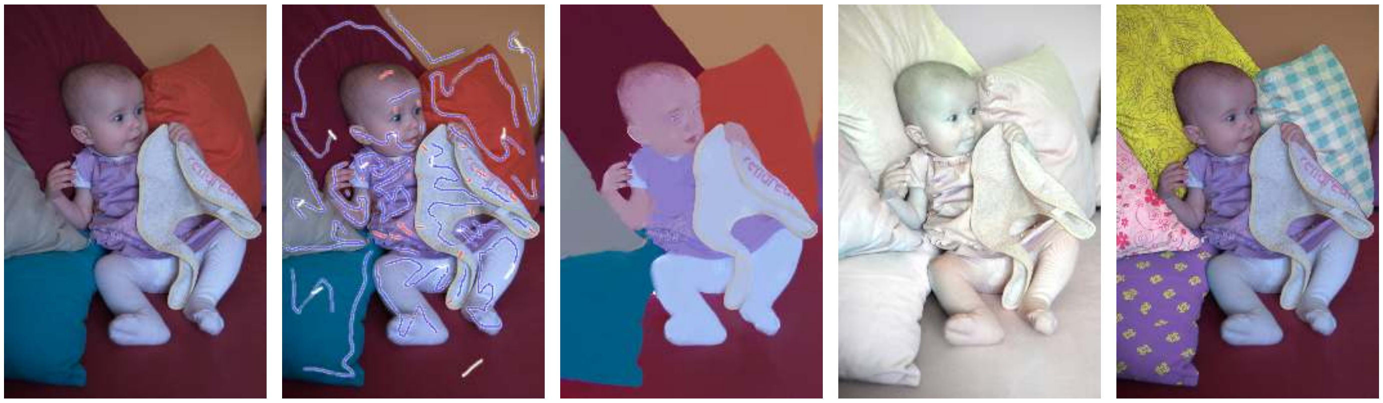
User-Assisted Intrinsic Images

Adrien Bousseau[1,2] Sylvain Paris[3] Frédo Durand[2]

[1]INRIA / Grenoble University [2]MIT CSAIL [3]Adobe Systems, Inc.



(a) Original photograph (b) User scribbles (c) Reflectance (d) Illumination (e) Re-texturing

Figure 1: *Our system relies on user indications, shown in (b), to extract from a single photograph its reflectance and illumination components (c-d). In (b), white scribbles indicate fully-lit pixels, blue scribbles correspond to pixels sharing a similar reflectance and red scribbles correspond to pixels sharing a similar illumination. This decomposition facilitates advanced image editing such as re-texturing (e).*

Abstract

For many computational photography applications, the lighting and materials in the scene are critical pieces of information. We seek to obtain *intrinsic images*, which decompose a photo into the product of an *illumination* component that represents lighting effects and a *reflectance* component that is the color of the observed material. This is an under-constrained problem and automatic methods are challenged by complex natural images. We describe a new approach that enables users to guide an optimization with simple indications such as regions of constant reflectance or illumination. Based on a simple assumption on local reflectance distributions, we derive a new propagation energy that enables a closed form solution using linear least-squares. We achieve fast performance by introducing a novel downsampling that preserves local color distributions. We demonstrate intrinsic image decomposition on a variety of images and show applications.

Keywords: computational photography, intrinsic images, reflectance-illumination separation

1 Introduction

Many computational photography tasks such as relighting, material alteration or re-texturing require knowledge of the lighting and materials in the scene. Unfortunately, in a photograph, illumination and reflectance are conflated through complex interaction and the separation of those components, called *intrinsic images* [Barrow and Tenenbaum 1978], has long been an open challenge. A pixel can be seen as the per-color-channel product of an *illumination* component, also called *shading*, and a *reflectance* component, also called *albedo*. Given a single image, the problem is severely ill-posed: a dark-yellow pixel can come from, e.g., a white material illuminated by a dark yellow light, or from a dark-yellow material illuminated by a bright white light.

In this paper, we introduce a new image decomposition technique that relies on sparse constraints provided by the user to disambiguate illumination from reflectance. Figure 1(b) illustrates these indications that can correspond to pixels of similar reflectance, similar illumination, or known illumination. Central to our technique is a new propagation method that estimates illumination from local constraints based on a few assumptions on image formation and reflectance distributions. In particular, we reduce the number of unknowns by assuming that local reflectance variations lie in 2D subspaces of the RGB color space. Although this simplification requires color images and cannot handle cases such as a black-and-white checkerboard texture, we show that it can handle a broad class of images with complex lighting. In order to enable fast and accurate solution, we also introduce a novel downsampling strategy that better preserves the local color distributions of an image and enables rapid multigrid computation. Finally, we illustrate the benefit of our image decomposition with image manipulations including reflectance editing and re-lighting.

To summarize, this paper makes the following contributions:
• we describe a user-assisted method to perform intrinsic image decomposition from a single image.
• we introduce a novel energy formulation that propagates user edits based on the local reflectance color distribution.
• we propose a new downsampling scheme that enforces the preservation of color distributions at reduced image resolutions.

1.1 Related Work

The decoupling of reflectance from illumination was introduced by Barrow and Tenenbaum [1978] as *intrinsic images*. The *reflectance* describes how an object reflects light and is also often called *albedo*. The *illumination* corresponds to the amount of light incident at a point (essentially irradiance). Although it is often refered to as *shading*, it includes effects such as shadows and indirect lighting.

Physically-based inverse rendering, such as Yu and Malik [1998], seeks to invert the image formation model in order to recover the lighting conditions of a scene, but requires known geometry.

Using several images of the same scene under different illuminations, Weiss [2001] proposes a method to estimate a reflectance image along with the illumination map of each input image. This approach has then been extended by Liu et al. [2008] to non-registered image collections in order to colorize black-and-white photographs. The use of varying illumination has also been applied to shadow removal in a flash/no-flash setup by Agrawal et al. [2006].

Due to its inherent ill-posedness, decomposition of intrinsic images from a single image cannot be solved without prior knowledge on reflectance and illumination. Based on the Retinex theory [Land and McCann 1971], Horn [1986] assumes that reflectance is piecewise constant while illumination is smooth. This heuristic allows for the recovery of a reflectance image by thresholding the small image gradients, assumed to correspond to illumination. Sinha and Adelson [1993] discriminate illumination from reflectance edges based on their junctions in a world of painted polyhedra: T junctions are interpreted as reflectance variations, while arrow and Y junctions correspond to illumination. Tappen et al. [2005] rely on a classifier trained on image derivatives to classify reflectance and illumination gradients. Despite these heuristics and classifiers, many configurations of reflectance and illumination encountered in natural images remain hard to disambiguate. Shen et al. [2008] propose to enrich these local approaches with non-local texture constraints. Starting from a Retinex algorithm, their texture constraints ensure that pixels that share similar texture will have the same reflectance.

A large body of work has been proposed for the specific problem of shadow removal, either automatically [Finlayson et al. 2002; Finlayson et al. 2004] or based on user interaction [Mohan et al. 2007; Wu et al. 2007; Shor and Lischinski 2008]. The common idea of these methods is to identify shadow pixels, either via boundary detection or region segmentation. Once shadows are detected, they can be removed by color correction or gradient domain filtering. These methods focus on cast shadow removal with clear boundaries, while we also target the removal of smooth shading where the boundary between lit and shaded regions cannot be delimited. Note that although the approach of Finlayson et al. [2002; 2004] relies on the estimation of an illumination-free image, this image is grayscale and does not represent the true reflectance.

Intrinsic images decomposition is also related to other image decompositions. Matting algorithms [Chuang et al. 2001; Levin et al. 2008] aim to separate the foreground and background layers of an image along with their opacity based on user indications. User-assisted approaches have been proposed to separate reflections from a single image [Levin and Weiss 2007]. Automatic decompositions have been introduced to restore pictures corrupted by haze [Fattal 2008], and to perform white balance of photographs taken under mixed lighting [Hsu et al. 2008]. Although all these methods do not directly target the extraction of illumination from a single image, our energy formulation is inspired by the *matting Laplacian* used in the work of Levin et al. [2007] and Hsu et al. [2008]. We rely on a similar assumption that, in natural images, material colors lie on subspaces of the RGB space [Omer and Werman 2004].

1.2 Overview

Our decoupling of illumination and reflectance is based on user-provided constraints and a new propagation model. The user can use sparse strokes to mark parts that share the same reflectance or where illumination does not vary, in particular at material boundaries. The user also needs to provide at least one fixed illumination value to solve for global scale ambiguity.

Based on the assumption that reflectance values are low-rank in local windows, we derive a closed-form equation that depends only on illumination. Adding the user constraints, we can solve a linear least-square system that provides the illumination. Reflectance is simply inferred by a division.

Our least-square solution works best with local windows of medium sizes, which raises computational challenges. We have found that standard multigrid approaches yield unsatisfactory results in our case, because traditional downsampling operators do not respect local color distributions. We introduce a new downsampling scheme for techniques like ours that rely on local color distributions. We show that it enables dramatic speedup and achieves high accuracy.

2 Reflectance-Illumination Decomposition

We first detail our assumptions about the observed image, in particular the fact that reflectance colors are locally planar. We show how it leads to a quadratic energy where illumination is the only unknown, and where user constraints can easily be included.

Image Formation Model As commonly done, we assume that the interaction between light and objects can be described using RGB channels alone. We focus on Lambertian objects and a single light color, although we show later that these hypotheses can often be alleviated in practice to handle colored light.

With this model, the observed color at a pixel is:

$$\mathbf{I} = s\,\mathbf{L} * \mathbf{R} \tag{1}$$

where s is the *illumination*, a non-negative scalar modeling the incident light attenuation due to factors such as light travel, occlusion and foreshortening, $\mathbf{L}$ is the RGB color of the light, $*$ denotes per-channel multiplication, and $\mathbf{R}$ is the material RGB *reflectance* that describes how objects reflect light. For clarity, we assume that the light is white, (or equivalently that the input image is white balanced). This means $\mathbf{L} = (1,1,1)^{\mathsf{T}}$ and Equation 1 becomes:

$$\mathbf{I} = s\mathbf{R} \tag{2}$$

2.1 Low-Rank Structure of Local Reflectance

Our objective is to retrieve the illumination s and the RGB components of $\mathbf{R}$ at each pixel. The difficulty is that Equation 2 provides only three equations, one per RGB channel. In addition, there is a scale ambiguity between illumination and reflectance, that is, if $\mathbf{R}_0$ and s_0 are a valid decomposition, then $k\mathbf{R}_0$ and s_0/k are also valid for any scalar factor $k > 0$.

We overcome the ill-posedness of the problem by a local low-rank assumption on reflectance colors. We are inspired by a variety of recent studies that show structure and sparsity in the distribution of colors in natural scenes. In particular, Omer and Werman [2004] show that the set of reflectance colors is sparse, which led to practical matting [Levin et al. 2008] and white balance [Hsu et al. 2008] techniques.

We build on this strategy and assume that, locally, the reflectance colors are low rank. Specifically, we assume that they lie in a 2D plane that does not contain the origin (black) in RGB space (Figure 2). They need not span a full plane and can, for example, consist of a single color or a color line. We acknowledge that this restriction prevents us from treating cases where only luminance variations occur, for example a black-and-white checkerboard, but it enables a practical algorithm that achieves satisfying results on a broad range of natural scenes as demonstrated by our results. In particular, this configuration encompasses cases such as colored textures, constant color objects (1 point in RGB space), edges between two objects (2 points), and T junctions (3 points).

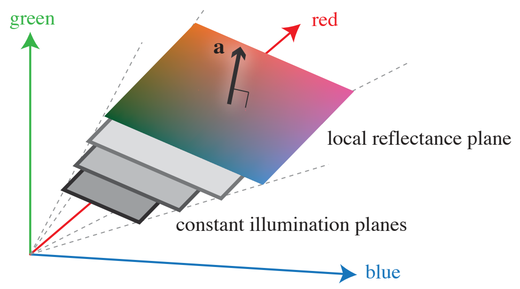


Figure 2: *Planar reflectance assumption. We assume that reflectance variations lie locally in a plane in color space. Pixels in planes parallel to the reflectance have constant illumination (Eq. 3).*

From this planar reflectance assumption, there exists a 3D vector $\mathbf{a}$ such that the reflectance values satisfy:

$$\mathbf{a} \cdot \mathbf{R} = 1 \tag{3}$$

Using Equations 2 and 3, we get

$$\mathbf{a} \cdot \mathbf{I} = s. \tag{4}$$

We have transformed the original equation with a product of unknowns into an equation that is linear in the unknowns $\mathbf{a}$ and s. We further show that we can eliminate $\mathbf{a}$ and directly solve for illumination s. Note also that, locally, there is no scale ambiguity anymore since $\mathbf{R}_0$ and $k\mathbf{R}_0$ cannot both be part of the same reflectance plane unless $k = 1$ (Eq. 3).

2.2 Reduction to Illumination Alone

We now seek to eliminate $\mathbf{a}$ and obtain an equation on s alone. We will then solve this equation in a least-squares fashion to account for user constraints and model deviations. We follow an approach inspired by the work of Levin et al. [2008] in the context of image matting. In a nutshell, we apply our low-rank assumption to all local windows (of e.g. 5×5 pixels) and assume that $\mathbf{a}$ is a constant over a window. We seek to minimize $(s - \mathbf{a} \cdot \mathbf{I})^2$ at each pixel of a window. Note that a pixel participates in many windows and that the unknowns are shared between those local energy terms. This overlap between windows is what enables information propagation. We can eliminate $\mathbf{a}$ from the equations because of the redundant information from the pixels in a window.

Energy Based on Local Windows For a pixel i, we formulate the following energy over its neighboring pixels in a local window $\mathcal{W}_i$:

$$\tilde{e}(s, \mathbf{a}_i) = \sum_{j \in \mathcal{W}_i} (s_j - \mathbf{a}_i \cdot \mathbf{I}_j)^2 \tag{5}$$

We add a regularizer to this energy so that the minimum is always well defined. Indeed, if there is a vector $\mathbf{b}$ such that $\mathbf{b} \cdot \mathbf{I}_j = 0$ for all $j \in \mathcal{W}_i$, then we have $\tilde{e}(s, \mathbf{a}_i) = \tilde{e}(s, \mathbf{a}_i + k\mathbf{b})$ for any real k. Such ambiguity occurs in common cases such as objects with constant reflectance $\mathbf{R}_0$ for which any $\mathbf{b}$ orthogonal to $\mathbf{R}_0$ yields $\mathbf{b} \cdot \mathbf{I}_j = \mathbf{b} \cdot s_j \mathbf{R}_0 = 0$. We address this with a regularizing term:

$$e(s, \mathbf{a}_i) = \sum_{j \in \mathcal{W}_i} (s_j - \mathbf{a}_i \cdot \mathbf{I}_j)^2 + \epsilon\, \mathbf{a}_i^2 \tag{6}$$

where we choose ϵ small so that it has an influence only in ambiguous cases ($\epsilon = 10^{-6}$ in our implementation). Summing over the image, we obtain the energy:

$$E(s, \mathbf{a}) = \sum_i e(s, \mathbf{a}_i) = \sum_i \left[\sum_{j \in \mathcal{W}_i} (s_j - \mathbf{a}_i \cdot \mathbf{I}_j)^2 + \epsilon\, \mathbf{a}_i^2 \right] \tag{7}$$

However, both s and $\mathbf{a}$ are unknown in the above equation. We follow a similar strategy to Levin et al. [2008] and show that $\mathbf{a}$ can be expressed as a function of s. We will see, however, that in our case the model is linear and not affine as with matting.

Illumination as Only Variable We rewrite Equation 6 in matrix form with two vectors S_i and A_i, a matrix M_i, n the number of pixels in $\mathcal{W}_i = \{j_1 \dots j_n\}$, and I^{r}, I^{g}, I^{b} and a^{r}, a^{g}, a^{b} being the RGB components of $\mathbf{I}$ and $\mathbf{a}$ respectively:

$$e(s, \mathbf{a}_i) = \left[\underbrace{\begin{pmatrix} s_{j_1} \\ \vdots \\ s_{j_n} \\ 0 \\ 0 \\ 0 \end{pmatrix}}_{S_i:\ (n+3)\times 1} - \underbrace{\begin{pmatrix} I^{\mathrm{r}}_{j_1} & I^{\mathrm{g}}_{j_1} & I^{\mathrm{b}}_{j_1} \\ \vdots & \vdots & \vdots \\ I^{\mathrm{r}}_{j_n} & I^{\mathrm{g}}_{j_n} & I^{\mathrm{b}}_{j_n} \\ \sqrt{\epsilon} & & \\ & \sqrt{\epsilon} & \\ & & \sqrt{\epsilon} \end{pmatrix}}_{M_i:\ (n+3)\times 3} \underbrace{\begin{pmatrix} a^{\mathrm{r}}_i \\ a^{\mathrm{g}}_i \\ a^{\mathrm{b}}_i \end{pmatrix}}_{A_i:\ 3\times 1} \right]^2 \tag{8}$$

For a fixed illumination $\bar{s}$, we have $e(\bar{s}, \mathbf{a}_i) = (\bar{S}_i - M_i\ A_i)^2$ which is a classical linear least-square functional with A_i as unknowns. The minimizer is given by: $A_i^{\min} = (M_i^{\mathsf{T}} M_i)^{-1} M_i^{\mathsf{T}} \bar{S}_i$. With this result, we rewrite the local energy as a function of the illumination only:

$$f(S_i) = \left(S_i - M_i (M_i^{\mathsf{T}} M_i)^{-1} M_i^{\mathsf{T}} S_i \right)^2 \tag{9}$$

Using the matrix $N_i = Id - M_i (M_i^{\mathsf{T}} M_i)^{-1} M_i^{\mathsf{T}}$ with Id the identity matrix, we obtain the global energy that is a function of the local illumination vectors $\{S_i\}$ only:

$$\sum_i (N_i S_i)^2 = \sum_i S_i^{\mathsf{T}} N_i^{\mathsf{T}} N_i\, S_i \tag{10}$$

This defines a functional in which each illumination value s appears in several local vectors S_i. To obtain a formula where each variable appears only once, we regroup all the s values into a large vector S that has as many elements as pixels. Then, we rearrange the terms of the $N^{\mathsf{T}} N$ matrices into a large matrix L. Each time the s_i and s_j variables interact in Equation 10, this contributes to the $(i, j)^{\text{th}}$ element of L with the element of the corresponding $N^{\mathsf{T}} N$ matrix[1]:

$$L(i, j) \;=\; \sum_{k \mid (i,j) \in \mathcal{W}_k} N_k^{\mathsf{T}} N_k (i_k, j_k) \tag{11}$$

where i_k and j_k are the local indices of i and j in $\mathcal{W}_k$. With L, we obtain the least-square energy that represents our image-formation model based on local reflectance planes:

$$F(S) = S^{\mathsf{T}} L\ S \tag{12}$$

[1]Source code is provided as supplemental material

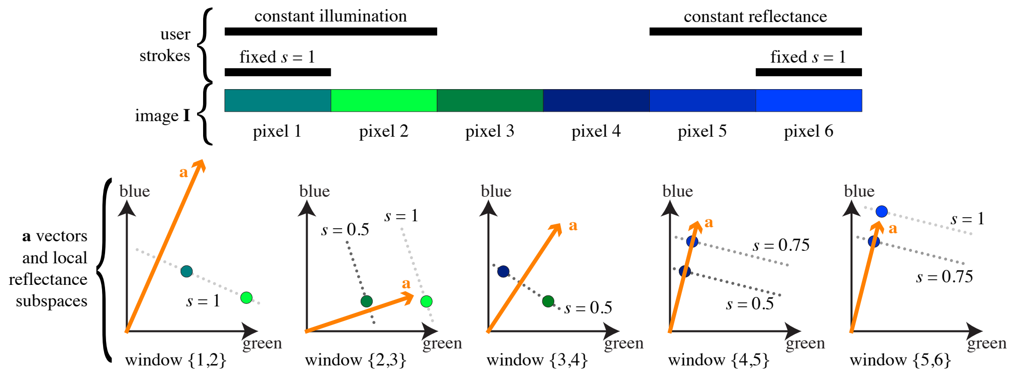


Figure 3: *We illustrate how our optimization scheme behaves on a 1D image. The orange* **a** *vectors define the local reflectance subspaces (lines in this example, planes on normal 2D images). The iso-illumination lines are shown with gray dotted lines. The* **a** *vectors are not explicitly computed by our optimization scheme. They are displayed for visualization purposes. Each window contains 2 pixels. Each pixel belongs to two windows, which constrains the reflectance subspaces. In addition, the user has specified the strokes shown above the pixels. The two end pixels are fixed to $s = 1$, which constrained the isoline $s = 1$ to go through them. The second pixel is constrained to have the same illumination as the first one. This fully defines the isoline $s = 1$ in the first window. In the second window, the isoline $s = 1$ is constrained by the second pixel. Since there is no constraint on the third pixel, the reflectance subspace is defined by the optimization process in order to minimize the least-squares cost. The third window with the pixels 3 and 4 is also fully determined by the optimization. The fourth window is partially constrained by the fifth pixel that has a constrained reflectance which defines its illumination relatively to the sixth pixel. The fifth and last window is fully constrained by the fixed illumination and constant reflectance strokes.*

Discussion By assuming a constant $\mathbf{a}$ vector in each window, we seek an illumination function s that can be locally expressed as a linear combination of the RGB channels. Even though each window has its own $\mathbf{a}$ vector, the choice of these vectors is constrained because the windows overlap. For instance, if we consider a pixel i_0 that belongs to two windows with vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ respectively, the illumination s_{i_0} at i_0 should minimize both $(s_{i_0} - \mathbf{a}_1 \cdot \mathbf{I}_{i_0})^2$ and $(s_{i_0} - \mathbf{a}_2 \cdot \mathbf{I}_{i_0})^2$, which couples the $\mathbf{a}_1$ and $\mathbf{a}_2$ vectors and ensures information propagation across windows (Figure 3).

2.2.1 User Strokes

We propose 3 types of tools so that the user can specify local cues about the reflectance and illumination. In this section we detail how these constraints are integrated in our energy formulation, while a typical interactive session is described in section 4. The first tool is a *constant-reflectance brush* (figure 4(a)). If two pixels share the same reflectance $\mathbf{R}_0$, then $\mathbf{I}_1 = s_1\mathbf{R}_0$ and $\mathbf{I}_2 = s_2\mathbf{R}_0$ that leads to $s_1\mathbf{I}_2 = s_2\mathbf{I}_1$. We define a least-square energy function at a pixel i covered by a reflectance stroke $\mathcal{B}_i^{\mathrm{R}}$:

$$z(\mathcal{B}_i^{\mathrm{R}}) \sum_{j \in \mathcal{B}_i^{\mathrm{R}}} (s_i\mathbf{I}_j - s_j\mathbf{I}_i)^2 \tag{13}$$

where $z(\cdot)$ is a normalization factor that ensures that strokes have an influence independent of their size, $z(\mathcal{B}) = 1/|\mathcal{B}|$ with $|\mathcal{B}|$ the number of pixels in $\mathcal{B}$. For convenience, if there is no stroke on i, we define $\mathcal{B} = \varnothing$ and $z(\mathcal{B}) = 0$. We sum over the image to obtain:

$$U^{\mathrm{R}}(s) = \sum_i z(\mathcal{B}_i^{\mathrm{R}}) \sum_{j \in \mathcal{B}_i^{\mathrm{R}}} (s_i\mathbf{I}_j - s_j\mathbf{I}_i)^2 \tag{14}$$

We also provide a *constant-illumination brush* (figure 4(b)) to the user to indicate regions with constant s values. Between a pair of pixels, this means $s_1 = s_2$, which translates into the energy $(s_1 - s_2)^2$. At the image level, this gives:

$$U^{\mathrm{S}}(s) = \sum_i z(\mathcal{B}_i^{\mathrm{S}}) \sum_{j \in \mathcal{B}_i^{\mathrm{S}}} (s_i - s_j)^2 \tag{15}$$

where $\mathcal{B}_i^{\mathrm{S}}$ is a constant-illumination stroke covering i and z is the same normalization factor as in the previous case.

Finally, we define a *fixed-illumination brush* (figure 4(c)) so that the user can specify absolute illumination values that are used as hard constraints in our optimization. In practice, we use this brush only to indicate fully lit pixels. These areas can be easily recognized by users. We do not use this brush for intermediate illumination values which would be harder to estimate for users. Formally, the brush defines a set $\mathcal{C}$ of pixels which illumination values are fixed, that is for all $i \in \mathcal{C}$, $s_i = \bar{t}_i$ with $\bar{t}_i$ the user-specified value at pixel i. For instance, $\bar{t} = 1$ for fully lit regions.

2.2.2 Constrained Least-square System

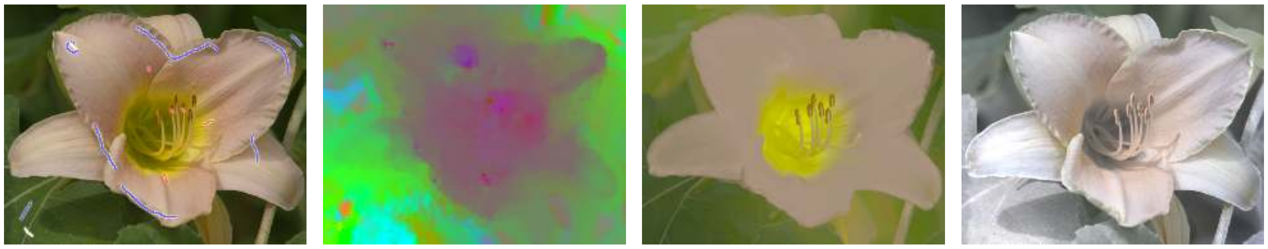
We combine the functional modeling the reflectance subspaces with the user-driven to obtain the following optimization:

$$\arg\min_s \; F(s) + w\left[U^{\mathrm{R}}(s) + U^{\mathrm{S}}(s)\right] \quad \text{such that } \forall i \in \mathcal{C}, s_i = \bar{t}_i \tag{16}$$

where w controls the importance of the strokes. In practice, we use $w = 1$ to give equal importance to our image model and to the user cues, which yields consistently satisfying results. Equation 16 defines a constrained least-square optimization since each term is a quadratic function of s. A minimizer can be obtained using classical linear solvers. For small neighborhoods $\mathcal{W}_i$, the system is sparse. As an example, for 3×3 windows the L matrix (Eq. 12) has only 25 non-zero entries per pixel and the overhead of the brushes is negligible. But for large neighborhoods, L is less sparse and the computation becomes expensive. Section 3 describes a multiscale scheme adapted to our problem.

2.3 Colored Illumination

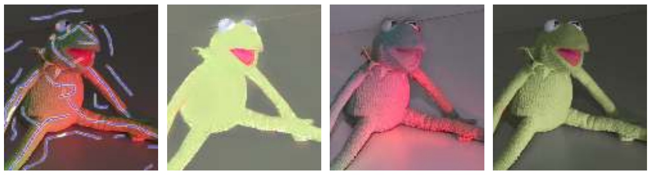
Although our method is derived under the assumption that illumination is monochromatic, that is, s is a scalar, we found that it is able to cope well with colored illumination. In this case, the illumination is a RGB vector $\mathbf{s} = (s^{\mathrm{r}}, s^{\mathrm{g}}, s^{\mathrm{b}})^{\mathsf{T}}$ and our image formation model becomes: $\mathbf{I} = \mathbf{s} * \mathbf{R}$ where $*$ denotes



(a) Input scribbles (b) $\mathbf{a}$ vectors (c) Reflectance (d) Illumination

Figure 4: *(a) The user can specify pixels sharing a constant reflectance (blue), a constant illumination (red), or a fixed illumination (white). Each type of scribble is represented with a single color for illustration purpose, while in our implementation strokes of similar color share the same constraint. (b) Visualization of the* $\mathbf{a}$ *vectors that vary across the image to fit the user constraints. (c,d) Resulting reflectance and illumination.*

per-channel multiplication. We use the previously described optimization method to compute each illumination component separately. The only difference is how we interpret the user strokes. For the channel $c \in \{\mathrm{r}, \mathrm{g}, \mathrm{b}\}$, the constant reflectance energy becomes: $\sum_i z(\mathcal{B}_i^{\mathrm{R}}) \sum_{j \in \mathcal{B}_i^{\mathrm{R}}} \left(\mathbf{s}_i^c I_j^c - s_j^c I_i^c\right)^2$ and the constant illumination energy: $\sum_i z(\mathcal{B}_i^{\mathrm{S}}) \sum_{j \in \mathcal{B}_i^{\mathrm{S}}} \left(s_i^c - s_j^c\right)^2$. Using these new definitions, we minimize Equation 16 for each RGB channel.



(a) User scribbles (b) Reflectance (c) Illumination (d) White balanced

Figure 5: *Our illumination estimation is robust to colored illumination (c). Mapping the estimated colored illumination to gray level and multiplying back with reflectance results in a white balanced image (d).*

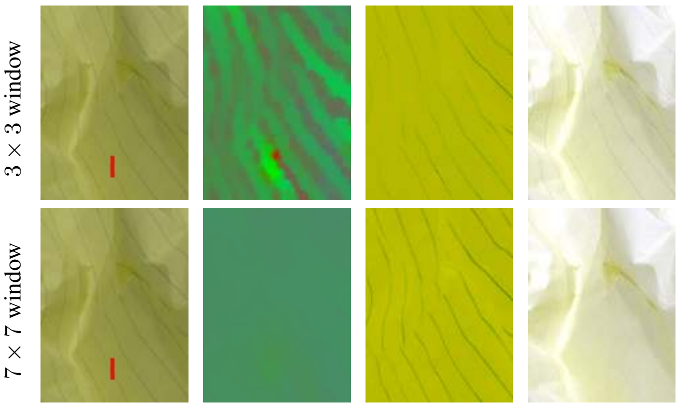
3 Distribution-Preserving Downsampling

The energy formulation described in the previous section relies on color distributions over local windows. However, to capture the color distribution of a textured reflector, the size of the window must be at least as big as the pattern. For example, in Figure 6 the blue lines are constrained as reflectance variations by a user scribble, but are interpreted as illumination variations away from the scribble because a 3×3 window does not cover two lines at a time (first row). 7×7 windows are required in order to capture the blue line texture away from the scribble (second row).

Unfortunately, large windows imply a less sparse least-square system and higher memory and computation costs. This suggests the use of multiresolution approaches such as multigrid, but we have found that standard methods perform poorly in our case. This is because the downsampling they rely on does not respect color distributions. For example, nearest neighbor or even bicubic interpolation can quickly discard colors that do not appear in enough pixels (Figure 8, first and second row). We introduce a new downsampling scheme that does not focus on SSD error but seeks to faithfully preserve the set of colors in a region.

Our new distribution-preserving downsampling is based on two key ideas. First, rather than define the computation of low-resolution pixels one at a time, we consider blocks of low-resolution pixels, because we need multiple pixels to capture a distribution of colors. That is, we downsample high-resolution blocks of $w \times w$ pixels into blocks of $v \times v$ pixels (typically from 4×4 to 2×2). Second, we select the best representatives of the color distribution of a block using clustering.

We integrate this downsampling scheme in a multigrid solver and show that coarse-level results can be upsampled accurately by upsampling the $\mathbf{a}$ vectors rather than the illumination itself. This requires explicitly extracting the per-window $\mathbf{a}$ values, but this is easy given s.



input $\mathbf{I}$ $\mathbf{a}$ vectors reflectance $\mathbf{R}$ (contrast incr.) illumination s

Figure 6: *A* 3×3 *window does not capture the color variations away from the scribble (red pixels). A* 7×7 *window ensures a better propagation of the* $\mathbf{a}$ *vectors modeling the reflectance planes.*

Downsampling Our scheme downsamples large $w \times w$ windows into small $v \times v$ windows such that $w = pv$ with w, v, and p integer numbers. In this configuration, each pixel in the low-resolution image corresponds to $p \times p$ high-resolution pixels. First, we divide the input image into $w \times w$ blocks and extract v^2 representative colors in each block with the K-means algorithm. Then, we rearrange these v^2 pixels into a $v \times v$ block. To preserve image structure, we test all the possible layouts and select the one that minimizes L^2 distance with the original image. The comparison between this downsampling scheme and standard nearest neighbor and linear downsampling is illustrated in Figure 7. We use our downsampled image $\mathbf{I}_{\downarrow}$ to compute illumination and reflectance as described in Section 2.

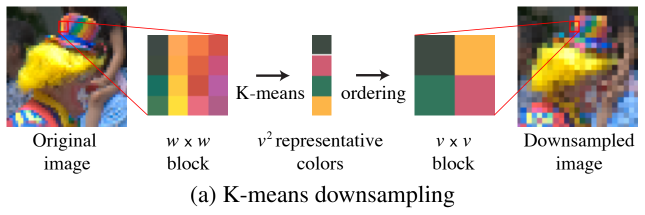


Figure 7: *For a downsampling factor p, we use K-means clustering to extract v^2 representative colors of each $w \times w$ image block. These representative colors are then ordered to form the downsampled image (a). In comparison, nearest neighbor (b) looses the green and gray color information, while linear downsampling (c) produces a loss of contrast.*

Upsampling Full-resolution illumination and reflectance could be obtained with standard upsampling scheme, but generic techniques ignore our image formation model. We leverage the planar reflectance structure to derive a high-quality and computationally efficient upsampling method. First, we use Equation 8 to compute the $\mathbf{a}_\downarrow$ vectors at low resolution. Then, we upsample these vectors using bilinear interpolation to obtain high-resolution $\mathbf{a}_\uparrow$. Finally, we compute the high-resolution illumination $s_\uparrow$ using the reflectance planes defined by $\mathbf{a}_\uparrow$, that is: $s_\uparrow = \mathbf{a}_\uparrow \cdot \mathbf{I}$. Levin et al. [2008] describe a similar approach to speed up their matting algorithm. The high-resolution reflectance $\mathbf{R}_\uparrow$ is then computed as $\mathbf{R}_\uparrow = \mathbf{I}/s_\uparrow$. Figure 8 (third row) illustrates the benefits of this strategy for textured surfaces: 3×3 windows are used on a downsampled input to compute a decomposition that closely match the solution obtained with 7×7 windows at fine resolution. In comparison, nearest neighbor (first row) and linear downsampling (second row) incorrectly include the blue lines in the illumination.

Multigrid Solver To further improve the performances of the linear solver, we apply the above multi-resolution approach recursively, in the spirit of multigrid algorithms [Briggs et al. 2000]. The idea behind multigrid methods is to efficiently solve for the low frequency components in a low resolution version of the domain, and refine the high frequency components at higher resolutions. In practice, we pre-compute a pyramid of the image at multiple scales using our distribution-preserving scheme that reduces 4×4 windows into 2×2 blocks. We compute the corresponding L matrix at each level (Eq. 12). We also cache the results of the K-means downsampling. When the user adds scribbles, we use this information to propagate the constraints in the pyramid at virtually no cost. At run time, we follow a coarse-to-fine strategy. The solution at a given level is upsampled to the next level and refined with a few Gauss-Seidel iterations. We iterate this scheme until we reach the finest resolution. Combining our distribution-preserving downsampling and smart upsampling, even the coarsest level yields a good approximation of the final solution. We use this property to provide early visual feedback to the user by directly upsampling the solution of the first pyramid level using the method described in the previous paragraph. The rest of the pyramid is processed in the background. With our unoptimized C++ implementation on dual-core 3 GHz PC, a 800×600 image requires 20 seconds to precompute the L matrices (this happens only once), visual feedback is available within a second and fully converged result after about 15 seconds. Table 2 details the computation time for the images in this paper.

Since we use a standard least-square approach, one could use an optimized solver such as the GPU-based method of Buatois et al. [2007] or McCann and Pollard [2008] to further speed up the computation.

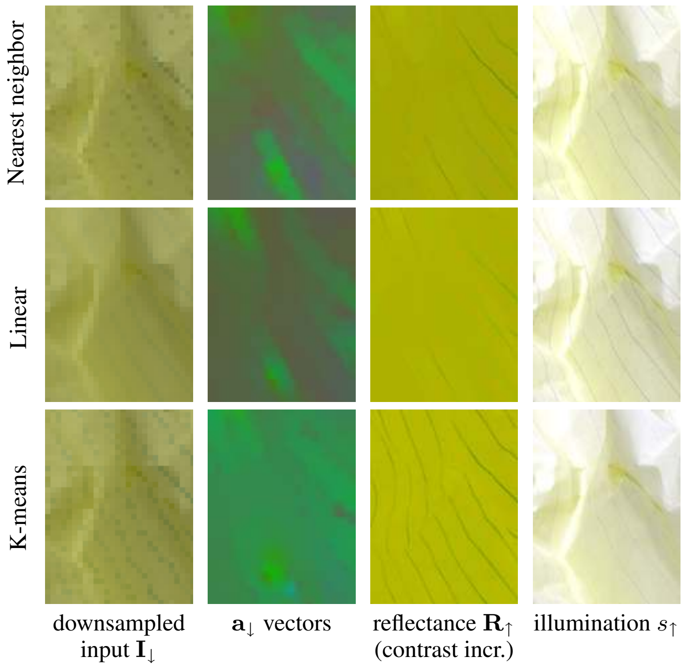

Nearest neighbor
Linear
K-means
downsampled input $\mathbf{I}_\downarrow$
$\mathbf{a}_\downarrow$ vectors
reflectance $\mathbf{R}_\uparrow$ (contrast incr.)
illumination $s_\uparrow$

Figure 8: *Multi-resolution strategy. We approximate large windows at fine resolution as small windows in a coarse version of the image. Nearest neighbor downsampling and linear downsampling fail to preserve the color variations at coarse resolution. Our K-means scheme preserves the blue and yellow color variations, leading to a better approximation of the reflectance plane.*

4 Results and Applications

Inputs The energy derivation described in section 2 assumes linear image values. When dealing with JPEG inputs (Figure 14), we approximate linear images by inverting the gamma, although this inversion can be sensitive to quantization and saturation. For these reasons, most of the results in this paper have been generated from RAW images, that provide linear values. Moreover, the 16 bit depth of RAW images allows a greater accuracy in dark areas of the image when computing reflectance as $\mathbf{R} = \mathbf{I}/s$. This is true for any intrinsic image method.

User Interactions In a typical session, a user starts by specifying a few fixed-illumination strokes in order to fix the global scale ambiguity between bright objects in shadow and dark objects in light. The resulting initial solution can then be iteratively refined by adding constant-reflectance and constant-illumination strokes. In theory, only one fixed-illumination stroke is enough to numerically solve the gobal scale ambiguity. However, if the scene contains disconnected components, for example a subject in front of a distant background, each component needs a fixed stroke since the illumination information cannot be propagated. For example in Figure 4, one fixed-illumination scribble is required for the pink flower, and one for the green leaves because these two regions form different connected components that do not share a similar reflectance or a similar illumination.

The fixed-illumination brush is often applied over the brightest points of objects, which are easy to identify for a human oberver. An inaccurate fixed illumination value only introduces a global scaling factor over connected color regions, which still produces a plausible result. The constant-reflectance and constant-illumination strokes are most effective when applied to regions with complementary variations, e.g. using the constant-illumination brush across reflectance edges often significantly improves the result. Similarly, the constant-reflectance brush is effective when applied over interreflections and other colored lighting variations. Our video illustrates these intermediate steps.

Figure 9 illustrates how the per-pixel error evolves with the number of strokes, computed from ground truth data with two sets of strokes scribbled by different users. Because no method exists to obtain ground truth decompositions from color photographs, we created a synthetic scene inspired by Figure 1 and computed a global illumination solution (Figure 11). Typically, the fixed-illumination strokes drastically reduce the error by fixing global ambiguities, while the constant-reflectance and illumination strokes correct small, but visually important, local errors. The two different sets of strokes quickly converge to decompositions that differ only by a global scaling factor.

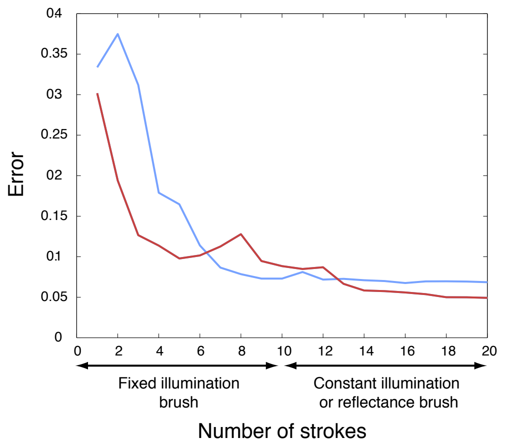


Figure 9: *Per-pixel error computed from a ground truth synthetic image, for two different sets of scribbles. Fixed illumination scribbles quickly produce a good estimate (10 first strokes), while constant illumination or reflectance scribbles are used to refine the result. The remaining error is due to the global scaling factor that has been over or under estimated.*

Our approach is robust to small variations in scribble placement and value, which makes it easy to use. To assess this, we have computed several results where we have randomly moved the scribbles up to 15 pixels and randomly changed the fixed illumination values up to 5%. All the results look equally plausible with objects appearing slightly brighter or darker, and remain usable in graphics applications (see results in supplemental materials). Table 1 reports the average per-pixel error for various scribble alterations on the synthetic image of Figure 11. While the amount of user scribbles required is comparable to other approaches [Levin et al. 2004; Levin et al. 2008], an important difference is that, in our approach, most of the scribbles do not require the users to specify numerical values. Instead, user only indicates similarity between reflectance or illumination, which is more intuitive to draw. Table 2 details the number of strokes for the examples in this paper.

Scribbles	Per-pixel error (%)
Fixed-illumination values set to ground truth values	2.28
Fixed-illumination values set to 1	5.98
Fixed-illumination values set to ground truth values, position randomly altered up to 15 pixels	7.12
Fixed-illumination values set to ground truth values randomly altered up to 5%	5.19

Table 1: *Average per-pixel error computed from a ground truth synthetic image with randomly altered scribbles.*

	Res.	User strokes	Matrix (s)	Solving (s)
Baby (fig. 1)	533 × 800	58	20.64	11.93
Flower (fig. 4)	750 × 500	15	18.21	7.87
Clown (fig. 10)	486 × 800	33	18.83	9.53
St Basile (fig. 14)	800 × 600	81	23.29	14.56
Paper (fig. 14)	750 × 500	36	18.17	9.2

Table 2: *Resolution, number of scribbles and computation time for matrix pre-computation and solving for the results of this paper. Note that we use the coarse level of the pyramid to display an approximate solution after 1 second (cf. text for detail).*

Intrinsic Decompositions Figure 10 illustrates the intrinsic image decomposition that our method produces for a colorful photograph. In comparison, a luminance computation produces different values for light and dark colors. We compare our approach with Tappen et al.'s [2005] algorithm in Figure 14 and in our supplementary materials. This method combines a Retinex classifier based on chromaticity variations and a Bayesian classifier trained on graylevel images. The result of these classifiers is a binary map that labels the image gradients as reflectance or illumination. The final reflectance and illumination images are reconstructed using a Poisson integration on each color channel. The main limitation of this automatic approach is that a binary labelling cannot handle areas where both reflectance and illumination variations occur, such as in highly textured areas, along occlusion boundaries or under mixed lighting conditions. In Figure 11 we show a similar comparison on ground truth data from a synthetic image. The information specified by the user together with our propagation algorithm allow us to extract fine reflectance and illumination, but our planar reflectance assumption prevents us from considering the black pixels of the eyes as reflectance. Because Tappen's method is automatic, there is a remaining scale ambiguity over the reflectance and illumination after Poisson reconstruction. We fix the brightest point of the illumination to a value of 1 in Figure 11(c) but the estimated illumination is still darker than the ground truth. The refletance in Figure 11(c) is not uniform because occlusion edges that are classified as reflectance also contain illumination variations.

Figure 16 compares our method with the automatic approach of Shen et al. [2008]. While their texture constraints greatly improve the standard Retinex algorithm, posterization artifacts are visible in the reflectance image due to the clustering that imposes that pixels of the same texture receive the same reflectance. Increasing the number of clusters reduces posterization but also reduces the benefit of the texture constraints. As any automatic approach, the result cannot be corrected in case of failure, such as in the St Basile image where different regions of the sky receive different reflectance. Moreover, their method cannot handle colored illumination.

Reflectance Editing One of the simplest manipulations offered by intrinsic images is editing one of the image component (reflectance or illumination) independently from the other. Figure 1(e) and 12 give examples of reflectance editing inspired by the retexturing approach of Fang and Hart [2004]. We use a similar normal-from-shading algorithm to estimate the normals of the ob-

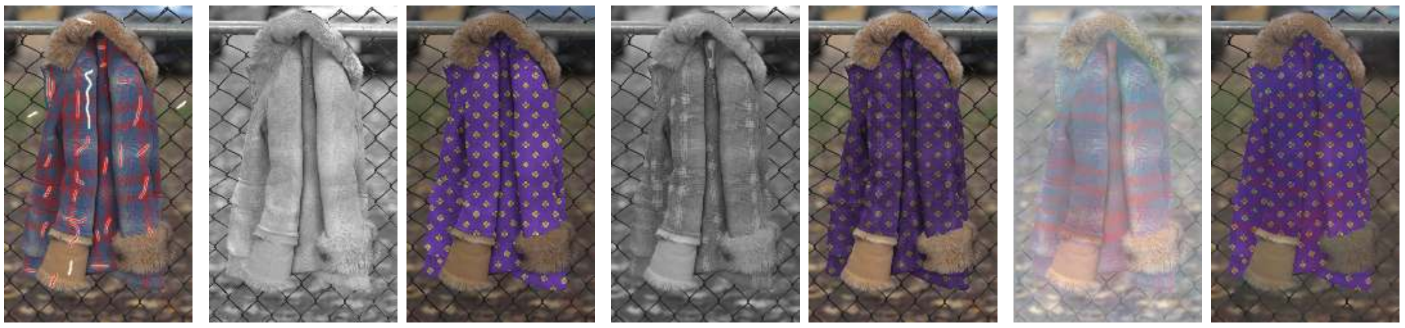


Figure 12: *Our approach (b) produces illumination maps that are more accurate than luminance (c) for re-texturing textured objects. On this highly textured image, the automatic classifier of Tappen et al.[2005] cannot decompose reflectance and illumination properly (d), which impacts the result of the manipulation.*

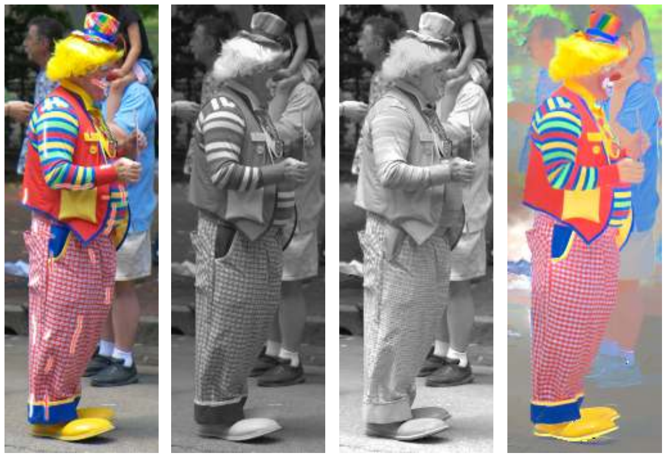


Figure 10: *While a naive luminance computation produces lower values for dark colors (b), our approach provides a convincing estimation of illumination (c) and reflectance(d).*

jects. Textures are then mapped in the reflectance image and displaced according to the normal map. We finally multiply the edited reflectance image by the illumination image to obtain a convincing integration of the new textures in the scene. While Fang and Hart obtained similar results using the luminance channel of the image, our illumination represents a more accurate input for their algorithm and other image based material editing methods [Khan et al. 2005] when dealing with textured objects (Figure 12).

Relighting Once reflectance and illumination have been separated, the scene can be relighted with a new illumination map. Figure 13 illustrates this concept with a simple yet effective manipulation. Starting from a daylight picture, we invert the illumination image to create a nighttime picture. Although this simple operation is not physically accurate, we found that it produces convincing results on architectural scenes because the areas that do not face the sun in the daylight image are the ones that are usually lit by night (interiors and surfaces oriented to the ground). We refine the result by adding a fake moon and mapping the gray level illumination to an orange to purple color ramp.

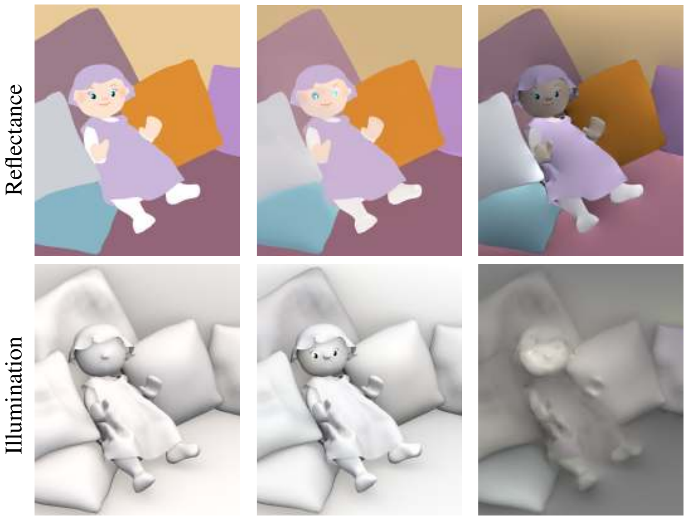
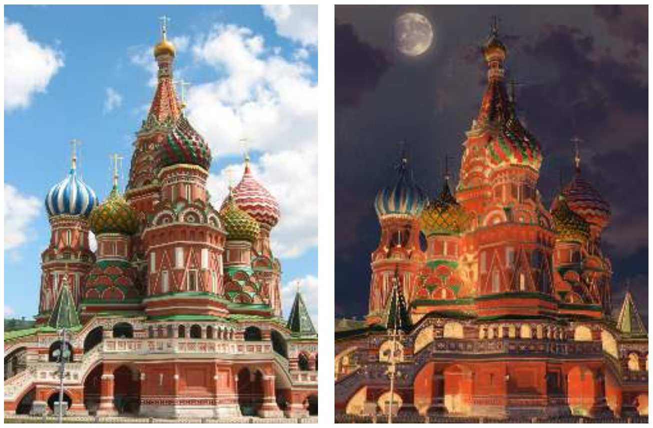

Reflectance
Illumination
(a) Ground truth
(b) Our method from a single image and user scribbles
(c) Tappen's method from a single image

Figure 11: *Comparison with ground truth data from a synthetic image. Compared to the automatic method of Tappen et al.[2005] (c), our user assisted approach produces finner results, but interprets the black pixels of the eyes as shadow (b).*

Discussion Although we found that our method works well in most cases, we acknowledge that similar to many inverse problems, the results can be sensitive to the input quality. JPEG artifacts and noise can become visible if one applies an extreme transformation to the illumination or reflectance, especially since JPEG compresses color information aggressively. For instance, inverting the illumination as in Figure 13 can reveal speckles on strongly compressed images. Over-exposure and color saturation are other sources of difficulty since information is lost. For instance, the clown in Figure 10 contains out-of-gamut colors that have been clipped by the camera. As a consequence, some residual texture can be discerned in the illumination component. Existing methods share all of these difficulties that are inherent to the intrinsic image decomposition problem. In Figure 15, we compare our result with Weiss' technique [2001]. Even with 40 images taken with a tripod and controlled lighting, illumination variations remain visible in the reflectance component, which is the opposite bias of our method.



(a) Original image (b) Nighttime relighting

Figure 13: *From a daylight picture (a), we invert the illumination and add a fake moon to create a nighttime picture (b). Note however that the process reveals saturation and blocky artifacts from the JPEG compression. Original image by Captain Chaos, flickr.com.*

5 Conclusions

We have described a method to extract intrinsic images based on user-provided scribbles. It relies on a low-rank assumption about the local reflectance distribution in photographs. This allows us to compute the illumination as the minimizer of a linear least-square functional. We have also presented a new downsampling scheme based on local clustering that focuses on preserving color distributions. Our method can handle complex natural images and can be used for applications such as relighting and re-texturing.

Acknowledgments

We thank Laurence Boissieux for creating the synthetic scene (Figure 11), and Sara Su for recording the voice-over. We are grateful to Marshall Tappen for sharing his code and to Li Shen and Ping Tan for running their algorithm on our images. Finally, we thank the anonymous reviewers and the MIT/ARTIS pre-reviewers for their constructive feedback and comments. This work was partially funded by an NSF CAREER award 0447561, by the MIT-Quanta T Party and by the INRIA Associate Research Team "Flexible Rendering". F. Durand acknowledges a Microsoft Research New Faculty Fellowship and a Sloan Fellowship.

References

AGRAWAL, A., RASKAR, R., AND CHELLAPPA, R. 2006. Edge suppression by gradient field transformation using cross-projection tensors. In *CVPR*, 2301–2308.

BARROW, H., AND TENENBAUM, J. 1978. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*.

BRIGGS, W. L., HENSON, V. E., AND MCCORMICK, S. F. 2000. *A multigrid tutorial (2nd ed.)*. Society for Industrial and Applied Mathematics.

BUATOIS, L., CAUMON, G., AND LÉVY, B. 2007. Concurrent number cruncher: An efficient sparse linear solver on the gpu. In *High Performance Computation Conference*.

CHUANG, Y.-Y., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2001. A bayesian approach to digital matting. In *CVPR*.

FANG, H., AND HART, J. C. 2004. Textureshop: Texture synthesis as a photograph editing tool. *ACM TOG (proc. of SIGGRAPH 2004) 23*, 3, 354–359.

FATTAL, R. 2008. Single image dehazing. *ACM TOG (proc. of SIGGRAPH 2008) 27*, 3, 72.

FINLAYSON, G. D., HORDLEY, S. D., AND DREW, M. S. 2002. Removing shadows from images. In *ECCV*.

FINLAYSON, G. D., DREW, M. S., AND LU, C. 2004. Intrinsic images by entropy minimization. In *ECCV*, 582–595.

HORN, B. K. 1986. *Robot Vision*. MIT Press.

HSU, E., MERTENS, T., PARIS, S., AVIDAN, S., AND DURAND, F. 2008. Light mixture estimation for spatially varying white balance. *ACM TOG (proc. of SIGGRAPH 2008) 27*, 3, 70.

KHAN, E., REINHARD, E., FLEMING, R., AND BÜLTHOFF, H. 2005. Image-based material editing. *ACM TOG (proc. of SIGGRAPH 2005) 24*, 3, 654 – 663.

LAND, E. H., AND MCCANN, J. J. 1971. Lightness and retinex theory. *Journal of the optical society of America 61*, 1.

LEVIN, A., AND WEISS, Y. 2007. User assisted separation of reflections from a single image using a sparsity prior. *IEEE Trans. PAMI 29*, 9, 1647–1654.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2004. Colorization using optimization. *ACM TOG (proc. of SIGGRAPH 2004) 23*, 689 – 694.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2008. A closed-form solution to natural image matting. *IEEE Trans. PAMI*.

LIU, X., WAN, L., QU, Y., WONG, T.-T., LIN, S., LEUNG, C.-S., AND HENG, P.-A. 2008. Intrinsic colorization. *ACM TOG (proc. of SIGGRAPH Asia 2008) 27*, 5, 152.

MCCANN, J., AND POLLARD, N. S. 2008. Real-time gradient-domain painting. *ACM TOG (Proc. of SIGGRAPH) 27*, 3, 93.

MOHAN, A., TUMBLIN, J., AND CHOUDHURY, P. 2007. Editing soft shadows in a digital photograph. *IEEE Computer Graphics and Applications 27*, 2, 23–31.

OMER, I., AND WERMAN, M. 2004. Color lines: Image specific color representation. In *CVPR*, 946–953.

SHEN, L., TAN, P., AND LIN, S. 2008. Intrinsic image decomposition with non-local texture cues. In *CVPR*.

SHOR, Y., AND LISCHINSKI, D. 2008. The shadow meets the mask: Pyramid-based shadow removal. *Computer Graphics Forum (Proc. of Eurographics) 27*, 3.

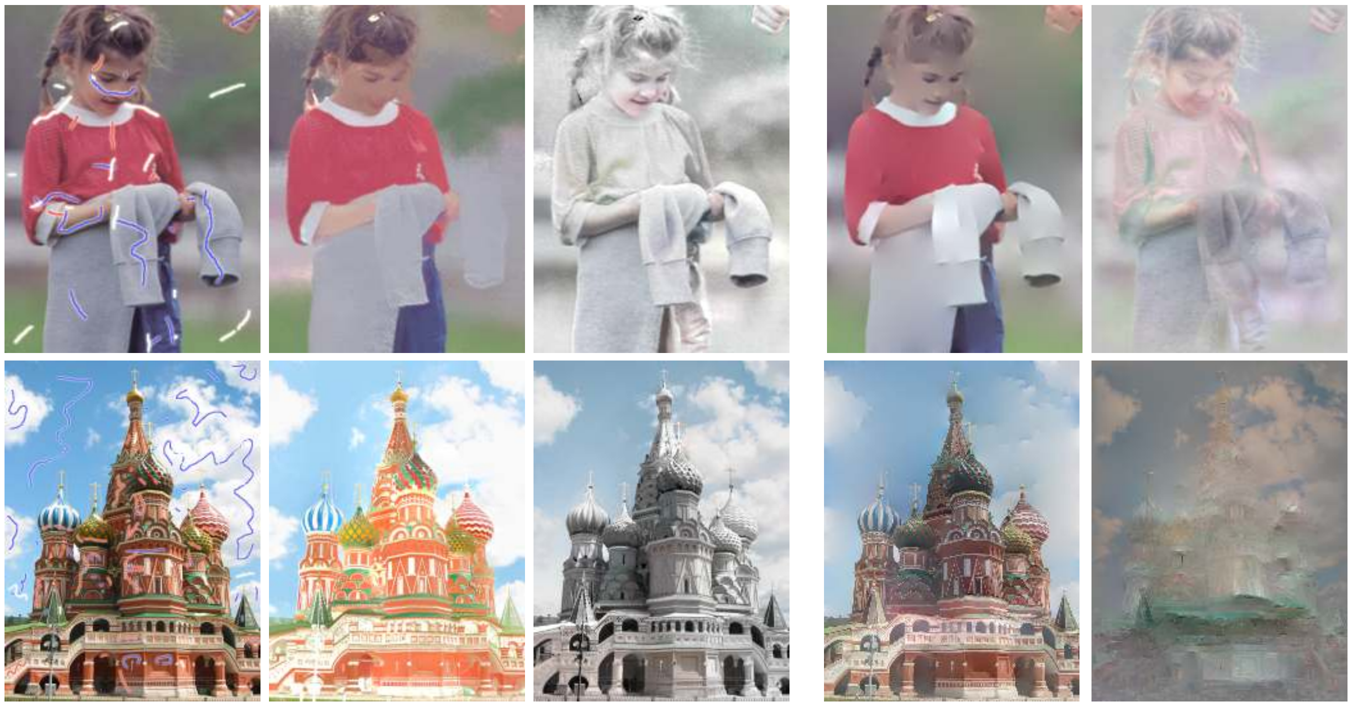
SINHA, P., AND ADELSON, E. 1993. Recovering reflectance and illumination in a world of painted polyhedra. In *ICCV*, 156–163.

TAPPEN, M. F., FREEMAN, W. T., AND ADELSON, E. H. 2005. Recovering intrinsic images from a single image. *IEEE Trans. PAMI 27*, 9.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *ICCV*, 68–75.

WU, T.-P., TANG, C.-K., BROWN, M. S., AND SHUM, H.-Y. 2007. Natural shadow matting. *ACM TOG 26*, 2, 8.

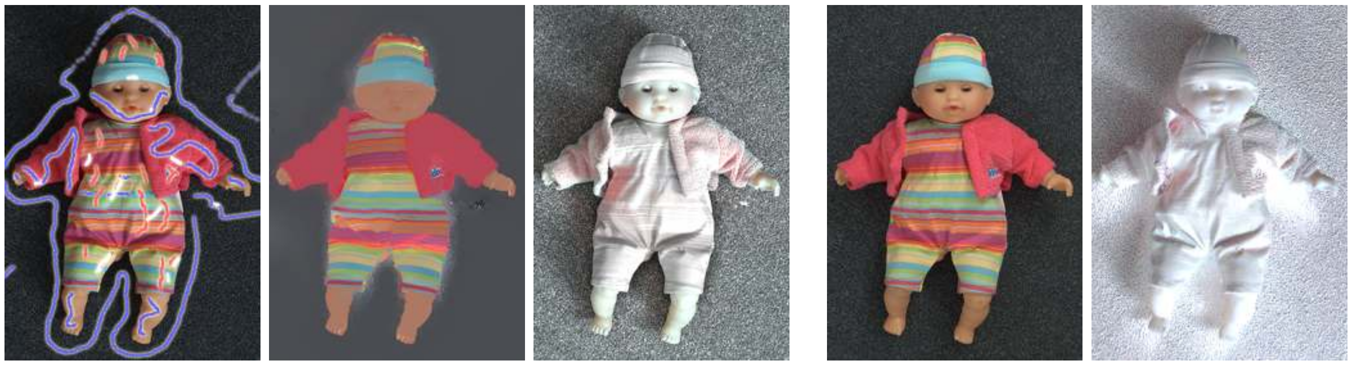
YU, Y., AND MALIK, J. 1998. Recovering photometric properties of architectural scenes from photographs. In *ACM SIGGRAPH 98*, 207–217.


(a) User scribbles

(b) Our reflectance and illumination from a single image and user scribbles

(c) Tappen's reflectance and illumination from a single image

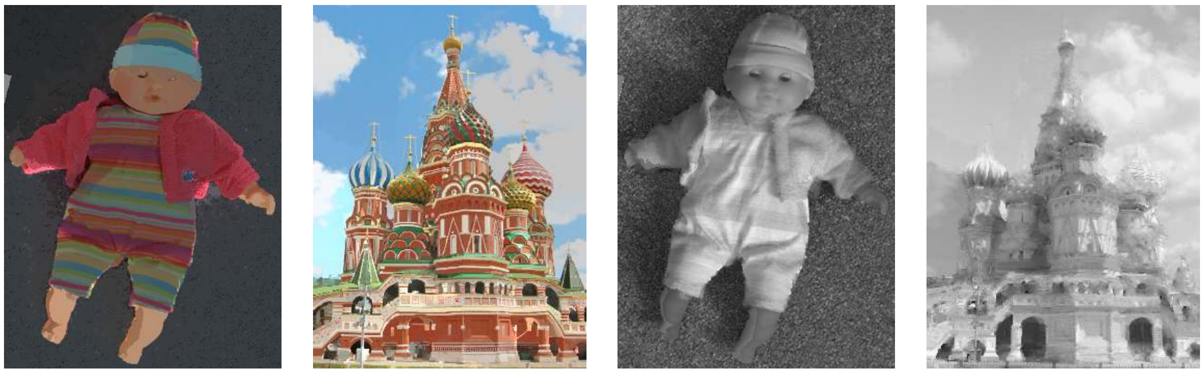
Figure 14: *Comparison with the automatic approach of Tappen et al.[2005]. St Basile Cathedral image by Captain Chaos, flickr.com.*


(a) User scribbles

(b) Our reflectance and illumination from a single image and user scribbles

(c) Weiss' reflectance and illumination from 40 images

Figure 15: *Comparison with the multiple image approach of Weiss[2001].*


(a) Shen's reflectance from a single image

(b) Shen's illumination from a single image

Figure 16: *Comparison with the automatic approach of Shen et al.[2008]. St Basile Cathedral image by Captain Chaos, flickr.com.*