Published in Proceedings of the IEEE International Conference on Computer Vision 2001

Color Eigenflows[*]: Statistical Modeling of Joint Color Changes

Erik G. Miller and Kinh Tieu
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

We develop a linear model of commonly observed *joint color changes* in images due to variation in lighting and certain non-geometric camera parameters. This is done by observing how all of the colors are mapped between two images of the same scene under various "real-world" lighting changes. We represent each instance of such a joint color mapping as a 3-D vector field in RGB color space. We show that the variance in these maps is well represented by a low-dimensional linear subspace of these vector fields. We dub the principal components of this space the *color eigenflows*. When applied to a new image, the maps define an image subspace (different for each new image) of plausible variations of the image as seen under a wide variety of naturally observed lighting conditions. We examine the ability of the eigenflows and a base image to reconstruct a second image taken under different lighting conditions, showing our technique to be superior to other methods. Setting a threshold on this reconstruction error gives a simple system for scene recognition.

1. Introduction

The number of possible images of an object or scene, even when taken from a single viewpoint with a fixed camera, is very large. Light sources, shadows, camera aperture, exposure time, transducer non-linearities, and camera processing (such as auto-gain-control and color balancing) can all affect the final image of a scene [5]. Humans seem to have no trouble at all compensating for these effects when they occur in small or moderate amounts. However, these effects have a significant impact on the digital images obtained with cameras and hence on image processing algorithms, often hampering or eliminating our ability to produce reliable recognition algorithms.

Addressing the variability of images due to these *photic parameters* has been an important problem in machine vision. We distinguish photic parameters from *geometric parameters*, such as camera orientation or blurring, that affect which parts of the scene a particular pixel represents. We also note that photic parameters are more general than "lighting parameters" that would typically only refer to light sources and shadowing. We include in photic parameters anything which affects the final RGB values in an image given that the geometric parameters and the objects in the scene have been fixed.

In this paper, we address the problem of whether two given images are digital photographs of the same scene under different photic parameter settings, or whether they are different physical scenes altogether. To do this, we develop a statistical linear model of *color change space*, by observing how the colors in static images change under naturally occurring lighting changes. This model describes how colors change *jointly* under typical (statistically common) photic parameter changes. Then, given a pair of images, we ask whether we can describe the difference between the two images, to within some tolerance, using the statistical color change model. If so, we conclude that the images are probably of the same scene.

Several aspects of our model merit discussion. First, it is obtained from video data in a completely unsupervised fashion. The model uses no prior knowledge of lighting conditions, surface reflectances, or other parameters during data collection and modeling. It also has no built-in knowledge of the physics of image acquisition or "typical" image color changes, such as brightness changes. It is completely data driven. Second, it is a single global model. That is, it does not need to be re-estimated for new objects or scenes. While it may not apply to all scenes equally well, it is a model of frequently occurring joint color changes, which is meant to apply to all scenes. Third, while our model is linear in *color change space*, each joint color change that we model (a 3-D vector field) is completely arbitrary, and is not itself restricted to being linear. That is, we define a linear space whose basis elements are *vector fields* that represent nonlinear color changes. This gives us great modeling power, while capacity is controlled through the number of basis fields allowed.

After discussing previous work in Section 2, we describe the form of the statistical model and how it is obtained

[*]For a color version of this paper, please see http://www.ai.mit.edu/people/emiller/color_flows.pdf.

from observations in Section 3. In Section 4, we show how our color change model and a single observed image can be used to generate a large family of related images. We also give an efficient procedure for finding the best fit of the model to the difference between two images, allowing us to determine how much of the difference between the images can be explained by typical joint color changes. In Section 5 we give results and comparisons with 3 other methods.

2. Previous Work

The color constancy literature contains a large body of work for estimating surface reflectances and various photic parameters from images. A common approach is to use linear models of reflectance and illuminant spectra (often, the illuminant matrix absorbs an assumed linear camera transfer function) [9]. A surface reflectance (as a function of wavelength λ) can be written as $S(\lambda) \approx \sum_{i=1}^{n} \alpha_i S_i(\lambda)$. Similarly, illuminants can be represented with a fixed basis as $E(\lambda) \approx \sum_{i=1}^{m} \beta_i E_i(\lambda)$. The basis functions for these models can be estimated, for example, by performing PCA on color measurements with known surface reflectances or illuminants. Given a large enough set of camera responses or RGB values, the surface reflectance coefficients can be recovered by solving a set of linear equations if the illuminant is known, again assuming no other non-linearities in the image formation.

A variety of algorithms have been developed to estimate the illuminant from a single image. This can be done if some part of the image has a known surface reflectance. Making strong assumptions about the distribution of reflectances in a typical image leads to two simple methods. Gray world algorithms [2] assume that the average reflectance of all the surfaces in a scene is gray. White world algorithms [10] assume that the brightest pixel corresponds to a scene point with maximal reflectance.

Some researchers have redefined the problem to one of finding the relative illuminant (a mapping of colors under an unknown illuminant to a canonical one). Color gamut mapping [4] models an illuminant using a “canonical gamut” or convex hull of all achievable image RGB values under the illuminant. Each pixel in an image under an unknown illuminant may require a separate mapping to move it within the “canonical gamut”. Since each such mapping defines a convex hull, the intersection of all such hulls may provide enough constraints to specify a “best” mapping. [3] trained a multi-layer neural network using backpropagation to estimate the parameters of a linear color mapping. The method was shown to outperform simpler methods such as gray/white world algorithms when trained and tested on artificially generated scenes from a database of surface reflectances and illuminants. A third approach by [6] works in the log color spectra space. In this space, the effect of a relative illuminant is a set of constant shifts in the scalar coefficients of linear models for the image colors and illuminant. The shifts are computed as differences between the modes of the distribution of coefficients of randomly selected pixels of some set of representative colors.

Note that in these approaches, illumination is assumed to be constant across the image plane. The mapping of RGB values from an unknown illuminant to a canonical one is assumed to be linear in color space. A diagonal linear operator is commonly used to adjust each of the R, G, and B channels independently. Not surprisingly, the gray world and white world assumptions are often violated. Moreover, a purely linear mapping will not adequately model non-linear variations such as camera auto-gain-control.

[1] bypasses the need to predict specific scene properties by proving statements about the sets of all images of a particular object as certain conditions change. They show that the set of images of a gray Lambertian convex object under all lighting conditions form a convex cone[1]. Only three non-degenerate samples from this cone are required to generate the set of images from this space. Nowhere in this process do they need to explicitly calculate surface angles or reflectances.

One aspect of this approach that we hoped to improve upon was the need to use several examples (in this case, 3) to apply the geometry of the analysis to a particular scene. That is, we wanted a model which, based upon a single image, could make useful predictions about other images of the same scene.

We present a paper in the same spirit, although it is a statistical method rather than a geometric one. Our main goal is: *given two different images, we wish to accept or reject the hypothesis that the two images are of the same scene, but taken under different lighting and imaging conditions*. We view this problem as substantially easier than the problem of estimating surface reflectances or illuminants.

3. Color Flows

In the following, let $\mathcal{C} = \{(r,g,b)^T \in \mathbb{R}^3 : 0 \leq r \leq 255, 0 \leq g \leq 255, 0 \leq b \leq 255\}$ be the set of all possible observable image color 3-vectors. Let the vector-valued color of an image pixel p be denoted by $\mathbf{c}(p) \in \mathcal{C}$.

Suppose we are given two N-pixel RGB color images I_1 and I_2 of the same scene taken under two different sets of photic parameters θ_1 and θ_2 (the images are registered). Each pair of corresponding image pixels p_1^k and $p_2^k, 1 \leq k \leq N$, in the two images represents a mapping $\mathbf{c}(p_1^k) \mapsto \mathbf{c}(p_2^k)$. That is, it tells us how a particular pixel's

[1]This result depends upon the important assumption that the camera, including the transducers, the aperture, and the lens introduce no non-linearities into the system. The authors' results on color images also do not address the issue of metamers, and assume that light is composed of only the wavelengths red, green, and blue.

color changed from image I_1 to image I_2. This single-color mapping is conveniently represented simply by the vector difference between the two pixel colors:

$$\mathbf{d}(p_1^k, p_2^k) = \mathbf{c}(p_2^k) - \mathbf{c}(p_1^k). \tag{1}$$

By computing N of these vector differences (one for each pair of pixels) and placing each vector difference at the point $\mathbf{c}(p_1^k)$ in the color space $\mathcal{C}$, we have created a vector field that is defined at all points in $\mathcal{C}$ for which there are colors in image I_1.

That is, we are defining a vector field Φ' over $\mathcal{C}$ via

$$\Phi'(\mathbf{c}(p_1^k)) = \mathbf{d}(p_1^k, p_2^k), \qquad 1 \leq k \leq N. \tag{2}$$

This can be visualized as a collection of N arrows in color space, each arrow going from a source color to a destination color based on the photic parameter change $\theta_1 \mapsto \theta_2$. We call this vector field Φ' a *partially observed color flow*. The "partially observed" indicates that the vector field is only defined at the particular color points that happen to be in image I_1.

To obtain a *full color flow*, i.e. a vector field Φ defined at all points in $\mathcal{C}$, from a partially observed color flow Φ', we must address two issues. First, there will be many points in $\mathcal{C}$ at which no vector difference is defined. Second, there may be multiple pixels of a particular color in image I_1 that are mapped to different colors in image I_2. We propose the following interpolation scheme[2], which defines the flow at a color point $(r, g, b)^T$ by computing a weighted proximity-based average of nearby observed "flow vectors":

$$\Phi(r, g, b) = \frac{\sum_{k=1}^{N} e^{-\|\mathbf{c}(p_1^k)-(r,g,b)^T\|^2/2\sigma^2} \Phi'(\mathbf{c}(p_1^k))}{\sum_{k=1}^{N} e^{-\|\mathbf{c}(p_1^k)-(r,g,b)^T\|^2/2\sigma^2}}. \tag{3}$$

This defines a color flow vector at every point in $\mathcal{C}$. Note that the Euclidean distance function used is defined in *color space*, not in the space defined by the [x,y] coordinates of the image. σ^2 is a variance term which controls the mixing of observed flow vectors to form the interpolated flow vector. As $\sigma^2 \rightarrow 0$, the interpolation scheme degenerates to a nearest-neighbor scheme, and as $\sigma^2 \rightarrow \infty$, all flow vectors get set to the average observed flow vector. In our experiments, we found empirically that a value of $\sigma^2 = 16$ (with colors on a scale from $0 - 255$) worked well in selecting a neighborhood over which vectors would be combined. Also note that color flows are defined so that a color point with only a single nearby neighbor will inherit a flow vector that is nearly *parallel* to its neighbor. The idea is that if a particular color, under a photic parameter change $\theta_1 \mapsto \theta_2$, is observed to get a little bit darker and a little bit bluer, for example, then its neighbors in color space are also defined to exhibit this behavior.

We have thus outlined a procedure for using a pair of corresponding images $\mathcal{I} = (I_1, I_2)$ to generate a full color flow. We will write for brevity $\Phi = \Phi(\mathcal{I})$ to designate the flow generated from the image pair $\mathcal{I}$.

3.1 Structure in the Space of Color Flows

Certainly an image feature appearing as one color, say blue, in one image could appear as almost any other color in another image. Thus the *marginal distribution* of mappings for a particular color, when integrated over all possible photic parameter changes, is very broadly distributed. However, when color mappings are considered jointly, i.e. as color flows, we hypothesize that the space of possible mappings is much more compact. We test this hypothesis by statistically modeling the space of joint color maps, i.e. the space of color flows.

Consider for a moment a flat Lambertian surface that may have different reflectances as a function of the wavelength. While in principle it is possible for a change in lighting to map any color from such a surface to any other color *independently of all other colors*[3], we know from experience that many such joint maps are unlikely. This suggests that there is significant structure in the space of color flows. (We will address below the significant issue of non-flat surfaces and shadows, which can cause highly "incoherent" maps.)

In learning color flows from real data, many common color flows can be anticipated. To name a few examples, flows which make most colors a little darker, lighter, or redder would certainly be expected. These types of flows can be well modeled with simple global linear operators acting on each color vector. That is, we can define a 3x3 matrix $\mathbf{A}$ that maps a color $\mathbf{c_1}$ in the image I_1 to a color $\mathbf{c_2}$ in the image I_2 via

$$\mathbf{c_2} = \mathbf{A}\mathbf{c_1}. \tag{4}$$

Such linear maps work well for many types of common photic parameter changes. However, there are many effects which these simple maps cannot model. Perhaps the most significant is the combination of a large brightness change coupled with a non-linear gain-control adjustment or brightness re-normalization by the camera. Such photic changes will tend to leave the bright and dim parts of the image alone, while spreading the central colors of color space toward the margins. These types of changes cannot be captured well by the simple linear operator described above,

[2]This scheme is analogous to a Parzen-Rosenblatt non-parametric kernel estimator for densities, using a 3-D Gaussian kernel. To be a good estimate, the true flow should therefore be locally smooth.

[3]By carefully choosing surface properties such as the reflectance of a point as a function of wavelength, $S(p, \lambda)$, and lighting conditions $E(\lambda)$, any mapping $\tilde{\Phi}$ can, in principle be observed even on a flat Lambertian surface. However, as noted in [12, 8], the metamerism which would cause such effects is uncommon in practice.

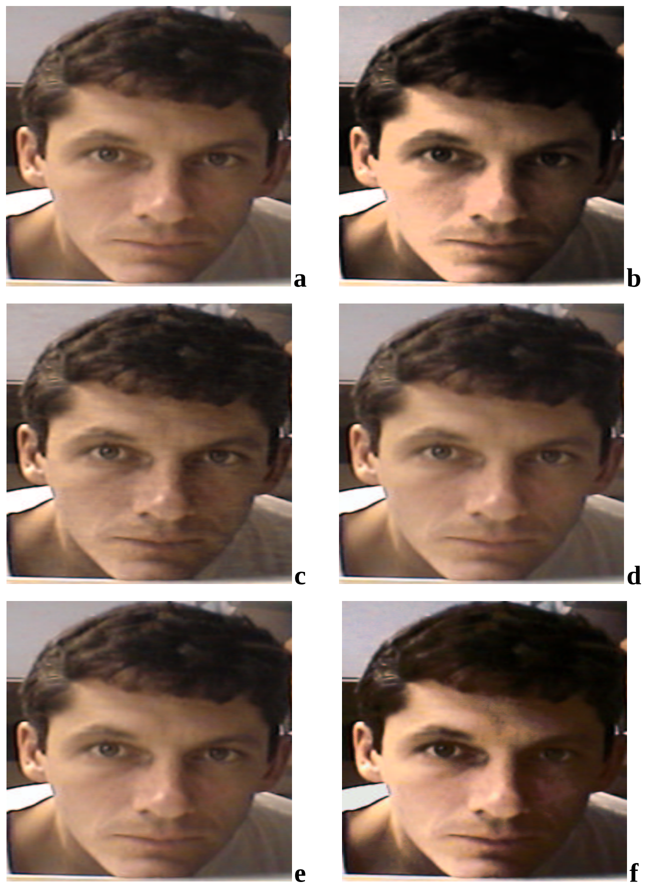


Figure 1: Image **b** is the result of applying a non-linear operator to the colors in image **a**. **c-f** are attempts to match **b** using **a** and four different algorithms. Our algorithm (image **f**) was the only one to capture the non-linearity.

but can be captured by modeling the space of color flows.

A pair of images exhibiting a non-linear color flow is shown in Figures 1**a** and **b**. Figure 1**a** shows the original image and **b** shows an image with contrast increased using a quadratic transformation of the brightness value. Notice that the brighter areas of the original image get brighter and the darker portions get darker. This effect cannot be modeled using a scheme such as that given in Equation 4. The non-linear color flow allows us to recognize that images **a** and **b** may be of the same object, i.e. to "match" the images.

3.2 Color Flow PCA

Our aim was to capture the structure in color flow space by observing real-world data in an unsupervised fashion. To do this, we gathered data as follows. A large color palette (approximately 1 square meter) was printed on standard non-glossy plotter paper using every color that could be produced by our Hewlett Packard DesignJet 650C pen plotter (see Figure 2). The poster was mounted on a wall in our office so that it was in the direct line of overhead lights and computer monitors, but not in the direct light from the single office window. An inexpensive video camera (the PC-75WR, Supercircuits, Inc.) with auto-gain-control was aimed at the poster so that the poster occupied about 95% of the field of view.

Images of the poster were captured using the video camera under a wide variety of lighting conditions, including various intervals during sunrise, sunset, at midday, and with various combinations of office lights and outdoor lighting (controlled by adjusting blinds). People used the office during the acquisition process as well, thus affecting the ambient lighting conditions. It is important to note that a variety of non-linear normalization mechanisms built into the camera were operating during this process.

Our goal was to capture as many common lighting conditions as possible. We did not use unusual lighting conditions such as specially colored lights. Although a few images that were captured probably contained strong shadows, most of the captured images were shadow-free. Smooth lighting gradients across the poster were not explicitly avoided or created in our acquisition process.

A total of 1646 raw images of the poster were obtained in this manner. We then chose a set of 800 image pairs $\mathcal{I}^j = (I_1^j, I_2^j), 1 \leq j \leq 800$, by randomly and independently selecting individual images from the set of raw images. Each image pair was then used to estimate a full color flow $\Phi(\mathcal{I}^j)$ as described in Equation 3.

Note that since a color flow Φ can be represented as a collection of $3P$ coordinates, it can be thought of as a point in $\mathbb{R}^{3P}$. Here P is the number of distinct RGB colors at which we compute a flow vector, and each flow vector requires 3 coordinates: dr, dg, and db, to represent the change in each color component. In our experiments we used $P = 16^3 = 4096$ distinct RGB colors (equally spaced in RGB space), so a full color flow was represented by a vector of $3 * 4096 = 12288$ components.

Given a large number of color flows (or points in $\mathbb{R}^{3P}$), there are many possible choices for modeling their distribution. We chose to use Principal Components Analysis since 1) the flows are well represented (in the mean-squared-error sense) by a small number of principal components (see Figure 3), and 2) finding the optimal description of a difference image in terms of color flows was computationally efficient using this representation (see Section 4).

The principal components of the color flows were computed (in MATLAB), using the "economy size" singular value decomposition. This takes advantage of the fact that the data matrix has a small number of columns (samples) relative to the number of components in a single sample.

We call the principal components of the color flow data

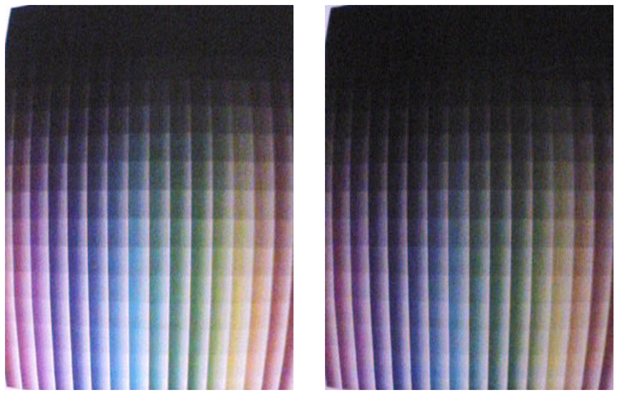


Figure 2: Images of the poster used for observing color flows, under two different "natural" office lighting conditions. Note that the variation in a single image is due to reflectance rather than a lighting gradient.

"color eigenflows", or just eigenflows[4], for short. We emphasize that these principal components of color flows have *nothing to do with the distribution of colors in images*, but only model the distribution of *changes in color*. This is a key and potentially confusing point. In particular, we point out that our work is very different from approaches that compute principal components in the intensity or color space itself, such as [13] and [11]. Perhaps the most important difference is that our model is a global model for all images, while the above methods are models only for a particular set of images, such as faces.

An important question in applying PCA is whether the data can be well represented with a "small" number of principal components. In Figure 3, we plot the eigenvalues associated with the first 100 eigenflows. This rapidly descending curve indicates that most of the magnitude of an average sample flow is contained in the first ten components. This can be contrasted with the eigenvalue curve for a set of random flows, which is also shown in the plot.

4. Using Color Flows to Synthesize Novel Images

How do we generate a new image from a source image and a color flow or group of color flows? Let $\mathbf{c}(p)$ be the color of a pixel p in the source image, and let Φ be a color flow that we have computed at a discrete set of P points according to Equation 3. For each pixel in the new image, its color $\mathbf{c}'$ can be computed as

$$\mathbf{c}'(p) = \mathbf{c}(p) + \alpha\Phi(\hat{\mathbf{c}}(p)), \tag{5}$$

[4]PCA has been applied to *motion* vector fields as in [7], and these have also been termed "eigenflows".

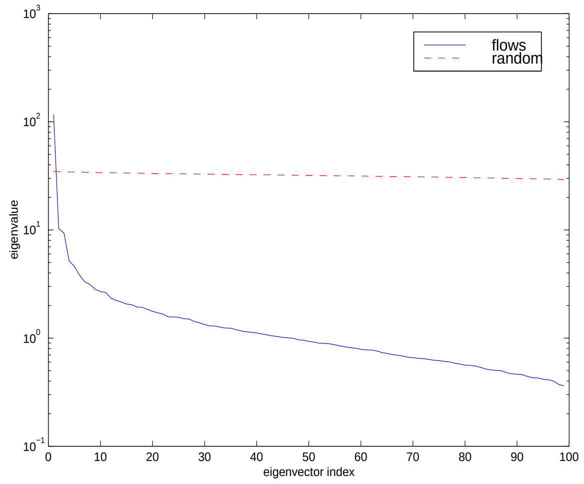


Figure 3: The eigenvalues of the color flow covariance matrix. The rapid drop off in magnitude indicates that a small number of eigenflows can be used to represent most of the variance in the distribution of flows.

where α is a scalar multiplier that represents the "quantity of flow". $\hat{\mathbf{c}}(p)$ is interpreted to be the color vector closest to $\mathbf{c}(p)$ (in color space) at which Φ has been computed. If the $\mathbf{c}'(p)$ has components greater than the allowed range of 0–255, then these components must be truncated.

Figure 4 shows the effect of each of the eigenflows on an image of a face. Each vertical sequence of images represents an original image (in the middle of the column), and the images above and below it represent the addition or subtraction of each eigenflow, with α varying between ± 8 standard deviations for each eigenflow.

We stress that the eigenflows were only computed once (on the color palette data),and that they were applied to the face image without any knowledge of the parameters under which the face image was taken.

The first eigenflow (on the left of Figure 4) represents a generic brightness change that could probably be represented well with a linear model. Notice, however, the third column in Figure 4. Moving downward from the middle image, the contrast grows. The shadowed side of the face grows darker while the lighted part of the face grows lighter. This effect cannot be achieved with a simple matrix multiplication as given in Equation 4. It is precisely these types of non-linear flows we wish to model.

4.1 From flow bases to image bases

Let $\mathcal{S}$ be the set of all images that can be created from a novel image and a set of eigenflows. Assuming no color truncation, we show how we can efficiently find the image in $\mathcal{S}$ which is closest (in an L_2 sense) to a target image.

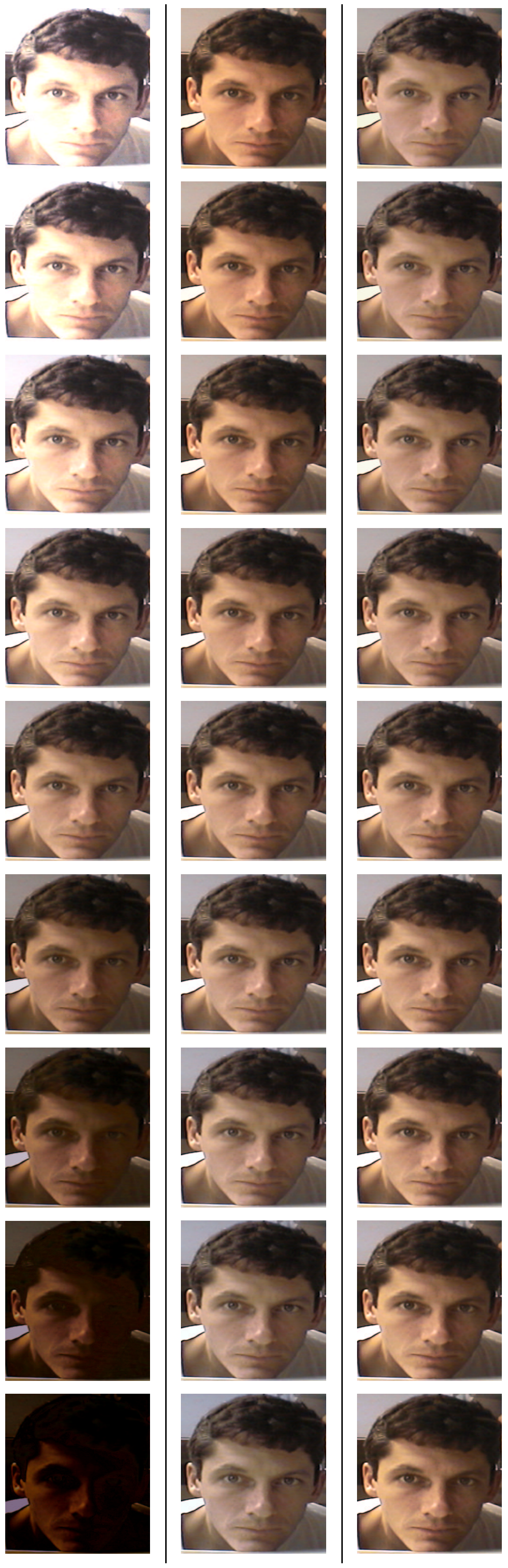


Figure 4: Effects of the first 3 eigenflows. See text.

Let $p_{x,y}$ be a pixel whose location in an image is at coordinates [x,y]. Let $I[x, y]$ be the vector at the location [x,y] in an image or in a difference image. Suppose we view an image I as a function that takes as an argument a color flow and that generates a difference image D by placing at each (x,y) pixel in D the color change vector $\Phi(\mathbf{c}(p_{x,y}))$. We denote this simply as

$$D = I(\Phi). \tag{6}$$

Then this "image operator" $I(\cdot)$ is linear in its argument since for each pixel (x,y)

$$\begin{aligned} (I(\Phi + \Psi))\,[x, y] &= (\Phi + \Psi)\,(\mathbf{c}(p_{x,y})) & (7) \\ &= \Phi(\mathbf{c}(p_{x,y})) + \Psi(\mathbf{c}(p_{x,y}))). & (8) \end{aligned}$$

The $+$ signs in the first line represent vector field addition. The $+$ in the second line refers to vector addition. The second line assumes that we can perform a meaningful component-wise addition of the color flows.

Hence, the difference pixels in a total difference image can be obtained by adding the difference pixels in the difference images due to each eigenflow (the *difference image basis*. This allows us to compute *any* of the possible image flows for a particular image and set of eigenflows from a (non-orthogonal) difference image basis. In particular let the difference image basis for a particular source image I and set of E eigenflows $\Psi_i, 1 \leq i \leq E$, be represented as

$$D_i = I(\Psi_i). \tag{9}$$

Then the set of images $\mathcal{S}$ that can be formed using a source image and a set of eigenflows is

$$\mathcal{S} = \{S : S = I + \sum_{i=1}^{E} \gamma_i D_i\}, \tag{10}$$

where the γ_i's are scalar multipliers, and here I is just an image and not a function. In our experiments, we used $E = 30$ of the top eigenvectors to define the space $\mathcal{S}$.

4.2 Flowing one image to another

Suppose we have two images and we pose the question of whether they are images of the same object or scene. We suggest that if we can "flow" one image to another then the images are likely to be of the same scene.

We can only flow image I_1 to another image I_2 if it is possible to represent the difference image as a linear combination of the D_i's, i.e. if $I_2 \in \mathcal{S}$. However, we may be able to get "close" to I_2 even if I_2 is not an element of $\mathcal{S}$.

Fortunately, we can directly solve for the optimal (in the least-squares sense) γ_i's by just solving the system

$$D = \sum_{i=1}^{E} \gamma_i D_i, \tag{11}$$

using the standard pseudo-inverse, where $D = I_2 - I_1$. This minimizes the error between the two images using the eigenflows. Thus, once we have a basis for difference images of a source image, we can quickly compute the best flow to any target image. We point out again that this analysis ignores truncation effects. While truncation can only reduce the error between a synthetic image and a target image, it may change which solution is optimal in some cases.

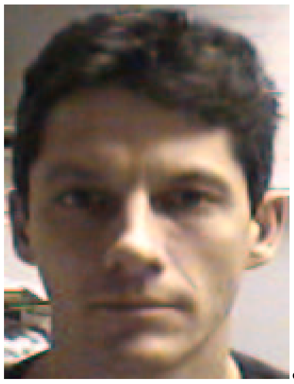
5. Experiments

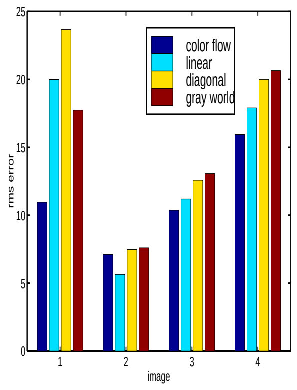
The goal of our system is to flow one image to another as well as possible when the images are actually of the same scene, but not to endow our system with enough capacity to be able to flow between images that do not in fact match. An ideal system would thus flow one image to a matching image with zero error, and have large errors for non-matching images. Then setting a threshold on such an error would determine whether two images were of the same scene.

We first examined our ability to flow a source image to a matching target image under different photic parameters. We compared our system to 3 other methods commonly used for brightness and color normalization. We shall refer to the other methods as **linear**, **diagonal**, and **gray world**. The **linear** method finds the matrix **A** according to Equation 4 that minimizes the L_2 fit between the synthetic image and the target image. **diagonal** does the same except that it restricts the matrix **A** to be diagonal. **gray world** adjusts each color channel in the synthetic image linearly so that the mean red, green, and blue values match the mean channel values in the target image.

While our goal was to reduce the numerical difference between two images using flows, it is instructive to examine one example which was particularly visually compelling, shown in Figure 1. Part **a** of the figure shows an image taken with a digital camera. Part **b** shows the image adjusted by squaring the brightness component (in an HSV representation) and re-normalizing it to 255. The goal was to adjust image **a** to match **b** as closely as possible (in a least squares sense). Images **c-f** represent the **linear, diagonal, gray world,** and **eigenflow** methods respectively. While visual results are somewhat subjective, it is clear that our method was the only method that was able to significantly darken the darker side of the face while brightening the lighter side of the face. The other methods which all implement linear operations in color space (ours allows non-linear flows) are unable to perform this type of operation. In another experiment, five images of a face were taken while changing various camera parameters, but lighting was held constant. One image was used as the source image (Figure 5**a**) in each of the four algorithms to approximate each of the other four images (see Figure 6).

Figure 5**b** shows the component-wise RMS errors between the synthesized images and the target image for each method. Our method outperforms the other methods in all but one task, on which it was second.


a



b

Figure 5: **a.** Original image. **b.** Errors per pixel component in the reconstruction of the target image for each method.

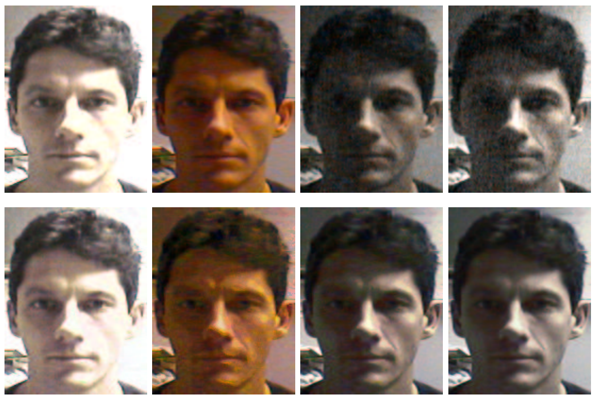


Figure 6: Test images. The images in the top row were taken with a digital camera. The images in the bottom row are the best approximations of those images using the eigenflows and the source image from Figure 5.

In another test, the source and target images were taken under very different lighting conditions (Figures 7**a** and **b**). Furthermore, shadowing effects and lighting direction changed between the two images. None of the methods could handle these effects when applied globally. To handle these effects, we used each method on small patches of the image. Our method again performed the best, with an RMS error of 13.8 per pixel component, compared with errors of $17.3, 20.1$, and 20.6 for the other methods. Figures 7**c** and **d** show the reconstruction of image **b** using our method and the best alternative method (**linear**). There are obvious visual artifacts in the linear method, while our method seems to have produced a much better synthetic image, especially in the shadow region at the edge of the poster.

One danger of allowing too many parameters in mapping one image to another is that images that do not actually match will be matched with low error. By performing synthesis on patches of images, we greatly increase the capacity of the model, running the risk of over-parameterizing or over-fitting our model. We performed one experiment to measure the over-fitting of our method versus the others. We horizontally flipped the image in Figure 7**b** and used this as a target image. In this case, we wanted the error to be large, indicating that we were unable to synthesize a similar image using our model. The RMS error per pixel component was 33.2 for our method versus 41.5, 47.3, and 48.7 for the other methods. Note that while our method had lower error (which is undesirable), there was still a significant spread between matching images and non-matching images.

We believe we can improve differentiation between matching and non-matching image pairs by assigning a cost to the *change in coefficients* γ_i across each image patch. For images which do not match, we would expect the γ_i's to change rapidly to accommodate the changing image. For images which do match, sharp changes would only be necessary at shadow boundaries or sharp changes in the surface orientation relative to directional light sources. We believe this can significantly enhance the method, by adding a strong source of information about how the capacity of the model is actually being used to match a particular image pair.

To use this method as part of an object recognition system, we clearly have to deal with geometric variation in addition to photic parameters. We are currently investigating the utility of our method on images which are out of alignment, which should aid in the incorporation of such a method into a realistic object recognition scenario.

Acknowledgements Support through ONR grant #'s N00014-00-1-0089 and N00014-00-1-0907 and through Project Oxygen at MIT.

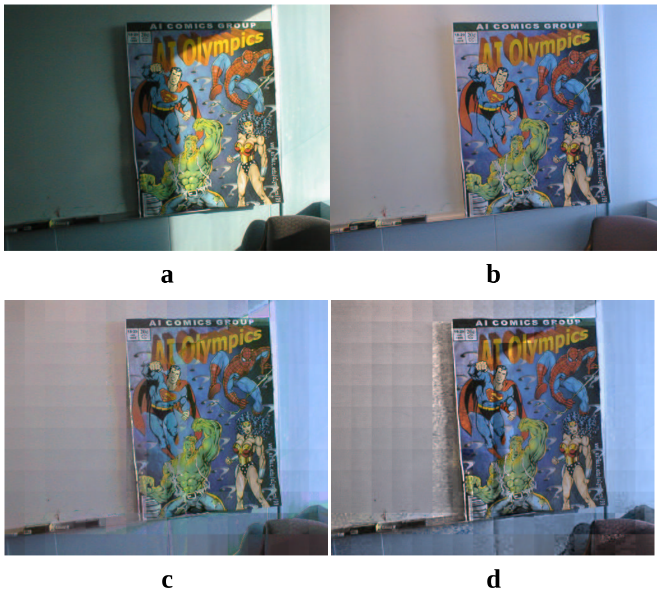


Figure 7: **a.** Image with strong shadow. **b.** The same image under more uniform lighting conditions. **c.** Flow from **a** to **b** using eigenflows. **d.** Flow from **a** to **b** using **linear**.

References

[1] P. N. Belhumeur and D. Kriegman. What is the set of images of an object under all possible illumination conditions? *International Journal of Computer Vision*, 28(3):1–16, 1998.

[2] G. Buchsbaum. A spatial processor model for object color perception. *Journal of the Franklin Institute*, 310, 1980.

[3] V. C. Cardei, B. V. Funt, and K. Barnard. Modeling color constancy with neural networks. In *Proceedings of the International Conference on Vision, Recognition, and Action: Neural Models of Mind and Machine*, 1997.

[4] D. A. Forsyth. A novel algorithm for color constancy. *International Journal of Computer Vision*, 5(1), 1990.

[5] B. K. P. Horn. *Robot Vision*. MIT Press, 1986.

[6] R. Lenz and P. Meer. Illumination independent color image representation using log-eigenspectra. Technical Report LiTH-ISY-R-1947, Linköping University, April 1997.

[7] J. J. Lien. *Automatic Recognition of Facial Expressions Using Hidden Markov Models and Estimation of Expression Intensity*. PhD thesis, Carnegie Mellon University, 1998.

[8] L. T. Maloney. Evaluation of linear models of surface spectral reflectance with small numbers of parameters. *Journal of the Optical Society of America*, A1, 1986.

[9] D. H. Marimont and B. A. Wandell. Linear models of surface and illuminant spectra. *Journal of the Optical Society of America*, 11, 1992.

[10] J. J. McCann, J. A. Hall, and E. H. Land. Color mondrian experiments: The study of average spectral distributions. *Journal of the Optical Society of America*, A(67), 1977.

[11] M. Soriano, E. Marszalec, and M. Pietikainen. Color correction of face images under different illuminants by rgb eigenfaces. In *Proceedings of the Second International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 148–153, 1999.

[12] W. S. Stiles, G. Wyszecki, and N. Ohta. Counting metameric object-color stimuli using frequency limited spectral reflectance functions. *Journal of the Optical Society of America*, 67(6), 1977.

[13] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cogntive Neuroscience*, 3(1):71–86, 1991.