Color-Constant Information Embedding

Fan Wang[1] and Roberto Manduchi[2]

[1] Stanford University, Stanford CA 94305, USA
fanw@stanford.edu
[2] University of California, Santa Cruz, Santa Cruz CA 95064, USA
manduchi@soe.ucsc.edu

Abstract. We propose a technique to embed information in the color of a printed surface. One or more reference surfaces are used to help compensate for the color changes due to varying illuminants. Seven different techniques, some of which are novel, are considered for color compensation. Experiments using different performance metrics are presented, providing a comparative assessment of the various algorithms and highlighting the importance of the correct choice of reference surfaces.

1 Introduction

There is a growing interest in technology to embed information in printed matter, in such a way that can be easily accessed by mobile devices such as cell phones. For example, 1-D and 2-D barcodes are often placed in advertisements for products or events, providing a means for anyone with a cell phone equipped with a camera, suitable software, and Internet access, to retrieve more information about the specific product or event. The density of information that can be embedded with this type of markers is limited by the camera resolution and viewing conditions. For example, low illumination requires large exposure time (resulting in motion blur) or high camera gain (resulting in noise). Also, limited depth of field requires precise focusing on the marker, which is sometime problematic with cell phone cameras (for those cell phones that indeed have focusing capabilities).

A simple strategy for adding more information to a marker of a given size is through the use of color. For example, Microsoft's High Capacity Color Barcode (HCCB) technology [1] uses 2-D barcodes enhanced with 4 different colors, resulting in a tremendous increase in the amount of information that can be embedded. By embedding information in a surface's reflectance signature, the spatial density of bars in a marker can be reduced, resulting in better readability from a distance, especially with mobile devices.

Unfortunately, cameras do not take direct reflectance measurements. The color measured by a camera is a function of the reflectance characteristics of the surface as well as of the spectra of the illuminant(s), and of the illumination and viewing geometry. One may partly reduce the influence of these variables by printing markers on Lambertian (opaque) surfaces, but the dependence of the measured color on the illuminant spectrum remains a major impediment to using a large

K.N. Kutulakos (Ed.): ECCV 2010 Workshops, Part II, LNCS 6554, pp. 13–26, 2012.

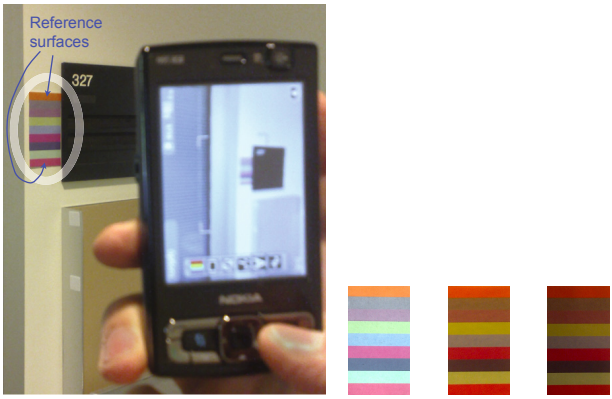

Fig. 1. Left: An example of application of the proposed technique. A marker with 9 color patches (two of which are used for reference) is taken by a cell phone camera. Each color patch contains 7 bits of information. Right: crop-outs from images of the same marker under different illuminants exemplify the dramatic color variations undergone by the patches.

number of colors, as two distinct surfaces may have the same color when lit by different illuminants.

Color constancy is a well known problem, and many partial solutions have been proposed. Our considered application, though, is different from most existing work: rather than normalizing for the unknown illuminant in a general, unconstrained scene, we *design* the marker so as to simplify color constancy. In particular, we add one or more *reference surfaces* next to the unknown, information-carrying surfaces. The color characteristics of the reference surfaces under different illuminants are assumed to be well known (by means of training samples). The idea is that the color of the reference surfaces should give enough information about the illuminant to enable color-constant measurement of the unknown surfaces.

The use of reference surfaces is not new: it is a standard practice in remote sensing, and even regular cameras implement white balancing based on the color of a white surface. Normally, colors are compensated after observation of the reference surface using a diagonal transformation. Although this may be suitable for narrowband channels in multispectral systems, diagonal color compensation is known to be suboptimal for the color matching functions of typical cameras. This is a problem for the quantitative analysis of colors. Thus, in this work we consider several other color compensation algorithm besides the diagonal transformation.

Identifying the unknown, information-carrying surface can be formally expressed as either an *indexing* or a *regression* problem. In the first case, the goal is to determine which among a set of possible surface types best represents the surface in the image. In the second case, we attempt to undo the effect of the specific illuminant, by "rendering" the color of the surface as if seen under a canonical illuminant. In summary, our problem can be expressed as follows:

Problem: Consider an image with R reference surfaces placed coplanar with and close to the "unknown" surface under analysis. All surfaces are Lambertian, and receive light from the same illuminant.
(1) Find the index of the unknown surface within a set of possible surfaces.
(2) Estimate the color of the unknown surface as seen under the canonical illuminant.

We note that the second formulation (regression) is more general in that it addresses the possibility of embedding information via surfaces that are not part of a known data set.

In this contribution, we consider a number of algorithms (some of which are well known, while some are novel) to solve this problem, and present comparative experimental results measuring the accuracy and error using a number R of reference surfaces from 1 to 3.

2 Previous Work

We denote the color of a surface s under illuminant i by $I^{(i)}_{(s)} = [I^{(i)}_{(s),1}, I^{(i)}_{(s),2}, I^{(i)}_{(s),3}]^T$, where $I^{(i)}_{(s),k}$ represents the k-th color channel. A popular model to represent the dependence of the color of the surface to the illuminant is based on the assumption that the spectra of all possible illuminants and of all possible surface reflectances form finite dimensional spaces (of dimension M_i and M_s respectively). For Lambertian (opaque) surfaces, this results in the following bilinear form:

$$I^{(i)}_{(s),k} = \alpha^{(i)T} Q_k \beta_{(s)} \tag{1}$$

where Q_k is an $M_i \times M_s$ matrix with positive entries that only depends on the camera; $\alpha^{(i)}$ only depends on the illuminant and the incidence angle; and $\beta_{(s)}$ only depends on the surface. In particular $\alpha^{(i)}$ and $\beta_{(s)}$ are independent of the color channel k. Let us define

$$\Phi_{(s)} = \left[Q_1\beta_{(s)} | Q_2\beta_{(s)} | Q_3\beta_{(s)}\right] \tag{2}$$

$$\Psi^{(i)} = \left[Q_1^T\alpha^{(i)} | Q_2^T\alpha^{(i)} | Q_3^T\alpha^{(i)}\right]$$

Note that knowledge of $\Phi_{(s)}$ enables *rendering* of the color of the surface s under any illuminant. Tsin *et al.* [2] showed that the matrices $\Phi_{(s)}$ can be estimated using SVD from a collection of images of surfaces under multiple illuminants. A similar idea was presented by Sunkavalli *et al.* [3].

Of special interest are rendering models that use color vectors directly, rather than resorting to an intermediate representation (e.g., the matrices $\Phi_{(s)}$ above). It is well known [4] that if $M_s = 3$, then $I^{(i_2)}_{(s)} = A^{(i_1)\to(i_2)} I^{(i_1)}_{(s)}$, where $A^{(i_1)\to(i_2)} = ((\Psi^{(i_1)})^{-1}\Psi^{i_2})^T$, and we assumed that $\Psi^{(i_1)}$ is full-rank. For a given pair of illuminants, the 3×3 matrix $A^{(i_1)\to(i_2)}$ can be estimated by observation of at least three different surface patches under both illuminants. If the dimension of the reflectance spectra space is further restricted to $M_s = 2$, then it can be shown

that the color transformation matrix $A^{(i_1)\to(i_2)}$ can be obtained by measuring the color change of at least two different surfaces.

Finlayson *et al.* [4] showed that, if $M_i = 3$ and $M_s = 2$ (the so–called 3–2 case), only one surface, seen under two different illuminants, is actually needed to estimate $A^{(i_1)\to(i_2)}$. The idea is to perform a suitable change of basis (*spectral sharpening*) in color space: $\hat{I}^{(i)}_{(s)} = TI^{(i)}_{(s)}$ where T is an invertible 3×3 matrix. The columns of T can be chosen as the eigenvectors of the matrix $((\Phi_{(s_1)})^{-1}\Phi_{s_2})^T$, where s_1 and s_2 represent any two different surfaces. It is shown in [4] that in the 3–2 case, the following rendering model applies:

$$\hat{I}^{(i_2)}_{(s)} = D^{(i_1)\to(i_2)}\hat{I}^{(i_1)}_{(s)} \tag{3}$$

where $D^{(i_1)\to(i_2)}$ is diagonal. Following [4], we will call this a *generalized diagonal* model. Given any two transformed colors $\hat{I}^{(i_1)}_{(s)}$ and $\hat{I}^{(i_2)}_{(s)}$ of the same surface observed under two different illuminants, the k–th diagonal entry of $D_{i_1\to I_2}$ can be computed as $\hat{I}^{(i_2)}_{(s),k}/\hat{I}^{(i_1)}_{(s),k}$.

A simpler version of the previous rendering model, the *diagonal* model, assumes that (3) holds even without the basis change induced by T.

$$I^{(i_2)}_{(s)} = D^{(i_1)\to(i_2)}I^{(i_1)}_{(s)} \tag{4}$$

It is not difficult to show that this assumption holds true only if each matrix Q_k has only one non–null column.

A different, fully data-driven approach to color constancy was proposed by Miller and Tieu [5]. This algorithm is based on the observation of joint color changes in images due to variation in lighting and other non-geometric camera parameters.

Our stated goal to *index* surfaces in a marker based on the measured color is reminiscent of other color-based indexing techniques (e.g., [6,7]). In contrast with these previous approaches, which focus on the general problem of object recognition in unconstrained environments, our work considers a very precise domain, with contextual information available in the form of reference surfaces.

3 Algorithms

In general, an algorithm to solve the problem stated in the Introduction has three components:

Illuminant estimation: Using the reference surfaces, a representation of the illuminant is obtained.

Surface estimation: A representation of the unknown surface is obtained. This step typically uses the representation of the illuminant found in the previous step.

Rendering/Indexing: The color of the unknown surface is rendered under the canonical illuminant, and the index of the unknown surface in the training data

set is found. Indexing is only feasible when the unknown surface is assumed to belong to the training data set.

Below we summarize the algorithms tested in our experiments. The original contributions of this work are marked by the letters **[O.C.]**.

The unknown illuminant (under which the surface is seen) and the canonical illuminant will be denoted by superscripts 'u' and 'c' respectively, while the unknown surface will be denoted by subscript 'u'. Thus, the color of the unknown surface in the image (under unknown illuminant) is $I_{(u)}^{(u)}$, while its color rendered under the canonical illuminant is $I_{(u)}^{(c)}$. We assume that images of a training set of N_s surfaces (which include the references surfaces as well as the unknown surface) under a number N_i of different illuminants (which include the canonical illuminant) have been captured off-line. The color values of the training data set are collected in the following matrices:

$$J_{(s)} = \left[I_{(s)}^{(1)}|I_{(s)}^{(2)}|\dots|I_{(s)}^{(N_i)}\right]^T \ , \ 1 \le s \le N_s \tag{5}$$

$$Y^{(i)} = \left[I_{(1)}^{(i)}|I_{(2)}^{(i)}|\dots|I_{(N_s)}^{(i)}\right]^T \ , \ 1 \le i \le N_i$$

$$J = \left[J_{(1)}|J_{(2)}|\dots|J_{(N_s)}\right] \ , \ Y = \left[Y^{(1)}|Y^{(2)}|\dots|Y^{(N_i)}\right]$$

We also define:

$$\Phi = \left[\Phi_{(1)}|\Phi_{(2)}|\dots|\Phi_{(N_s)}\right], \Psi = \left[\Psi^{(1)}|\Psi^{(2)}|\dots|\Psi^{(N_i)}\right] \tag{6}$$

$$A = [\alpha^{(1)}|\alpha^{(2)}|\dots|\alpha^{(N_i)}]^T \ , \ B = [\beta_{(1)}|\beta_{(2)}|\dots|\beta_{(N_s)}]^T \tag{7}$$

where the matrices $\Phi_{(s)}$ and $\Psi^{(i)}$ were defined in (2). The bilinear model (1) implies that:

$$J_{(s)} = A\Phi_{(s)} \ , \ Y^{(i)} = B\Psi^{(i)} \ , \ J = A\Phi \ , \ Y = B\Psi \tag{8}$$

Without loss of generality, we assume that the reference surfaces have indices in $[1, 2, \dots, R]$, and therefore their colors under the unknown illuminant are $I_{(1)}^{(u)}, I_{(2)}^{(u)}, \dots, I_{(R)}^{(u)}$.

Algorithm 1: Diagonal Model

Illuminant estimation: The illuminant is characterized by the diagonal rendering matrix $D^{(u)\to(c)}$. To find this matrix, we solve the three least-squares problems:

$$D_{[k]}^{(u)\to(c)} = \arg\min_a \sum_{s=1}^{R} \left(I_{(s),k}^{(c)} - aI_{(s),k}^{(u)}\right)^2 \tag{9}$$

where $D_{[k]}^{(u)\to(c)}$ is the (k,k) entry of the matrix.

Rendering/Indexing: $I_{(u)}^{(c)} = D^{(u)\to(c)} I_{(u)}^{(u)}$. Then indexing is accomplished by finding the surface with the closest color to $I_{(u)}^{(c)}$among the training surface seen under the canonical illuminant:

$$s_u = \underset{1\le s\le N_s}{\arg\min} \|I_{(u)}^{(c)} - I_{(s)}^{(c)}\|^2 \tag{10}$$

Note that this algorithm does not require the "surface estimation" step.

Algorithm 2: Generalized Diagonal Model

This algorithm is identical to Algorithm 1, except that the color vectors are pre-multiplied by a "spectral sharpening" matrix T.
[O.C.] We propose a novel approach to derive a suitable "spectral sharpening" matrix T. We begin by observing that a good matrix T is such that $TY^{(c)}$ is well approximated by $D^{(i)\to(c)}Y^{(i)}$ for all illuminants i in the training data set, where $D^{(i)\to(c)}$ are suitable diagonal matrices. It is easily seen that this is identical to the problem of approximating $Y^{(c)}Y^{(i)\dagger}$ (where $Y^{(i)\dagger}$ is the pseudo-inverse of $Y^{(i)}$) with $T^{-1}D^{(i)\to(c)}T$. Thus, the problem is one of finding the matrix T that produces the best approximate joint diagonalization of the 3×3 matrices $Y^{(c)}Y^{(i)^\dagger}$. In our experiments, we solved the joint diagonalization problem using the publicly available Matlab function `rjd.m`, developed by Jean-Francois Cardoso, which is based on a Jacobi-like iterative technique [8].

Algorithm 3: Surface-to-Surface Diagonal Model [O.C.]

This algorithm is based on the simple observation that, under the same hypothesis in which the diagonal model (4) holds true, the following diagonal relationship holds between the colors of two surfaces s_1, s_2 seen under the same illuminant:

$$I_{(s_2)}^{(i)} = D_{(s_1)\to(s_2)} I_{(s_1)}^{(i)} \tag{11}$$

where $D_{(s_1)\to(s_2)}$ is independent of the illuminant. The proof is trivial, and relies on the fact that diagonal matrices commute.

Surface estimation: The unknown surface is represented by the set of R diagonal matrices $D_{(s)\to(u)}$ mapping the colors $I_{(s)}^{(u)}$ to $I_{(u)}^{(u)}$, where $D_{(s)\to(u),k} = I_{(u),k}^{(u)}/I_{(s),k}^{(u)}$. These matrices represent the relationship between the unknown surface and the reference surfaces in the training data set when seen under the same illuminant.

Rendering/Indexing: The matrices $D_{(s)\to(u)}$ are used to map the color of each reference surface $I_{(s)}^{(c)}$ in the training data set, seen under the canonical illuminant, to a prediction of the color of the unknown surface under the canonical illuminant. These predictions are then averaged together. In formulas:

$$I_{(u)}^{(c)} = \sum_{s=1}^{R} D_{(s)\to(u)} I_{(s)}^{(c)} / R \tag{12}$$

Indexing is accomplished as by (10). Note that this algorithm does not require the "illuminant estimation" step.

Algorithm 4: Surface-to-Surface Multiple Diagonal Model [O.C.]

Illuminant estimation: Rather than considering just the canonical illuminant, as in Algorithms 1 and 2, we look at all illuminants i in the training data set and compute all diagonal matrices $D^{(i)\to(u)}$ mapping the color of any surface under illuminant i to the color of the same surface under the unknown illuminant:

$$D_{[k]}^{(i)\to(u)} = \arg\min_{a} \sum_{s=1}^{R} \left(I_{(s),k}^{(u)} - a I_{(s),k}^{(i)} \right)^2 \tag{13}$$

Surface estimation: We first render the colors of all surfaces in the data set, as seen under the unknown illuminant:

$$I_{(s)}^{(u)} = \sum_{i=1}^{N_i} D^{(i)\to(u)} I_{(s)}^{(i)} / N_i \tag{14}$$

As in Algorithm 3, the unknown surface is represented by the set of all diagonal matrices $D_{(s)\to(u)}$ mapping the colors $I_{(s)}^{(u)}$ to $I_{(u)}^{(u)}$, where $D_{(s)\to(u),k} = I_{(u),k}^{(u)} / I_{(s),k}^{(u)}$. However, with respect to Algorithm 3, now there are N_s such diagonal matrices.

Rendering/Indexing: This part is identical to Algorithm 3, except that the average prediction $I_{(u)}^{(c)}$ is computed using all N_s images in the data set.

Algorithm 5: Bilinear Model

Let us define $\gamma^{(u)} = (A^\dagger)^T \alpha^{(u)}$ and $\delta_{(u)} = (B^\dagger)^T \beta_{(u)}$, where A, B were defined in (6) and $\alpha^{(u)}$, $\beta_{(u)}$ were defined in (1). Then one easily sees that, as long as $\text{rank}(J) \geq M_i$ and $\text{rank}(Y) \geq M_s$: $I_{(s)}^{(u)} = J_{(s)}^T \gamma^{(u)}$, $I_{(u)}^{(i)} = Y^{(i)T} \delta_{(u)}$. This formulation allows us to express a surface color as a linear function of the training data.

Illuminant estimation: Define $J_{\text{ref}} = \left[J_{(1)} | J_{(2)} | \ldots | J_{(R)} \right]$. The unknown illuminant is represented by the vector $\gamma^{(u)}$, where $\gamma^{(u)} = \left(\left[I_{(1)}^{(u)T} | I_{(2)}^{(u)T} | \ldots | I_{(R)}^{(u)T} \right] J_{\text{ref}}^\dagger \right)^T$.

Surface estimation: First render all surfaces in the data set, as seen by the unknown illuminant:

$$Y^{(u)} = [J_{(1)}^T \gamma^{(u)} | J_{(2)}^T \gamma^{(u)} | \ldots | J_{(N_s)}^T \gamma^{(u)}]^T \tag{15}$$

The unknown surface is represented by the vector $\delta_{(u)}$, computed as follows: $\delta_{(u)} = \left(I_{(u)}^{(u)T} Y^{(u)\dagger} \right)^T$.

Rendering/Indexing: $I_{(u)}^{(c)} = Y^{(c)T} \delta_{(u)}$. Indexing is accomplished as by (10).

Algorithm 6: Bilinear Model (Compact) [O.C.]

A problem with Algorithm 5 is that it requires storage and processing of a conspicuous portion of the training data at run time. In fact, storage requirements and computation would be much lighter if the matrices $\{Q_k\}$ in (1) were available. We introduce in the following an algorithm to estimate matrices $\{\tilde{Q}_k\}$ that are algebraically similar to $\{Q_k\}$. To our knowledge, this is the first time that a direct estimation of these matrices has been proposed.

Let us define: $J_{(s),k} = \left[I^{(1)}_{(s),k}, I^{(2)}_{(s),k}, \ldots, I^{(N_i)}_{(s),k}\right]^T$ and $J_k = \left[J_{(1),k}|J_{(2),k}|\ldots|J_{(N_s),k}\right]$. We observe that $J_k = AQ_kB$ for $1 \leq k \leq 3$. Our algorithm for finding matrices $\{\tilde{Q}_k\}$ that are similar to $\{Q_k\}$ proceeds as follows. We first compute $\tilde{Q}_k(1) = A^\dagger J_k B^\dagger$ for $1 \leq k \leq 3$. Then we iterate the following two steps:

Step 1: For $1 \leq s \leq N_s$, $1 \leq i \leq N_i$, compute:

$$\tilde{\Phi}_{(s)}(n) = \left[\tilde{Q}_1(n)\beta_{(s)}|\tilde{Q}_2(n)\beta_{(s)}|\tilde{Q}_3(n)\beta_{(s)}\right] \tag{16}$$

$$\tilde{\Phi}(n) = \left[\tilde{\Phi}_{(1)}(n)|\tilde{\Phi}_{(2)}(n)|\ldots|\tilde{\Phi}_{(N_s)}(n)\right]$$

$$\tilde{\Psi}^{(i)}(n) = \left[\tilde{Q}_1^T(n)\alpha_{(i)}|\tilde{Q}_2^T(n)\alpha_{(i)}|\tilde{Q}_3^T(n)\alpha_{(i)}\right]$$

$$\tilde{\Psi}(n) = \left[\tilde{\Psi}^{(1)}(n)|\tilde{\Psi}^{(2)}(n)|\ldots|\tilde{\Psi}_{(N_i)}(n)\right]$$

$$\tilde{A}(n) = J\tilde{\Phi}^\dagger(n)\ ,\ \tilde{B}(n) = Y\tilde{\Psi}^\dagger(n)$$

Step 2: Compute $\tilde{Q}_k(n+1) = \tilde{A}^\dagger(n) J_k \tilde{B}^\dagger(n)$ $(1 \leq k \leq 3)$.

It is easy to see that at each iteration, the Frobenius norm $e_F(n)$ of the error matrix $J - A\tilde{\Phi}(n)$ can only decrease or remain the same. Step 1 and 2 are iterated until the $e_F(n)/e_F(n-1)$ is lower than a fixed threshold. Note that this algorithm is performed off-line with training data. Once the matrices $\{\tilde{Q}_k\}$ and $\{\tilde{\Phi}_{(s)}\}$ have been computed, on-line color analysis is performed as follows.

Illuminant estimation: Define $\tilde{\Phi}_{\text{ref}} = \left[\tilde{\Phi}_{(1)}|\tilde{\Phi}_{(2)}|\ldots|\tilde{\Phi}_{(R)}\right]$. The unknown illuminant is represented by the vector $\alpha^{(u)}$, computed as follows:

$$\alpha^{(u)} = \left(\left[I^{(u)T}_{(1)}|I^{(u)T}_{(2)}|\ldots|I^{(u)T}_{(R)}\right]\Phi^\dagger_{\text{ref}}\right)^T \tag{17}$$

Surface estimation: The unknown surface is represented by the vector $\beta_{(u)}$, computed as follows:

$$\beta_{(u)} = \left(I^{(u)T}_{(u)}\left[\tilde{Q}_1^T\alpha^{(u)}|\tilde{Q}_2^T\alpha^{(u)}|\tilde{Q}_3^T\alpha^{(u)}\right]^\dagger\right)^T \tag{18}$$

Rendering/Index.: $I^{(c)}_{(u)} = \left[\tilde{Q}_1\beta_{(u)}|\tilde{Q}_2\beta_{(u)}|\tilde{Q}_3\beta_{(u)}\right]^T \alpha^{(c)}$ where $\alpha^{(c)}$ is computed off-line. Indexing is accomplished as by (10).

Note that the matrices $\{\tilde{Q}_k\}$ have dimension $M_i \times M_s$. Appropriate values M_i and M_s can be estimated via eigenvalue analysis. In our experiments, we used $M_i = 12$ and $M_s = 9$.

Algorithm 7: Nearest Neighbor [O.C.]

This algorithm is fully data-driven and doesn't use a model of color formation.

Illuminant estimation: The unknown illuminant is represented by the index i^u defined by $i^u = \underset{1\le i\le N_i}{\arg\min} \sum_{s=1}^{R} \|I^{(u)}_{(s)} - I^{(i)}_{(s)}\|^2$.

Surface estimation: The unknown surface is represented by the index s_u defined by $s_u = \underset{1\le s\le N_s}{\arg\min} \|I^{(u)}_{(u)} - I^{(i^u)}_{(s)}\|^2$.

Rendering/Indexing: The color of the unknown surface is rendered as seen under the canonical illuminant by $I^{(c)}_{(s_u)}$. The index of the unknown surface is s_u.

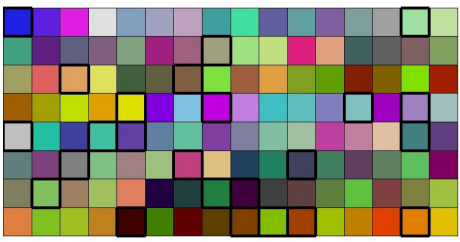

Fig. 2. The $N_s = 128$ surface patches used in the experiments. The 24 patches used for reference are shown with a thick border.

4 Experiments

In order to test our algorithms, we printed a color checkerboard with 512 colors, uniformly sampled in (R,G,B) space. Some of the printed colors were practically indistinguishable from each other, so we selected 128 colors by greedy iterative exclusion from an image of the checkerboard taken by a camera. More precisely, we selected the two patches whose colors (in the image) were most similar, and removed one of them. This operation was repeated until only 128 colors were left. The selected colors are shown in Fig. 2.

Images of the checkerboard were taken under 75 different illumination conditions (one of which is selected as the canonical illuminant). The illumination conditions considered included direct sunlight, diffuse natural light under different overcast conditions, and various types of artificial light (incandescence lamps, neon light, etc.) Two different cameras were used. One was a low-quality

camera from a cell phone (Sony Ericsson W580i with white balance set to a fixed value). The other camera was a high-end Canon EOS 350D, producing images in raw (CR2) format. In each image, the color values within a patch (typically covering a few hundred pixels) were averaged together to reduce noise.

The reference colors are chosen from a sample of 24 colors, selected from the 128 colors using k-means. The selected colors are shown with a black border in Fig. 2. Note that this set contains two gray patches. For a given number R, with $1 \leq R \leq 3$, we use all combinations of these 24 colors taken R at a time as reference surface sets. For each choice of reference patch set, all algorithms were tested over each one of the remaining $128-R$ patches which thus represented the "unknown" patch. In each test, one illuminant is selected, and each unknown patch is analyzed based on the color of the reference patches under the same illuminant.

We devised a cross-validation procedure with a sequence of ten rounds. In each round, half of the illuminants were selected at random to represent the "training" illuminants, meaning that the algorithms were trained based on the color of all surfaces as seen under such illuminants. The algorithms were then tested with each one of the remaining ("test") illuminants. For each test illuminant i chosen in each cross-validation round r, and for each choice of the reference patch set P, we define by *prediction error rate* $E_{\mathrm{p}}^{(i,r,P)}$ the number of unknown patches that were incorrectly indexed by the algorithm, divided by the number of patches considered (in our experiments, 128), and by *rendering error* $E_{\mathrm{r}}^{(i,r,P)}$ the mean (over all considered patches) of the Euclidean error between the rendered color of the unknown patch under canonical illuminant and its correct value. We then compute the average prediction and rendering error ($E_{\mathrm{p,av}}^{(P)}$, $E_{\mathrm{r,av}}^{(P)}$), along with the maximum prediction and rendering error ($E_{\mathrm{p,max}}^{(P)}$, $E_{\mathrm{r,max}}^{(P)}$), over all rounds and all illuminants in each round. Note that these values are a function of the choice of reference patch set P. Finally, for each metric considered, we select the reference patch set that minimizes the corresponding error, obtaining $E_{\mathrm{p,av}}$, $E_{\mathrm{r,av}}$, $E_{\mathrm{p,max}}$, or $E_{\mathrm{r,max}}$. We believe that both maximum (worst case) and average errors are important for performance assessment. The results of our experiments are shown in Figs. 3 and 4, along with the optimal reference patches for each metric considered.

4.1 Discussion

When comparing the different algorithms, it is important to bear in mind both performance and computational cost. In terms of implementation, a look-up table could be used to store all possible colors of the reference patches and produce the index of the unknown patch or its color under canonical illumination. Using B bits to represent each color, this table would occupy $2^{RB} \log_2 N_s$ bits of memory for the color indices, and $2^{RB}B$ for the rendered colors. For example, with a 8-bit camera, using $N_s = 128$ as in our experiments and $R = 3$ reference patches, the index table can be stored in 15 Mbytes, while the rendered color table requires 17 Mbytes. This amount of storage is durable even in a hand-held

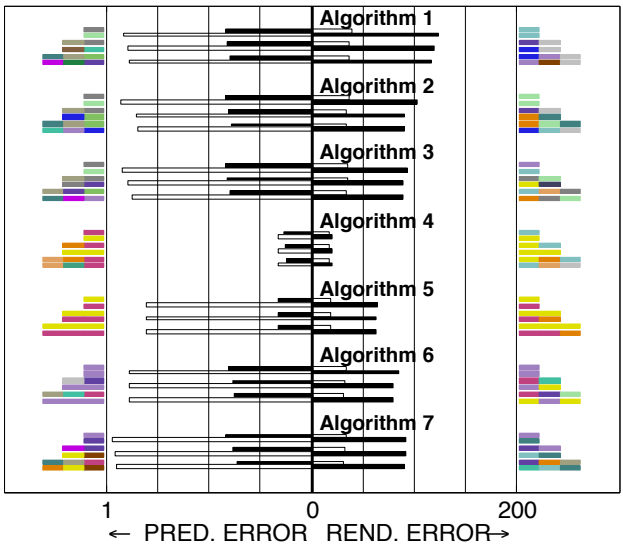

Fig. 3. Results from the experiments using the cell phone camera. For each algorithm, we show results using a number R from 1 to 3 of reference patches. The optimal choice of reference patches for each illuminant and each metric considered is also shown. The left part of the plot shows $E_{\mathrm{p,max}}$ (white bars) and $E_{\mathrm{p,ave}}$ (black bars). The right part shows $E_{\mathrm{r,max}}$ (black bars) and $E_{\mathrm{r,av}}$ (white bars).

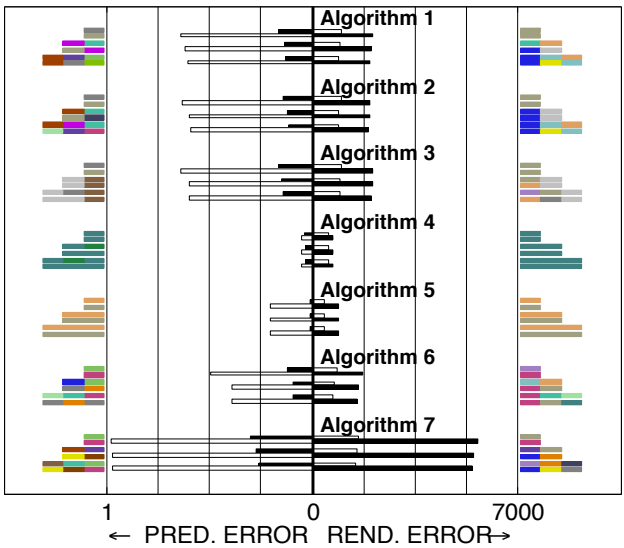

Fig. 4. Results from the experiments using the high-end camera. See caption of Fig. 3

device such as a cell phone. For a larger pixel depth, feasibility of a look-up table depends on the number of reference patches being used.

If online computation is necessary, the fastest techniques are Algorithms 1, 2 and 3, which have negligible complexity. Algorithm 4 and 7 have complexity that is linear with $N_s N_i$. The bulk of complexity of Algorithm 5 is in the computation of the pseudoinverse of a matrix of size $N_s \times 3$ (remember that $N_s = 128$ in our experiments). Algorithm 6 requires computation of the pseudoinverse of a matrix of size $M_s \times 3$ where, as mentioned earlier, M_s was set to 9 in our experiments.

As expected, the tests with the high-end camera produced better accuracy. Note that the the images with this camera had 16 bits per pixel, which explains the higher error values recorded (since each color channels take values from 0 to $2^{16} - 1$). The finer quantization, and a presumably more linear sensor response than the cell phone camera, are likely to account for the better performance of this camera. Note in passing that we have not considered photometric camera calibration, a technique that could improve the quality of the results.

Comparative analysis of the algorithms show that the Algorithms 1, 2 and 3 give similar performances. As expected, the "sharpening" technique of Algorithm 2 improves on the simple diagonal model of Algorithm 1, although only by a small amount. Note that the worst-case scenario prediction error rate $E_{\mathrm{p,max}}$ for these algorithms is above 0.5 for both camera. This means that, even with the best choice of reference patches, there exists an illuminant such that half of the patches are misclassified.

The best performing technique is clearly Algorithm 4, which provides the smallest error under all metrics. The worst case prediction error rate $E_{\mathrm{p,max}}$ is equal to 0.16 using three reference patches for the camera cellphone case, and 0.03 for the high-end camera. The worst-case rendering error $E_{\mathrm{r,max}}$ is equal to 20 for the 8-bit camera and 720 for the 16-bit camera.

The bilinear model (Algorithm 5) gives good results in terms of average errors, but the worst-case prediction error rates are much higher. Its "compact" version (Algorithm 6) gives worse results, likely due to the some inaccuracy in the computation of the matrices $\{Q_k\}$. However, note that Algorithm 6 still performs better than the diagonal models algorithms 1–3, at least for the high-end camera, and has lower complexity than the bilinear model (Algorithm 5). The data-driven approach (Algorithm 7) gives unsatisfactory results overall.

In general, increasing the number of reference patches improves performance, but this improvement is surprisingly small. In fact, in very few cases, adding one more reference patch does not reduce the error at all. In Figs. 3 and 4, these situations lead to some colors being repeated in the optimal reference patch set.

In order to highlight the dependence of the system's performance on the choice of a reference patch, we show in Fig. 5 the values of $E_{\mathrm{r,av}}$ for Algorithm 4 for $R = 1$, as a function of the chosen reference patch (all 128 possible patches are considered here, rather than just the 24 shown in Fig. 2) and for a chosen set of training illuminants. Note that the best reference patches have low color saturation values. Indeed, the 20% best performing patches have median saturation equal to 0.5, while the 20% worst performing patches have median

saturation equal to 1 (where 'saturation' is defined according to the HSV color space). This confirms the intuitive notion that colors with a broad spectrum are preferable to colors with a narrow spectrum for this application. However, our experiments also shows that white or gray are not necessarily the best reference colors.

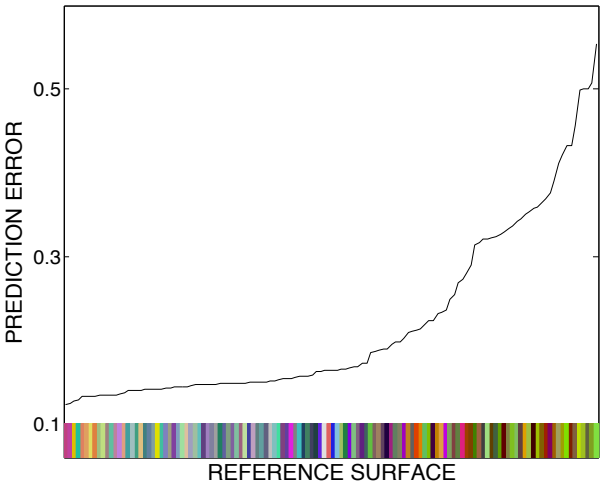

Fig. 5. Values of $E_{\mathrm{p,av}}$ for Algorithm 4 with $R = 1$ as a function of the chosen reference patch (images taken by the cell phone camera).

5 Conclusions

Embedding information in printed color is a promising technique, but special care is required to solve the "color constancy" problem. We have presented experimental results comparing seven different algorithms that use one or more reference surfaces to undo the effect of the unknown illuminant. The results, in terms of accuracy and rendering error, are, in our opinion, encouraging. In particular, one of the original algorithms proposed here (Algorithm 4) has excellent accuracy.

Of course, there are other issues besides color constancy that may hinder this approach. For example, printed colors may fade with time unless specialized inks and substrate are used. Noise is also an important factor, especially when the color marker is seen from a distance (resulting in a small foreshortened area) or with low illumination. Still, we believe that this technology has serious potential as a practical means for information embedding, by itself or in conjunction with other techniques such as barcodes.

Acknowledgement. This material is based upon work supported by the National Science Foundation under Grant No. IIS - 0835645.

References

1. Parikh, D., Jancke, G.: Localization and segmentation of a 2D high capacity color barcode. In: Proceedings of the 2008 IEEE Workshop on Applications of Computer Vision (WACV 2008), pp. 1–6. IEEE Computer Society, Washington, DC (2008)
2. Tsin, Y., Collins, R.T., Ramesh, V., Kanade, T.: Bayesian color constancy for outdoor object recognition. In: IEEE Computer Vision and Pattern Recognition (CVPR), pp. 1132–1139 (2001)
3. Sunkavalli, K., Romeiro, F., Matusik, W., Zickler, T., Pfister, H.: What do color changes reveal about an outdoor scene? In: Proc. IEEE CVPR (2008)
4. Finlayson, G., Drew, M., Funt, B.: Diagonal transforms suffice for color constancy. In: Proceedings of Fourth International Conference on Computer Vision, pp. 164–171 (1993)
5. Miller, E., Tieu, K.: Color eigenflows: Statistical modeling of joint color changes. In: Proc. of International Conference on Computer Vision (ICCV), vol. 1, pp. 607–614 (2001)
6. Swain, M.J., Ballard, D.H.: Color indexing. International Journal of Computer Vision 7, 11–32 (1991)
7. Funt, B.V., Finlayson, G.D.: Color constant color indexing. IEEE Transactions on Pattern Analysis and Machine Intelligence 17, 522–529 (1995)
8. Cardoso, J.F., Souloumiac, A.: Jacobi angles for simultaneous diagonalization. SIAM J. Matrix Anal. Appl. 17, 161–164 (1996)