we avoid sampling the BRDFs for different surface orientations. Therefore the precomputed datasets are quite compact compared to the huge storage requirements by [NRH03]. We typically sample the view direction ω_o in a 128×128 resolution to avoid run-time interpolation across the view. The artifacts due to this simplification are rarely noticeable. This requires only $16 \sim 32$ MB per BRDF. For low-frequency BRDFs, a view sampling of 64×64 usually suffices, requiring only $4 \sim 8$ MB storage per BRDF. Thus we can easily store multiple BRDFs in memory at the same time, enabling fast dynamic switching of BRDFs.

4.3. Rendering Performance

We report our rendering performance in two parts. The first part is for dynamically changing the light: it is recomputed every time the user rotates the lighting or changes the environment map. Since this step computes local lighting by multiplying the source lighting with all rotation matrices for the sampled normals, its computational cost is independent of the view or the model; instead, it depends strongly on the number of non-zero elements in both the precomputed matrices and the source lighting. Tables 1 and 2 both list the lighting change frame rates for various settings.

The second part of the rendering algorithm computes the vertex colors based on the view-dependent BRDF slice (selected per-vertex at run-time), and the local lighting interpolated from the local lighting of available normals (computed in the first step). Thus its computational cost primarily depends on the size (number of vertices) of the model and the number of non-zero terms in the BRDF. Table 4 lists the rendering speed due to view change.

We achieve interactive frame rates for both lighting change and view change, except when the lighting resolution goes up to 128×128, making the precomputed matrix data too large to maintain interactivity. Our typical settings are: lighting resolution $N = 64 \times 64$, normal tessellation $ntess = 32 \times 32$, and matrix truncation threshold $1 \sim 4$. To increase the rendering realism, we attenuate vertex colors by precomputed ambient occlusion values to produce global shadowing effects. The performance overhead for ambient occlusion is negligible.

4.4. Discussion

The proposed rotation technique provides a unique tradeoff between memory consumption and rendering fidelity. For environment mapping, we can use low-resolution matrices with a very small memory footprint if we desire only low-frequency reflections comparable in quality to previous SH-based techniques. For example, if we use low resolution $N = 16^2$ source lighting, the precomputed rotation matrices only require 11.8 MB storage (with 32^2 normal sampling); and real-time performance is achieved with sufficient accuracy for low-frequency lighting and reflections.

Higher quality reflections than previously possible can be achieved at the cost of more memory. For example, $N = 64^2$ rotation matrices require more storage size (266MB), but in return we are able to handle high frequency reflections at near interactive rates ($\sim$2fps) with superior quality over previous techniques. In addition, the lighting, viewpoint and BRDFs can all be modified interactively on the fly, eliminating the need for prefiltering environment maps.

It is important to notice that the memory required for rotation matrices is only allocated once, not per-BRDF or per-model, so the cost can be amortized over many models with varying reflectance models. Since the available memory on a standard PC is increasing rapidly, we believe the memory usage required by our technique will not be a concern in the near future.

As discussed earlier, our rotation matrices are indexed by the surface normal, because each local frame is uniquely defined from the normal. One limitation with this simplification is that we cannot currently model the twist of local frame around the normal, or in other words, we do not handle user defined local frames. This means for rendering anisotropic BRDFs, such as the 'brushed metal', we are limited to brushed directions that are automatically generated by our system rather than defined by the user. The ability to handle arbitrary local frame would require an additional sampling around the normal. However, this is a limitation only of our implementation, not the technique itself.

5. Conclusions and Future Work

We have presented a system for rotating functions directly in the wavelet domain, and shown that this can be used to achieve interactive environment mapping with dynamic, high-frequency BRDF and lighting content. One of the main contributions of this paper is that wavelets can be rotated quickly because the projection of each rotated basis function back onto the basis is sparse. Although obvious in retrospect, this observation does not seem to have been widely

ntrunc	Error	Size	Spars.	avg.#	L sp.
1	0.051	266 MB	1.52%	62.2	1.8 fps
8	0.067	242 MB	1.45%	59.4	1.9 fps
16	0.068	219 MB	1.3%	54	2.1 fps
32	0.077	189 MB	1.1%	46.3	2.5 fps
128	0.102	105 MB	0.6%	25.3	3.6 fps

Table 1: *Truncation of small matrix elements. The columns list the truncation threshold (on a quantized 16-bit integer scale), accuracy (relative RMS error of the rendered image as in Figure 5), the size, sparsity, and the average number of elements per row of the precomputed matrices. The rightmost column lists the relighting speed due to dynamically changing light. All matrices are precomputed with* $N = 64^2$.

Res. N	P.T.	Size	Spars.	avg.#	L sp.
16^2	2 sec	11.8 MB	18%	46.2	23 fps
32^2	24 sec	58.4 MB	5.6%	57.5	7.5 fps
64^2	10 min	266 MB	1.5%	62.2	1.8 fps
128^2	3 hrs	1.1 GB	0.38%	63.2	0.4 fps

Table 2: *Precomputation profiles for different source lighting resolution N (M is always equal to* $N/4$*). The columns list the precomputation time, storage size, sparsity and the average number of elements per row in the precomputed matrices. We use our default normal sampling rate* 32^2 *and truncation threshold* 1. *The rightmost column lists the corresponding relighting speed for lighting change.*

ntess	Error	Pr. Time	Size
8^2	0.225	38 sec	16.6 MB
16^2	0.087	2.5 min	66.5 MB
32^2	0.062	10 min	266 MB
64^2	0.051	40 min	1 GB

Table 3: *Experiments with different normal sampling rates. The error refers to the relative RMS error of the rendered images as in Figure 6. The precomputation time and storage size grows linearly with the sampling rate. All matrices are precomputed with* $N = 64^2$.

Model	# verts	# faces	View Speed
Bird	30.6K	61K	$21 \sim 25$ fps
Head	49.2K	98.3K	$13 \sim 16$ fps
Armadillo	100K	200K	$5 \sim 7$ fps
Dragon	120K	240K	$4 \sim 5$ fps

Table 4: *View-dependent rendering performance. We list the size of each model, and rendering frame rates for view change (when lighting stops changing).*

recognized. In fact, one of the arguments for avoiding the use of wavelets has been the presumption that no fast or efficient rotation mechanism exists (as opposed to spherical or zonal harmonics). Such a mechanism enables us to stay in the wavelet domain, retaining the full benefits of that space for superior non-linear approximation (compression and computation reduction). One note is that a fast algorithm for wavelet rotation implies similar fast algorithms for translation in other domains such as 1D sound, 2D images, or 3D volumes. Although we have chosen environment map rendering as our driving application, we believe that the wavelet rotation technique is in general extendable to many other applications such as local deformable PRT (in the spirit of [SLS05]), wavelet-based importance sampling [CJAMJ05], and image processing etc.

The results in this paper suggest a challenging but compelling future research direction: a general signal-processing framework in which all processing takes place in the wavelet domain. As a concrete example of why this would be useful, many modern image compression schemes are wavelet-based, and yet image editing systems are forced to decompress and apply basic approximations in the pixel basis. One goal would be to design a full-fledged image processing system that operates efficiently entirely on the compressed coefficients. This could have significant implications for our ability to interactively manipulate extremely large imagery, a growing problem with the advent of super-high resolution digital photography.

The key barrier to such an image-editing system is that it would require at least three basic operations: addition, multiplication, and convolution, all operating directly on wavelet-compressed signals. While addition is easy, the other two are highly non-trivial. Nevertheless, previous work has suggested a solution to the multiplication operation in terms of so-called tripling coefficients [NRH04]. This paper presents the next major motivation, which is the suggestion that convolution can be handled efficiently, since it is simply a translation and integration (e.g., a dot product). With these components, one can imagine developing new signal processing systems that work efficiently and compactly directly on compressed signals.

Acknowledgements

The authors would like to thank Jiajun Zhu for editing the video for this paper, Nolan Goodnight for his help and suggestions, Paul Debevec for his light probe images, the Stanford University Computer Graphics Laboratory for the dragon model, and the anonymous reviewers for their constructive comments.

References

[BN76] BLINN J. F., NEWELL M. E.: Texture and reflection in computer generated images. *Commun. ACM 19*,

10 (1976), 542–547.

[CIGR99] CHIO C., IVANIC J., GORDON M., RUEDENBERG K.: Rapid and stable determination of rotation matrices between spherical harmonics by direct recursion. *The Journal of Chemical Physics 111*, 19 (1999), 8825–8831.

[CJAMJ05] CLARBERG P., JAROSZ W., AKENINE-MÖLLER T., JENSEN H. W.: Wavelet importance sampling: efficiently evaluating products of complex functions. *ACM Trans. Graph. 24*, 3 (2005), 1166–1175.

[CON99] CABRAL B., OLANO M., NEMEC P.: Reflection space image based rendering. In *Proc. of SIGGRAPH '99* (1999), pp. 165–170.

[DM97] DEBEVEC P. E., MALIK J.: Recovering high dynamic range radiance maps from photographs. In *Proc. of SIGGRAPH '97* (1997), pp. 369–378.

[Edm60] EDMONDS A.: *Angular Momentum in Quantum Mechanics*. Princeton University, 1960.

[GKMD06] GREEN P., KAUTZ J., MATUSIK W., DURAND F.: View-dependent precomputed light transport using nonlinear gaussian function approximations. In *ACM Symposium on Interactive 3D graphics* (2006), pp. 7–14.

[GKPB04] GAUTRON P., KŘIVÁNEK J., PATTANAIK S. N., BOUATOUCH K.: A novel hemispherical basis for accurate and efficient rendering. In *Proc. of the 15th Eurographics Symposium on Rendering* (June 2004), pp. 321–330.

[Gre86] GREENE N.: Environment mapping and other applications of world projections. *IEEE Comput. Graph. Appl. 6*, 11 (1986), 21–29.

[HS99] HEIDRICH W., SEIDEL H.-P.: Realistic, hardware-accelerated shading and lighting. In *Proc. of SIGGRAPH '99* (1999), pp. 171–178.

[KKB*06] KŘIVÁNEK J., KONTTINEN J., BOUATOUCH K., PATTANAIK S., ŽÁRA J.: Fast approximation to spherical harmonic rotation. In *Proc. of the 22nd spring conference on Computer graphics (to appear)* (2006).

[KM00] KAUTZ J., MCCOOL M. D.: Approximation of glossy reflection with prefiltered environment maps. In *Proc. of Graphics Interface 2000* (2000), pp. 119–126.

[KSS02] KAUTZ J., SLOAN P.-P., SNYDER J.: Fast, arbitrary brdf shading for low-frequency lighting using spherical harmonics. In *Proc. of the 13th Eurographics Rendering Workshop* (2002), pp. 291–296.

[KVHS00] KAUTZ J., VÁZQUEZ P.-P., HEIDRICH W., SEIDEL H.-P.: Unified approach to prefiltered environment maps. In *Proc. of the 11th Eurographics Rendering Workshop* (2000), pp. 185–196.

[LK03] LEHTINEN J., KAUTZ J.: Matrix radiance transfer. In *ACM Symposium on Interactive 3D graphics* (2003), pp. 59–64.

[LSSS04] LIU X., SLOAN P.-P., SHUM H.-Y., SNYDER J.: All-Frequency Precomputed Radiance Transfer for Glossy Objects. In *Proc. of the 15th Eurographics Symposium on Rendering* (2004), pp. 337–344.

[NRH03] NG R., RAMAMOORTHI R., HANRAHAN P.: All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans. Graph. 22*, 3 (2003), 376–381.

[NRH04] NG R., RAMAMOORTHI R., HANRAHAN P.: Triple product wavelet integrals for all-frequency relighting. *ACM Trans. Graph. 23*, 3 (2004), 477–487.

[PH03] PRAUN E., HOPPE H.: Spherical parametrization and remeshing. *ACM Trans. Graph. 22*, 3 (2003), 340–349.

[RH01] RAMAMOORTHI R., HANRAHAN P.: An efficient representation for irradiance environment maps. In *Proc. of SIGGRAPH 2001* (2001), pp. 497–500.

[RH02] RAMAMOORTHI R., HANRAHAN P.: Frequency space environment map rendering. In *ACM Trans. Graph.* (2002), vol. 21, pp. 517–526.

[SDS96] STOLLNITZ E., DEROSE T., SALESIN D.: *Wavelets for Computer Graphics*. Morgan Kaufmann, 1996.

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *ACM Trans. Graph.* (2002), vol. 21, pp. 527–536.

[SLS05] SLOAN P.-P., LUNA B., SNYDER J.: Local, deformable precomputed radiance transfer. *ACM Trans. Graph. 24*, 3 (2005), 1216–1224.

[WTL04] WANG R., TRAN J., LUEBKE D.: All-Frequency Relighting of Non-Diffuse Objects using Separable BRDF Approximation. In *Proc. of the 15th Eurographics Symposium on Rendering* (2004), pp. 345–354.

[WTL06] WANG R., TRAN J., LUEBKE D.: All-frequency relighting of glossy objects. *ACM Trans. Graph. 25*, 2 (2006), 293–318.