EUROGRAPHICS 2009 / P. Dutré and M. Stamminger
(Guest Editors)

Volume 28 (*2009*), *Number 2*

Shadowing Dynamic Scenes with Arbitrary BRDFs

Derek Nowrouzezahrai Evangelos Kalogerakis Eugene Fiume

Dynamic Graphics Project, University of Toronto

Abstract

We present a real-time relighting and shadowing method for dynamic scenes with varying lighting, view and BRDFs. Our approach is based on a compact representation of reflectance data that allows for changing the BRDF at run-time and a data-driven method for accurately synthesizing self-shadows on articulated and deformable geometries. Unlike previous self-shadowing approaches, we do not rely on local blocking heuristics. We do not fit a model to the BRDF-weighted visibility, but rather only to the visibility that changes during animation. In this manner, our model is more compact than previous techniques and requires less computation both during fitting and at run-time. Our reflectance product operators can re-integrate arbitrary low-frequency view-dependent BRDF effects on-the-fly and are compatible with all previous dynamic visibility generation techniques as well as our own data-driven visibility model. We apply our reflectance product operators to three different visibility generation models, and our data-driven model can achieve framerates well over 300Hz.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Color, shading, shadowing, and texture—

1. Introduction

Soft-shadowing and realistic bidirectional reflectance distribution functions (BRDFs) are essential for the rendering of convincing images. These effects are challenging to compute in real-time since the integration of many lighting directions as well as a compact representation of the reflectance data are required. The problem is compounded if the camera, lighting and BRDF are all allowed to change while the geometry is animating and deforming.

We present a general formulation for arbitrary BRDF shading of dynamic and deforming geometry that can readily be applied to many existing dynamic visibility generation techniques. Furthermore, our approach allows BRDFs to be swapped dynamically at no additional run-time cost. We use spherical harmonics (SH), leveraging their orthogonality and rotational invariance properties. This restricts us to low-frequency soft shadows and glossy reflections.

We introduce *reflection product operators* (RPOs) to efficiently convert visibility to view-dependent transfer by combining BRDF factorization [KM99] with SH product matrices. RPOs require only three seconds to compute and less than 10 kilobytes of storage each. BRDFs can be swapped in real-time, under dynamic viewing, lighting and shadowing conditions, all while maintaining framerates over 300 Hz on mid-range workstations.

We derive an efficient self-shadowing algorithm using compact, data-driven models that accurately predict per-vertex SH visibility for complex deforming geometry, such as cloth. RPOs can be applied to several SH visibility generation approaches such as SH exponentiation by Ren et al. [RWS*06], dynamic height field shadowing by Snyder and Nowrouzezahrai [SN08], shadow fields by Zhou et al. [ZHL*05], and the data-driven model we present in this work. We apply RPOs to several of these approaches.

Our goal is to relight and shadow animated and deforming geometries with dynamically changing BRDFs, while the lighting and camera move. Our main contributions include:

1. A simple and modular reflectance representation, decoupled from dynamic visibility generation, that efficiently incorporates view-dependent BRDF shading effects on animating geometry.
2. Compact and accurate regression models for temporally coherent self-shadows on complex deformable geometry.

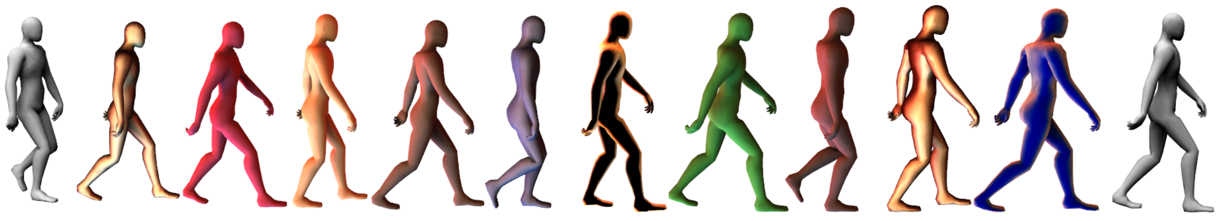

Figure 1: *An animating character with self-shadows under varying lighting, view and BRDF (326 Hz.)*

2. Previous Work

Classic precomputed radiance transfer techniques compute and project the BRDF-weighted visibility for static scene geometry into a compact basis prior to online rendering. At run-time, an arbitrary lighting function can be projected into the same basis and used to shade the scene. Spherical harmonics [SKS02, KSS02], wavelets [NRH03, NRH04, MHL*06] and radial basis function (RBF) bases [TS06] have been used to this end. Liu et al. [LSSS04] and Wang et al. [WTL04, WTL06] apply the BRDF factorization of Kautz et al. [KM99] to precomputed radiance transfer (PRT) so as to reduce the storage requirements for non-diffuse BRDFs. Our RPOs are motivated by their approaches.

Dynamic visibility generation techniques for animated scene geometry originated in Zhou et al.'s work [ZHL*05]. They tabulate a set of SH visibility functions in a volume surrounding each rigid scene object. At run-time, to shadow a receiver point, the *k*-nearest samples in the shadow fields of each object are interpolated and combined using an SH product operation. Kautz et al. [KLA04] rasterize the blocking triangles from each receiver point into a low-resolution buffer, projecting this visibility data into the SH basis and shading the scene under varying lighting conditions. Real-time rendering is limited to small scenes with few triangles, but view-dependent BRDFs can also be supported at interactive rates for these scenes. Sloan et al. [SLS05] represent local shading effects, such as translucency, on deformable models in the Zonal Harmonics (ZH) basis, which allows rapid rotation into arbitrarily deforming frames. This technique cannot capture global shadowing effects. Wang et al. [WXZ*06] approximate an analytical SH operator for angular scaling and shadow deforming geometry with diffuse BRDFs.

Ren et al. [RWS*06] represent articulated characters as a set of spheres and accumulate visibility in *log SH* space. Cheap SH additions and exponentiation replace expensive SH products for visibility calculations. The accumulated visibility is used for diffuse shading under varying distant lighting. Sloan et al. [SGNS07] extend this technique with an image-based splatting approach. Snyder and Nowrouzezahrai [SN08] calculate the shadowed diffuse shading for dynamic height fields with a multi-scale image pyramid visibility algorithm. While their approaches are best suited for objects that can easily be approximated by spheres, our data-driven visibility models can also handle more complex deformable objects, such as cloth. We demonstrate the versatility of our RPOs by applying them to these two visibility generation techniques, as well as the technique we will introduce shortly.

Dachsbacher et al. [DSDD07] and Dong et al. [DKTS07] reformulate the rendering equation, treating visibility *implicitly* with a GPU-accelerated finite element solver. They compute direct and indirect lighting with non-diffuse BRDFs for static scenes and simple animations at interactive framerates. Ritschel et al. [RGK*08] couple low-resolution shadow mapping with a GPU-accelerated instant radiosity algorithm in order to render complex and dynamic animations with direct and indirect illumination under varying illumination and BRDFs at interactive framerates. Annen et al. [ADM*08] approximate environment lighting with a set of area lights and use a fast shadow mapping algorithm to shade dynamic scenes at interactive to real-time framerates. Our self-shadowing models run at real-time framerates and RPOs allow the BRDF to be changed on-the-fly.

Data-driven visibility techniques precompute shading for several frames of an animation and fit a model to this data. Kirk and Arikan [KA07] and Kontkanen and Aila [KA06] fit a linear model of the joint angles of an articulating character to ambient occlusion (AO) data. Nowrouzezahrai et al. [NSK*07, NKSF08] apply a similar approach, fitting a reduced dimensional linear model to SH diffuse transfer vectors and ZH visibility for simple articulating characters.

Feng et al. [FPJY07] precompute transfer matrices at the vertices of a deforming mesh, for many frames of an animation. A novel incremental clustering and dimensionality reduction approach is used to compress the original dataset by a factor of 100. We, on the other hand, decouple the visibility from the BRDF and only fit a data-driven model to the visibility component that varies during animation. In doing so, we achieve an order of magnitude higher performance, orders of magnitude less storage and precomputation time, and the ability to change the BRDF on-the-fly.

3. Terminology and Review

At every point x we wish to shade, the hemispherical binary visibility function, $V_x(s)$, is defined as

$$V_x(s) = \begin{cases} 0 & \text{if } \mathrm{ray}(s) = 1 \text{ and } s \cdot N_x \geq 0, \\ 1 & \text{otherwise,} \end{cases} \quad (1)$$

where s is a point on the unit sphere $\mathbb{S}^2$, $ray(s)$ is a function that is 1 if a ray in direction s is occluded by geometry, and N_x is the normal at x. Projecting this function onto a finite, predetermined subset of the SH basis functions $\boldsymbol{y}_i$ yields the coefficient vector $\boldsymbol{v} = \{v_1, \ldots, v_k\}$ such that $v_i = \langle V_x, \boldsymbol{y}_i \rangle = \int V_x(s)\, \boldsymbol{y}_i(s)\, ds$.

Accumulating visibility across multiple blockers can be performed, in the most general case in SH, by taking the product of many SH visibility functions [ZHL*05]. The product of two SH functions, $\boldsymbol{a} \times \boldsymbol{b}$, requires the application of the SH triple product order-3 tensor to the two vectors [RWS*06]: $\langle \boldsymbol{a} \times \boldsymbol{b}, \boldsymbol{y}_i \rangle = \int a(s) b(s) \boldsymbol{y}_i(s)\, ds = \sum_{jk} \Gamma_{ijk} \boldsymbol{a}_j \boldsymbol{b}_k$. For r rigid objects, the accumulated visibility vector is

$$\begin{aligned} \boldsymbol{v}_i &= (\boldsymbol{v}_1 \times \ldots \times \boldsymbol{v}_r)_i \\ &= \sum_{j_1 k_1} \Gamma_{i j_1 k_1} \boldsymbol{v}_{j_1} \boldsymbol{v}_{k_2} \;\ldots \sum_{j_{r-1} k_{r-1}} \Gamma_{k_{r-2} j_{r-1} k_r} \boldsymbol{v}_{j_{r-1}} \boldsymbol{v}_{k_r} \ . \end{aligned} \quad (2)$$

SH products are costly, in practice limiting the number of terms, r, accumulated in the visibility vector. Alternatively, blocking spheres can be accumulated in a *log SH* space:

$$\boldsymbol{v}_i = \exp\left(\sum_b \boldsymbol{v}_b^{\log}\right)_i \quad (3)$$

where $\boldsymbol{v}_i^{\log}$ represents the log SH shadowing due to a single blocking sphere tabulated as a function of the ratio of the sphere's radius to its distance from the receiver point.

Dynamic height field visibility can be generated in real-time with a multi-resolution pyramid of horizon map values, $\{\omega_0, \ldots, \omega_n\}$, taken at many discretized azimuthal directions, $\{\phi_0, \ldots, \phi_n\}$ [SN08]. The visibility function is

$$V_x(s) = \begin{cases} 0 & \text{if } s_\theta \leq \omega_i \text{ for } i \text{ such that } s_\phi \in [\phi_i, \phi_{i+1}], \\ 1 & \text{otherwise.} \end{cases} \quad (4)$$

Canonically tabulated visibility wedges are rotated using fast SH Z-rotations and combined to form the final SH visibility

$$\boldsymbol{v} = \sum_{n-1} \mathrm{R}_z(\phi_i) \boldsymbol{v}_{[\omega_i, \omega_{i+1}]}. \quad (5)$$

Shading in SH is based on the rendering Equation [Kaj86]

$$L_x(\omega_o) = \int_{\mathbb{S}^2} L_{in}(\omega) V_x(\omega) \underbrace{f_x(\omega, \omega_o) \max(\omega \cdot N_x, 0)}_{\bar{f}_x(\omega, \omega_o)} d\omega \quad (6)$$

where ω_o is the outgoing viewing direction, L_x is the outgoing radiance at x, L_{in} is the incoming radiance at x, f_x is the BRDF, and $\bar{f}_x$ is the cosine–weighted BRDF restricted to the upper hemisphere defined by N_x. We denote the SH projections of L_{in} and $\bar{f}_x$ (evaluated at the outgoing direction) as $\boldsymbol{l}$ and $\boldsymbol{f}^o = \langle \bar{f}_x(\omega, \omega_o), \boldsymbol{y}(\omega) \rangle$. We will use subscripts to index the elements of these SH coefficient vectors.

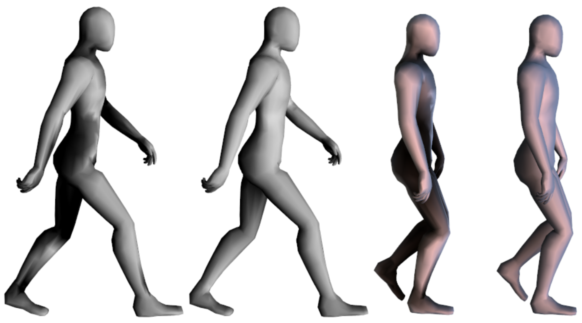

Figure 2: *Self-shadowing using the replacement rules of [RWS*06] (1st and 3rd images), and our data-driven results.*

In the most general setting, calculating the outgoing radiance with SH projections of each of the terms in the integrand requires a costly application of the triple product tensor

$$L_x(\omega_o) \approx \sum_{ijk} \boldsymbol{l}_i\, \boldsymbol{v}_j\, \boldsymbol{f}_k^o\, \Gamma_{ijk} \ . \quad (7)$$

Alternatively, the BRDF and visibility terms can be "baked" into a *transfer* term during precomputation, resulting in a more efficient double-product integral computation:

$$\boldsymbol{t}_i^o = \sum_{jk} \boldsymbol{v}_j\, \boldsymbol{f}_k^o\, \Gamma_{ijk} \;\Rightarrow\; L_x(\omega_o) \approx \sum_i \boldsymbol{l}_i\, \boldsymbol{t}_i^o \ . \quad (8)$$

Wang et al. [WTL04, WTL06] and Liu et al. [LSSS04] factor BRDFs using the method of Kautz et al. [KM99] to combine the *light factor* terms, $U^b(\omega_i)$, with the visibility, and evaluate the *view factor* terms, $W^b(\omega_o)$, at the outgoing direction

$$\begin{aligned} f_x(\omega_i, \omega_o) \approx \sum_b U^b(\omega_i)\, W^b(\omega_o) \;&\Rightarrow\; \tilde{\boldsymbol{t}}_i^b = \sum_{jk} \boldsymbol{v}_j\, \boldsymbol{u}_k^b\, \Gamma_{ijk} \\ L_x(\omega_o) \approx \sum_b W^b(\omega_o) \sum_k \boldsymbol{l}_k\, \tilde{\boldsymbol{t}}_k^b \ . \end{aligned} \quad (9)$$

The disadvantage of these approaches to the triple-product formulation is that the BRDF cannot be changed at run-time. Feng et al. [FPJY07] fit a non-linear model to these large transfer datasets for all vertices and animation frames.

In contrast, RPOs combine BRDF factorization and SH product matrices, resulting in three key advantages:

1. The representation is compact, requiring orders of magnitude less computation and storage than previous work.
2. We can change the BRDFs during run-time while avoiding the expensive triple-product evaluation.
3. Our data-driven model (Sec. 5) need only fit the varying visibility, not the larger, more complicated transfer data.

We apply our RPOs to visibility generated using the hybrid SH exponential method (HYB), combining an optimal linear approximation with scaling and squaring to compute the SH exponential (see Section 4.3 in [RWS*06]), and a modified version of a dynamic height field visibility approach [SN08]. Furthermore, our accurate data-driven self-shadowing model complements previous work, such as that of Ren et al. [RWS*06]: while their model robustly handles cast shadows from articulated and deforming geometry, Figure 2 shows that our method more accurately captures self-shadows.

4. Reflection Product Operators

We introduce a simple representation for BRDFs that allows us to dynamically change the material properties of our scene. This representation is compact (less than 10 kilobytes per BRDF) and is used to convert dynamically generated SH visibility to SH transfer while avoiding explicit storage and run-time evaluation of the triple product tensor.

We start by factoring BRDF data, represented by a matrix of incident and outgoing directional samples, using the SVD factorization technique of Kautz and McCool [KM99]. Note that any factorization technique that generates light-only and view-only factors is suitable. Next, we project the k light and view terms, $U^b(\omega_i)$ and $W^b(\omega_o)$, into the SH basis. We subsequently compute and store the SH product matrix for each of the projected light factors

$$\boldsymbol{u}^b = \langle U^b, \boldsymbol{y} \rangle,\ \boldsymbol{w}^b = \langle W^b, \boldsymbol{y} \rangle,\ \mathsf{M}^b_{ij} = \sum_k \boldsymbol{u}^b_k\, \Gamma_{ijk}\,. \qquad (10)$$

To calculate the outgoing radiance, assuming we have synthesized the dynamic SH visibility, we explicitly decouple the visibility and BRDF from transfer in Equation 9

$$L_x(\omega_o) \approx \sum_b \boldsymbol{w}^b \cdot \boldsymbol{y}(\omega_o) \sum_k \left[\boldsymbol{l}_k \left(\mathsf{M}^b \boldsymbol{v} \right)_k \right]. \qquad (11)$$

Here we perform the SH product of the BRDF light-factors and the visibility (inside the square brackets) and dot them with the SH light vector (right most summation), and the left most summation over the k BRDF factors also includes the SH expansion of the BRDF view terms. The key benefit of this reformulation is that many M^bs and $\boldsymbol{w}^b$s can be precomputed and swapped in at run-time while $\boldsymbol{v}$ can be generated dynamically. To accelerate the run-time calculation of Equation 11 we pre-compile a GPU shader function, F, (for each BRDF) that efficiently evaluates the matrix-vector multiplications and takes the product with the expanded view factor terms: F^b has inputs $\boldsymbol{v}$, M^b, $\boldsymbol{w}^b$, $\boldsymbol{y}(\omega_o)$, and a single output $\boldsymbol{t}^b$ representing the b^{th} BRDF factor of the transfer. These functions are the *reflection product operators*. Projecting the BRDF factors into SH and generating the shader functions is performed nearly interactively for each BRDF during initialization.

The storage required for an RPO with a k term BRDF factorization is, at most (depending on sparsity of the light-factor), $k \times n^2(1+n^2)$ floats, for an n-order SH expansion. We typically use $k = 8$ and $n = 4$, for a total of only 8.5 kilobytes per BRDF assuming 32-bit floats. We only reproduce low-frequency shadowing and reflectance effects, and so an 8 term expansion is suitable for our needs [WTL06]. Figure 3 illustrates the application of our RPOs for analytic and measured BRDFs [MPBM03] under various lighting conditions.

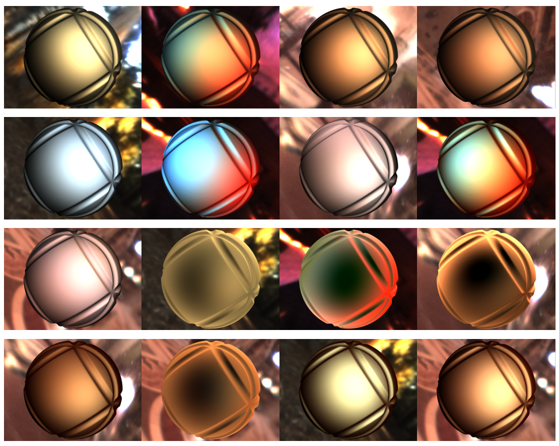
Figure 3: *Various BRDFs rendered with RPOs (8.5 kilobytes each) under environmental and directional lighting.*

5. Data-driven Visibility Generation

We propose a flexible, data-driven model for synthesizing per-vertex visibility data on articulated and deformable geometries, such as character animations and cloth simulations. In this section, we will discuss several of the key issues involved in successfully applying data-driven approaches to the problem of low-frequency shadow synthesis. Among these are the parameterization of the input and output spaces, clustering and dimensionality reduction considerations, shading data generation, and most importantly, the choice of a compact, predictive model.

5.1. Input Parameterization

Data-driven models map multi-dimensional inputs to outputs. A suitable set of input parameters that fully describe the current state, $\mathbf{x}$, of an animation is required. For skinned characters, a natural parameterization of this input space is the set of all joint angles in the character's skeleton. For deformable models, such as cloth animations, no such natural parameterization exists. Similar to Kalogerakis et al. [KNS*09], for cloth simulations we use the full set of dihedral angles as the current state and then reduce the size of this space using dimensionality reduction techniques (see below). We have found that this parameterization allows our

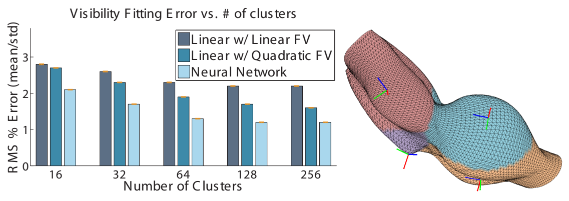

Figure 4: *Right: example clusters with associated PCA directions. Left: how the # of clusters affects the visibility error.*

data-driven models to capture the complex shadows between folds and wrinkles, and to generalize for similar cloth states.

Dimensionality reduction is performed over the input parameterization to find a linear lower-dimensional subspace suitable for representing the input state. We apply principal components analysis (PCA) to the input parameters defined at each training data point, where $\mathcal{X} = \mathsf{P}_i(\mathbf{x} - \bar{\mathbf{x}})$ is the reduced dimensional input vector, P_i is the input projection matrix composed of the n retained eigenvectors of the input space, and $\bar{\mathbf{x}}$ is the mean of the input dataset. For typical animation sequences we retain enough dimensions to capture $\sim$90% of the input variance. At run-time (Sec. 6), the full-dimensional input vector is projected into the lower-dimensional vector prior to being mapped into an appropriate output space (Sec. 5.2.)

Generating suitable training data is important to ensure that our model can generalize to novel animation sequences. In our experience, there are two effective methods to do so. Firstly, if an animation sequence exists that spans a wide range of input scenarios, then it may be used. One such example would be an animation sequence for an articulated character containing many stretching movements, covering the plausible motion of all the joint angles. Alternatively, regularly sampling the space of input parameters also generates a suitable training data set for our model.

5.2. Output Parameterization

Ultimately, we must reconstruct a spherical signal (represented by its SH coefficients) at each vertex of a mesh, and so a natural choice for the output space is the mesh vertices. We leverage the spatial coherence and redundancy in the high-dimensional output without sacrificing visual accuracy.

Clustering and dimensionality reduction can be used to improve the performance of our model and reduce its storage requirements. The SH visibility at adjacent vertices does not vary significantly, and we leverage this redundancy with a spatial clustering approach similar to [SHHS03, KA06], although other clustering techniques [FPJY07] may also be suitable. Our clustering scheme also addresses the issue of maintaining *spatially and temporally* smooth local tangent frames on our mesh surface, which improves the coherence of the data and is necessary if a local frame representation of the visibility and BRDF is to be used (see Section 5.4). From our experiments, naïve approaches such as defining tangent directions according to local triangle edge directions fail. We segment one frame of the animation (a single mesh) into its rigid components. For articulated characters we perform mean-shift clustering based on the skinning weights of the skeleton, while for other models we use the technique of [JT05]. As a result, triangles with similar rotation sequences are clustered together into as-rigid-as-possible mesh components. We then perform PCA over mesh vertices in each cluster. Projecting the PCA directions into the tangent planes of each vertex yields tangent directions that are temporally coherent when animated and spatially smooth. Figure 4 illustrates an example clustering with its PCA directions for a deforming muscle, and the relationship between the number of clusters and the model error.

We also perform PCA over the SH coefficients at vertices within each cluster, typically retaining $\sim$95% of the variance. Here, $\mathcal{Y} = \mathsf{P}_o(\mathbf{y} - \bar{\mathbf{y}})$ is the reduced dimensional output space, P_o is the PCA output projection, $\mathbf{y}$ is the output vector (SH coefficients defined at mesh vertices) and $\bar{\mathbf{y}}$ the data mean. Figure 5 illustrates the effect output variance on shadow reconstruction and model error.

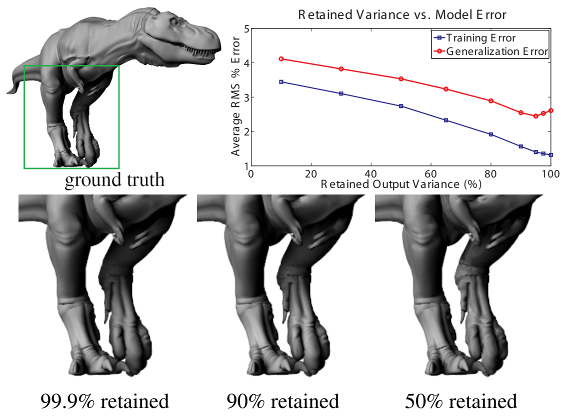

Figure 5: *Top right: the effect of output variance on visibility error. Bottom: notice how shadow details from the left leg onto the right leg, and just above the left knee, are affected by the reduced percentage of retained variance.*

5.3. Linear vs. Non-linear Predictive Models

Our predictive models, fit to training data, will map reduced dimensional inputs to reduced dimensional outputs. Unlike the local interpolation of [FPJY07], our regression models generalize to unseen animations with low storage requirements and few training poses. We investigate the suitability of both linear and non-linear models in this regard.

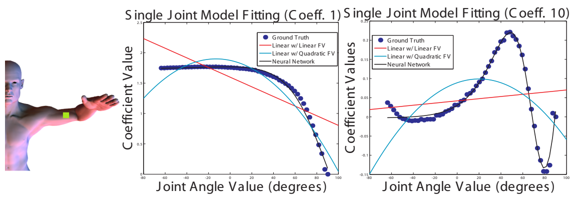

Figure 6: *Fitting a model to single joint visibility variation.*

Linear models are the simplest to address and, in certain cases, may generate suitable results [KA07, KA06, NSK*07, NKSF08]. We have found that a linear model, combined with a prior to ensure that small changes in the input result in small changes in the output, is suitable for reconstructing visibility for articulated characters with mild deformations. However, for highly deformable animations, this model did not yield acceptable results. Linear models with higher-order feature vectors (FVs) tend to overfit the data and make generalization difficult (see Figure 7.)

Non-linear models are better suited to our datasets. Figure 6 shows how, for a single joint of an articulated character, the visibility data at a vertex behaves like a smoothed step function. Various linear and non-linear model fits to this case are illustrated. A non-linear model (per cluster) is especially necessary if the data will be visualized with a non-diffuse BRDF, since diffuse BRDFs smooth the discrepancy between a linear model's results and the ground truth. For each SH coefficient, we fit a regularized single-layer neural network with tanh neurons:

$$\tilde{\mathcal{Y}}(\mathcal{X}) = \sum_{n=1}^{N} \boldsymbol{c}^n \tanh(\boldsymbol{d}^n \cdot \mathcal{X} + d^0) + c^0 , \qquad (12)$$

where $\tilde{\mathcal{Y}}$ is the reduced dimensional generated output, $\mathcal{X}$ is the reduced dimensional input, N is the number of neurons, $\boldsymbol{c}^n$ and $\boldsymbol{d}^n$ are the model's weight coefficient vectors and c^0 and d^0 are bias terms. The parameters of the model are fit by minimizing the following objective function using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimizer:

$$\sigma(\boldsymbol{c}^n, \boldsymbol{d}^n) = \sum_i \|\tilde{\mathcal{Y}} - \mathcal{Y}\|^2 + \alpha \sum_{n=1}^{N} (\|\boldsymbol{c}^n\|^2 + \|\boldsymbol{d}^n\|^2) , \quad (13)$$

where the regularization term, α, and the number of neurons are chosen with cross-validation. This model is well-suited for our data. The BFGS optimizer is well-suited for this non-linear optimization problem and we have found that 5000 iterations are typically enough for the optimizer to converge onto a satisfactory minimum. Figure 7 compares model errors on representative animations. In all cases, we use fewer than 200 training frames.

Regularization is applied using 4-fold cross-validation to avoid overfitting the training data and to increase model generalization. We have targeted our models for applications which require generalization, such as interactive games where a character model might interact unexpectedly with the environment through a physics engine. Alternatively, our models may also be used for data-compression purposes, and thus generalization would not be necessary since only the training data need be reproduced at run-time. In this case, regularization should be avoided since the goal of compression can be thought of as requiring the training data to be reproduced as accurately as possible.

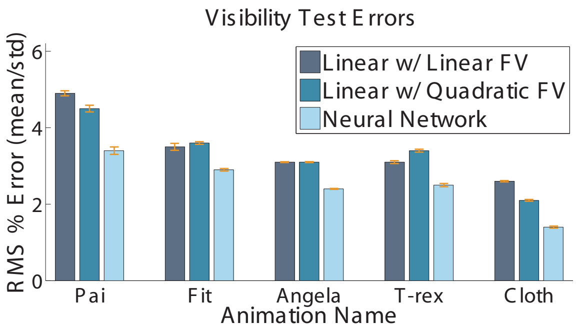

Figure 7: *Visibility error of each model on novel poses.*

5.4. Shading Dataset Considerations

There are many suitable representations of the SH data to which we can fit our model. We will analyze, in the context of model fitting and reconstructing the outgoing radiance, how to handle the cosine foreshortening term, and local versus global co-ordinate frame representations of the visibility data.

The cosine foreshortening term in the rendering equation can either be absorbed into the BRDF (and thus, the reflection product operator), or it can be analytically re-incorporated at run-time using the following product matrix

$$\mathrm{C}_{ij} = \sum_k \left[\int_{s \in S} \max\left(N_x \cdot s\right) \boldsymbol{y}_k(s)\, ds \right] \Gamma_{ijk} \qquad (14)$$

which has a simple analytic form. Although absorbing the cosine with the BRDF can in some cases adversely affect the effectiveness of the factorization, in our experience this was not the case for low-frequency BRDFs. Furthermore, Figure 8 shows that fitting a model to the non-cosine weighted visibility results in much lower reconstruction errors [NKSF08]. For these reasons, we fit our models to the visibility and use cosine-weighted BRDF data for the RPOs.

Choosing the coordinate frame used to represent the SH visibility is a critical decision. With a local co-ordinate frame

Geometry (# vertices)	Dataset Size	Input/Ouput Variance %	Clustered Model Size (MB)			FPS			Fitting Time (H:M)		
			Linear	Quadratic	NN	Linear	Quadratic	NN	Linear	Quadratic	NN
Pai (11.6k)	258.7 MB	90/95	8.3	8.5	9.0	325	312	268	0:12	0:16	14:32
Fit (20.1k)	412.8 MB	80/90	16.1	16.9	17.9	165	160	141	0:17	0:23	16:03
Angela (24.8k)	137.5 MB	90/95	14.0	14.5	16.1	127	121	108	1:02	1:38	32:14
T-rex (21.6k)	109.5 MB	90/95	16.1	16.5	17.4	112	105	98	0:38	0:48	26:43
Cloth (7.7k)	26.1 MB	90/99	2.8	3.1	3.6	332	328	321	0:27	0:45	12:08
Human (1.6k)	13.3 MB	90/99	1.4	1.5	1.8	348	339	326	0:08	0:11	5:11

Table 1: *Sizes, execution times and fitting times of the reduced dimensional linear and neural network models. Since we use RPOs, our original datasets only consist of SH visibility and are much smaller than animated transfer datasets.*

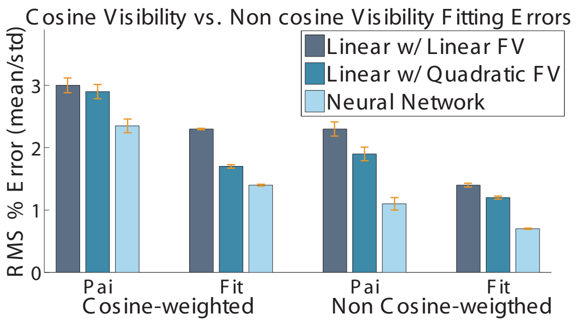

Figure 8: *Model errors with and without the cosine term.*

representation of the visibility, the BRDF data may be expressed in its natural 4D parameter space, however at runtime the lighting would need to be rotated into the local frame of each shading point to obtain $\boldsymbol{l}$ in Equation 11. Alternatively, with visibility expressed in a global co-ordinate frame, the lighting need not be rotated but the BRDF becomes a normal-dependent 6D function and an RPO would have to be tabulated for a discrete set of normals (or tangent-frames for anisotropic BRDFs.) We compare our models with each type of data and summarize the results in Figure 9.

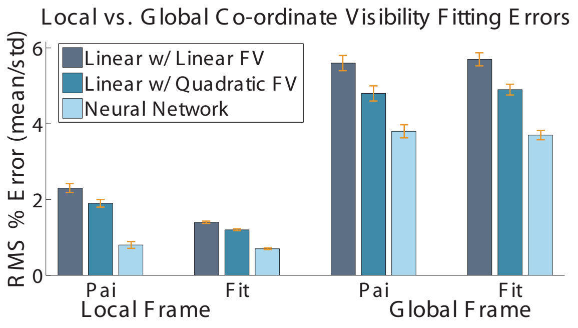

Figure 9: *Model errors with different SH coordinate frames.*

We chose local SH visibility since our models consistently fit this data with lower error. We will address the issue of efficiently rotating the lighting into the tangent frame.

Zonal harmonics lighting is used to solve the local-frame lighting rotation problem. SH Environmental lighting is approximated with a set of ZH lobes since rotating ZH signals into the per-vertex local co-ordinate frames is much more efficient than rotating general SH signals [SLS05]. We perform ZH lobe fitting in a slightly different manner than previous work [Slo08, SLS05]. Given the SH coefficients of an environment light, $\boldsymbol{l}$, we sample a small number (e.g. 1500) of uniformly distributed directions on the unit sphere. Each of these directions is treated as a potential lobe direction and we calculate the residual of a single-lobe ZH fit, *for each color channel*, and for each candidate lobe direction. The coefficients are calculated using Equation 18 of [SLS05]

$$\boldsymbol{l}_l^* = \frac{\sum_{m=-l}^{l} \boldsymbol{y}_{lm}(s^*)\boldsymbol{l}_{lm}}{\sum_{m=-l}^{l} \boldsymbol{y}_{lm}^2(s^*)} \tag{15}$$

where s^* is the candidate lobe direction and $\boldsymbol{l}_l^*$ is the ZH band coefficient. The best fit direction and coefficients obtained from this procedure are used to seed a BFGS optimization that minimizes an error function for the single-lobe ZH fit to the lighting. Both the lobe coefficients and candidate direction are free variables during optimization and, from our experience, it is not necessary to use gradients during optimization. We suspect this is because the initial sampling seeds the optimizer very close to the final result. The process is repeated for each ZH lobe. In all our examples, we found 3 ZH lobes to be sufficient for low-frequency lighting. Figure 10 illustrates 3-lobe ZH fits to SH environment maps. The entire process takes 3 seconds per environment map.

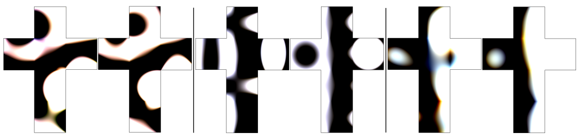

Figure 10: *SH lighting and 3-lobe ZH fits (in pairs.)*

6. Precomputation and Run-time Algorithms

Generating the reflection product operators requires a straightforward implementation of Section 4:

1. Each BRDF is tabulated as a matrix of values for (local) incoming and outgoing directions.
2. Any SVD algorithm (we use MATLAB's svd function) executed on the matrix yields the light and view factors.
3. Each of the k retained factor pairs are projected into SH.
4. The light factor coefficients are used to determine the entries of the symmetric SH product matrix, M^i.
5. A pre-compiled shader function optimizes the matrix-vector product and view factor expansion: $\mathsf{F}^b(\,\boldsymbol{v},\,\mathsf{M}^b,\,\boldsymbol{w}^b,\,\mathbf{y}(\omega_o),\,\boldsymbol{t}^b\,)$ with $\boldsymbol{t}^b = \left[\boldsymbol{w}^b \cdot \boldsymbol{y}^o\right]\left(\mathsf{M}^b \boldsymbol{v}\right)$.

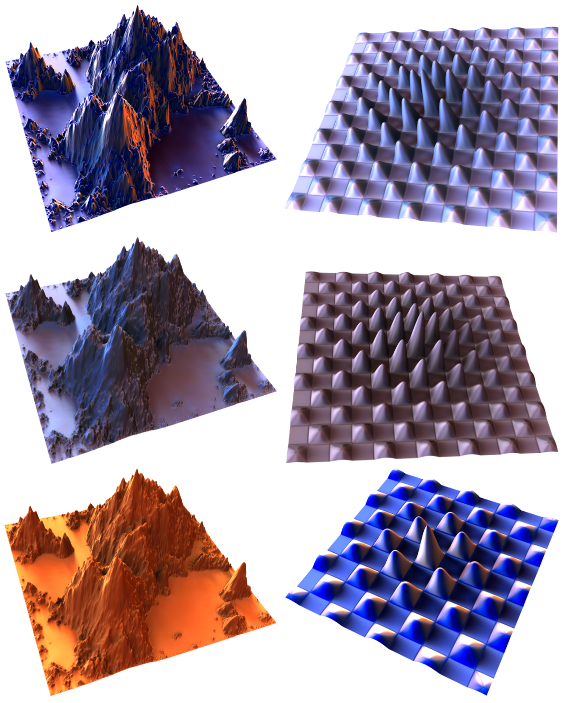

Figure 11: *RPOs applied to dynamic height fields (32 Hz.)*

This process takes no longer than 3 minutes per BRDF. At run-time, the following steps are executed in order to shade:

1. Query the input parameters of the current frame of the animation, $\mathbf{x}$, and project them into their reduced dimensional representation: $\mathcal{X} = \mathsf{P}_i(\mathbf{x} - \bar{\mathbf{x}})$.
2. The data-driven model mapping is applied: $\mathcal{Y} = \tilde{\mathcal{Y}}(\mathcal{X})$.
3. The reduced dimensional output is projected to the full dimension (mesh vertex) space: $\boldsymbol{v} = \mathbf{y} = [\mathsf{P}_o]^T \mathcal{Y} + \bar{\mathbf{y}}$.
4. Equation 11 is computed with the RPOs.

Our visibility models can be fit in a fraction of the time of previously proposed methods. Table 1 summarizes model statistics for various datasets. Our models achieve high compression *as well as* predictability for novel animations.

7. Results

All of our results are captured at 800x600 on a P4 3.0 GHz CPU, with 2 GB RAM and an nVidia GeForce 8800 GTX with 768 MB VRAM. All data-driven shadows are generated using the NN model with 64 clusters. All RPOs use 8-term BRDF factorizations. The HYB technique of [RWS*06] is used for cast shadows and, except for the humanoid datasets, we generate the spherical blocker approximations using k-means clustering over the mesh vertices (Fig. 12). For dynamic height fields, a modified version of the algorithm presented in [SN08] is used (Fig. 11): we sub-sample the visibility and shade with RPOs at full resolution (512^2, or 522k triangles). Our visibility model compares favorably to ground truth renderings even for highly deformable models (Fig. 13.)

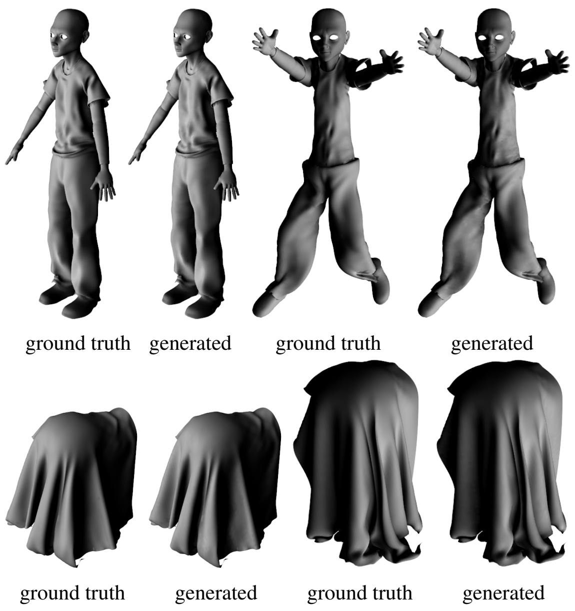

Figure 13: *Ground truth comparisons for novel poses.*

8. Discussion and Future Work

We have presented both linear and non-linear approaches to synthesizing visibility data. The linear models require less precomputation time and storage, are slightly more efficient to evaluate at run-time, and can provide plausible results (especially under diffuse shading) which are temporally coherent. These models are suitable for games with many articulated characters lit by dynamic lighting.

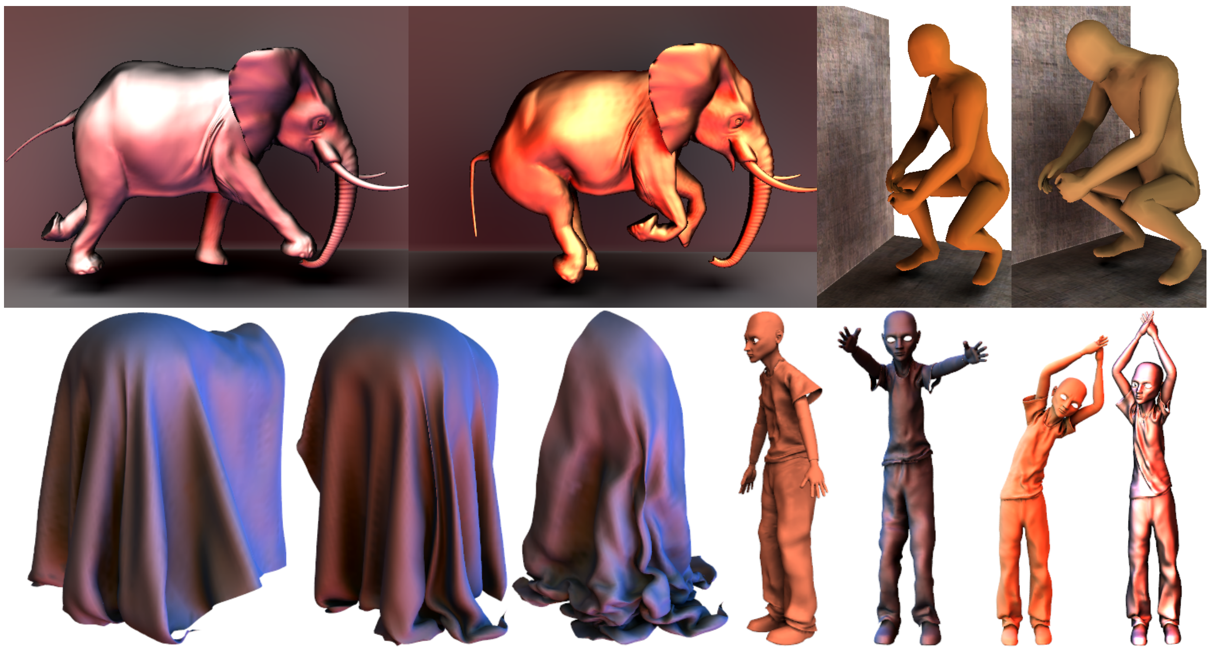

Figure 12: *RPOs applied to our data-driven model (108 to 326 Hz) and SHexp (62 Hz) with dynamic view, lighting and BRDFs.*

The non-linear model provides a much better trade-off between accuracy and speed, capturing the non-linear trend (sharp discontinuities) of local-coordinate SH visibility with high fidelity and few neurons, and can generalize to novel animations just as well as the linear models. This model, however, is slightly more complex to implement than linear models and requires more precomputation. For very complex deformations, such as our cloth examples, we recommend using the neural network models.

For all models, providing training data that samples the input space sufficiently is a very important requirement. For example, given a training dataset composed of a walking animation sequence where the arms never cross above the plane of the shoulder blades, the user can expect the models to perform very well on running sequences. However, for any sequences where the character raises its arms above its head, the models will not be able to completely predict the shadows cast by the arms onto the head.

Currently, our data-driven models only reproduce self-shadowing effects. We address this limitation in our examples by combining many different visibility generation approaches; however, a promising area of future work is the application of data-driven techniques to cast shadows. This is a challenging problem since a parameterizable volumetric representation of visibility is required, resulting in much larger training datasets and models. Moreover, incorporating indirect illumination with varying BRDFs is another area of future work with challenging problems, since the incident radiance at shading points is a complex, non-linear function of the BRDFs in the scene.

Acknowledgments We thank Peter-Pike Sloan and John Snyder for discussions regarding ZH lobe optimization strategies, Patricio Simari for his suggestions regarding model reduction techniques, Ryan Schmidt for the ball model, Simon Breslav for help with the figures, Christian Lessig for proofreading, Chris Landreth for the Angela dataset, Karan Singh for the Fit datasets, Eitan Grinspun and Rony Goldenthal for the cloth dataset, Joel Anderson for the T-rex dataset, Michael Comet for the muscle dataset, Robert Wang for joint angle datasets, Zhong Ren for the humanoid datasets, Doug James, Christopher Twigg, Robert Sumner and Jovan Popovic for the animal datasets, and Alexis Angelidis for the Pai datasets. We thank Paul Debevec for the light probes. The Eurographics reviewers provided many thoughtful comments, corrections and suggestions. The motion capture data used on the Fit and Pai datasets was obtained from mocap.cs.cmu.edu. The database was created with funding from NSF EIA-0196217. This research is partially funded by 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.

References

[ADM*08] Annen T., Dong Z., Mertens T., Bekaert P., Seidel H.-P., Kautz J.: Real-time, all-frequency shadows in dynamic scenes. *ACM Trans. Graph.* (2008).

[DKTS07] Dong Z., Kautz J., Theobalt C., Seidel H.-P.: Interactive global illumination using implicit visibility. In *Pacific Graphics* (2007), IEEE.

[DSDD07] Dachsbacher C., Stamminger M., Drettakis G., Durand F.: Implicit visibility and antiradiance for interactive global illumination. *ACM Trans. Graph.* (2007).

[FPJY07] Feng W.-W., Peng L., Jia Y., Yu Y.: Large-Scale Data Management for PRT-Based Real-Time Rendering of Dynamically Skinned Models. In *EGSR* (2007).

[JT05] James D. L., Twigg C. D.: Skinning mesh animations. *ACM Trans. Graph.* (2005).

[KA06] Kontkanen J., Aila T.: Ambient occlusion for animated characters. In *EGSR* (2006).

[KA07] Kirk A. G., Arikan O.: Real-time ambient occlusion for dynamic character skins. In *I3D* (NY, USA, 2007), ACM.

[Kaj86] Kajiya J. T.: The rendering equation. *SIGGRAPH* (1986).

[KLA04] Kautz J., Lehtinen J., Aila T.: Hemispherical rasterization for self-shadowing of dynamic objects. In *EGSR* (Norrköping, Sweden, 2004), EG Association.

[KM99] Kautz J., McCool M. D.: Interactive rendering with arbitrary BRDFs using separable approximations. In *EGRW* (1999).

[KNS*09] Kalogerakis E., Nowrouzezahrai D., Simari P., McCrae J., Hertzmann A., Singh K.: Data-driven curvature for real-time line drawing of dynamic scenes. *ACM Trans. Graph. (To Appear)* (2009).

[KSS02] Kautz J., Sloan P.-P., Snyder J.: Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In *EGSR* (2002), EG Association.

[LSSS04] Liu X., Sloan P. P., Shum H. Y., Snyder J.: All-frequency precomputed radiance transfer for glossy objects. In *EGSR* (Norrköping, Sweden, 2004), EG Association.

[MHL*06] Ma W.-C., Hsiao C.-T., Lee K.-Y., Chuang Y.-Y., Chen B.-Y.: Real-time triple product relighting using spherical local-frame parameterization. *Vis. Comput.* (2006).

[MPBM03] Matusik W., Pfister H., Brand M., McMillan L.: A data-driven reflectance model. *SIGGRAPH* (2003).

[NKSF08] Nowrouzezahrai D., Kalogerakis E., Simari P., Fiume E.: Shadowed relighting of dynamic geometry with 1D BRDFs. In *Eurographics: Short Papers* (2008).

[NRH03] Ng R., Ramamoorthi R., Hanrahan P.: All-frequency shadows using non-linear wavelet lighting approximation. In *SIGGRAPH* (2003), ACM.

[NRH04] Ng R., Ramamoorthi R., Hanrahan P.: Triple product wavelet integrals for all-frequency relighting. In *SIGGRAPH* (2004), ACM.

[NSK*07] Nowrouzezahrai D., Simari P., Kalogerakis E., Singh K., Fiume E.: Compact and efficient generation of radiance transfer for dynamically articulated characters. In *GRAPHITE* (2007), ACM.

[RGK*08] Ritschel T., Grosch T., Kim M. H., Seidel H.-P., Dachsbacher C., Kautz J.: Imperfect shadow maps for efficient computation of indirect illumination. In *SIGGRAPH Asia '08* (2008), ACM.

[RWS*06] Ren Z., Wang R., Snyder J., Zhou K., Liu X., Sun B., Sloan P.-P., Bao H., Peng Q., Guo B.: Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. In *SIGGRAPH* (2006), ACM.

[SGNS07] Sloan P.-P., Govindaraju N. K., Nowrouzezahrai D., Snyder J.: Image-based proxy accumulation for real-time soft global illumination. In *Pacific Graphics* (2007), IEEE.

[SHHS03] Sloan P.-P., Hall J., Hart J., Snyder J.: Clustered principal components for precomputed radiance transfer. *ACM Trans. Graph.* (2003).

[SKS02] Sloan P.-P., Kautz J., Snyder J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH* (2002), ACM.

[Slo08] Sloan P.-P.: Stupid spherical harmonics (sh) tricks. In *Game Developers Conference* (2008).

[SLS05] Sloan P.-P., Luna B., Snyder J.: Local, deformable precomputed radiance transfer. In *SIGGRAPH* (2005), ACM.

[SN08] Snyder J., Nowrouzezahrai D.: Fast soft self-shadowing on dynamic height fields. In *EGSR* (2008).

[TS06] Tsai Y.-T., Shih Z.-C.: All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. In *SIGGRAPH* (2006), ACM.

[WTL04] Wang R., Tran J., Luebke D. P.: All-frequency relighting of non-diffuse objects using separable BRDF approximation. In *EGSR* (2004).

[WTL06] Wang R., Tran J., Luebke D.: All-frequency relighting of glossy objects. *ACM Trans. Graph. 25*, 2 (2006).

[WXZ*06] Wang J., Xu K., Zhou K., Lin S., Hu S., Guo B.: Spherical harmonics scaling. *Vis. Comput.* (2006).

[ZHL*05] Zhou K., Hu Y., Lin S., Guo B., Shum H.-Y.: Precomputed shadow fields for dynamic scenes. In *SIGGRAPH* (2005), ACM.