Inverse Shade Trees for Non-Parametric Material Representation and Editing

Inverse Shade Trees for Non-Parametric Material Representation and Editing

Jason Lawrence[1] Aner Ben-Artzi[2] Christopher DeCoro[1] Wojciech Matusik[3] Hanspeter Pfister[3]
Ravi Ramamoorthi[2] Szymon Rusinkiewicz[1]

[1]Princeton University [2]Columbia University [3]MERL

Abstract

Recent progress in the measurement of surface reflectance has created a demand for non-parametric appearance representations that are accurate, compact, and easy to use for rendering. Another crucial goal, which has so far received little attention, is *editability*: for practical use, we must be able to change both the directional and spatial behavior of surface reflectance (e.g., making one material shinier, another more anisotropic, and changing the spatial "texture maps" indicating where each material appears). We introduce an *Inverse Shade Tree* framework that provides a general approach to estimating the "leaves" of a user-specified shade tree from high-dimensional measured datasets of appearance. These leaves are sampled 1- and 2-dimensional functions that capture both the directional behavior of individual materials and their spatial mixing patterns. In order to compute these shade trees automatically, we map the problem to matrix factorization and introduce a flexible new algorithm that allows for constraints such as non-negativity, sparsity, and energy conservation. Although we cannot infer every type of shade tree, we demonstrate the ability to reduce multi-gigabyte measured datasets of the Spatially-Varying Bidirectional Reflectance Distribution Function (SVBRDF) into a compact representation that may be edited in real time.

Keywords: Light Reflection Models, Non-Parametric, Data-Driven, Matrix Factorization, SVBRDF, BRDF

1 Introduction

The use of measured surface reflectance has the potential to bring new levels of photorealism to renderings of complex materials. Such datasets are becoming common, with recent work on acquiring dense measurements of both individual materials [Marschner et al. 1999; Matusik et al. 2003] and spatially-dependent reflectance [Dana et al. 1999; McAllister 2002; Lensch et al. 2003; Han and Perlin 2003; Marschner et al. 2005]. The availability of such data, however, has highlighted the difficulty of representing complex materials accurately using conventional analytic reflectance models [Ngan et al. 2005]. Non-parametric representations provide greater accuracy and generality, but so far have not incorporated the important design goal of *editability*. That is, in order to be useful in a practical production pipeline, an appearance representation must let the designer change both the spatial and directional behavior of surface reflectance. This paper proposes a compact tree-based representation (Figure 1) that provides the intuitive editability of parametric models while retaining the accuracy and flexibility of general linear decomposition methods.

The concept of composing a complex shading function from a tree-structured collection of simpler functions and masks was introduced in the seminal "shade trees" work of Cook [1984]. We develop an *Inverse Shade Tree* (IST) framework that takes as input a measured appearance dataset and a (user-supplied) tree structure, and fills in the leaves of the tree. In our trees, the leaves are sampled curves and maps (i.e., 1-D and 2-D) representing intuitive concepts such as specular highlight shape or texture maps. They are combined at interior nodes, of which the most common is a sum-of-products "mixing" node (other node types considered in this paper

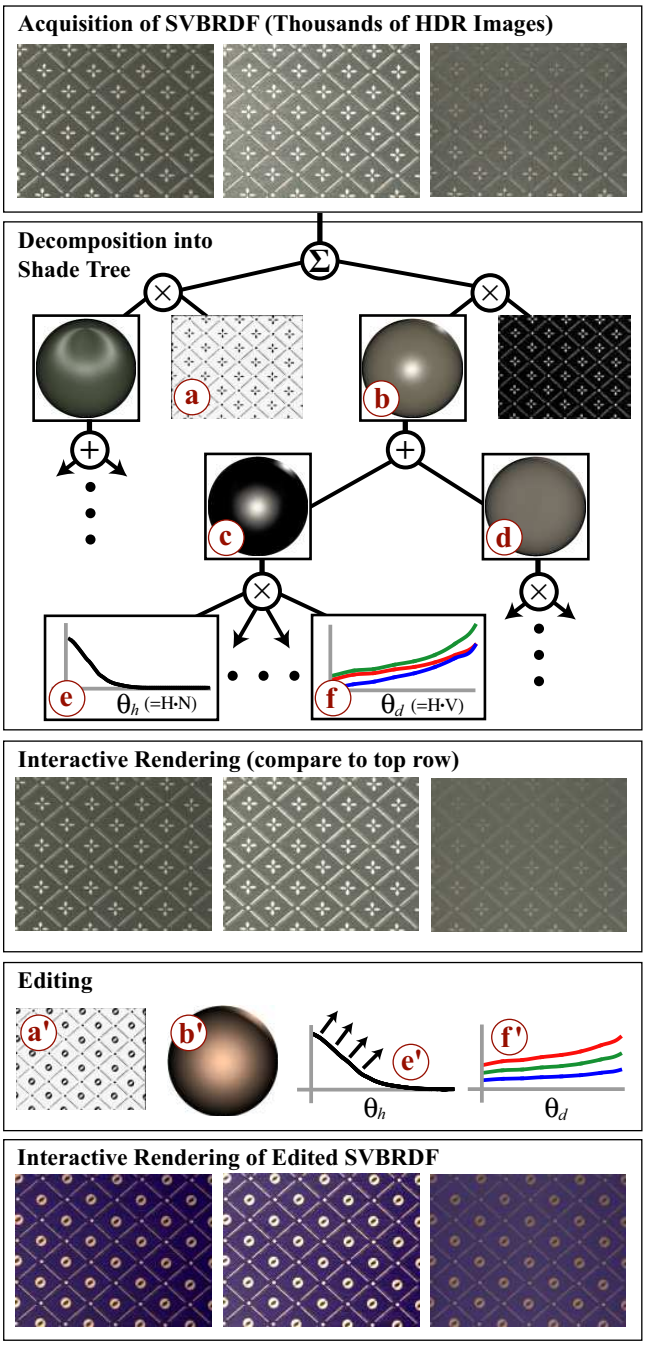


Figure 1: *We introduce a non-parametric framework for decomposing measured SVBRDF data into a set of (a) spatially-varying blending weight maps and (b) basis BRDFs. The basis BRDFs are factored into sampled 2D functions corresponding to (c) specular and (d) diffuse components of reflectance (we show lit spheres rendered with these factors, not the 2D factors). These 2D functions are further decomposed into (e & f) 1D curves. In addition to providing accurate interactive rendering of the original SVBRDF, this representation also supports editing either (a') the spatial distribution of the component materials or (b') individual material properties. The latter is accomplished by editing (e' & f') the sampled curves.*

include normal and tangent maps, as well as compositing operations such as "over"). For example, in the first level of the tree in Figure 1, the Spatially-Varying Bidirectional Reflectance Distribution Function or SVBRDF is composed of a sum of products of spatial mixing weights (a) and basis BRDFs (b). The IST decomposition proceeds top-down, at each stage decomposing the current dataset according to the type of node encountered in the tree.

The editability of the resulting shade trees depends on having their leaves correspond to pieces that are meaningful to the user. For example, when decomposing an SVBRDF, we would like the resulting BRDFs to correspond to our intuitive notion of separate materials, instead of being arbitrary linear combinations. For this reason, this paper focuses on the "unmixing" problem, showing how to map it to matrix factorization (Section 4). We introduce flexible algorithms based on linear constrained least squares that are designed to produce intuitive decompositions. These algorithms can incorporate constraints such as non-negativity, and provide control over the sparsity in the decomposition (resulting in a continuous tradeoff between pure factorization and clustering). As compared to existing methods, we maintain accuracy while producing editable *parts-based* separations. When the original function is a SVBRDF, these "parts" correspond to different materials; when the function is a BRDF, these "parts" correspond to different scattering phenomena, such as diffuse reflection, specularity, or back-scattering. In addition, our algorithms incorporate domain-specific constraints such as energy conservation, and deal with practical issues such as large datasets and confidence weighting. We expect these techniques to be generally applicable to data dimensionality reduction applications, beyond the task of material representation addressed here.

We explore inverse shade trees in a prototype system that begins with densely measured spatially-varying reflectance (with raw dataset sizes of several gigabytes), and generates compact and intuitive trees. We demonstrate that the resulting trees permit real-time non-parametric editing (Section 6) of materials and their spatial distribution, and analyze the accuracy (Section 7) of both the material separation and BRDF decomposition stages.

2 Relationship to Previous Work

Parametric Models for Reflectance: Fitting analytic reflectance models to data has been a widely-adopted approach, and some models were in fact developed specifically for fitting to measurements [Ward 1992; Lafortune et al. 1997]. Thus, one possible representation of a measured SVBRDF is a collection of analytic BRDF parameters at each surface location [McAllister 2002; Gardner et al. 2003]. Such a representation provides for easy editing of materials, and with the addition of a clustering step [Lensch et al. 2003] allows editing a single material everywhere it appears on a surface.

These approaches, however, have several key drawbacks. Reducing a dense set of measurements to a handful of parameters may introduce significant error [Ngan et al. 2005]. Moreover, it requires non-linear optimization, which is computationally expensive and numerically unstable. Finally, clustering the values of the BRDF parameters [Lensch et al. 2003] does not generate a desirable separation of the component materials in the presence of blending on the surface (even, in some cases, the trivial pixel-level blending present at antialiased material edges). This is both because the problem is underconstrained and because the parameters of most BRDF models are not linearly related.

The work of Goldman et al. [2005] is most similar to our own. They fit a convex combination of two analytic BRDFs (along with surface normals) at each surface location. This results in a sparse, non-negative representation, although their sparsity constraint is less general than the one we introduce in Section 4.2.2. Moreover, they use an isotropic Ward BRDF model which is more restrictive than our data-driven approach.

In this paper, we solve the *material separation* problem using the measurements directly, before fitting any secondary models to individual BRDFs. This allows for arbitrary blending of materials, giving correct results when parametric approaches fail (see Section 7 for a comparison of accuracy). In addition, we use a non-parametric representation of BRDFs based on a small set of intuitive curves, providing both generality and greater accuracy for some classes of materials that exhibit anisotropy or retroreflection.

Non-Parametric Models and Matrix Decomposition: Non-parametric approaches, including basis function decomposition [Dana et al. 1999] and standard matrix rank-reduction algorithms such as PCA, can retain high fidelity to the original data. In the context of appearance representation, researchers have explored a variety of rank reduction algorithms, including variants of PCA [Kautz and McCool 1999; Furukawa et al. 2002; Vasilescu and Terzopoulos 2004], homomorphic factorization [McCool et al. 2001], ICA [Tsumura et al. 2003], k-means clustering [Leung and Malik 2001], and NMF [Chen et al. 2002; Lawrence et al. 2004; Peers et al. 2006].

Though our approach also falls in the category of dimensionality reduction, we build on prior work by performing a multi-level sequence of decompositions, rather than just a single one. More importantly, motivated by an evaluation of existing methods, we introduce a set of new matrix factorization algorithms (Section 4) specifically designed to provide *editable* decompositions, a criterion for which existing methods are not optimized.

Non-Parametric Material Editing: A data-driven approach to BRDF editing has been proposed by Matusik et al. [2003], in which a user labels directions (e.g., "shininess", "rustiness", etc.) in a high-dimensional space of measured materials. The large number of materials, and the uncompressed representation of each BRDF, contribute to large storage requirements and inhibit interactive control. Though our BRDF editing system is also based on measured data, we provide for direct manipulation of curves controlling the reflectance instead of focusing on higher-level behaviors. In addition, our compressed representation and interactive renderer permit real-time manipulation of materials, including spatial variation.

To our knowledge, only two systems have demonstrated material editing via direct manipulation of low-dimensional factors of the BRDF [Ashikhmin et al. 2000; Jaroszkiewicz and McCool 2003]. In these systems the user edits a 2D image, which is then used as a component in a parametric or homomorphic-factored BRDF model. We generalize this by allowing interactive control over all the 1D and 2D factors necessary to specify a full SVBRDF.

3 System Overview

Although the shade tree framework could in principle represent many types of appearance data, including BTFs, BSSRDFs, light fields, and time-varying textures, this paper focuses on SVBRDFs. Here we provide a brief overview of our full pipeline, ranging from measurement through representation, rendering, and editing.

Appearance Acquisition: We used a spherical gantry with computer control over camera and light source direction [Marschner et al. 2005] to record a set of high-dynamic-range images of planar samples with spatially-varying material under many illumination and view directions. The choice to measure mostly-planar samples simplified the calibration and registration, but the techniques presented here apply to arbitrary geometry. We acquired five different SVBRDFs that include layered materials, anisotropy, retroreflection, and spatially-varying normal and tangent directions [1].

After geometric and photometric calibration, the images are reprojected onto the best-fit plane of the surfaces, yielding a uniform spatial sampling (at approximately 500×500 points) of reflectance measurements. Because the SVBRDF is a function of six variables (i.e., it can be written $S(u, v, \omega_i, \omega_o, \lambda)$, where λ is discretized into RGB or HSV bands), sampling the illumination and view directions uniformly and densely is impractical. Therefore, we sampled the forward- and backward-scattering lobes of the reflectance more densely than the other regions of the domain, yielding a total of between 2,000 and 6,000 reflectance measurements for each point on the surface. The scattered data are conceptually resampled onto a uniform grid using the push-pull algorithm [Gortler et al. 1996] with a Gaussian reconstruction kernel (as we shall see later, we avoid a complete reconstruction by using a subsampling method).

[1] Complete dataset available at: `http://ist.cs.princeton.edu`

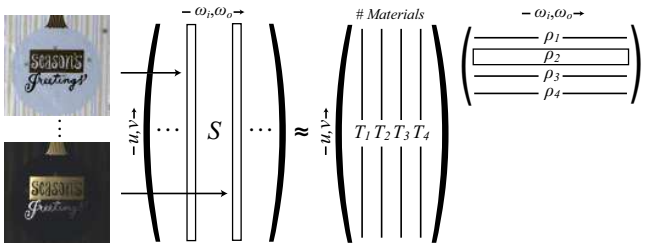
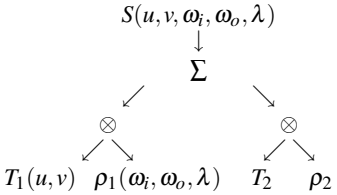


Figure 2: *We cast the SVBRDF decomposition depicted in Tree 1 as the factorization of a matrix. (a) High-dynamic range images of a SVBRDF captured with a spherical gantry are (b) organized into a matrix with rows that vary along spatial position and columns that vary along incident and outgoing angles. This matrix is factored into the outer product of (c) functions of spatial position ("blending weights") and (d) functions of incident and reflected directions (i.e., "basis BRDFs" in tabular form). In this example, we factor the SVBRDF of a holiday greeting card into four terms.*

Decomposition: We produce a shade tree with a series of decompositions of the SVBRDF and, using the same algorithms, the component BRDFs. For example, consider Figure 1(top), which shows a few images from a dense set of measurements of the SVBRDF of an anisotropic wallpaper sample. The first level of our decomposition separates the SVBRDF into 4D functions that depend on directions of incidence and reflection ("basis BRDFs," shown in Figure 1b as lit spheres) and 2D functions of spatial position ("blending weights," shown in Figure 1a). We represent the decomposition with this tree diagram:

Tree 1:

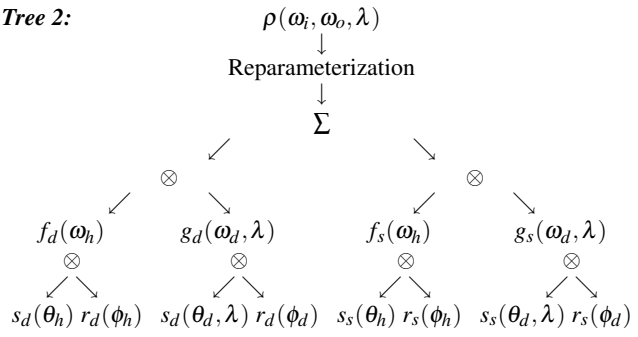


Note that we have chosen to associate color (i.e., λ) with the BRDFs. However, if it were more convenient for later editing, we could have associated color with the spatial blending weights instead, resulting in a *color* texture and *colorless* basis BRDFs.

Although we have reduced the size of the original SVBRDF, these basis BRDFs are still tabular representations of a 4D function, making them unsuitable for interactive rendering or editing. To address this, we further reduce the basis materials through a series of decompositions into 2D functions and eventually into 1D curves. Although we introduce these decompositions in the context of representing the materials contained within a SVBRDF, they provide, in general, an editable non-parametric representation of the BRDF.

For the example in Figure 1, each basis BRDF is decomposed into two terms, each a product of 2D functions of half (ω_h) and difference (ω_d) angles [Rusinkiewicz 1998]. For the shiny gold material, one term corresponds roughly to the specular component of the reflectance (Figure 1c), while the other represents near-Lambertian diffuse (Figure 1d). For improved editability, we may further factor the 2D functions into products of 1D curves defined on the corresponding elevation and azimuthal angles (Figures 1e and f):

Tree 2:



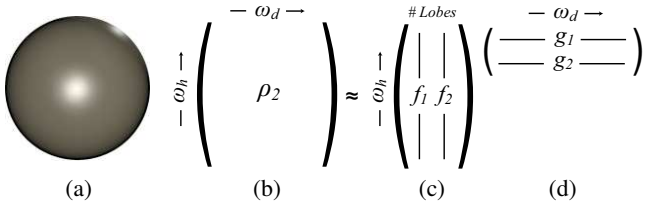


Figure 3: *We cast the BRDF decomposition in Tree 2 to matrix factorization. Each (a) tabular BRDF (shown as a lit sphere) is (b) re-parameterized and then rasterized into a matrix with rows that vary along the half-angle and columns that vary with difference-angle. We factor this matrix into (c) functions of the half-angle and (d) functions of the difference angle.*

While the decomposition into 1D curves results in a simpler representation and is desirable for isotropic BRDFs (which are invariant to ϕ_h), we have found that it may reduce accuracy for some complex anisotropic materials. In these cases, we terminate the decomposition one level higher, resulting in 2D maps resembling those of Ashikhmin et al. [2000] and Ngan et al. [2005].

Rendering and Editing: The measured materials may now be rendered by re-composing the shade trees in a pixel shader; this provides interactive feedback during editing. In most cases, the 1D functions or curves at the leaves of the tree correspond naturally to physical phenomena controlled by parameters in existing analytic models. For example, the $s_s(\theta_h)$ curve is related to the distribution of microfacets on the surface, and hence determines the shape of the specular highlight. The $s_s(\theta_d,\lambda)$ curve describes the behavior of the specular lobe as the view direction moves from normal incidence to grazing, capturing Fresnel effects such as color shifts, increased specular reflection, and a reduced diffuse term. Color variation, such as in $s_s(\theta_d,\lambda)$, can be represented with separate curves for each RGB color component (Figure 1f), or in an alternate colorspace such as HSV.

4 Algorithms for Matrix Factorization

We can cast the tree-structured decompositions described previously as a sequence of matrix factorizations. At the top-level (Tree 1), we organize samples of a SVBRDF into a matrix that is factored into the outer product of 2D blending weights and 4D basis BRDFs (see Figure 2). At the second-level (Tree 2), we decompose each basis BRDF into appropriate 2D factors by computing another matrix factorization (see Figure 3).

There are a variety of algorithms available for computing these factorizations. In Section 4.1, we compare existing approaches and discuss the conditions under which they fail to provide a meaningful decomposition. In Section 4.2, we introduce a new factorization algorithm based on linearly constrained optimization that improves the separation in challenging cases. One key benefit of this new algorithm is that it can incorporate domain-specific constraints for decomposing appearance data (Section 4.3). Lastly, we address practical considerations related to scattered input data and matrices whose sizes exceed the capacity of main memory (Section 4.4).

We will adopt the notation of [Lee and Seung 2000] to discuss matrix factorizations. Specifically, a $n \times m$ input matrix V is approximated as the product of a $n \times k$ matrix W and a $k \times m$ matrix H ($V \approx WH$). Rank reduction occurs when k is smaller than n and m, and we are most interested in cases of extreme compression (e.g., n and m are hundreds or thousands, while k is between 1 and 5). We consider factorizations that minimize the Euclidean error,

$$\|V - WH\|^2 = \sum_{ij} \left(V_{ij} - (WH)_{ij}\right)^2. \quad (1)$$

To begin, we focus on SVBRDF decomposition, so that the original data is in matrix V and the mixing weights and basis BRDFs end up in W and H, respectively. Later, we will consider BRDFs, and W and H hold sampled half-angle and difference-angle maps.

Figure 4: *Two images (originals are HDR) from the "Season's Greetings" dataset, together with hand-generated mixing masks that would be produced by an ideal decomposition. Notice that the separation is* ***soft****, with significant blending between the gold foil and both the blue and white paper. Since the blending weights in the bottom row are colorless scalars (the color for this shade tree is in the BRDFs), we use grayscale images to visualize them.*

Algorithm Groups	Properties:		
	Linear	Positive	Sparse
SVD / ICA	Yes	No	No
Homomorphic	No	Yes	No
Clustering	No	Yes	Yes
NMF / pLSI	Yes	Yes	No
Our Method: ACLS	Yes	Yes	Yes

Table 1: *Comparison of matrix factorization algorithms. Existing methods do not satisfy the three properties of linearity, positivity, and control over sparsity, which are critical for a meaningful editable decomposition.*

4.1 Evaluation of Existing Algorithms

We compare several classes of factorization algorithms suitable for accurately representing measured appearance data, evaluating their performance on the *Season's Greetings* dataset shown in Figure 4. This measured SVBRDF is of a holiday greeting card with four materials (blue and white paper, and gold and silver foil), as shown in the ideal separation at bottom. Note that the materials are smoothly blended over the surface. For example, the gold foil is present in different amounts at the boundary between the gold border around the word "Season's" and the paper background (Figure 4, T_2 and T_3). Also, the stripes in the background were created by halftoning the gold material over the paper background. This is visible as spatial blending between materials T_2 and T_4.

To represent these effects, while providing an editable decomposition, we have identified three key properties of a factorization algorithm. First, it should allow for a basis consisting of linear combinations of the input to resolve the blending of different materials. Second, the algorithm should guarantee non-negativity to produce a physically plausible result and favor parts-based decompositions. Third, the algorithm should provide control over the sparsity of the solution, favoring a representation that uses individual materials, where possible, to represent the SVBRDF (as opposed to blending between materials across the *entire* surface). Table 1 summarizes these properties for different algorithm classes.

PCA/ICA: Two popular rank reduction algorithms are Principal Component Analysis (PCA) and Independent Component Analysis (ICA), along with extensions such as multilinear tensor factorization [Vasilescu and Terzopoulos 2004]. The main advantage of PCA is that it yields a *global* minimum of Equation 1. However, these algorithms recover a basis that is orthonormal (for PCA) or statistically independent (for ICA). These restrictions are not sufficient to produce a meaningful description of the data. In particular,

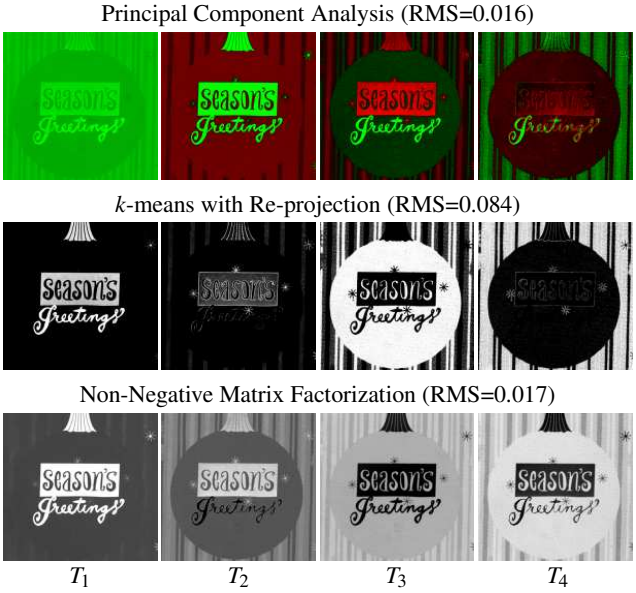


Figure 5: *Blending weights computed from the "Season's Greetings" dataset using the factorization algorithms discussed in Section 4. For PCA, these terms are visualized as images where red and green correspond to positive and negative values with luminance proportional to the magnitude. For k-means and NMF, all values are non-negative and are visualized as grayscale images. Note that neither PCA nor NMF provide a separation of the data into distinct* parts *suitable for editing. Although clustering performs better, it too fails to recover the desirable separation into the four component materials present in this sample (Figure 4,bottom row). In particular, k-means assigns both the gold and silver foil to a single cluster (T_1) and combines the gold foil and paper into a separate term (T_2).*

they allow negative values in W and H, resulting in a representation whose terms cannot be edited independently (Figure 5, top).

Homomorphic Factorization: Introduced in the context of representing non-parametric BRDFs, Homomorphic Factorization [McCool et al. 2001], decomposes a high-dimensional function into a single *product* of an arbitrary number of lower dimensional functions. Although it can support an arbitrary number of factors, it does not allow linear combinations. Hence, this algorithm is not appropriate for representing the SVBRDF as a *sum of products* of basis materials and spatial blending weights, or decomposing a BRDF into a *sum* of diffuse, retroreflective and specular lobes.

Clustering: One popular method for clustering data is the k-means algorithm [Hartigan and Wong 1979]. Like all clustering algorithms, k-means partitions the input into disjoint sets, associating each point with a representative point called the *cluster center*. This can be interpreted as a factorization of the SVBRDF, where the cluster centers are stored in the matrix H and W is computed by re-projecting the data onto this basis (using gradient descent for example). In our experiments, clustering performs well on input with a small basis that is well-separated over the surface. However, when the SVBRDF exhibits blending of its component materials, clustering typically fails to recover a useful basis. For example, in Figure 5, middle, k-means has incorrectly assigned a single cluster to the *combination* of the gold foil and paper (T_2) while grouping the gold and silver foils into a separate cluster (T_1).

Non-Negative Matrix Factorization: Another matrix decomposition approach is Non-Negative Matrix Factorization (NMF) [Lee and Seung 2000]. Together with similar algorithms such as Probabilistic Latent Semantic Indexing [Hofmann 1999], NMF guarantees that both resulting factors contain only non-negative values. One motivation for this constraint is to encourage the algorithm to describe the input data as the sum of positive *parts*, thereby producing a more meaningful factorization. In our experiments, however, the character of the decomposition is sensitive to small changes in the data (including those due to measurement noise and misalignment), and the non-negativity constraint is not always enough to guarantee an editable separation (see Figure 5, bottom).

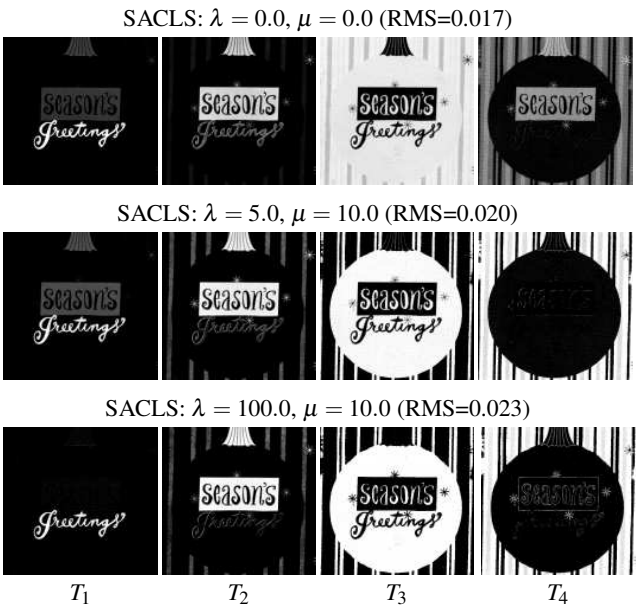


Figure 6: *The blending weights computed from the "Season's Greetings" dataset using SACLS with different settings of λ and μ. Increasing values of λ force the algorithm to behave more like clustering, trading numerical accuracy for a more meaningful separation.*

4.2 Our Method: Alternating Constrained Least Squares

We have seen that existing matrix factorization methods do not fulfil the three properties (linearity, positivity, sparsity) needed to produce meaningful, editable decompositions. We now describe a new suite of algorithms that allow for these, while also supporting additional domain-specific constraints such as energy conservation in the SVBRDF and monotonicity of the BRDF (Section 4.3).

Our algorithm is built upon efficient numerical methods for solving linear constrained least squares (LCLS) problems of the form:

$$\underset{x \in R^n}{\text{minimize}} \ \frac{1}{2}\|b - Mx\|^2 \quad \text{subject to} \quad l \leq \begin{Bmatrix} x \\ Ax \end{Bmatrix} \leq u \qquad (2)$$

The n-element vector x is called the *vector of unknowns*, M is called the *least-squares matrix* and b is the *vector of observations*. The vectors u and l provide the upper and lower *bound constraints* of both x and the linear combinations encoded in the matrix A, called the *general constraints*. There are several algorithms available for solving these types of problems. We use an inertia-controlling method that maintains a Cholesky factorization of the reduced Hessian of the objective function [Gill et al. 1984]. We use an implementation of this algorithm from the Numerical Algorithms Group (NAG) library set, called `nag-opt-lin-lsq` [NAG 2005].

4.2.1 Non-Negative Factorization

As with NMF, we initialize W and H to contain positive random values, and minimize Equation 1 by alternately updating these two matrices. This problem is known to be convex in either W or H separately, but not simultaneously in both. As a consequence, we will present an algorithm that finds a *local* minimum of Equation 1.

Without loss of generality, we consider the case where both V and W are row vectors ($v \approx wH$). We later extend the discussion to consider the entire matrix W. For a fixed H, we update our current estimate of w by minimizing Equation 1, subject to the linear constraint $w \geq 0$. To accomplish this, we solve the LCLS problem in Equation 2, with $M = H^T$, $b = v^T$, and $x = w^T$. To constrain the solution to be non-negative, we set $l = 0$ and $u = \infty$.

We update the entire matrix W by computing the above solution for each of its rows in turn. Similarly, we can transpose the problem, take W to be the *least-squares matrix* M, and update our estimate of H one column at a time. By alternating between estimating W and H, we achieve a non-negative factorization of the input matrix V. Because we are guaranteed never to increase Equation 1 after either update, this algorithm, which we call Alternating Constrained Least Squares (ACLS), is guaranteed to converge to a local minimum.

Compared to NMF, for which each iteration requires only a few matrix multiplications, each iteration of ACLS is considerably more expensive. On the other hand, each iteration of ACLS results in a greater decrease in error and it converges with an order of magnitude fewer iterations. In our experiments, we have found the overall computation time for these two algorithms to be comparable.

4.2.2 Sparsity

A non-negativity constraint is frequently not enough to provide an intuitive *parts-based* decomposition. We introduce a modification that considers the *sparsity* of the solution, providing a continuous tradeoff between non-negative matrix factorization and clustering. We have found this flexibility to be effective for decomposing SVBRDFs exhibiting complex blending of multiple materials.

In order to define sparsity, consider the SVBRDF factorization shown in Figure 2. A sparse decomposition is one in which there is a linear combination of relatively few basis materials at each surface location. That is, each *row* of W has few non-zero entries. Although there are several expressions that quantify this notion, we require one that leads to a linear least-squares problem: it must be quadratic in the elements of the row. Therefore, we define the *sparsity penalty* for a row w as the sum of the squares of all but one of the coordinates of w (i.e., $\sum_{i \neq j} w_i^2$, where the selection of j is discussed below). For a fixed H, we can combine this sparsity penalty, weighted by a parameter λ, with the approximation error (1), which gives a new error to be minimized:

$$\|v - wH\|^2 + \lambda \sum_{i \neq j} w_i^2. \qquad (3)$$

One potential problem with this formulation is that we can decrease the overall error simply by decreasing the magnitude of *all* the elements of w. To address this, we introduce an additional soft constraint that the L_1 norm of w should be *close* to unity. As before, we can add this penalty, weighted by the parameter μ, to the error:

$$\|v - wH\|^2 + \lambda \sum_{i \neq j} w_i^2 + \mu \left(1 - \sum_i w_i\right)^2. \qquad (4)$$

Starting from Equation 4, we can write out the corresponding least-squares matrix and observation vector to be used in Equation 2:

$$b = \begin{pmatrix} v^T \\ \sqrt{\mu} \\ 0 \end{pmatrix} \quad M = \begin{pmatrix} — & H^T & — \\ — & \sqrt{\mu} \ldots \sqrt{\mu} & — \\ \sqrt{\lambda} \ldots \sqrt{\lambda} & 0 \ \sqrt{\lambda} \ldots \sqrt{\lambda} & \end{pmatrix}. \qquad (5)$$

The 0 in the bottom row of M will be at the j^{th} position.

Putting things together, we estimate w by iterating over the possible values of j (in practice this corresponds to the rank of the approximation and is small) and minimizing Equation 4; we retain the w that corresponds to the selection of j with the smallest error. The entire matrix W is estimated one row at a time in this fashion. We alternate between updating our estimate of W and H until the error converges to a local minimum. Assuming that both W and H are initially inside the feasible region, each iteration cannot increase Equation 4, so this algorithm, which we call Sparse Alternating Constrained Least Squares (SACLS), is guaranteed to converge. This is a critical property, and one not shared by some previous approaches to sparse factorization (such as that of Hoyer [2002]), which include a normalization step that can increase error.

The parameter λ influences the sparsity of the factorization, ranging from pure matrix factorization ($\lambda = 0$) to pure clustering ($\lambda \to \infty$). The parameter μ, in contrast, determines the extent to which we insist that the sum of material contributions at each location is 1. We have found the algorithm to be somewhat less sensitive to the selection of this parameter. As with previous work in low entropy coding [Olshausen and Field 2002], we define both λ and μ in units of the variance in V. This provides more intuitive control over these values, but trial and error is still required to determine their appropriate settings. Figure 6 illustrates the impact of different settings of λ on the decomposition.

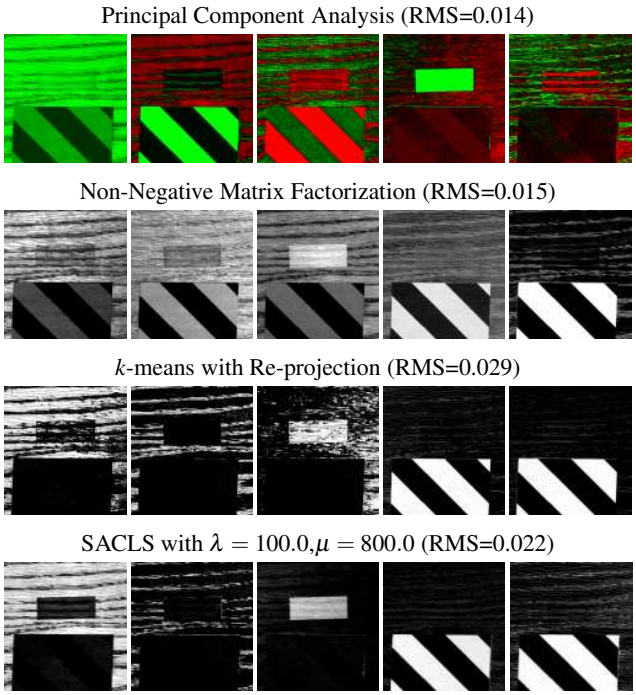


Figure 7: *Visual comparison of the spatial blending weights computed by several linear factorization algorithms on the "Wood+Tape" dataset. Our method (bottom row) provides control over sparsity and guarantees the component BRDFs are physically plausible (energy conserving and reciprocal). This aids in providing automatic separation of the measured data into its component materials and provides a final representation that can be edited (Figure 13).*

4.3 Domain-Specific Constraints

One advantage of the ACLS algorithm is that it can be easily extended to include additional linear constraints beyond non-negativity and sparsity. In this section, we introduce several useful constraints in the context of representing the SVBRDF and BRDF, including energy conservation and monotonicity.

4.3.1 SVBRDF Constraints: Energy Conservation

When factoring the SVBRDF, we can extend ACLS to guarantee that the basis BRDFs conserve energy. For convenience, suppose that H contains values of the BRDF for different light positions and a *single* viewing direction (these techniques can readily be extended to multiple viewing directions). In this simplified case, we can constrain the BRDF at the j^{th} row of H to conserve energy by bounding the sum of its values, each weighted by the solid angle:

$$\sum_i H_{ji}\,\Delta\omega_i \;\leq\; 1. \tag{6}$$

This constraint is incorporated into the ACLS framework by first linearizing the matrices V and H into column vectors $\tilde{v} = (V_{11}\,V_{12}\ldots V_{mn})^T$ and $\tilde{h} = (H_{11}\,H_{12}\ldots H_{km})^T$. From Equation 2, we set $b = \tilde{v}$, $x = \tilde{h}$, and define M and A as follows:

$$M = \begin{pmatrix} w_{11}\,0\,\cdots\,0\,w_{12}\,0\,\cdots\,0 \\ \vdots \\ 0\,\cdots\,0\,w_{m1}\,0\,\cdots\,w_{mk} \end{pmatrix} \quad A = \begin{pmatrix} \Delta\omega_1\cdots\Delta\omega_m\;\;0\;\;\cdots\;\;0 \\ \vdots \\ 0\;\;\cdots\;\;0\;\;\Delta\omega_1\cdots\Delta\omega_m \end{pmatrix}$$

Finally, we set the boundary constraints (i.e., l and u) to guarantee that H is non-negative and the sums encoded in the matrix A lie between 0 and 1. By solving Equation 2 under these substitutions, we guarantee that the BRDFs encoded in H conserve energy. With the added constraint on the parameterization of the BRDF that $\phi_d + \pi \rightarrow \phi_d$, we can also guarantee reciprocity. To our knowledge, this is the first factorization algorithm that guarantees an SVBRDF decomposition into *physically plausible* non-parametric BRDFs.

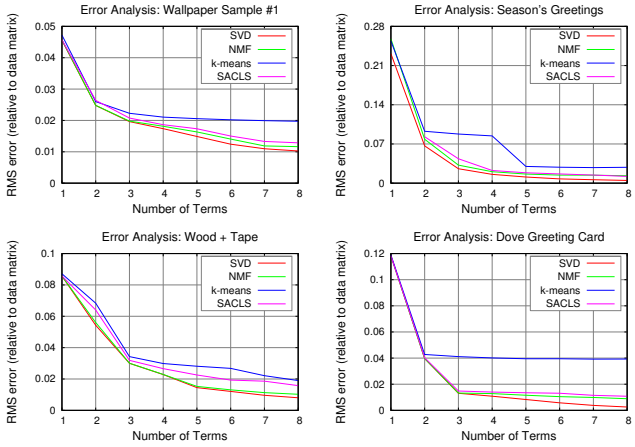


Figure 8: *Accuracy of representing four different SVBRDFs with four possible linear decomposition algorithms. For each dataset, the SACLS algorithm provides a representation with comparable numerical accuracy to existing data-driven approaches.*

Reciprocity and energy conservation constraints were used to perform material separation on all the samples considered in this paper. Figure 7 shows our separation for the *Wood+Tape* dataset. This particularly challenging SVBRDF consists of a piece of oak partially covered by semi-transparent tape and retro-reflective tape. Note that the tape completely disappears at certain incident and reflected directions (Figure 13, left column). On this data, PCA and NMF produce decompositions with significant mixing, while clustering improperly groups regions of the wood grain with the tape (Figure 7). On the other hand, SACLS correctly separates the SVBRDF into two different types of wood grain, a tape layer smoothly blended over the wood, and two separate terms for the retroreflective materials. We have observed similarly intuitive material separation results for all the datasets. Moreover, this intuitive separation comes at little or no decrease in the numerical accuracy. Figure 8 shows the cosine-weighted RMS error (defined as the square root of the sum of squared differences between the original images and the reconstruction, weighted by $cos(\theta_i)$) produced by four decomposition algorithms, for a range of different terms (number of materials).

4.3.2 BRDF Constraints: Value and Monotonicity

At the second level in our tree-structured decomposition (Tree 2), we factor a tabular BRDF into the sum of terms, each a product of functions of half- and difference-angle. As with the SVBRDF, this is equivalent to factoring a matrix (Figure 3).

Factoring the BRDF into multiple 2D terms using standard non-negative factorization (Section 4.2) generally yields factors that are arbitrary linear combinations whose values should not be independently edited. To address this, we allow for two types of constraints on these factors. First, we can constrain one of the half-angle terms in Figure 3c to remain at a constant value while allowing the rest of the factorization to update normally. This has the effect of separating the BRDF into a lobe with uniform θ_h dependence (typically diffuse-like terms, though not restricted to be perfectly Lambertian) plus a lobe with arbitrary half-angle distribution (usually a specular lobe). In all cases, the dependence on the difference angle is retained, allowing for Fresnel effects such as color shifts, increased specular reflection, and a reduced diffuse term.

Additionally, we can constrain the half-angle dependence of the "specular" term to be monotonically decreasing in θ_h, resulting in more physically plausible highlights. We constrain the derivative of f_i to be negative at each sample. Because this is a linear operator (e.g., central differences), it can be directly encoded into the matrix A along with the settings $l = -\infty$ and $u = 0$ in Equation 2. Figure 9 provides an example of using both value and monotonicity constraints for the shiny gold foil ρ_2 from Figures 2 and 3.

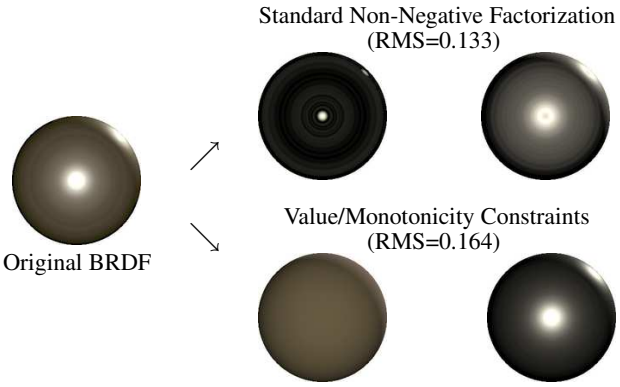


Figure 9: *The ACLS algorithm can be extended to incorporate value and monotonicity contraints. We factor a tabular BRDF (left) into the sum of two terms. At top, we use basic non-negative factorization (Section 4.2). At bottom, the two terms are computed by constraining one term to have uniform θ_h dependence while the other is monotonically decreasing in θ_h.*

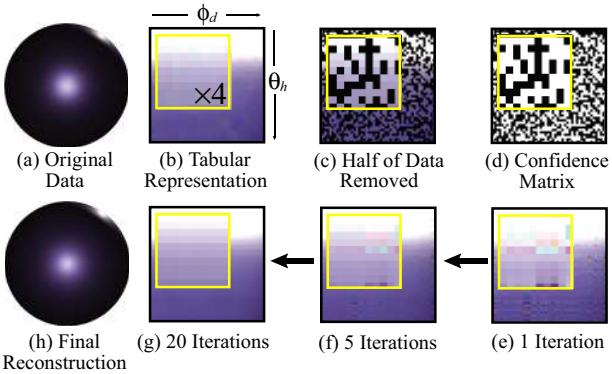


Figure 10: *Using a measured BRDF as input (a) we construct a data matrix indexed by θ_h and ϕ_d shown in (b). (Note: we show only one section of the complete data matrix by omitting variation in θ_d). For this test, we removed 50% of the data values to produce the matrix in (c) and compute a confidence matrix (d) where measured values have a confidence of 1 and missing values have a confidence of 0. We show the factorization computed by confidence-weighted ACLS after one (e) and five (f) iterations. After 20 iterations (g), we produce a matrix that approximates the original.*

4.4 Practical Considerations

To make these factorization algorithms practical, there are several issues we must consider. First, for most real-world data, the level of confidence of each measurement is not uniform across the input matrix V. For example, reflectance measurements from grazing angles will be less accurate then those from perpendicular angles. Additionally, some regions of the domain are not measured, producing "holes" in the input. We would like to associate a "confidence" with each value in the matrix in order to allow scattered data interpolation across missing regions in the input. Second, because of the high dimensionality and resolution of our datasets, the sizes of the matrices we factor often exceed the capacity of main memory. Finally, to help avoid incorrect local minima we initialize ACLS from multiple starting positions.

4.4.1 Missing Data and Confidence Weighting

We incorporate a confidence matrix C into our objective function to weight the contribution of each element in V towards the error:

$$\sum_{ij}\Big(C_{ij}\left(V_{ij}-(WH)_{ij}\right)\Big)^2. \tag{7}$$

An element with a confidence of 0 will have no effect on the factorization, seamlessly allowing for missing data. We can incorporate C into any ACLS variant through a simple modification to the least-squares matrix M and the observation vector b in Equation 2. For convenience, consider estimating a single row in W (denoted w) for a fixed H according to the corresponding rows in V and C (denoted v and c respectively). The related linear constrained problem from Equation 2 will have $b_j = c_j v_j$ and $M_{jk} = c_j H_{kj}$. Note that this reduces to standard ACLS for $c_j = 1$. Figure 10 shows the performance of confidence-weighted ACLS on a controlled example, where 50% of samples of a BRDF are removed.

In practice, we use the push-pull algorithm [Gortler et al. 1996] with a Gaussian kernel to reconstruct V from scattered data. We associate a confidence with each matrix cell that is proportional to its total weight at the top-level of the reconstruction pyramid. The confidence of cells that correspond to incident or reflected directions below the horizon are set to zero. The choice of interpolation method will affect the final output. We experimented with several techniques and found the push-pull algorithm to provide the best trade-off between quality and computational effort.

4.4.2 Subsampling for Large Datasets

Due to the high-dimensionality of the datasets we are interested in factoring, V often exceeds main memory. However, its rank is significantly smaller then the resolution of our measurements (i.e., $k \ll m$). We exploit this rank-deficiency by computing a basis for a subset of the complete data (call this V'). We use whichever variant of ACLS is appropriate to compute: $V' \approx W'H$. Because we discarded only *complete rows* of V, the matrix H can be thought of as an estimate of its basis. The original data is projected onto H using ACLS to estimate W while holding H fixed. This procedure requires storing only one row of V, one row of W and the complete H matrix in main memory at any given time. We can similarly reduce V by computing a factorization over a subset of the columns.

This strategy converges at least as quickly as standard ACLS even for aggressive downsampling. If the sample size is too small, however, the basis of V' will not accurately represent V, and the error will increase. For the datasets we consider, this situation only arises when using less than 0.01% of the original matrix.

In practice, we reconstruct the SVBRDF datasets containing isotropic materials at an angular resolution of $100 \times 30 \times 15$ ($\theta_h \times \theta_d \times \phi_d$), while representing anisotropic datasets with a 64×64 parabolic map [Heidrich and Seidel 1999] for the half-angle term at 30×15 different positions of θ_d and ϕ_d respectively. The spatial resolution of the samples are approximately 500^2. If we were to represent the SVBRDF matrix explicitly, this would require 125GB of memory for the isotropic case, and 5,149GB for anisotropic samples. Instead, we rely on subsampling: we compute blending weights using 50 columns of the original matrix at qualitatively different positions (i.e., specular highlights, back-scattering, perpendicular and grazing angles), and reconstruct full resolution tabular BRDFs at 100 randomly selected positions.

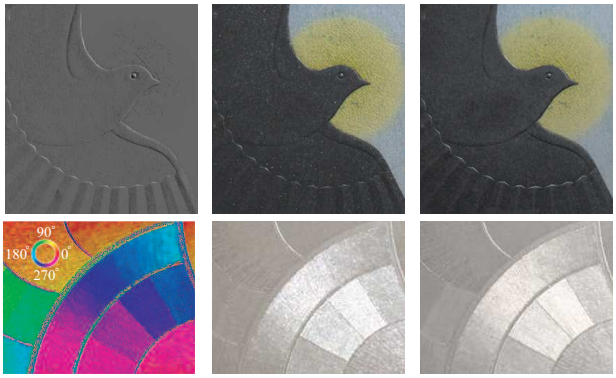
The top-level decomposition takes ∼2 hours for each dataset and those at lower levels in the tree require ∼30 minutes. In our experiments, these running times scale linearly with the input.

4.4.3 Initialization of ACLS

While the simplest strategy for all ACLS variants is to initialize the matrices W and H with random positive values (subject, of course, to any additional desired constraints such as monotonicity), the fact that ACLS performs local minimization leaves it susceptible to local minima. We have found that a more robust strategy is to first run k-means clustering with a relatively large k (for example, 20), then initialize ACLS with a random subset of the cluster centers. For even greater robustness, we repeat the ACLS minimization with many randomly-chosen subsets of cluster centers, and take as our final result the one with smallest error. In our experiments, this strategy is robust in avoiding incorrect local minima, and ameliorates the undesirable lack of provable global convergence (shared by all algorithms considered here, except PCA).

5 Normal and Tangent Estimation

For materials containing normal variation or (for anisotropic materials) variation in tangent direction, we can augment our SVBRDF decomposition tree with the addition of normal and/or tangent maps. If both are present, we are effectively estimating a full rotated coordinate system at each spatial location, thus capturing effects similar to those recently demonstrated by Marschner et al. [2005].

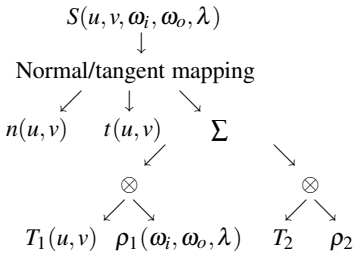


Normal/Tangent Map | Original Image | Shade Tree Reconstruction

Figure 11: *Normal and tangent maps.* **Left:** *We show $N \cdot L$ of the Dove normal map for a near grazing light direction. For tangent maps, we set the hue of each pixel to be the tangent direction. Note that this direction is undefined in regions with isotropic reflectance.* **Middle:** *original images.* **Right:** *images rendered using a 3-term shade tree with normal and tangent maps (note: blending weights not shown).*

All these effects are captured with the following tree, which we use instead of Tree 1:

Tree 3:



We estimate normal and tangent directions at each spatial position in three stages. First, we fit a generic BRDF with an anisotropic Gaussian specular lobe (i.e., a Ward BRDF) at each location, with the rotation angles defining the coordinate system as free parameters to the fit. Using these initial orientation estimates, we build the matrix described in the previous section and compute its k-term factorization. We then refine our estimates using this factorization, again solving for the best-fit normal and tangent. We can repeat these steps until the overall error converges, though in practice we found that two iterations are sufficient to accurately recover the fine geometric surface detail present in our samples and required $\sim$10 hours of total processing time. We show the final normal maps and tangent maps for two datasets in Figure 11.

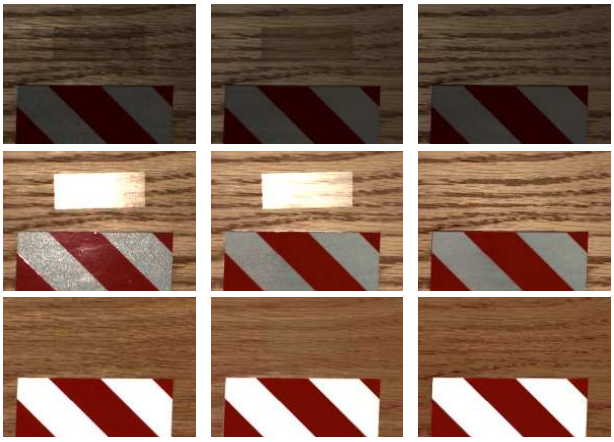
6 Results: Editing

The benefit of obtaining a decomposition of the SVBRDF into a meaningful shade tree is that any leaf node may be independently edited. In this section, we describe several possible edits at both the material level (altering the spatial texture of which component material appears where) and at the BRDF level (changing salient aspects of a material's reflectance using our curve-based model). The supplementary video shows further real-time editing results. While most of these edits are straightforward given our intuitive shade tree representation, they are to our knowledge the first demonstration of non-parametric editing of spatially-varying measured materials, and would not be easy with alternative matrix factorization methods, which do not provide a meaningful separation of materials or individual BRDFs.

6.1 SVBRDF Editing

Changing Spatial Distribution of Materials: Perhaps the most obvious edit is to change the spatial distribution or "texture" of the basis materials. In Figures 1 and 12 we have changed the texture by re-painting the blending weight maps. To achieve the edit shown in Figure 13, we first define a transparency mask for the tape as the product of its blending weights and a user-set constant. Our resulting shade tree composites the tape *over* the additional layers using this mask. Because our separation was not perfect for this challenging case, we manually repaired some error in the wood blending weights below the tape. We can also interactively re-position the tape (shown in the supplemental video) or remove it altogether. This edit would not be possible using the decompositions achieved by alternative algorithms (Figures 7 and 18).

Figure 12: *A key benefit of our framework is that it supports editing. Here we change the normal map and blending weights in the* Season's Greetings *shade tree: we retain the original BRDFs, but spell a slightly embossed SIGGRAPH 2B0ST0N6 (compare to Figure 4, top row).*



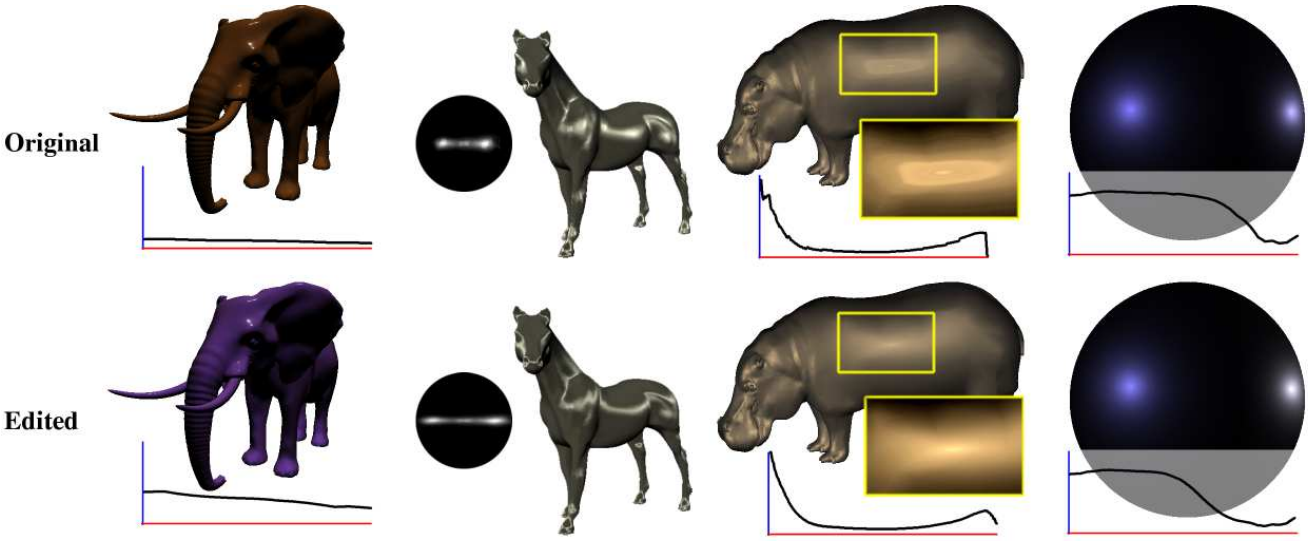
Original | Reconstruction | Removed Tape

Figure 13: *The* Wood+Tape *dataset consists of a piece of oak partially covered by semi-transparent tape and retroreflective bicycle tape.* ***Left:*** *Three original images illustrate that the tape disappears for some incident and reflected directions, making this a challenging dataset to separate into its component materials.* ***Middle:*** *Reconstruction provided by a shade tree with five terms, computed using the ACLS algorithm.* ***Right:*** *We edit the weight maps to remove the tape. Although our separation was not perfect, the resulting edits display few artifacts.*

Original | Metallic-Blue | Nickel

Figure 14: *An example of* ***material transfer****: one of the subtrees of our decomposition is completely replaced with a different one. Here, we replace one of the component BRDFs with several materials from Matusik's database [2003], while retaining spatial texture and normal maps.*

Changing or Combining Basis Materials: Figure 1 shows an edit in which one of the component materials was made less shiny (using the BRDF curve editing techniques discussed below), while the hue of the other material was changed. A related edit involves completely replacing basis materials with other measured BRDFs. In Figure 14 we replace the metallic-silver BRDF in the *Dove* data with several measured BRDFs from the Matusik [2003] database.



(a) Changing diffuse color by editing $s_d(\theta_d, \lambda)$ curve (b) Varying highlight shape by editing $f_s(\omega_h)$ map (c) Smoothing noisy data by editing $s_s(\theta_h)$ curve (d) Desaturating toward grazing by editing $s_s(\theta_d, \lambda)$ curve

Figure 15: *Our system allows for BRDF edits similar to those available with parametric representations, implemented by moving or warping the 1D curves and 2D maps defined in Tree 2.*

6.2 BRDF Editing

The BRDF edits available using our representation include many that have become familiar to users of parametric models, but have thus far not been easy to perform with non-parametric BRDFs. Several possibilities are shown in Figure 15:

a. The diffuse color is changed by editing the $s_d(\theta_d, \lambda)$ curve. Since this is represented in HSV space, it is easy to make changes to the overall color while maintaining any desaturation, color shift, or Fresnel effects present in the original data.
b. The shape of the highlight is represented by the $f_s(\omega_h)$ map (or the $s_s(\theta_h)$ and $r_s(\phi_h)$ curves if a decomposition into 1D factors has been performed). Warping this maintains the shape of the highlight while making it narrower or wider, or varying the amount of anisotropy.
c. One drawback of measured data is that it contains noise that may be difficult to remove when using previous non-parametric representations. Although our curve-based model is faithful to the measured data, we can remove noise by smoothing the 1D curves. In the figure, we demonstrate this by smoothing the $s_s(\theta_h)$ curve to remove some noise in the specular highlight.
d. The Fresnel reflection law predicts that specular highlights will become stronger and less saturated towards grazing incidence. We may introduce such an effect, or exaggerate it, by editing the $s_s(\theta_d, \lambda)$ curve.

Additional effects possible in our framework include changing retroreflective behavior (via the $s_d(\theta_d)$ curve), simulating the color shift of gonio-apparent paints (via the $s_s(\theta_d, \lambda)$ curve), and introducing nonphotorealistic behavior by quantizing the curves.

7 Comparison to Analytic Models

Our ACLS algorithm was designed to create meaningful decompositions into non-parametric shade trees that can be edited. We have compared to previous matrix factorizations algorithms that do not provide separations useful for *editing*—indeed, this was not their design goal. Previous methods for creating intuitive decompositions are those that fit a parametric BRDF model at each point, followed by clustering to give a user control of individual materials everywhere they appear on the surface [Lensch et al. 2003]. In this section, we compare to these methods, showing our higher qualitative and quantitative accuracy.

For these comparisons, we use the Ward BRDF model (as did Goldman et al. [2005]). We use the anisotropic version and augment it for retro-reflective materials with a back-scattering lobe consisting of a Gaussian function of θ_d. While other models like Lafortune can have an arbitrary number of terms, it is difficult to stably fit more than 2-3 lobes, and the form of this model does not represent complex materials and anisotropy well [Ngan et al. 2005].

SVBRDF Accuracy: Figure 16 shows a comparison of our algorithm and approximating the SVBRDF as unique parametric fits at each surface position, as well as the result of clustering these fits. Due to the inherent flexibility of our non-parametric representation, it introduces less error than clustering at any given term count. In fact, with only 2 materials, we are more accurate than fitting an independent Ward model to each position.

Because the RMS error is dominated by large values of the BRDF, arising from either shiny materials or measurements near grazing angles, it is important to also provide a qualitative comparison as done in Figure 17. Note that the Ward model is unable to match the irregular shaped anisotropic highlight in the wallpaper (the supplemental video contains additional comparisons). In the bottom of Figure 17, it also poorly approximates the shiny materials for positions outside their specular highlights. This is a common problem that occurs when the error of the analytic fit is dominated by the large values of the BRDF near specularities and grazing angles. Because we represent the BRDF as a sampled function, our shade tree is flexible enough to match the measured appearance of these datasets qualitatively better than the analytic model.

SVBRDF Material Separation and Editability: We also evaluate our approach in its ability to provide a final separation of materials that is suitable for editing. We present qualitative comparisons of the separation achieved using our techniques, and parametric clustering, for two particularly challenging cases.

The top rows of Figure 18 show the separation of the *Season's Greetings* dataset into four blending weights computed from clustering Ward fits (top row), and using our ACLS algorithm (second row). Note that clustering the parameters improperly assigns a cluster to the *combination* of the gold foil and white paper (top row, second image) in addition to incorrectly combining the gold and silver foils into a single cluster (top row, left). ACLS correctly associates the four component materials with unique terms, providing a final separation that can be edited (Figure 12).

The bottom pair of rows in Figure 18 show separation results for the *Wood+Tape* dataset. In this case, the parametric approach is unable to recover the transparent tape layer, as its reflectance is always observed in combination with the underlying wood. This results in a separation that incorrectly assigns the same cluster to regions of the wood grain *and* the tape (second to bottom, third column). On the other hand, ACLS (bottom row) automatically separates the

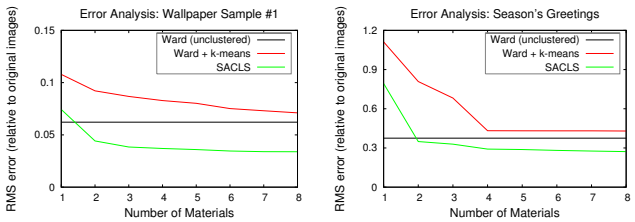
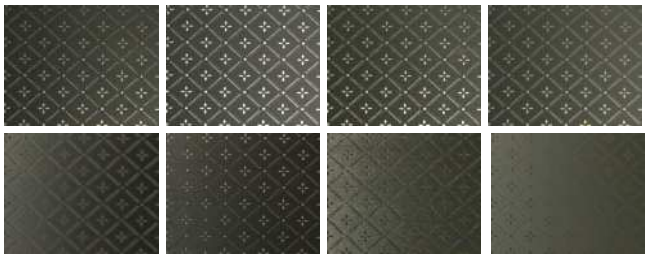


Figure 16: *Quantitative comparison of representing a measured SVBRDF using the Ward BRDF and k-means clustering vs. our method for increasing number of clusters/terms. For reference, Figure 17 shows a visual comparison of the reconstructions of these methods for different numbers of terms.*

Visual Comparison: Wallpaper #1



Ward Unclustered RMS=0.062	Ward 2 Clusters RMS=0.087	Original Image	SACLS 2 Terms RMS=0.038

Visual Comparison: Season's Greetings

Ward Unclustered RMS=0.375	Ward 4 Clusters RMS=0.432	Original Image	SACLS 4 Terms RMS=0.291

Figure 17: *Visual comparison between unclustered Ward model (fit independently to each spatial location), clustered Ward fits, original images, and our method. We also list the RMS values shown in Figure 16.*

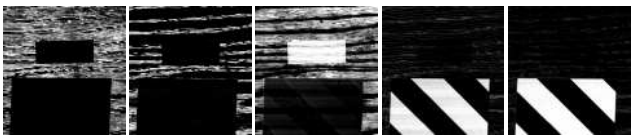
SVBRDF into two distinct types of wood grain, a separate layer for the semi-transparent tape, and separate terms for the two colors of the retroreflective bicycle tape. This produces a shade tree with components appropriate for rendering and editing (Figure 13).

BRDF Accuracy: We compare the accuracy of our decomposition of the BRDF into 2- and 1-D factors with fitting the anisotropic Ward BRDF model to the original measurements. In Figure 19, we show both numerical and qualitative analysis of the error in using these techniques to represent (top) retroreflective bicycle tape and (bottom) green anisotropic wallpaper. Recall that the original measurements are rasterized into a uniformly spaced table of values organized into a matrix. This introduces error into the approximation, which is quantified between the second and third columns in Figure 19. Further decomposing this tabular BRDF into 2D factors and 1D curves introduces additional error, as shown in the two rightmost columns. Along the leftmost column, we show qualitative comparisons and error numbers for using the Ward BRDF model to approximate these original measurements. The fixed form of the parametric model leads to higher qualitative and quantitative error for some light sources and views. In particular, the analytic model overestimates the magnitude of the back-scattering lobe near $\theta_d = 0$. Moreover, the precise shape of the anisotropic highlights (for wallpaper) is not well approximated by an elliptical Gaussian.

Separation using *k*-means on Ward Parameters

Separation using ACLS

Separation using *k*-means on Ward Parameters



Separation using ACLS

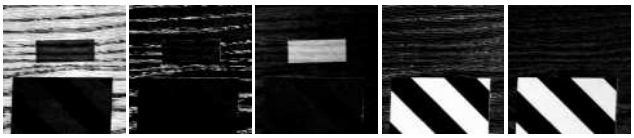


Figure 18: *Visual comparison of the separation achieved by applying k-means clustering to the fits of a Ward BRDF and that computed by the ACLS algorithm for two different datasets. We computed four terms (resp. clusters) for the (top two rows) "Season's Greetings" dataset and (bottom two rows) the "Wood+Tape" dataset. For the ACLS algorithm, we weighted the sparsity and L_1 norm constraints with $\lambda = 100.0$ and $\mu = 10.0$ for Season's Greetings, and $\lambda = 100.0$ and $\mu = 800.0$ for Wood+Tape.*

8 Limitations

The approach in this paper is designed for a variety of real-world spatially-varying materials. An important assumption, however, is that BRDFs are blended linearly over the surface, as in most real materials. It is theoretically possible for the 6D SVBRDF to vary smoothly, but not be easily expressible as a *linear* combination of basis materials or 4D BRDFs. In these cases, alternative representations may be more compact but not editable, since this has not been addressed by previous techniques. Another limitation on this work is that we must build a regularly-sampled data matrix before applying our factorization. By contrast, methods such as nonlinear parameter fitting, homomorphic factorization, or radial basis function interpolation operate directly with scattered input data. In practice, our use of confidence weighting and subsampled reconstruction minimizes the resampling error and additional computational time associated with our use of regularly-sampled matrices.

9 Conclusions and Future Work

We have introduced a non-parametric *Inverse Shade Tree* framework for representing and editing measured spatially- and directionally-dependent surface appearance. The representation is more accurate than parametric models, more intuitive than other non-parametric methods, and well-suited for interactive rendering and editing.

As future work, we would like to investigate algorithms that automatically infer the structure of the tree according to the data, including automatic selection of the number of terms to use at each decomposition. In addition, we may simultaneously decompose the same dataset into multiple trees, any of which may be edited depending on the desired change. Another direction for future work is to incorporate additional aspects of reflectance variation such as displacement maps, sub-surface scattering, or fine geometric detail typically represented as Bidirectional Texture Functions (BTFs).

Another avenue of future work is related to the ACLS techniques we have proposed. Their flexibility and provable local convergence make them ideal candidates for a broad range of dimensionality

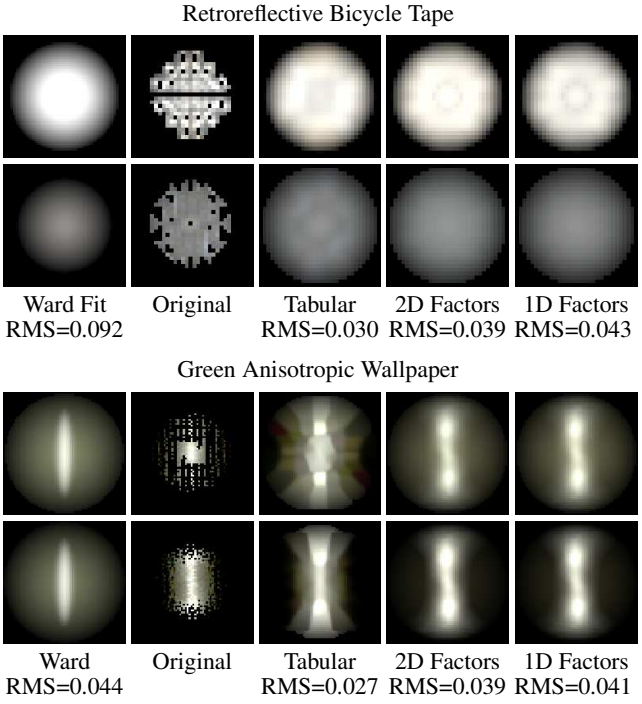


Figure 19: *Analysis of the error introduced by several levels of our tree-structured decomposition for BRDFs, and comparison with Ward fits. For each material, the top and bottom rows show parabolic maps of ω_h distributions at ($\theta_d = 15°, \phi_d = 90°$) and ($\theta_d = 45°$, $\phi_d = 90°$) respectively.*

reduction applications in data mining and other machine learning contexts. We wish to evaluate the efficiency and noise-tolerance properties of ACLS, and investigate the impact of various types of additional linear constraints.

10 Acknowledgements

This research was funded by the Sloan Foundation and National Science Foundation grants CCF-0347427, CCF-0305322 and CCF-0446916. The idea of performing a hierarchical decomposition of measured reflectance data originated in conversations with Marc Levoy. The authors wish to thank Steve Marschner for letting us use the spherical gantry at Cornell along with the SIGGRAPH and TIGGRAPH reviewers for their helpful and constructive feedback.

References

ASHIKHMIN, M., PREMOŽE, S., AND SHIRLEY, P. 2000. A microfacet-based BRDF generator. In *Proceedings of ACM SIGGRAPH 2000*.

CHEN, W.-C., BOUGUET, J.-Y., CHU, M. H., AND GRZESZCZUK, R. 2002. Light field mapping: efficient representation and hardware rendering of surface light fields. In *ACM Transactions on Graphics (SIGGRAPH 2002)*.

COOK, R. L. 1984. Shade trees. In *Computer Graphics (Proceedings of ACM SIGGRAPH 1984)*, 223–231.

DANA, K., VAN GINNEKEN, B., NAYAR, S., AND KOENDERINK, J. 1999. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics 18*, 1, 1–34.

FURUKAWA, R., KAWASAKI, H., IKEUCHI, K., AND SAKAUCHI, M. 2002. Appearance based object modeling using texture database: acquisition, compression and rendering. In *Eurographics Workshop on Rendering*, 257–266.

GARDNER, A., TCHOU, C., HAWKINS, T., AND DEBEVEC, P. 2003. Linear light source reflectometry. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3, 749–758.

GILL, P., MURRAY, W., SAUNDERS, M., AND WRIGHT, M. 1984. Procedures for optimization problems with a mixture of bounds and general linear constraints. In *ACM Transactions on Mathematical Software*.

GOLDMAN, D. B., CURLESS, B., HERTZMANN, A., AND SEITZ, S. M. 2005. Shape and spatially-varying BRDFs from photometric stereo. In *IEEE International Conference on Computer Vision*.

GORTLER, S., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. 1996. The lumigraph. In *Proceedings of ACM SIGGRAPH 1996*.

HAN, J. Y., AND PERLIN, K. 2003. Measuring bidirectional texture reflectance with a kaleidoscope. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3, 741–748.

HARTIGAN, J. A., AND WONG, M. A. 1979. A k-means clustering algorithm. *Applied Statistics 28*, 100–108.

HEIDRICH, W., AND SEIDEL, H.-P. 1999. Realistic, hardware-accelerated shading and lighting. In *Proceedings of ACM SIGGRAPH 1999*.

HOFMANN, T. 1999. Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence*.

HOYER, P. O. 2002. Non-negative sparse coding. In *IEEE Workshop on Neural Networks for Signal Processing*, 557–565.

JAROSZKIEWICZ, R., AND MCCOOL, M. D. 2003. Fast extraction of BRDFs and material maps from images. In *Graphcs Interface*.

KAUTZ, J., AND MCCOOL, M. 1999. Interactive rendering with arbitrary BRDFs using separable approximations. In *Eurographics Workshop on Rendering*, 247–260.

LAFORTUNE, E. P. F., FOO, S.-C., TORRANCE, K. E., AND GREENBERG, D. P. 1997. Non-linear approximation of reflectance functions. In *Proceedings of ACM SIGGRAPH 1997*, 117–126.

LAWRENCE, J., RUSINKIEWICZ, S., AND RAMAMOORTHI, R. 2004. Efficient BRDF importance sampling using a factored representation. *ACM Transactions on Graphics (SIGGRAPH 2004) 23*, 3.

LEE, D., AND SEUNG, H. S. 2000. Algorithms for non-negative matrix factorization. In *Proceedings of Neural Information Processing Systems*, 556–562.

LENSCH, H. P. A., KAUTZ, J., GOESELE, M., HEIDRICH, W., AND SEIDEL, H.-P. 2003. Image-based reconstruction of spatial appearance and geometric detail. *ACM Transactions on Graphics 22*, 2.

LEUNG, T., AND MALIK, J. 2001. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision 43*, 1, 29–44.

MARSCHNER, S., WESTIN, S., LAFORTUNE, E., TORRANCE, K., AND GREENBERG, D. 1999. Image-Based BRDF measurement including human skin. In *Eurographics Workshop on Rendering*, 139–152.

MARSCHNER, S. R., WESTIN, S. H., ARBREE, A., AND MOON, J. T. 2005. Measuring and modeling the appearance of finished wood. *ACM Transactions on Graphics (SIGGRAPH 2005) 24*, 3, 727–734.

MATUSIK, W., PFISTER, H., BRAND, M., AND MCMILLAN, L. 2003. A data-driven reflectance model. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3, 759–769.

MCALLISTER, D. 2002. *A Generalized Surface Appearance Representation for Computer Graphics*. PhD thesis, UNC.

MCCOOL, M. D., ANG, J., AND AHMAD, A. 2001. Homomorphic factorization of BRDFs for high-performance rendering. In *Proceedings of ACM SIGGRAPH 2001*, 185–194.

NAG, 2005. Numerical Algorithms Group C Library.

NGAN, A., DURAND, F., AND MATUSIK, W. 2005. Experimental analysis of BRDF models. In *Proceedings of the Eurographics Symposium on Rendering*, 117–226.

OLSHAUSEN, B. A., AND FIELD, D. J. 2002. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature 381*, 607–609.

PEERS, P., VOM BERGE, K., MATUSIK, W., RAMAMOORTHI, R., LAWRENCE, J., RUSINKIEWICZ, S., AND DUTRÉ, P. 2006. A compact factored representation of heterogeneous subsurface scattering. *ACM Transactions on Graphics (SIGGRAPH 2006) 25*, 3.

RUSINKIEWICZ, S. 1998. A new change of variables for efficient BRDF representation. In *Eurographics Workshop on Rendering*, 11–22.

TSUMURA, N., OJIMA, N., SATO, K., SHIRAISHI, M., SHIMIZU, H., NABESHIMA, H., AKAZAKI, S., HORI, K., AND MIYAKE, Y. 2003. Image-based skin color and texture analysis/synthesis by extracting hemoglobin and melanin information in the skin. *ACM Transactions on Graphics (SIGGRAPH 2003) 22*, 3, 770–779.

VASILESCU, M. A., AND TERZOPOULOS, D. 2004. TensorTextures: Multilinear image-based rendering. *ACM Transactions on Graphics (SIGGRAPH 2004) 23*, 3.

WARD, G. J. 1992. Measuring and modeling anisotropic reflection. In *Computer Graphics (Proceedings of ACM SIGGRAPH 1992)*, 265–272.