Intrinsic Scene Properties from a Single RGB-D Image

Jonathan T. Barron, *Member, IEEE,* and Jitendra Malik, *Fellow, IEEE*

Abstract—In this paper, we present a technique for recovering a model of shape, illumination, reflectance, and shading from a single image taken from an RGB-D sensor. To do this, we extend the SIRFS ("shape, illumination and reflectance from shading") model, which recovers intrinsic scene properties from a single image [1]. Though SIRFS works well on neatly segmented images of objects, it performs poorly on images of natural scenes which often contain occlusion and spatially-varying illumination. We therefore present Scene-SIRFS, a generalization of SIRFS in which we model a scene using a mixture of shapes and a mixture of illuminations, where those mixture components are embedded in a "soft" segmentation-like representation of the input image. We use the noisy depth maps provided by RGB-D sensors (such as the Microsoft Kinect) to guide and improve shape estimation. Our model takes as input a single RGB-D image and produces as output an improved depth map, a set of surface normals, a reflectance image, a shading image, and a spatially varying model of illumination. The output of our model can be used for graphics applications such as relighting and retargeting, or for more broad applications (recognition, segmentation) involving RGB-D images.

Index Terms—computer vision, machine learning, intrinsic images, shape from shading, shape estimation, illumination estimation, segmentation, normalized cuts, depth sensing

1 Introduction

The broad fields of computer vision and computer graphics trace their heritage back to the two dual problems of inferring a model of the physical world from an image, and rendering an image from a model of the physical world. The graphics problem of photorealistic rendering, though challenging, has seen a remarkable amount of progress in the past half-century to the point where we now have very accurate and efficient models, well reviewed in [2]. The inverse of this problem, however, has seen surprisingly little success in comparison. In this paper, we will present a model for taking an RGB-D image of a natural scene (a standard RGB image augmented with the output of a depth sensor, such as the Kinect) and inverting the formation of that image to estimate the physical world that lies beneath the observed image. Doing this requires marrying rendering machinery from the computer graphics literature with inference and optimization techniques from the computer vision community. The rich physical model of the world that we recover has a variety of uses, from simple graphics applications such as relighting or rotating the camera, to providing other vision systems with a deeper understanding of the physical world than is immediately evident in simple pixel intensities.

This core problem of inferring the properties of a scene (shape, surface normals, illumination, reflectance, etc) that together reproduce a single observed image was first posed as the "intrinsic images" problem by Barrow and Tenenbaum [3]. But over time, this term has become shorthand for the task of decomposing an image into a "shading" image and a "reflectance" image, best exemplified by the Retinex algorithm [4], [5]. In recent years much progress has been made on this subset of the intrinsic images problem [6], [7], [8], but the most dramatic progress has come from the SIRFS ("shape, illumination, and reflectance from shading") model [1], [9], [10], [11], which recovers shape and illumination in addition to shading and reflectance, and outperforms standard "intrinsic image" approaches. SIRFS is severely limited by its assumption that input images are segmented images of single objects, illuminated under a single global model of illumination. Natural images, in contrast, contain many shapes which may occlude or support one another, as well as complicated, spatially-varying illumination in the form of shadows, attenuation, and interreflection.

In this paper, we address the problem of inferring a *mixture* of shapes and a *mixture* of illuminations (and implicitly, a shading image and a reflectance image) which explain a natural *scene*. Initially, this problem may seem trivial: why not use segmentation techniques to decompose an image into its constituent objects or illuminations, and then apply SIRFS to each segment? But this is a classic "chicken-or-the-egg" problem, as we cannot reliably segment an image into its constituent shapes and illuminations without first inferring shape and illumination, and vice versa.

• *J.T. Barron and J. Malik are with the Department of Electrical Engineering and Computer Science, University of California at Berkeley, Berkeley, CA 94720.*
E-mail: barron,malik@eecs.berkeley.edu

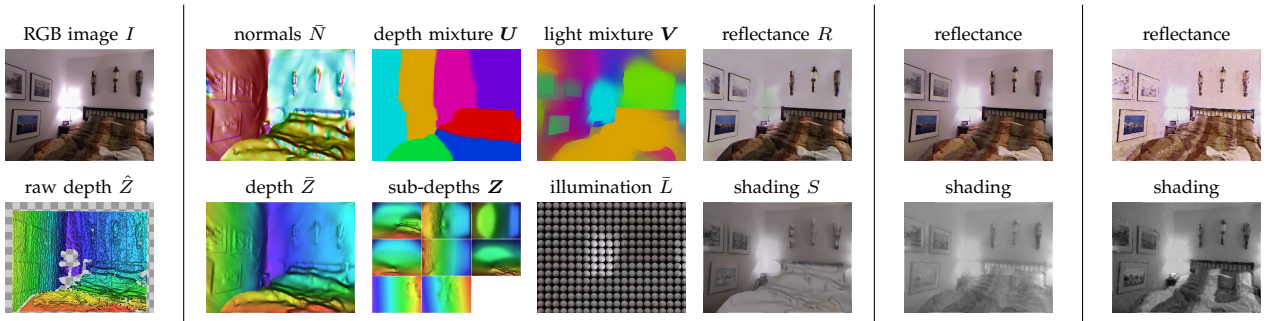


Fig. 1. In 1(a) we have the input to our model: an RGB image and a Kinect depth map from the NYU Depth Dataset [17]. In 1(b) we have the output of our model. Depth maps are visualized with hue corresponding to depth and luminance corresponding to slant, and surface normals are visualized with hue corresponding to orientation, and saturation and luminance corresponding to slant. Mixtures are visualized with hue corresponding to component in the mixture, and intensity corresponding to the probability assigned to that component. Illumination is visualized by rendering a coarse grid of spheres under the spatially-varying illumination. In 1(c) and 1(d) we show the reflectance and shading images produced by two intrinsic image techniques, where 1(d) is the state-of-the-art. See Figures 6 and 7 for more examples.

Additionally, regions of a scene viewed in isolation are often ambiguous, which suggests that information must be shared between regions. We must therefore unify the problems of reconstruction (inferring intrinsic scene properties) and reorganization (grouping an image into meaningful regions), by jointly optimizing over a mixture of shapes, a mixture of illuminations, and the corresponding embedding of each mixture component in the image.

For our technique to work, our shape and light mixtures must respect the structure of the image. We therefore embed our mixtures in the normalized Laplacian of the image, building on normalized cuts [12], as well as Laplacian embeddings [13] and spectral graph theory [14]. This is motivated by the observation that variation in shape and illumination tends to produce gradients and contours in the image, and so our mixtures of shapes and illuminations should be embedded in a space that respects such image variation.

Using shading cues to infer shape, as we are attempting, is understood to work poorly for recovering low-frequency (coarse) shape information [15]. Thankfully, depth data from sensors such as the Kinect [16] is becoming increasing commonplace, and is complementary to shading: binocular disparity (the principle by which the Kinect computes depth) is accurate at coarse scales and inaccurate at fine scales. We will therefore assume the input to our model is an RGB-D image, where "D" is the depth map produced by a sensor such as the Kinect. This makes our problem easier, but in no way trivial — depth maps from sensors such as the Kinect are noisy and incomplete for many reasons: occlusion of the structured light, dark objects, sensor noise, alignment errors, quantization, and the inherent physical limitations of binocular disparity. Attempts to use raw depth maps from the Kinect for photometric applications therefore often fail badly. See Figures 6-9 for demonstrations of how noisy these depth maps are compared to the depth maps that our model produces.

In Figures 6 and 7 we show the output of our model on an RGB-D image from the NYU Depth Dataset [17]. Our model's depth map is a clear improvement over the raw sensor depth map (missing regions have been filled in, noise has been removed, detail has been added), our output shading and reflectance images look better than those of the best "intrinsic image" algorithms, our shape mixture has separated the bed in the foreground from the walls in the background, and our recovered mixture of illuminations captures the complicated illumination in the scene produced by the lamp. Even "mistakes" produced by our model are compelling: our model has attempted to reconstruct the shape of the contents of the photos on the wall, and has modeled these contents with a different illumination environment than the rest of the scene, similarly to how a human might perceive an image within an image.

Some past work has addressed similar problems to our own. Forsyth [18] used a spatially-varying model of illumination to address complicated illumination and interreflection, but did not address reflectance or scene-like shape occlusion. Yu *et al.* [19] and Karsch *et al.* [20] have attempted to recover the reflectance and illumination of a scene, but assume known geometry and multiple images, or a user annotation of geometry and illumination, respectively. Hoeim *et al.* [21] and Saxena *et al.* [22] present algorithms for determining the "spatial layout" of a scene, but these shape estimates are coarse, and these models do not recover illumination, reflectance, or shading. Lee *et al.* [23]

produces shading and reflectance images given RGB-D data, but requires a video and a fused depth map, and does not produce an illumination model or a refined shape. Chen and Koltun [24] produce shading and reflectance maps from an RGB-D image, but also do not estimate illumination or a refined shape, nor do they require that shading be a rendering of some shape and illumination.

This paper is as follows: in Section 2 we review SIRFS, in Section 3 we formalize Scene-SIRFS, and in Section 4 we introduce the embedding used by our shape and illumination mixtures. In Sections 5 and 6 we present our priors on shape and illumination (our shape prior incorporates the input depth map from the Kinect), and in Section 7 we show how we optimize the resulting inference problem. In Sections 8.1 and 8.3 we present experiments on pseudo-synthetic and real RGB-D data.

2 SIRFS

Our model builds upon the "shape, illumination, and reflectance from shading" (SIRFS) model [1], [9], [10], [11], which is a framework for recovering intrinsic scene properties from a single image of a segmented object. SIRFS can be thought of as an extension of classic shape-from-shading models [25] in which reflectance and illumination are recovered in addition to shape. Formally, the SIRFS problem formulation is:

$$\begin{aligned} \underset{R,Z,L}{\text{minimize}} \quad & g(R) + f(Z) + h(L) \\ \text{subject to} \quad & I = R + S(Z, L) \end{aligned} \tag{1}$$

Where R is a log-reflectance image, Z is a depth-map, L is a spherical-harmonic model of illumination [26], and $S(Z, L)$ is a "rendering engine" which produces a log-shading image given Z and L. Z and R are "images" with the same dimensions as I, and L is a 27-dimensional vector. $g(R)$, $f(Z)$, and $h(L)$ are cost functions for reflectance, shape, and illumination respectively, and can be viewed (roughly) as negative log-likelihoods. We constrain the rendering of our scene properties $R + S(Z, L)$ to be equal to the observed log-intensity image I. Solving this problem corresponds to searching for the least costly (or most likely) explanation $\{Z, R, L\}$ for image I.

Though this model outperforms other intrinsic image techniques, it is critically limited by the requirement that the image contain a single, segmented object. This limitation is due to several factors: 1) SIRFS considers shapes to be composed of a single smooth depth-map Z, and therefore cannot model depth discontinuities, occlusion, etc. 2) SIRFS has a single global model of illumination L, but natural scenes contain spatially-varying illumination due to attenuation, interreflection, cast and attached shadows, etc. 3) SIRFS uses the occluding contour of the object [27], information which is not explicit in natural scenes.

To address these shortcomings, we will use a mixture of shapes and illuminations embedded in a soft segmentation of the scene, and we will use a Kinect depth-map as a surrogate for the information provided by the missing contour cue and to address the inherent limitation of shading for low-frequency shape reconstruction [11], [15]. We call our resulting technique "Scene-SIRFS", as it extends SIRFS from objects to scenes.

3 SCENE-SIRFS

The problem formulation of Scene-SIRFS is:

$$\begin{aligned} \underset{R,\boldsymbol{Z},\boldsymbol{\psi},\boldsymbol{L},\boldsymbol{\omega}}{\text{minimize}} \quad & g(R) + \sum_{n=1}^{|\boldsymbol{Z}|} f'(Z^n, U^n) + h'\left(\sum_{m=1}^{|\boldsymbol{L}|} V^m L^m\right) \\ \text{subject to} \quad & I = R + S'(\boldsymbol{Z}, \boldsymbol{U}, \boldsymbol{L}, \boldsymbol{V}) \\ & U^n = \frac{\exp(B\psi^n)}{\sum_{n'} \exp(B\psi^{n'})}, \forall_n \\ & V^m = \frac{\exp(B\omega^m)}{\sum_{m'} \exp(B\omega^{m'})}, \forall_m \end{aligned} \tag{2}$$

Where $\boldsymbol{Z} = \{Z^n\}$, $\boldsymbol{U} = \{U^n\}$, $\boldsymbol{L} = \{L^m\}$, and $\boldsymbol{V} = \{V^m\}$. This is similar to Equation 1, except that we have sets of shapes and lights instead of a single shape and light, and we have introduced $\boldsymbol{U}$ and $\boldsymbol{V}$, two sets of "images" that define distributions over shapes and lights, respectively. We can think of $\boldsymbol{U}$ as a "visibility" map or a soft relaxation of a $2.5D$ shape representation: if $U^n_{i,j} = 1$, then $Z^n_{i,j}$ is visible at pixel (i, j). Similarly, $\boldsymbol{V}$ is the "ownership" of each illumination in $\boldsymbol{L}$, such that if $V^m_{i,j} = 1$ then pixel (i, j) is entirely illuminated by L^m. Our prior on shape is now a sum of priors over individual depth maps, where each Z^n in $\boldsymbol{Z}$ is regularized independently (see Section 5). In contrast, our prior on illumination is over the expected illumination of the entire scene, the per-pixel weighted combination of each illumination (see Section 6). Our shape and light mixture probabilities $\boldsymbol{U}$ and $\boldsymbol{V}$ are "images" (where each image corresponds to one mixture component) parametrized by the matrices $\boldsymbol{\psi}$ and $\boldsymbol{\omega}$, respectively, where each column (ψ^n or ω^m) is a 17-dimensional vector parametrizing the "ownership" of that shape or light mixture in the scene. The probabilities $\boldsymbol{U}$ and $\boldsymbol{V}$ are the product of each weight matrix with B (the eigenvectors of the normalized Laplacian of the RGB image, explained in later) passed through a softmax function[1]. We use 8 shapes and illuminations in our mixtures for all experiments ($|\boldsymbol{L}| = |\boldsymbol{Z}| = 8$) though this is arbitrary. See Figures 5, 6, and 7 for visualizations of these mixtures.

For the purpose of optimization, we need to define the normal field of this mixture of shapes $N'(\boldsymbol{Z}, \boldsymbol{U})$. We cannot use the surface normals of the expected

1. in a slight abuse of notation, $\boldsymbol{U}$ and $\boldsymbol{V}$ are simultaneously treated as sets of images and as matrices whose columns are vectorized images.

depth map $N(\sum Z^n U^n)$ as this cannot model depth-discontinuities. We also cannot use the expected surface normals of the mixture of shapes $\sum U^n N(Z^n)$ as this normal field may have vectors of non-unit length. We will therefore linearize each Z^n into a set of partial derivatives in x and y, take the expectation of those with respect to $\boldsymbol{U}$, and then construct a normal field from those expected partial derivatives. This gives us a proper normal field where each Z^n's influence at pixel (i,j) is proportional to $U^n_{i,j}$. Formally:

$$N'(\boldsymbol{Z},\boldsymbol{U}) = \left\{ \frac{D^x}{D^m}, \frac{D^y}{D^m}, \frac{1}{D^m} \right\}$$

$$D^x = \sum_{n=1}^{|\boldsymbol{Z}|} U^n (Z^n * h^x), \quad D^y = \sum_{n=1}^{|\boldsymbol{Z}|} U^n (Z^n * h^y)$$

$$D^m = \sqrt{1 + (D^x)^2 + (D^y)^2}$$

$$h^x = \frac{1}{8}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad h^y = \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{3}$$

Let $S'(\cdot)$ be our rendering engine for our mixtures, which computes the normal field of the mixture of shapes and renders it such that the spherical harmonic illumination at pixel (i,j) is a linear combination of all L^m, weighted by $V^m_{i,j}$:

$$S'(\boldsymbol{Z},\boldsymbol{U},\boldsymbol{L},\boldsymbol{V}) = S\left(N'(\boldsymbol{Z},\boldsymbol{U}), \sum_{m=1}^{|\boldsymbol{L}|} V^m L^m \right) \tag{4}$$

Where $S(\cdot)$ is the rendering engine in SIRFS.

Though the spatially varying illumination parametrized by $\{\boldsymbol{L},\boldsymbol{V}\}$ is capable of explaining away shadows, specularities, and interreflections, no attempt has been made to ensure that the illumination is globally consistent. Though this may seem unsettling, the human visual system has similar properties: people tend not to notice inconsistent shadows or impossible illumination [29]. It would be very satisfying to instead construct a more sophisticated model of illumination which captures the various global phenomena that real-world illumination displays, but doing so requires both a very efficient "forward" mechanism for rendering an image from a model of the world, and an efficient (and easily optimized) "backward" mechanism for backpropagating the gradient of a prior onto that model. Unfortunately, efficient techniques for computing that "forward" half of this problem (photorealistic rendering) are still an active research area [2], and only very simple models (such as the linear model used here) appear to easily lend themselves to efficient backpropagation.

4 Mixture Embedding

Using a mixture of shapes and illuminations is necessary to model depth discontinuities and spatially varying illumination, both of which tend to produce variation in the image in the form of contours, intensity variation, texture gradients, etc. It therefore follows that we should embed the shape and light mixtures in some space where the "ownership" of each mixture adheres to the segmentation of the scene. This simplifies inference, as we restrict our attention to only mixtures of the shapes and lights that are supported by evidence in the image. This is similar to the motivation for the use of superpixels as a substrate for inference in CRF-type models, though our experience suggests that superpixels are a poor embedding for this task, as they are too "hard". We will instead embed each mixture component in a more "soft" embedding: the eigenvectors of the normalized Laplacian of a graph corresponding to the input RGB image, a la normalized cuts [12].

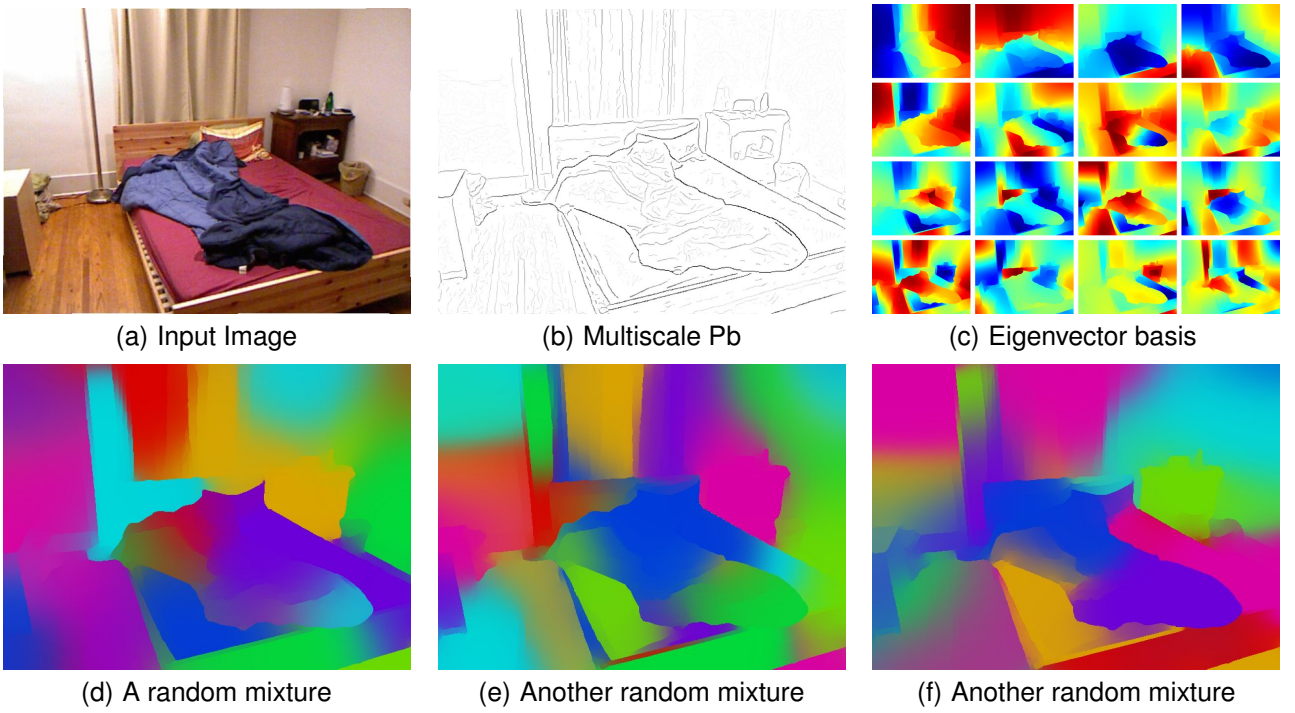
The motivation behind normalized cuts is that we wish to partition an image into segments by making "cuts" in the graph implied by the image. Let us define our image graph as $G = (V, E)$, where V are the nodes in the graph (which in our case are pixels) and E are the edges in the graph. We wish to split our image into two partitions, A and B, to maximize the normalized cut disassociation measure:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \tag{5}$$

Intuitively, this measure means that we want to minimizing the disassociation between groups while also maximizing the association within each group. This formulation as stated is intractable, but can be tractably approximated with the following generalized eigenvalue problem:

$$y = \arg\min_y \frac{y^{\mathrm{T}}(D-W)y}{y^{\mathrm{T}}Dy} \tag{6}$$

Where W is the affinity matrix used in our graph (where $W_{i,j}$ a measure of similarity between pixels i and j) and $D = W\mathbf{1}$. In the conventional normalized cuts formulation, image segments are produced by constructing a graph of an image, solving this generalized eigenvalue to get the eigenvector corresponding to the second-smallest eigenvalue (the smallest eigenvector is all 1's and so should be ignored), splitting the image graph along the dimension of the recovered eigenvector, and then repeating this process on the two subgraphs recursively. However, later work has found it effective to instead find the k smallest eigenvectors of the image graph, and use those k vectors as a basis for a different clustering algorithm such as K-means [30]. We will build upon this idea, and use these normalized-cut eigenvectors as a basis for embedding our shape and light mixtures into the graph of the image. This "embedding" approach avoids issues such as the artifacts that usually result from "hard" segment assignments, while still



(a) Input Image (b) Multiscale Pb (c) Eigenvector basis

(d) A random mixture (e) Another random mixture (f) Another random mixture

Fig. 2. A visualization of the embedding used in our shape and light mixtures. In 2(a), we have an input image. In 2(b) we have the output of multiscale Pb on the input image, and in 2(c) we have the 16 smallest eigenvectors (ignoring the eigenvector that is all 1's) of mPb using the intervening contour cue [28]. Each shape's and light's "ownership" of the image is parametrized by a 17-dimensional vector, which is projected onto the eigenvector basis and passed through a softmax function to yield the probability of each pixel belonging to each mixture component. 2(d), 2(e), and 2(f) are visualizations of three random mixtures with 8 components (such as $\boldsymbol{U}$ or $\boldsymbol{V}$) where the weight vectors (ψ or ω) are generated randomly (sampled from a Gaussian).

giving us a low-dimensional space with which we can parametrize our problem.

We construct our embedding as follows: given an image, first we compute the multiscale Pb of that image [28], [31]. We then form an affinity matrix W from mPb using the intervening contour cue [32], and compute the 17 eigenvectors $\{\mathbf{u}_i\}$ corresponding to the smallest eigenvalues. Computing these eigenvectors is generally considered to be computationally fairly expensive, but it can be made much more efficient ($\approx 20\times$ faster) using the "downsampled" normalized cuts variant of the generalized eigenvector problem [33]. The first eigenvector we recover is all 1's, and is normally ignored, but our model actually benefits from a "bias" term in this eigenvector parametrization, and so we do not discard this eigenvector. For eigenvectors 2 through 17, we subtract the mean from each $\mathbf{u}_i$ and divide each by its standard deviation, and then concatenate these normalized eigenvectors into a matrix B, with one column per-pixel. B is our embedding space, in that each mixture component is defined by a 17-dimensional vector, whose inner product with B defines how dominant that mixture component is at every pixel in the input image. A similar embedding is used in [34], for the purpose of combining recognition and segmentation. See Figure 2 for a visualization of this embedding.

Working in this embedding space has a number of advantages: a mixture component that spans every pixel in the image (of which there are hundreds of thousands) is compactly described by only 17 numbers, which is computationally very convenient, makes optimization very easy, and obviates the need for placing "smoothness" priors on our mixture embeddings. Our embedding space respects the structure of the image, forcing our shape and light mixtures to naturally follow the image and to never be discontinuous at any point in the image where the RGB image does not itself contain a discontinuity. This model choice does however have the obvious downside that our model is incapable of expressing certain solutions, but this makes sense: the vast majority of possible soft-segmentations of an image are poor choices, as images are extremely high-dimensional structured spaces.

It may seem unusual that we construct our embedding using only RGB information, instead of using the complete RGB-D image. We do this because the depth images are often mis-aligned and noisy enough that it is challenging to construct a single accurate contour signal from both sources of information. Using only the image to create an embedding circumvents the

noise in the depth map and forces the reconstructed shape to be aligned with the image.

Our prior on reflectance $g(\cdot)$ is exactly the same as in SIRFS. In Sections 5 and 6 we will define $f'(\cdot)$ and $h'(\cdot)$, our priors on our shape and illumination mixtures, respectively.

5 Shape Priors and RGB-D Images

Our prior on shape is a modification of that of SIRFS. We use a linear combination of the smoothness and isotropy terms $f_k(Z)$ and $f_i(Z)$ from [1] and we replace the $f_o(\cdot)$ term to incorporate knowledge from the raw sensor depth map produced by the Kinect $\hat{Z}$:

$$f'(Z,U) = \lambda_k f_k(Z) + \lambda_f f_i(Z) + \lambda_o f_o(Z,U) \tag{7}$$

Where $f_k(Z)$ minimizes the local variation of mean curvature of Z, encouraging Z to be smooth, and $f_i(Z)$ minimizes the slant of Z, encouraging Z to be fronto-parallel. We introduce $f_o(Z,U)$, which encourages Z to be similar to the raw sensor depth map if Z is thought to be "visible" according to U. Crucially, we apply this prior to each individual depth map in our mixture rather than to some average depth map. This encourages the scene's constituent depth maps to be smooth while allowing the expected depth map implied by the mixture to vary abruptly, thereby allowing us to model depth discontinuities and occlusion.

We use version 2 of the NYU Depth Dataset [17], which consists of RGB images and aligned Kinect depth maps. Because Kinect depth maps often have gaps, the dataset also provides inpainted depth maps. We will use the raw depth maps rather than the inpainted ones, as our algorithm will implicitly denoise and inpaint depth during inference. In addition to gaps, Kinect depth maps have different kinds of noise. First: the depth and RGB images are often not well-aligned — not enough to matter for most recognition tasks, but enough to affect photometric or reconstruction tasks. Second: the disparity recovered by the Kinect is often noisy, presumably due to sensor noise or errors in the Kinect's stereo algorithm. Third: the disparity is quantized, which leads to step-like artifacts in depth.

We must construct a loss function to encourage our recovered depth Z to resemble the raw sensor depth $\hat{Z}$. First, let us approximate the upper bound of the error introduced by quantizing the disparity corresponding to $\hat{Z}$:

$$Z_{i,j}^{err} = (1.4233 \times 10^{-5})\hat{Z}_{i,j}^2 + 2 \tag{8}$$

where $\hat{Z}$ and Z^{err} are in centimeters. The first term is derived from the baseline of the Kinect, and the second term is additional ad-hoc slack. We assume that if the difference between $Z_{i,j}$ and $\hat{Z}_{i,j}$ at pixel (i,j) is less than $Z_{i,j}^{err}$, then that difference is due to quantization and therefore should be ignored. Errors greater than $Z_{i,j}^{err}$ will be robustly penalized, as they probably are due to sensor noise or alignment errors. Our loss function is:

$$f_o(Z,U) = \sum_{i,j} U_{i,j} \max\left(0, \left|Z_{i,j} - \hat{Z}_{i,j}\right| - Z_{i,j}^{err}\right)^{\alpha_o} \tag{9}$$

Minimizing this is equivalent to assuming noise is uniformly distributed over a region of size $2Z_i^{err}$ and is hyper-Laplacian outside of that range. The loss is proportional to $U_{i,j}$, which means that $Z_{i,j}$ need only resemble $\hat{Z}_{i,j}$ if our model believes that this depth map is in the foreground at pixel (i,j). α_o controls the shape of the tail of the distribution, and is tuned with cross-validation on the training set (along with λ_k, λ_f, and λ_o), which sets $\alpha_o = 0.7$.

6 Priors on Illumination

Our prior on illumination is a simple extension of the illumination prior of [1] to a mixture model, in which we regularize the expectation of a set of illuminations instead of a single illumination. Given $\boldsymbol{L}$ (our set of spherical harmonic illuminations) and $\boldsymbol{V}$ (our set of "images" that define a per-pixel distribution over our illuminations), we can compute the expectation of this model at each pixel of the image:

$$\bar{L}_{i,j} = \sum_{m=1}^{|\boldsymbol{L}|} V_{i,j}^m L^m \tag{10}$$

Where $\bar{L}_{i,j}$ is a 27-dimensional vector describing the effective illumination at pixel (i,j) in the image. Our prior on illumination is the negative log-likelihood of a multivariate normal distribution, applied to each 27-dimensional "pixel" in $\bar{L}$:

$$h'(\bar{L}) = \lambda_L \sum_{i,j} (\bar{L}_{i,j} - \boldsymbol{\mu}_L)^{\mathrm{T}} \Sigma_L^{-1} (\bar{L}_{i,j} - \boldsymbol{\mu}_L) \tag{11}$$

Where $\boldsymbol{\mu}_L$ and Σ_L are the parameters of the Gaussian we learn on the training set, and λ_L is the multiplier on this prior (learned through cross-validation on the training set).

This prior was chosen for a number of reasons. Intuitively it makes sense to regularize the illumination at every pixel of the image, especially since the rest of our Scene-SIRFS formulation is also phrased in terms of per-pixel priors. Specifically, placing priors on the expectation of illumination is compatible with our other model choices, as this same expectation is used to render the shading image used by the model. It may seem a bit odd to use spherical harmonics for this purpose, as the primary merit of spherical harmonic illumination is that it has been demonstrated empirically and theoretically to be a sound model of *global* illumination, not spatially-varying local illumination [26]. That being said, spherical harmonics appear to work well empirically in this context, though simpler models (standard directional + ambient illumination)

could likely be used as well. Another reason why we chose spherical harmonics is to remain compatible with the illumination prior from basic SIRFS [1], which allows us to use the same learned prior from that model here. Again, it may seem strange to use priors on global illumination to constrain local illumination, but acquiring training data for spatially-varying local illumination is extremely challenging, and our model does not seem to suffer due to the use of an illumination model which may be more expressive or powerful (as global illumination is generally considered more difficult to model than local illumination) than is necessary.

7 Initialization & Optimization

Optimization of our model is similar to that of [1]. We absorb the $I = R + S(\cdot)$ constraint in Equation 2 by rewriting $g(R)$ as $g(I - S(\cdot))$, thereby eliminating R as a free parameter. In optimization we internally represent each depth map Z^n as a pyramid, and whiten each illumination L^m according to $\{\boldsymbol{\mu}_L, \Sigma_L\}$. We vectorize our pyramid-depths, whitened illuminations, and mixture weights $\{\boldsymbol{\psi}, \boldsymbol{\omega}\}$ into one state vector, and then minimize the loss in Equation 2 with respect to that state vector using L-BFGS.

This optimization problem is non-convex, and so it is sensitive to initialization. Because the scenes in the NYU dataset are mostly composed of planar surfaces, we will initialize each depth map Z^i in $\boldsymbol{Z}$ to a plane such that the scene is well-described by the set of planes. To do this, we fit a mixture of Gaussians to the (x, y, z) coordinates of each pixel in $\hat{Z}$ (in image coordinates) using EM with 50 random restarts. Once EM converges we have n multivariate Gaussians, each parametrized by a mean μ and a covariance matrix Σ. If a Gaussian does indeed describe a roughly-planar surface, then Σ will be elongated in two directions, and narrow in the third. This means that the Gaussian is well described by the plane satisfying $v^{\mathrm{T}}([x, y, z] - \mu) = 0$, where v is the eigenvector corresponding to the smallest eigenvalue of Σ. We initialize each surface in our mixture to its corresponding plane in our mixture of Gaussians, by solving for z at every pixel. See Figure 3 for a visualization.

This plane-fitting sometimes produces poor results on our synthetic dataset, because our synthetic scenes contain mostly fronto-parallel objects stacked on top of each other. Therefore, in our synthetic experiments we initialize the depth maps by doing K-means (with 50 random restarts) on just the z values in the scene, and then initializing each depth map to be a centroid, thereby constraining the initial depth-planes to be fronto-parallel.

8 Experiments

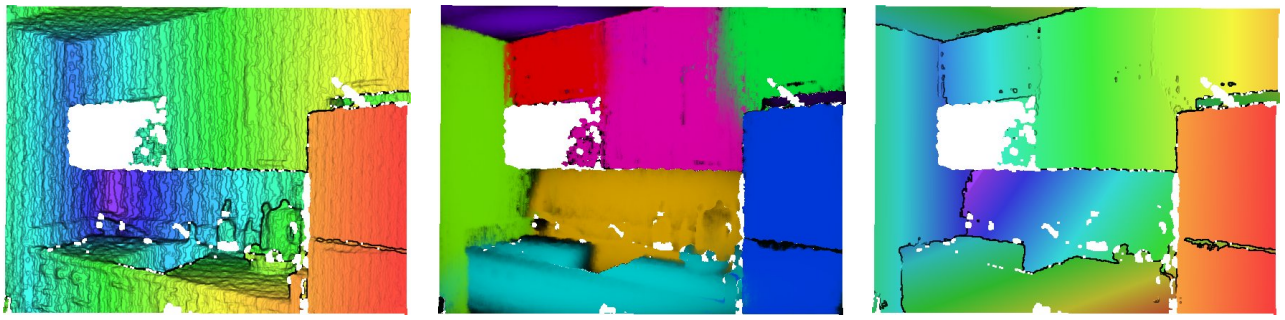
8.1 Pseudo-synthetic Dataset

The primary goal of this paper is to produce a model that works well on actual Kinect images. However, it is extremely difficult to produce ground-truth shape, reflectance, shading, and illumination models for real-world natural scenes. Thankfully, using the MIT-Berkeley Intrinsic Images dataset we can compose pseudo-synthetic scenes that emulate natural scenes. We will do this by layering the objects in the dataset into a scene, and then rendering the resulting shapes using a randomly sampled spatially-varying illumination. For each scene we also produce a noisy Kinect-like depth map for use as input to our model. With a dataset of "scene"-like images which contain occlusion and spatially varying illumination, we can quantitatively evaluate how our model might perform on actual Kinect imagery. This dataset also allows us to tune hyperparameters (on the training set) and compare our model to others (on the test set).

The creation of our dataset is as follows: The 20 objects in the MIT dataset are downsampled by a factor of two, such that our resulting scenes can be reasonably sized. We split the objects into training and test sets, using the same split as in [1]. We then synthesize 10 training and 10 test scenes, where training scenes are composed of training objects, and test scenes are composed of test objects. To generate a scene, we iteratively layer objects on top of each other, taking care to place each object in the largest part of the scene that is still empty. For each scene we generate a layered depth map, normal map, and reflectance image. Given these, we then synthesize a natural illumination to generate a shading image. All of our scenes are 256×256 pixels.

Natural scenes have complicated, spatially-varying illumination due to attenuation, shadowing, inter-reflection, etc. Therefore, to make this dataset a somewhat realistic surrogate for real images of natural scenes, we cannot simply use one global model of illumination, as was done in [1]. We will instead synthesize our own spatially-varying illumination as a mixture of many attenuated point-light sources. For each scene, we generate 50 lights, each of whose position is generated in a uniform region twice the size of the volume enveloped by the objects in the scene, and whose color c_i is randomly sampled from the average RGB value of an image in the sIBL Archive[2]. For each light, we compute the distance d_i of each depth-map pixel to the light source and compute an attenuation $a_i = 1/(1 + d_i^2/40000)$, where 40000 (which controls the amount of attenuation) is chosen manually such that the resulting scenes look reasonable. For each pixel, we compute the unit vector ℓ_i from that pixel's location in the scene to the light source's position.

2. http://www.hdrlabs.com/sibl/archive.html



(a) Kinect Depth Map (b) Mixture of Gaussians (c) Plane-Fit Depth Map

Fig. 3. We initialize the depth maps in our shape mixture by fitting a mixture of Gaussians to the (x, y, z) coordinates of depth-map pixels, and then fitting a plane to each Gaussian. 3(a) shows the raw depth map, 3(b) shows the posterior probability of each pixel under each mixture component, and 3(c) shows the fitted planes composed into one depth map according to hard assignments under the mixture of Gaussians.

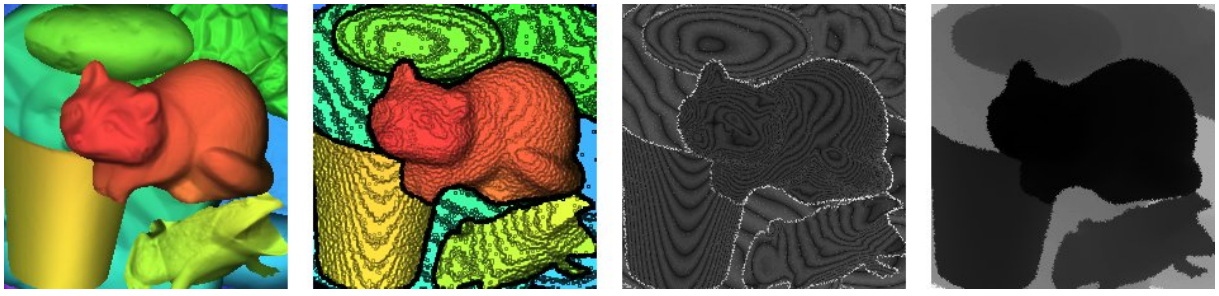


Fig. 4. A visualization of how we introduce noise to Kinect images. In the first column we have a ground-truth depth-map, and in the second we have our corrupted version of it that we will use as a proxy for Kinect data. The third column shows the difference between the two, where we see the stripe-like noise introduced by quantization, as well as the noise near the boundaries of the objects introduced by shuffling and mis-aligning the image. The fourth column is a visualization of our error model, where we see error increases with depth, in accordance with our understanding of binocular stereo.

We then render the scene according to attenuated Lambertian reflectance $(c_i \times \max(0, a_i(n_i \cdot \ell_i)))$ for each light source, and take the sum of all of these renderings (after dividing by the max intensity) as the complete shading image. The final image, which is this shading image multiplied by the previously-generated reflectance image, will be the RGB input to our system.

In parallel with rendering these images, we render a "light probe" surface that has the same depths as the scene, but whose normal field is chosen to uniformly sample the space of orientations. We will evaluate the fidelity of our recovered illumination by rendering this "light probe" surface according to our recovered illumination, and comparing it to the ground-truth produced during this synthesis.

In Section 5 we described a probabilistic model of the noise present in Kinect depth maps, for use during inference. This noise model is not sufficient for our experiments: we must be able to synthetically generate noisy depth maps that are similar to Kinect depth maps, for the purposes of tuning our model parameters and empirically evaluating the accuracy of our model on pseudo-synthetic data. We therefore present a generative model to construct a noisy depth map: Given a "true" depth-map Z^* in centimeters, we first construct a disparity map. We replace each pixel in the disparity map with the bilinearly interpolated value of a location near that pixel where the shift is drawn from a normal distribution $\sigma = 1/2$, then we add IID Gaussian noise to each pixel of the disparity map ($\sigma = 1/6$), and then we translate all pixels in the disparity map by a shift drawn from a normal distribution ($\sigma = 1/4$). The first shuffling and injection of noise is intended to simulate sensor noise in the Kinect, and the second image-wide shuffling is intended to simulate a slightly inaccurate alignment between the image and the depth map, which we often see in even well-aligned Kinect data. We quantize the shuffled disparity by rounding it to the nearest integer, and then convert the disparity measurements back into depth measurements. Formally, our procedure for corrupting a ground-truth depth map is:

$$\hat{Z} \leftarrow \frac{35130}{\lfloor 35130/(\text{shuffle}(Z^*)) + \mathcal{N}(0, (1/6)^2) + 0.5 \rfloor} \quad (12)$$

The constant 35130 is derived from the baseline of the Kinect sensors. See Figure 4 for a visualization of

these synthetic noisy depth maps. Though our model for generating noise is different from our model for inference in the face of noise, manual investigation of the data suggests that the models largely agree — the marginal distribution of the noisy depth maps generated from Equation 12 appears to match the prior posed earlier, in terms of the width of the uniform distribution and the shape of the hyper-laplacian tail. Perhaps more importantly, our synthetically noisy depth maps look similar to the real depth maps in the NYU Depth Dataset [17].

Table 1 compares our model's performance to other intrinsic images techniques and to ablations of our model. The error metrics we use to evaluate performance are reviewed in Section 8.2. The shading and reflectance images produced by our model beat or match the best intrinsic image algorithms. The surface normals produced by our model have half of the error of the input, though for absolute depth error we do not improve. This is consistent with the limits of shading, as shading directly informs the surface normal, but only implicitly informs absolute depth. Our performance is similar to that of SIRFS in terms of reflectance, but much better in all other respects. A naive extension of SIRFS to scenes (in which we use normalized cuts to segment each image into 16 segments and run SIRFS on each segment in isolation) performs similarly to basic SIRFS. The source of our advantage over SIRFS is shown through our ablations — removing either the shape or the illumination mixture components hurts performance on every error metric, and removing the Kinect depth map hurts performance on the depth and normal error metrics, though not the shading or reflectance metrics. The degenerate case of our model which only denoises the depth map and ignores the RGB image performs surprisingly well in terms of error relative to the ground-truth shape and normal field. However, we believe this mostly reflects a bias in our error metrics towards overly smooth shapes, which the shape-denoising ablation produces (see Figure 9).

8.2 Error Metrics

Our error metrics are a revision of those in [1]. We use six error metrics: two for shape, one for shading, one for reflectance, one joint error metric for shading and reflectance introduced in [7], which we will refer to as rs-MSE (though which the original authors call "LMSE"), and one for illumination.

Our first shape error metric is the shift-invariant mean absolute error between our recovered shape $\hat{Z}$ and the true shape Z^*:

$$Z\text{-MAE}(\hat{Z}, Z^*) = \frac{1}{n}\min_{\beta} \sum_{x,y} \left| \hat{Z}_{x,y} - Z^*_{x,y} + \beta \right| \quad (13)$$

Our second shape error metric is the mean angle (in radians) between our recovered normal field $\hat{N}$ and the true normal field N^*:

$$N\text{-MAE}(\hat{N}, N^*) = \frac{1}{n} \sum_{x,y} \arccos\left(\hat{N}_{x,y} \cdot N^*_{x,y} \right) \quad (14)$$

Together these two error metrics measure the important quantities of the recovered depth map. Z-MAE is sensitive to the overall shape of the depth map, modulo any global shift (which cannot be directly observed except in the Kinect depth map) and N-MAE is sensitive to the local orientation of each pixel in the depth map.

For shading and reflectance, we use the scale-invariant mean squared error of our recovered shading image $\hat{s} = \exp(\hat{S})$ and reflectance image $\hat{r} = \exp(\hat{R})$

$$s\text{-MSE}(\hat{s}, s^*) = \frac{1}{n}\min_{\alpha} \sum_{x,y} \left\| \alpha \hat{s}_{x,y} - s^*_{x,y} \right\|_2^2 \quad (15)$$

$$r\text{-MSE}(\hat{r}, r^*) = \frac{1}{n}\min_{\alpha} \sum_{x,y} \left\| \alpha \hat{r}_{x,y} - r^*_{x,y} \right\|_2^2 \quad (16)$$

These error metrics are invariant to absolute scaling of the entire image, but not to each RGB channel individually, and therefore measure errors in overall color and white-balance.

We also use the error metric introduced with the MIT intrinsic images dataset [7], which the authors refer to as LMSE, but which we will call rs-MSE to avoid confusion with L-MSE. This metric measures local scale-invariant error for each channel of both reflectance and shading, thereby measuring high-frequency errors in both that are insensitive to overall intensity and color.

Our error metric for illumination is the scale-invariant MSE of a rendering of our recovered illumination $\{\hat{\boldsymbol{L}}, \hat{\boldsymbol{V}}\}$ on a "light-probe"-like surface, compared to a ground-truth light-probe surface that was rendered alongside the image when generating the data:

$$L\text{-MSE}(\hat{\boldsymbol{L}}, \hat{\boldsymbol{V}}, S^p) = \frac{1}{n}\min_{\alpha} \sum_{x,y} ||\alpha S(N^p, \hat{\boldsymbol{L}}, \hat{\boldsymbol{V}})_{x,y} - S^p_{x,y}||_2^2 \quad (17)$$

Where N^p is the light-probe normal field we use, and S^p is a rendering of that normal field under the ground-truth attenuated point light-sources used in generating our pseudo-synthetic imagery.

When evaluating these error metrics, we take the geometric mean of each metric over each image in the test set. When producing our "average" error metric, we take the geometric mean of each error metric. The geometric mean is used because it is insensitive to scalings of the constituent error metrics, so arbitrarily large error metrics or unusually difficult images don't have more influence. The geometric mean is therefore difficult to trivially minimize in practice.

Algorithm	Z-MAE	N-MAE	s-MSE	r-MSE	rs-MSE	L-MSE	Avg.
(1) Color Retinex [7], [4]	-	-	0.0230	0.0364	0.0354	-	-
(2) Tappen *et al.* 2005 [8]	-	-	0.0281	0.0337	0.0387	-	-
(3) Gehler *et al.* 2011 [6]	-	-	0.0181	0.0224	**0.0216**	-	-
(4) Kinect Only	**5.09**	0.5799	-	-	-	-	-
(5) SIRFS	114.82	0.6841	0.0181	0.0202	0.0289	0.0241	0.1647
(6) SIRFS + Segmentation	57.43	0.7600	0.0176	0.0200	0.0296	0.0210	0.1458
(A) Scene-SIRFS (Complete)	10.91	**0.2618**	**0.0101**	**0.0184**	0.0227	**0.0166**	**0.0764**
(B) Scene-SIRFS ($\lambda_o = 0$)	122.67	0.6454	0.0134	0.0203	0.0256	0.0199	0.1491
(C) Scene-SIRFS (No Initialization)	11.06	0.3000	0.0113	0.0233	0.0263	0.0176	0.0860
(D) Scene-SIRFS ($\|\boldsymbol{Z}\| = 1$)	22.72	0.5123	0.0179	0.0284	0.0348	0.0237	0.1302
(E) Scene-SIRFS ($\|\boldsymbol{L}\| = 1$)	11.64	0.2754	0.0163	0.0313	0.0269	0.0211	0.0988
(F) Scene-SIRFS ($\|\boldsymbol{Z}\| = \|\boldsymbol{L}\| = 1$)	24.59	0.5285	0.0292	0.0587	0.0523	0.0213	0.1708
(G) Scene-SIRFS (Z only)	9.82	0.2877	-	-	-	-	-
(H) Scene-SIRFS (Z only, $\|\boldsymbol{Z}\| = 1$)	24.69	0.5552	-	-	-	-	-

TABLE 1
Our results on the test set of our pseudo-synthetic dataset. Shown are the geometric means of six error metrics (detailed in the 8.2) across the test set, and an "average" error (the geometric mean of the other error metrics). Z-MAE measures shape errors, N-MAE measures surface-normal errors, s-MSE, r-MSE, and rs-MSE measure shading and reflectance errors, and L-MSE measures illumination errors. If an algorithm does not produce a certain scene property, its error is left blank. (1)-(3) are intrinsic image algorithms, which produce shading and reflectance images from an RGB image, where (3) is the current state-of-the-art. (4) evaluates the error of the noisy Kinect-like depth maps we use as input. (5) is the SIRFS model that we build upon, and is equivalent to our model without any mixture models or a Kinect depth map. (6) is SIRFS run in isolation on the segments produced by normalized cuts. In addition to our complete model (A), we present several ablations. (B) has no Kinect information, (C) has no initialization, and (D)-(F) omit one or both of the shape or light mixtures. (G) is a shape-denoising algorithms in which we omit the RGB image and just optimize over shape with respect to our prior on shapes, and (H) is (G) with a single depth map, instead of a mixture model.

8.3 Kinect Data

To qualitatively evaluate our model, we sampled several images from version 2 of the NYU Depth Dataset [17], and ran them through our model (all using the same hyperparameter setting as in our pseudo-synthetic experiment). The output of our model can be seen in Figures 6 and 7. We compare against two intrinsic image algorithms: Retinex [7], [4] and Gehler *et al.* [6].

Our shading and reflectance images generally look much better than those produced by the intrinsic image algorithms, and our recovered depth and surface normals look much better than the input Kinect image. Our spatially varying illumination captures shadowing and interreflections, and looks reasonable. The primary cause of errors in our model appears to be over-smoothing of the depth map, which we believe is because the error metrics with which we cross-validate our model tend to favor conservative parameter settings, and because MAP estimation for tasks such as ours tends to produce overly conservative output [35].

One way to evaluate the accuracy of our model is to use it in graphics applications. In Figures 8 and 10 we use our output to re-render the input image under different camera viewpoints and under different illumination conditions. Our renderings look significantly better than renderings produced with the inpainted Kinect depth map provided by the NYU dataset. Changing the viewpoint with the raw Kinect depths creates jagged artifacts at the edges of shapes, while our depth (which is both denoised and better-aligned to the image) looks smooth and natural at object boundaries. Relighting the raw Kinect depth produces terrible artifacts, as the surface normals of the raw depth are very inaccurate due to noise and quantization, while relighting our output looks reasonable, as the surface normals are cleaner and reflectance has been separated from shading. In Figure 9 we see that the depth maps our model produces are less noisy than the NYU depth maps, and more detailed than the output of the shape-denoising ablation of our model, demonstrating the importance of the complete model.

9 Conclusion

We have presented Scene-SIRFS, a variant of SIRFS that takes as input images of natural scenes rather than images of segmented objects. We have done this by generalizing SIRFS into a mixture model of shapes and illuminations, and by embedding those mixtures into a soft segmentation of an image. We additionally use the noisy depth maps in RGB-D data to improve low-frequency shape estimation.

The output of our model can be used for graphics applications such relighting or re-orienting the camera, and it is easy to imagine other applications such as inserting objects, modifying reflectances, or white balancing. Our model improves the initial depth map by removing noise, adding fine-scale shape detail,

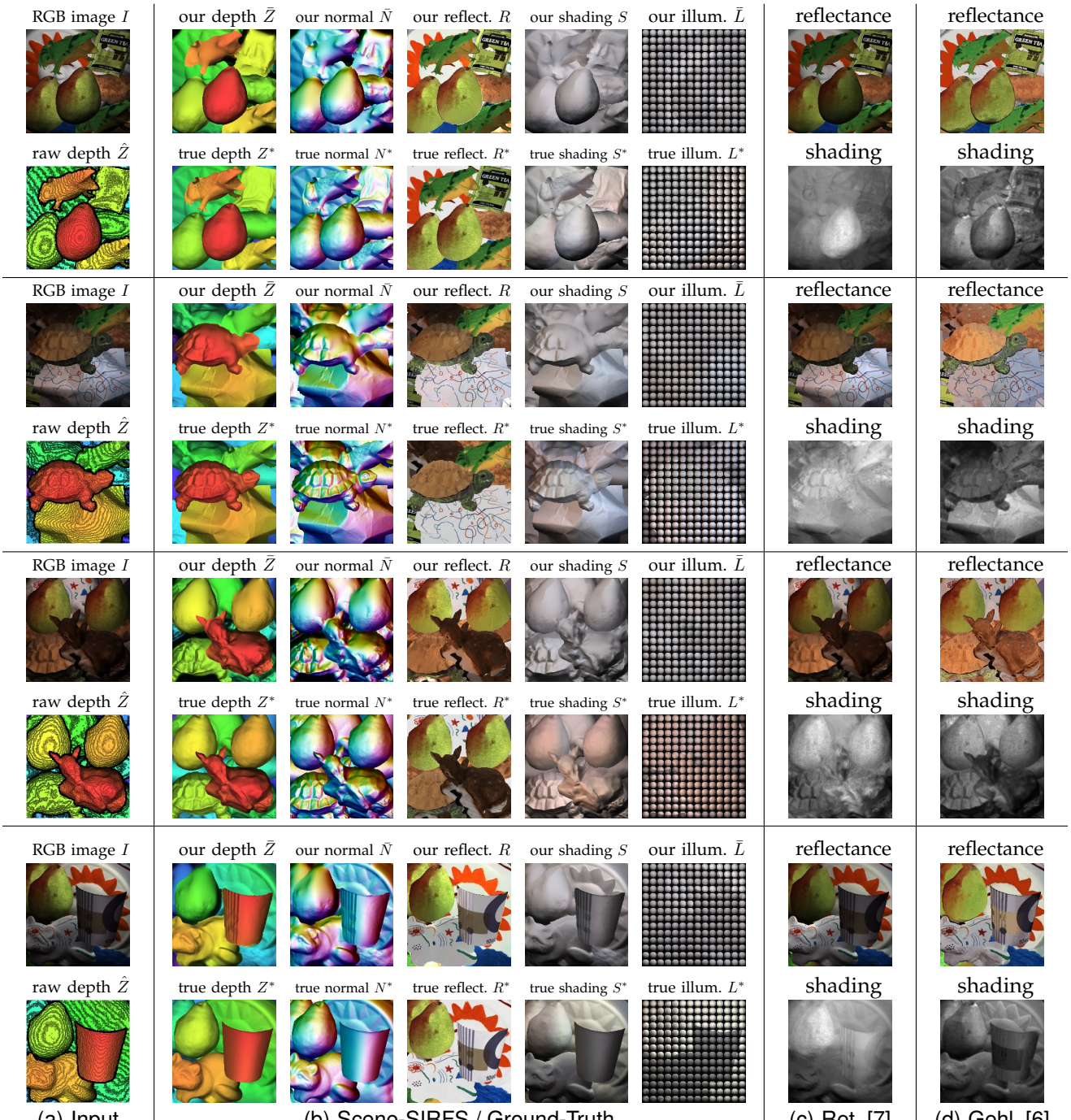


Fig. 5. Some test-set scenes from our pseudo-synthetic scene dataset. In 5(a) we have the input to our model: an RGB image and a noisy Kinect-like depth map. In 5(b) we have the depth map, surface normals, reflectance, shading, and spatially-varying illumination that our model produces, and the corresponding ground-truth scene properties on the bottom. In 5(c) and 5(d) we show the shading and reflectance images produced by the best-performing intrinsic image algorithms.

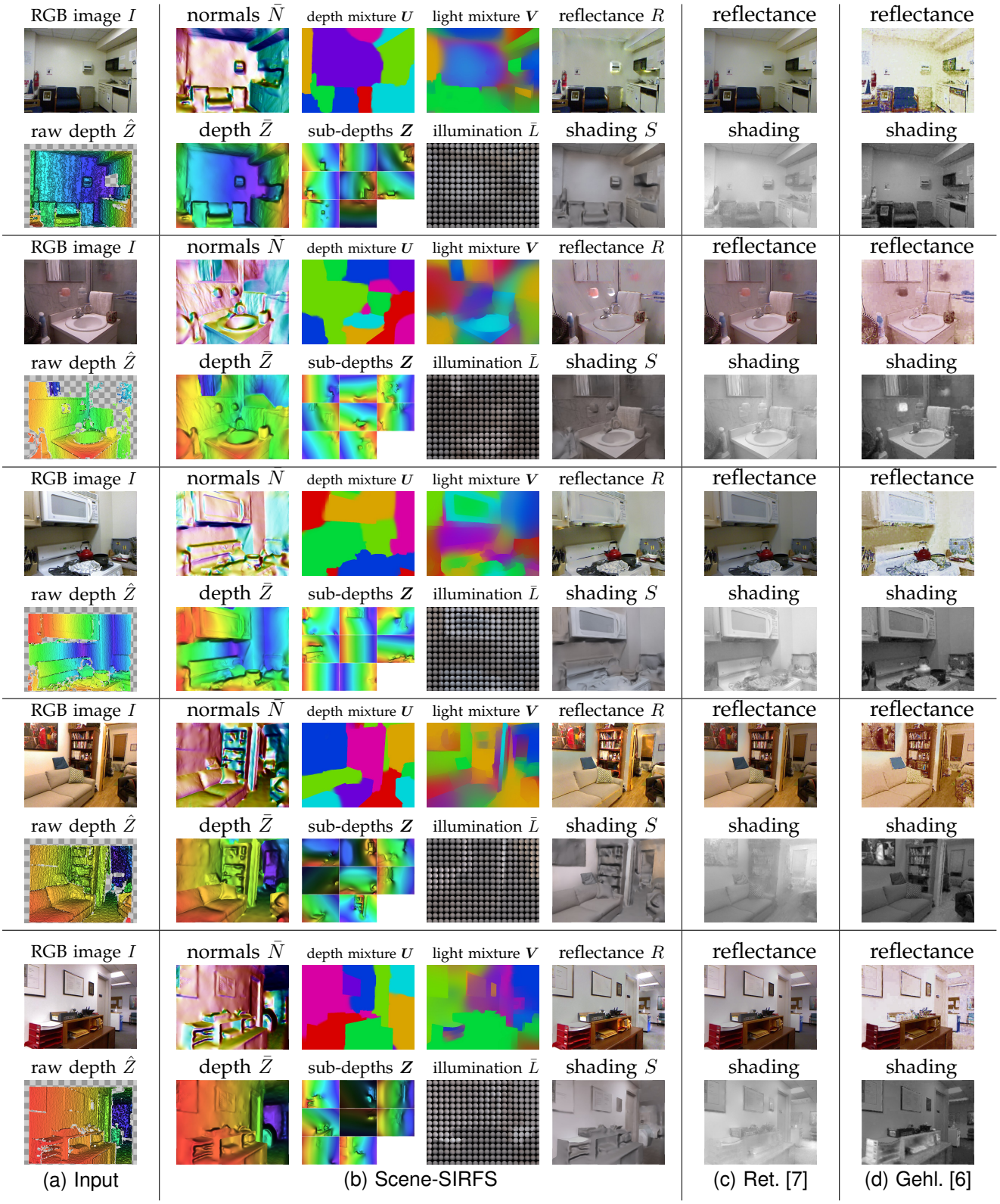


Fig. 6. In 6(a) we have the input to our model: an RGB image and a Kinect depth map from the NYU Depth Dataset [17]. In 6(b) we have the output of our model. Mixtures are visualized with hue corresponding to component, and intensity corresponding to probability. Illumination is visualized by rendering a coarse grid of spheres.

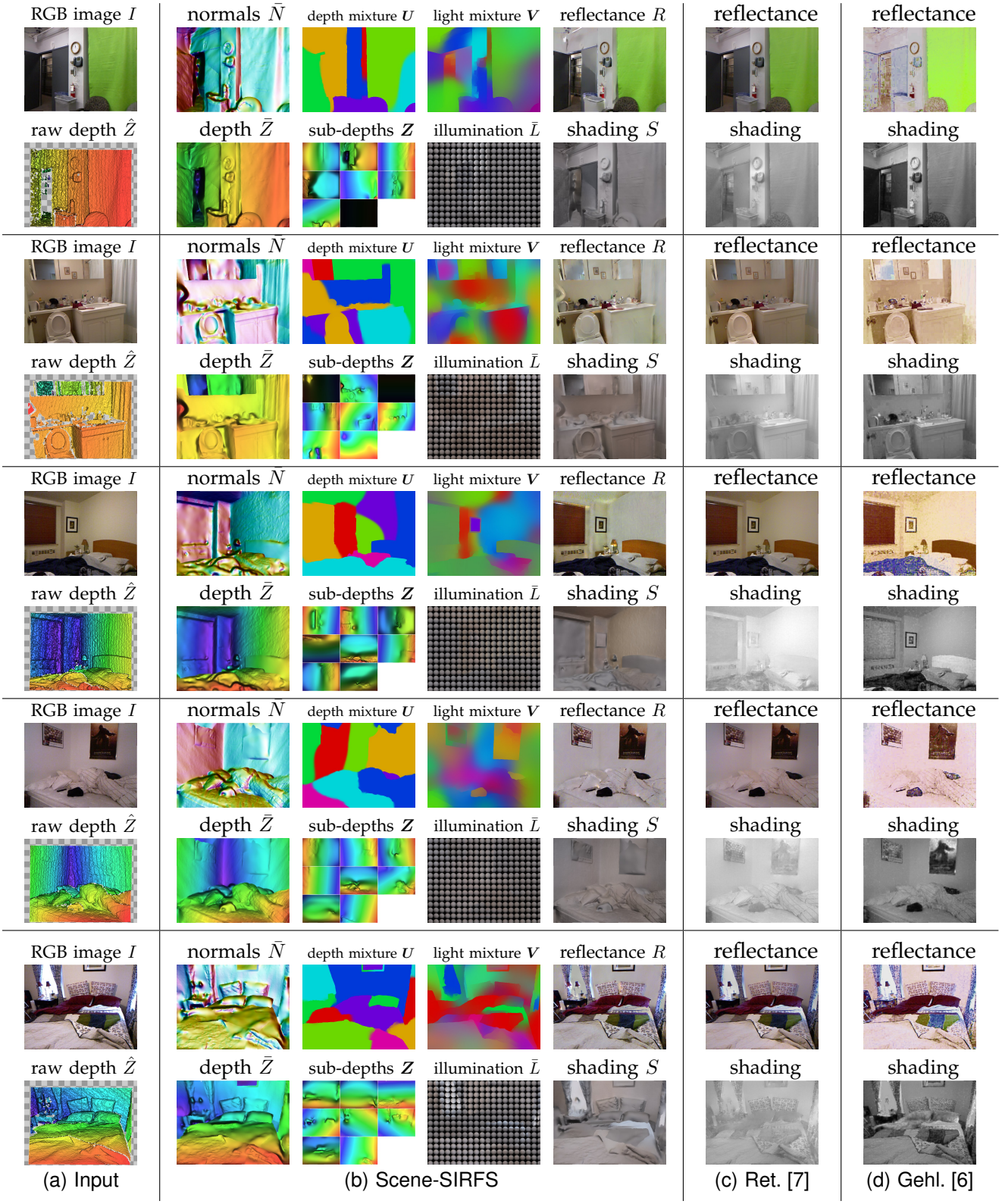
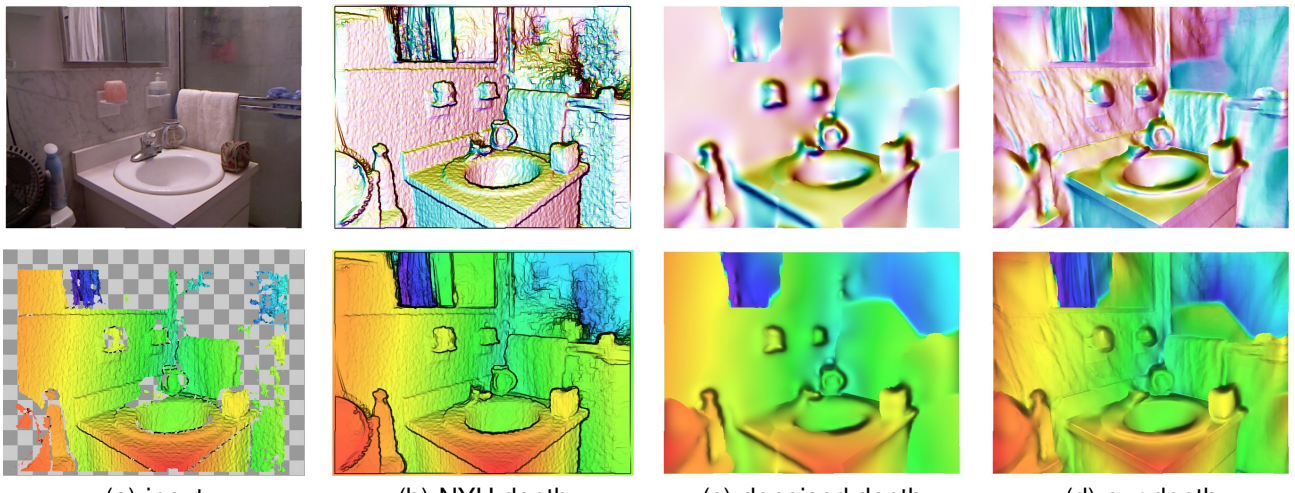


Fig. 7. More results from the NYU Depth Dataset [17].

Fig. 8. After a model is recovered, the camera can be moved and the input image (left) can be shown from a different viewpoint (right). Such a warping could be produced using just the smoothed Kinect depth maps provided in the NYU dataset (middle), but these images have jagged artifacts at surface and normal discontinuities. Both renderings, of course, contain artifacts in occluded regions.



(a) input (b) NYU depth (c) denoised depth (d) our depth

Fig. 9. One output of our model is a denoised depth-map. In 9(a) we have the RGB-D input to our model, demonstrating how noisy and incomplete the raw Kinect depth map can be. 9(b) shows the inpainted normals and depth included in the NYU dataset [17], where holes have been inpainted but there is still a great deal of noise, and many fine-scale shape details are missing. 9(c) is from an ablation of our model in which we just denoise/inpaint the raw depth map ("model H" in our ablation study), and 9(d) is from our complete model. The NYU depth map is noisy and does not model depth discontinuities, and our "denoising" model tends to oversmooth the scene, but our complete model has little noise while recovering much of the detail of the scene and correctly separating objects into different layers.

Fig. 10. After a model is recovered, the spherical harmonic illuminations can be replaced (here we use randomly generated illuminations) and the input image (left) can shown under a different illumination (right). The middle image is our attempt to produce similar re-lit images using only the inpainted depth maps in the NYU dataset, which look noticeably worse due to noise in the depth image and the fact that illumination and reflectance have not been decomposed.

and aligning the depth to the RGB image, all of which presumably would be useful in any application involving RGB-D images. Perhaps most importantly, our model takes an important step towards solving one of the grand challenges in vision — inferring all intrinsic scene properties from a single image.

ACKNOWLEDGMENTS

J.B. was supported by NSF GRFP and ONR MURI N00014-10-10933.

REFERENCES

[1] J. T. Barron and J. Malik, "Shape, illumination, and reflectance from shading," *Tech Report*, 2013.
[2] M. Pharr and G. Humphreys, *Physically Based Rendering, Second Edition: From Theory To Implementation*. Morgan Kaufmann Publishers Inc., 2010.
[3] H. Barrow and J. Tenenbaum, "Recovering intrinsic scene characteristics from images," *Computer Vision Systems*, 1978.
[4] B. K. P. Horn, "Determining lightness from an image," *Computer Graphics and Image Processing*, 1974.
[5] E. H. Land and J. J. McCann, "Lightness and retinex theory," *JOSA*, 1971.
[6] P. Gehler, C. Rother, M. Kiefel, L. Zhang, and B. Schoelkopf, "Recovering intrinsic images with a global sparsity prior on reflectance," *NIPS*, 2011.
[7] R. Grosse, M. K. Johnson, E. H. Adelson, and W. T. Freeman, "Ground-truth dataset and baseline evaluations for intrinsic image algorithms," *ICCV*, 2009.
[8] M. F. Tappen, W. T. Freeman, and E. H. Adelson, "Recovering intrinsic images from a single image," *TPAMI*, 2005.
[9] J. T. Barron and J. Malik, "Color constancy, intrinsic images, and shape estimation," *ECCV*, 2012.
[10] ——, "Shape, albedo, and illumination from a single image of an unknown object," *CVPR*, 2012.
[11] ——, "High-frequency shape and albedo from shading using natural image statistics," *CVPR*, 2011.
[12] J. Shi and J. Malik, "Normalized cuts and image segmentation," *TPAMI*, 2000.
[13] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, 2002.
[14] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.
[15] A. Blake, A. Zisserman, and G. Knowles, "Surface descriptions from stereo and shading," *IVC*, 1986.
[16] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, "Depth mapping using projected patterns," *US Patent*, 2009.
[17] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," *ECCV*, 2012.

[18] D. A. Forsyth, "Variable-source shading analysis," *IJCV*, 2011.
[19] Y. Yu, P. Debevec, J. Malik, and T. Hawkins, "Inverse global illumination: recovering reflectance models of real scenes from photographs," *SIGGRAPH*, 1999.
[20] K. Karsch, V. Hedau, D. Forsyth, and D. Hoiem, "Rendering synthetic objects into legacy photographs," *SIGGRAPH Asia*, 2011.
[21] D. Hoiem, A. A. Efros, and M. Hebert, "Recovering surface layout from an image," *IJCV*, 2007.
[22] A. Saxena, M. Sun, and A. Ng, "Make3d: learning 3d scene structure from a single still image," *TPAMI*, 2008.
[23] K. J. Lee, Q. Zhao, X. Tong, M. Gong, S. Izadi, S. U. Lee, P. Tan, and S. Lin, "Estimation of intrinsic image sequences from image+depth video," *ECCV*, 2012.
[24] Q. Chen and V. Koltun, "A simple model for intrinsic image decomposition with depth cues," *ICCV*, 2013.
[25] B. K. P. Horn, "Shape from shading: A method for obtaining the shape of a smooth opaque object from one view," MIT, Tech. Rep., 1970.
[26] R. Ramamoorthi and P. Hanrahan, "An Efficient Representation for Irradiance Environment Maps," *CGIT*, 2001.
[27] J. Koenderink, "What does the occluding contour tell us about solid shape?" *Perception*, 1984.
[28] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *TPAMI*, 2011.
[29] Y. Ostrovsky, P. Cavanagh, and P. Sinha, "Perceiving illumination inconsistencies in scenes." *Perception*, 2005.
[30] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *NIPS*, 2001.
[31] M. Maire, P. Arbelaez, C. C. Fowlkes, and J. Malik, "Using contours to detect and localize junctions in natural images," *CVPR*, 2008.
[32] T. K. Leung and J. Malik, "Contour continuity in region based image segmentation," *ECCV*, 1998.
[33] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marqués, and J. Malik, "Multiscale combinatorial grouping," *CVPR*, 2014.
[34] S. Maji, N. Vishnoi, and J. Malik, "Biased normalized cuts," *CVPR*, 2011.
[35] A. Levin, Y. Weiss, F. Durand, and W. Freeman, "Understanding and evaluating blind deconvolution algorithms," *CVPR*, 2009.

Jonathan T. Barron is a senior research scientist in Google, working on computer vision and computational photography. He received a PhD in Computer Science from the University of California, Berkeley in 2013, where he was advised by Jitendra Malik, and he received a Honours BSc. in Computer Science from the University of Toronto in 2007. His research interests include computer vision, machine learning, computational photography, shape reconstruction, and biological image analysis. He received a National Science Foundation Graduate Research Fellowship in 2009, and the C.V. Ramamoorthy Distinguished Research Award in 2013.

Jitendra Malik was born in Mathura, India in 1960. He received the B.Tech degree in Electrical Engineering from Indian Institute of Technology, Kanpur in 1980 and the PhD degree in Computer Science from Stanford University in 1985. In January 1986, he joined the university of California at Berkeley, where he is currently the Arthur J. Chick Professor in the Computer Science Division, Department of Electrical Engg and Computer Sciences. He is also on the faculty of the department of Bioengineering, and the Cognitive Science and Vision Science groups. During 2002-2004 he served as the Chair of the Computer Science Division and during 2004-2006 as the Department Chair of EECS. He serves on the advisory board of Microsoft Research India, and on the Governing Body of IIIT Bangalore.

He received the gold medal for the best graduating student in Electrical Engineering from IIT Kanpur in 1980 and a Presidential Young Investigator Award in 1989. At UC Berkeley, he was selected for the Diane S. McEntyre Award for Excellence in Teaching in 2000, a Miller Research Professorship in 2001, and appointed to be the Arthur J. Chick Professor in 2002. He received the Distinguished Alumnus Award from IIT Kanpur in 2008. He was awarded the Longuet-Higgins Prize for a contribution that has stood the test of time twice, in 2007 and in 2008. He is a fellow of the IEEE and the ACM, and a member of the National Academy of Engineering.