Machine Vision and Applications manuscript No.
(will be inserted by the editor)

Marc Ebner

Color Constancy Based on Local Space Average Color

Abstract Light, which is reflected from an object, varies with the type of illuminant used. Nevertheless, the color of an object appears to be approximately constant to a human observer. The ability to compute color constant descriptors from reflected light, is called color constancy. In order to solve the problem of color constancy, some assumptions have to be made. One frequently made assumption is that on average, the world is gray. We address the problem of color constancy and focus on the use of space average color for color constancy. Instead of computing global space average color we suggest to use local space average color as the illuminant frequently varies across an image. We discuss several different methods on how to compute local space average color. The performance of the different algorithms as well as related algorithms is evaluated on an object recognition task. Algorithms based on local space average color are simple, yet highly effective for the problem of color constancy. Such algorithms are particularly suited for object recognition tasks.

Keywords Color Constancy, Space Average Color, Color Perception, Vision, Color, Reflectance, Object Recognition

Marc Ebner
Universität Würzburg, Lehrstuhl für Informatik II
Am Hubland, 97074 Würzburg, Germany
Tel.: (+49)931/888-6612
Fax: (+49)931/888-6603
E-mail: ebner@informatik.uni-wuerzburg.de
http://wwwi2.informatik.uni-wuerzburg.de/staff/ebner/welcome.html

1 Motivation

Let us take some kind of sensor such as analog film or a digital sensor array with which we take an image from a scene. The sensor measures the light which is reflected from the objects. The reflected light depends on the response function of the sensor, on the material properties of the objects and the type of illuminant used. Assuming a narrow band response function for the sensors, the response is roughly proportional to the reflectance of the object times the intensity of the illuminant for a particular wavelength. Suppose that we have a wall which reflects light uniformly for all wavelengths, i.e. a white wall, and that the wall is illuminated using light from a light bulb with more energy in the red and green parts of the spectrum compared to the blue part of the spectrum, i.e. a slightly yellowish light source. When the light is reflected from the wall and measured by the sensor it will appear to be yellow. Thus, if we take an image of the white wall which is illuminated by the yellowish light source, then the wall will appear yellow instead of white in a photograph. However, it is clear to a human observer, that the wall is white and not yellow. The human visual system is somehow able to compute color constant descriptors from the light entering the eye. This ability is called color constancy (Zeki, 1993; Ebner, 2007a).

Computing color constant descriptors is obviously very important for consumer photography. The colors shown in a photograph should be exactly the same colors as perceived by the observer at the time when he or she took the photograph. If we had a correct model of human color vision Ebner (2007b), we would be

able to produce prints which would satisfy amateur photographers all around the world.

However, color constancy is also very important for object recognition tasks as well as a variety of methods from machine vision which rely on color images. Quite often algorithms work in one particular lighting condition but not in another one. Color based object recognition becomes much simpler if reflectances were known. If the reflectances are not known it pays to first obtain an estimate of the reflectances and then use this estimate instead of the measured color of the objects for recognition. Color constancy algorithms are important for any computer vision algorithm based on color.

The reader should take note that the two objectives just described, mimicking human color constancy and obtaining an estimate of the reflectances are not the same. Even though human color perception correlates with integrated reflectance as shown by McCann et al. (1976), the human visual system does not actually estimate the reflectance of objects as the experiments of Helson (1938) have shown. Both objectives may be followed in their own right and both are valuable research questions.

A number of different algorithms have been developed to address the problem of color constancy – from Land and McCann's original Retinex algorithm (Land and McCann, 1971) to gamut constraint methods (Barnard et al., 1997; Forsyth, 1988, 1990), color constancy based on Bayesian decision theory (Brainard and Freeman, 1997), color by correlation (Barnard et al., 2000; Finlayson et al., 1997, 2001), comprehensive color normalization (Finlayson et al., 1998), use of neural networks (Funt et al., 1996) or computation of intrinsic images (Weiss, 2001; Finlayson and Hordley, 2001b; Tappen et al., 2002; Finlayson et al., 2004). Some algorithms have also been based on the dichromatic reflection model (Tominaga, 1991; D'Zmura and Lennie, 1992; Finlayson and Schaefer, 2001; Risson, 2003; Ebner and Herrmann, 2005). In order to address the problem of color constancy, many authors assume that the illuminant is constant across the image.

Only a few authors have considered a spatially varying illuminant. A notable exception is the original Retinex algorithm of Land and McCann (1971). Horn (1974, 1986) extended the algorithm to two dimensions. The algorithm works by first applying a Laplacian and then using a threshold operation to distinguish between a change in reflectance from a change of the illuminant. Changes of the illuminant are suppressed after which the application of the Laplacian is undone through an integrative step. Blake (1985) extended Horn's algorithm by separating the Laplacian into two gradient operations and applying the threshold operation in between. Land also proposed an alternative formulation of his original Retinex algorithm (Land, 1986). In this version of the Retinex algorithm, color constant descriptors are computed by first taking the logarithm of the input and then subtracting the logarithm of the average color of the surround. Moore et al. (1991) implemented a variant of this algorithm in VLSI. However, this algorithm requires global operation which is carried out over all pixels in order to compute a color corrected output. Rahman et al. (1999) suggested to perform the color correction on multiple scales. The method may not only be used for color constancy but also for dynamic range compression. The two-dimensional gamut constraint algorithm has been extended by Barnard et al. (1997) to work with images which were taken under a spatially varying illuminant. Algorithms for color constancy assuming a constant illuminant throughout the image as well as algorithms which allow for a spatially varying illuminant are addressed in a recent book on color constancy (Ebner, 2007a). An overview over color constancy algorithms is also given by Agarwal et al. (2006).

Most algorithms are computationally quite complex. For instance, in order to implement the gamut constraint methods one needs to compute the convex hull of a set of points which is a very difficult problem if implemented using finite precision arithmetic. In contrast, the methods presented in this contribution are easy to implement and - as the experimental results show - quite effective at computing color constant descriptors. We show that the method may even be integrated into imaging chips or display devices because the method is completely parallel. In addition, a variant of the method which is presented here is also a candidate for the process which computes color constant descriptors within the human brain (Ebner, 2007b).

We previously reported on a parallel algorithm for color constancy (Ebner, 2004) based on the computation of local space average color. Here, we we describe how this parallel algorithm may be implemented using a resistive grid which

allows integration into imaging chips and display devices. We also show how the resistive grid can be quickly simulated using successive over-relaxation. We will evaluate the algorithms described in this contribution along with other algorithms known from the literature using a variety of test images from different sources. We will see that algorithms based on local space average color are particularly suited for object recognition tasks.

The main contribution of this article is as follows. (1) We give an overview on how to compute local space average color depending on the computational architecture which is available, (2) we show how color constancy may be performed directly inside an imaging chip using a simple resistive grid without any need of a global scaling operation, (3) we show how to extend Finlayson's comprehensive color constancy algorithm (Finlayson et al., 1998) to work with varying illuminants within a scene by using local space average color, and (4) we also present a thorough evaluation of the performance of algorithms based on the computation of local space average color. In particular, the accuracy of the computed data is evaluated and algorithms based on local space average color are compared to more complex color constancy algorithms.

The article is structured as follows. First, we will briefly review some theory of color image formation. Then we will review the gray world assumption. We will show how the gray world assumption may be used to estimate the illuminant locally for each image pixel. Next, we show how a resistive grid may be used to do this estimation quickly. A resistive grid may be used inside an image chip to compute a color corrected output before a CPU has retrieved the data from the imaging chip. All algorithms are evaluated on a standard set of images to be used for color constancy experiments. We also evaluate all algorithms on a particularly difficult set of real world images with multiple differently colored illuminants. The performance of algorithms based on the computation of local space average color is compared to additional well known algorithms from the literature. At the end of the article we will give some conclusions.

2 Theory of Color Image Formation

Consider an array of sensors which measures the incident light. Each sensor responds to light from some part of the visible spectrum. Let $\mathbf{S}(\lambda)$ be a vector which describes the response characteristics of the sensor, i.e. $S_i(\lambda)$ describes the response curve of the i-th sensor for wavelength λ. Most sensor arrays use three different sensors which respond to light in the red, green, and blue part of the spectrum. In this case, we have $i \in \{r, g, b\}$. Some sensor arrays with 4 different sensors also exist. Light from an object patch enters the lens of the camera and falls onto an infinitesimal small patch on the sensor. Let us consider two corresponding locations $\mathbf{x}_{\text{obj}}$ on the object and $\mathbf{x}_I$ on the image sensor. Let $E(\lambda, \mathbf{x}_I)$ be the irradiance of wavelength λ falling onto an infinitesimal small patch on the sensor array. The energy measured by the sensor is given by

$$\mathbf{I}(\mathbf{x}_I) = \int E(\lambda, \mathbf{x}_I)\mathbf{S}(\lambda)d\lambda \tag{1}$$

where the integration is done over all wavelengths. Let $L(\lambda, \mathbf{x}_{\text{obj}})$ be the radiance from a light source falling onto the corresponding object position $\mathbf{x}_{\text{obj}}$. The amount of light reflected from the surface is specified by the reflectance function $R(\lambda, \mathbf{x}_{\text{obj}})$. Thus, for a Lambertian surface which reflects the light equally in all directions, we have

$$E(\lambda, \mathbf{x}_I) = R(\lambda, \mathbf{x}_{\text{obj}})L(\lambda, \mathbf{x}_{\text{obj}}). \tag{2}$$

The object geometry at position $\mathbf{x}_{\text{obj}}$ introduces a scaling factor $G(\mathbf{x}_{\text{obj}})$. A Lambertian surface illuminated by a point light source can be modeled using

$$G(\mathbf{x}_{\text{obj}}) = \cos\alpha = \mathbf{N}_{\text{obj}}\mathbf{N}_L \tag{3}$$

where α is the angle between the normal vector of the surface $\mathbf{N}_{\text{obj}}$ and a vector which points into the direction of the light source $\mathbf{N}_L$ at position $\mathbf{x}_{\text{obj}}$. Thus, the intensity $\mathbf{I}$ measured by the sensor is given by

$$\mathbf{I}(\mathbf{x}_I) = G(\mathbf{x}_{\text{obj}}) \int R(\lambda, \mathbf{x}_{\text{obj}})L(\lambda, \mathbf{x}_{\text{obj}})\mathbf{S}(\lambda)d\lambda. \tag{4}$$

This model of color image formation is frequently used by many color constancy algorithms (Buchsbaum, 1980; Finlayson, 1996; Finlayson et al., 1995; Finlayson and Hordley,

2001a; Finlayson et al., 1994a, 2004; Finlayson and Hordley, 2001b; Finlayson et al., 2002; Finlayson and Schaefer, 2001; Finlayson et al., 1998; Forsyth, 1988, 1992; Novak and Shafer, 1992).

Even though each sensor responds to a range of wavelengths, the sensor's response characteristics can often be approximated by Dirac's delta functions (Finlayson and Hordley, 2001b)

$$S_i(\lambda) = \delta(\lambda - \lambda_i) \tag{5}$$

with $i \in \{r, g, b\}$. In this model, each sensor responds to light of a single wavelength. If the sensor's response characteristics are not narrow band they may be sharpened (Finlayson et al., 1994a,b; Finlayson and Funt, 1996; Barnard et al., 2001). Assuming that the sensor can be modeled by delta functions, simplifies the lighting equation considerably, we obtain

$$I_i(\mathbf{x}_I) = G(\mathbf{x}_{\text{obj}})R(\lambda_i, \mathbf{x}_{\text{obj}})L(\lambda_i, \mathbf{x}_{\text{obj}}) \tag{6}$$

for the intensity $I_i(\mathbf{x}_I)$ measured by sensor i at position $\mathbf{x}_I$ on the sensor array. From now on, we will denote the sensor position as well as the corresponding object position using image coordinates (x, y). We use the index i to denote the color band, i.e. wavelength. Let

$$\mathbf{c}(x, y) = [c_r(x, y), c_g(x, y), c_b(x, y)] \tag{7}$$

be the color of the image at position (x, y). Assuming a linear relationship between the energy measured by the sensor $\mathbf{I}$ and the image data, we have

$$\mathbf{c}(x, y) = \mathbf{I}(x, y) = G(x, y)\mathbf{R}(x, y)\mathbf{L}(x, y) \tag{8}$$

where $\mathbf{R}(x, y) = [R_r(x, y), R_g(x, y), R_b(x, y)]$ is a vector describing the reflectances and $\mathbf{L}(\mathbf{x}, \mathbf{y}) = [L_r(x, y), L_g(x, y), L_b(x, y)]$ is a vector describing the illuminant. The multiplication is carried out component-wise, i.e. if $\mathbf{c} = \mathbf{ab}$ then $c_i = a_i b_i$ with $i \in \{r, g, b\}$. In other words, the intensity of each color channel is proportional to the product of the reflectance and the light falling onto the object scaled by a geometry factor. Since we only have three measurements but at least 6 unknowns, some assumptions have to be made in order to estimate the reflectance from an image.

Note, that here we have assumed a linear relationship between the energy measured by the sensor and the image data. In practice one often needs to process images which are stored as JPEG or in some other file format. If nothing is known about the process on how the image was created, then it is usually assumed that the image was stored using the sRGB color space (Stokes et al., 1996). The sRGB color space is device independent and allows exchange of images between different machines. It stores images with a gamma correction applied. Many digital cameras store their images using the sRGB color space. In order to restore the linear relationship between image data and measured intensities we have to undo any gamma correction before processing.

3 The Gray World Assumption

A simple scaling of the color channels suffices to obtain a color corrected image if the response curves of the receptors are very narrow-band and we assume that the illuminant is uniform over the entire image. The required scaling factor can be computed using the gray world assumption. The gray world assumption was proposed by Buchsbaum (1980). It estimates the illuminant by computing global space average color. The fact that the illuminant can be estimated by computing some kind of average of the light reaching the observer was known for a long time. It was also suggested by Land (see Judd (1960)). Buchsbaum was the first to formalize the method. The assumption is that, on average, the world is gray. Buchsbaum estimates the illuminant by assuming that a certain standard spatial spectral average exists for the entire visual field. This average can be used to estimate the illuminant. Once the illuminant is estimated, the reflectances can be computed. Quite a large number of algorithms have been proposed that use the gray world assumption in one way or another (Ebner, 2003, 2004, 2006; Finlayson et al., 1998; Gershon et al., 1987; Moore et al., 1991; Paulus et al., 1998; Pomierski and Groß, 1995; Rahman et al., 1999; Tominaga, 1991).

Our derivation of the gray world assumption differs from the one given by Buchsbaum. Buchsbaum assumed overlapping response characteristics of the sensor array. We will assume non-overlapping response characteristics. It has been shown that only in some specific cases a more general transform (Barnard et al., 2001; Finlayson et al., 1994a) is helpful. We will also include geometry information into the reflection

model. Buchsbaum's model did not use any geometry information. As we have already seen above, the intensity $c_i(x,y)$ of color channel i at position (x,y) of the image can be approximated by

$$c_i(x,y) = G(x,y)R_i(x,y)L_i(x,y). \tag{9}$$

The first term, $G(x,y)$, is a factor which depends on the scene geometry at the corresponding object point. The second term, $R_i(x,y)$, is the amount of light reflected at the corresponding object position for the wavelength λ_i. The third term, $L_i(x,y)$, is the intensity of the light emitted by the light source. The illuminant scales the product of the geometry factor and the reflectance of the object. Thus, we can achieve color constancy by dividing this expression by the illuminant. An independent scaling of the color bands suffices for a uniform illuminant.

Let us assume that the colors of the objects shown in the image are uniformly distributed over the entire color range and that we have a single uniform illuminant $L_i(x,y) = L_i$. Let the image intensities be in the range $[0,1]$. If we have a number of objects in the scene with a sufficiently large number of different colors, then the average of a sufficiently large number of pixels will be close to $\frac{1}{2}\mathbf{L}$. Using the above derivation for the image data, space average color $\mathbf{a} = [a_r(x,y), a_g(x,y), a_b(x,y)]$ of an image with n pixels is given by

$$a_i = \frac{1}{n}\sum_{x,y} c_i(x,y) \tag{10}$$

$$= \frac{1}{n}\sum_{x,y} G(x,y)R_i(x,y)L_i \tag{11}$$

$$= L_i\frac{1}{n}\sum_{x,y} G(x,y)R_i(x,y). \tag{12}$$

We can consider both the object geometry and the reflectance to be independent random variables, i.e. there is no correlation between the shape of an object and its color. For a large number of image pixels, we have

$$a_i = L_i E[GR_i] \tag{13}$$

where $E[GR_i]$ denotes the expected value of the geometry factor times the reflectance. Because of the independence assumption, we have

$$a_i = L_i E[G]E[R_i]. \tag{14}$$

Let us now assume that reflectances R_i are uniformly distributed over the interval $[0,1]$, i.e. some objects absorb most of the incident light ($R_i \approx 0$) and some reflect most of the incident light ($R_i \approx 1$). This basically means that all colors are equally likely provided that we have narrow band sensors and all the different sensor responses are perceived as different colors. If we make this assumption, we obtain

$$a_i = L_i E[G]\frac{1}{2}. \tag{15}$$

Using this equation, we can estimate the color of the illuminant as

$$L_i \approx \frac{2}{E[G]}a_i = f a_i \tag{16}$$

where $f = \frac{2}{E[G]}$ is a scaling factor. This scaling factor depends on the scene viewed. Because we are computing an average pixel value, it is very important that the relationship between pixel colors and measured sensor values is linear. If the relationship is not linear (possibly due to a gamma correction) it has to be linearized by undoing the gamma correction.

Once the color of the illuminant is known, it can be used to estimate the product of the geometry factor times the reflectances. Let o_i be the computed output color which is independent of the illuminant. We have,

$$o_i(x,y) = \frac{c_i(x,y)}{f a_i} \approx \frac{c_i(x,y)}{L_i} = G(x,y)R_i(x,y). \tag{17}$$

The factor f scales all color channels by the same amount. It only affects the lightness of the image but not the colors. If we set $f = 2$, we assume a perpendicular orientation between object and camera, i.e. E[G]=1. Since the factor f scales all channels equally, this factor basically only adjusts the brightness of the image. Using $f = 2$ will improve overall brightness in extended dark areas of the image. It automatically causes the resulting average of the image to be moved to $\frac{1}{2}$ which is necessary for a completely parallel algorithm. A suitable value for this factor can also be estimated directly from the image. We simply need to rescale all channels such that only 1% of all pixels are clipped.

If we have two or more light sources then the assumption, that the illuminant is uniform over the entire image, is violated. In this case,

one needs to estimate the illuminant locally for each image pixel. The intensity of color band i at position (x, y) in the image is given by

$$c_i(x, y) = G(x, y)R_i(x, y)L_i(x, y). \tag{18}$$

If we estimate the illuminant $L_i(x, y)$ locally for each image pixel then we can again compute a color corrected image by dividing each pixel by the estimated color of the illuminant

$$\frac{c_i(x, y)}{L_i(x, y)} = \frac{G_i(x, y)R_i(x, y)L_i(x, y)}{L_i(x, y)} \tag{19}$$

$$= G_i(x, y)R_i(x, y). \tag{20}$$

The result is a color corrected image which is independent of the color of the illuminant. The geometry factor could be removed by normalizing the colors. Instead of computing global space average color we can compute local space average color as an estimate of the illuminant. In this case, the estimate is given by

$$L_i(x, y) \approx f a_i(x, y) \tag{21}$$

where f is a scaling factor as described above, e.g. $f = 2$.

Figure 1 shows results for the gray world assumption for two sample images. The image on the left shows a typical office scene. For this image, the gray world assumption works nicely. The reader should take note of the fact that the gray world assumption only works (assuming a sample scene which is not achromatic) if there are a sufficiently large number of different colors in the scene. This can be seen in the image shown on the right. This image shows a leaf from a banana plant which mostly contains green colors and a bit of yellow. Its average is $[0.19, 0.27, 0.04]$. It is obvious that the gray world assumption does not work if there are only very few different colors in the scene.

A variant of the gray world hypothesis was recently proposed by van de Weijer and Gevers (2005); van de Weijer et al. (2007). Note that for the gray world assumption it is assumed that the black level is set correctly for the imaging device. Van de Weijer et al. suggest to apply the gray world hypothesis to edges. This has the advantage that the black level does not have to be set correctly. The color of the illuminant is estimated by first computing local edges at scale σ. The edge differences are averaged over the entire image. The illuminant $\mathbf{L}$ is thus given by

$$\mathbf{L} = f\left(\int \left|\frac{\delta^n \mathbf{c}^\sigma(\mathbf{x})}{\delta \mathbf{x}^n}\right|^p d\mathbf{x}\right)^{\frac{1}{p}} \tag{22}$$

where f is a scaling factor, n is the order of the derivative, p is the Minkowski norm and σ is the scale parameter. Use of the Minkowski norm is due to Finlayson and Trezzi (2004). Van de Weijer et al. (2007) report very good illuminant estimation results for $n = 2$, $p = 7$ and $\sigma = 5$.

4 Estimating the Illuminant Locally

Local space average color can be computed easily using a convolution. Let $g(x, y)$ be a kernel function then local space average color $a_i(x, y)$ is given by

$$a_i(x, y) = \tag{23}$$

$$k(x, y) \int \int c_i(x, y)g(x - x', y - y')dx'dy'$$

the scaling factor $k(x, y)$ is chosen such that

$$k(x, y) \int \int g(x', y')dx'dy' = 1. \tag{24}$$

A possible choice for a kernel function would be either a Gaussian, i.e. $g(x, y) = e^{-\frac{r^2}{2\sigma^2}}$ with $r = \sqrt{x^2 + y^2}$ or an exponential function, i.e. $g(x, y) = e^{-\frac{|x|+|y|}{\sigma}}$. For the gray world assumption to hold, the support of the Gaussian has to be sufficiently large. A suitable choice in case of a Gaussian kernel is $\sigma = 0.18s$ where $s = \max\{n_x, n_y\}/2$ and n_x and n_y is the width and height of the input image. In case of an exponential kernel, the kernel should be slighly smaller in order for both kernels to produce the same estimate of the illuminant. A suitable choice is $\sigma = 0.17s$. The scaling factors were chosen such that the support for the kernel is larger than 0.1 for 34% of the image. This choice is near optimal for the images contained in dataset no. 1 which is used in the experiments below. Figure 2 shows the dependence of the average standard deviation of the pixels of several scenes contained in this data set after the color constancy algorithm based on local space average color has been applied.

Given an arbitrary input image with a spatially varying illuminant, the scaling parameter σ can be tuned to this illuminant. The above

Fig. 1 Results using the gray world assumption for two different input images. The image on the left shows a typical office scene. The image on the right shows a leaf from a banana plant. The gray world assumption only works if there are a sufficiently large number of different colors in the scene.

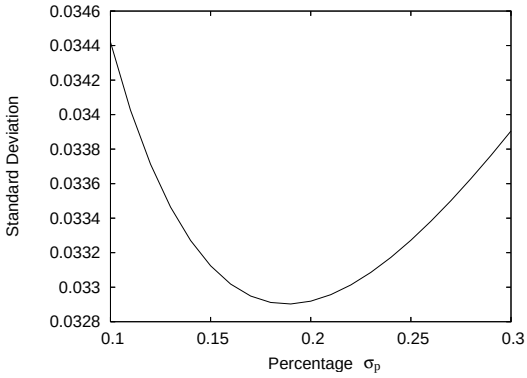

Fig. 2 Dependence of the average per pixel standard deviation on the size of support. The standard deviation is computed across several images showing the same scene after the color constancy algorithm based on local space average color has been applied with an exponential kernel $e^{-\frac{r}{\sigma_p s}}$ and s $s = \max\{n_x, n_y\}/2$ and n_x and n_y is the width and height of the input image.

choice for the support of the kernel essentially means that the kernel extends over one third of the image. Hence, this kernel removes a slowly changing illuminant from the image. Implicit in this choice of support is that the illuminant is either constant or linearly changing over an area of 34% of the image. This is a quite reasonable assumption for many photographs as Figure 2 shows.

Since the illuminant has to be estimated locally for each image pixel we can search for a parallel algorithm (Ebner, 2004, 2006). There are two major incentives for parallelizing this color constancy algorithm: (1) A parallel algorithm for color constancy can be mapped to what is known about the human stages of color perception. Many algorithms for color constancy which have been developed cannot readily be mapped to what is known about the human visual system. Some algorithms, such as the gamut constraint algorithms are computationally very complex and it is not apparent how the human visual system would compute or work with complex hulls of the observed gamut. (2) A parallel algorithm for color constancy may be integrated directly into imaging chips or even output devices such as flat panel displays. Currently digital cameras usually apply color constancy algorithms after the imaging chip has acquired the data.

Let us now assume that we have a grid of processing elements, i.e. a single instruction

multiple data (SIMD) type of architecture (Almasi and Gottlieb, 1994). Each processing element is running the same instructions but processing different data. We will be using one processing element per image pixel. This is not strictly required as we could resize images before processing. Each processing element will estimate local space average color.

Once we have an estimate of local space average color, we can perform a rescaling of the color channels as described above to arrive at a color constant image. Each processing element is connected to its neighbors. We can either work with connections to the 4 nearest neighbors or the 4 diagonal neighbors. We can also use a neighborhood of 8. Since we only have local connections between processing elements, the algorithm can be scaled to arbitrary image sizes just by extending or reducing the number of processing elements used.

Let $\mathbf{c}(x, y)$ be the measured color at location (x, y) of the grid. We assume that three different color bands are processed independently, i.e.

$$\mathbf{c}(x, y) = [c_r(x, y), c_g(x, y), c_b(x, y)]. \tag{25}$$

Initially, we do not have an estimate of local space average color. Let us just suppose for a moment that we did and let $\mathbf{a}$ be this estimate of local space average color.

$$\mathbf{a}(x, y) = [a_r(x, y), a_g(x, y), a_b(x, y)] \tag{26}$$

Let $N(x, y)$ be the neighboring processing elements of element (x, y). We then iterate the following update equations for all three color bands

$$\mathbf{a}'(x, y) := \frac{1}{|N(x, y)|} \sum_{(x',y')\in N(x,y)} \mathbf{a}(x', y') \tag{27}$$

$$\mathbf{a}(x, y) := \mathbf{c}(x, y) \cdot p + \mathbf{a}'(x, y) \cdot (1 - p) \tag{28}$$

where p is a small percentage. The first operation (Eqn 27) simply averages the data from the neighboring elements. The second operation (Eqn 28) slowly fades the measured color of the current element into the computed average. The result is a new estimate of local space average color $\mathbf{a}$. If we iterate the above equations indefinitely, the data diffuses between neighboring elements. This process converges to local space average color for each pixel. The parameter p determines the extent over which local space average color will be computed. It is similar though not quite identical to the smoothing of the input image using an exponential kernel as will be described in Section 5.

The parameter p, just as the scaling parameter σ, depends on the size of the image. The parameter has to be chosen such that the averaging is done over a reasonably large area of the entire image. If the parameter p is large, then local space average color will be computed for a small neighborhood. If the parameter p is small, then local space average color will be computed for a large neighborhood. Setting the parameter $p = 0.0005$ for images of size 256×256 gives reasonable results. We describe below how this parameter can be set in practice by showing the correspondence between the resistive grid implementation and the smoothing using an exponential kernel.

We can view the second step of the averaging operation as a biased average, e.g.

$$\mathbf{a}(x, y) = \frac{\mathbf{a}'(x, y) \cdot (\frac{1}{p} - 1) + \mathbf{c}(x, y)}{\frac{1}{p}}. \tag{29}$$

If we set $p = 0.0005$, we obtain

$$\mathbf{a}(x, y) = \frac{\mathbf{a}'(x, y) \cdot 1999 + \mathbf{c}(x, y)}{2000}. \tag{30}$$

In other words, the average has a carrying capacity of 2000. We have 1999 components of $\mathbf{a}'$ and one component of $\mathbf{c}$.

Note that the current average is multiplied by $(1-p)$ each iteration. Since $(1-p)$ is smaller than 1, this component will slowly decay towards zero. Thus, we now see that the initialization of local space average color can be arbitrary. Whatever value we use to initialize local space average color, after a large number of iterations, little will be left of the original value.

Figure 3 illustrates the averaging operation for a small 3×3 mesh of processing elements. It is assumed here that only a single band is processed. The elements are initialized with the numbers from 1 to 9. If we keep averaging the data from neighboring elements, we arrive at the correct global average after a small number of iterations. So far, we did not include the center element into the averaging operation. Not including the center element may lead to oscillations. The oscillatory behavior disappears if the center element is included.

Figure 4 shows local space average color computed for a sample image. The images show snapshots after 1, 100, and 500 iterations. For

step 1 step 100 step 500

Fig. 4 Snapshots of local space average color after 1, 100, and 500 time steps.

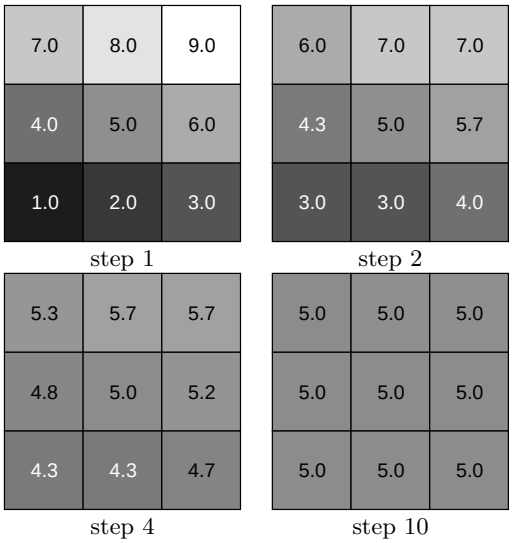

Fig. 3 By iteratively averaging data from neighboring processing elements, we compute the global average of the values. The sequence shows what happens if we initialize a 3×3 mesh of processing elements with numbers from 1 to 9.

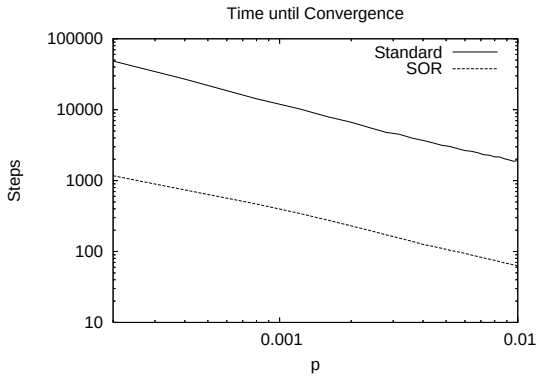

Fig. 6 Time to convergence is shown for different values of p. Successive over-relaxation provides a considerable speed-up.

this example, we have initialized the original estimate of local space average color at time step 0 to be zero. Thus, the image after the first time step is just the input image (except for scaling). Over time, the algorithm refines its estimate of local space average color. The color diffuses through the processing elements until a stable state is reached.

The parameter p may be used to control the extent of the averaging operation. Figure 5 shows local space average color computed from the same image but with different values of p. From left to right, the parameter p was set to 0.05, 0.005, 0.0005. If p is large, then local space average color will be computed for a small neighborhood. If p is small, local space average color will be computed for a large neighborhood. This begs the question to what value should this parameter be set. The area has to be large enough such that the gray world assumption is valid. In other words, the area over which local space average color will be computed has to be large enough such that the colors contained in this region are sufficiently diverse. Thus, we have to take care that the parameter p is set to a small enough value. What happens if we set p to a value which is too small? Assuming an infinitesimally small value for p, the algorithm computes global space average color. In this case, we would not be able to remove small linear changes of the illuminant which may be present across the image. We will describe below how to compute the parameter p from the smoothing factor σ which we used for the exponential kernel. The correspondence between the two is

$$\sigma = \sqrt{\frac{1-p}{4p}}. \tag{31}$$

The number of iterations required until convergence obviously also depends on the choice of the parameter p. If p is large, convergence will be achieved earlier. Convergence will take longer

p=0.05 p=0.005 p=0.0005

Fig. 5 Local space average color (after convergence) computed for different values of p.

if p is small. At each step, the color of the current pixel is multiplied by p and then added to the intermediate average. Therefore, we need at least $\frac{1}{p}$ iterations to bring local space average color into the range of the color $\mathbf{c}$ of the current pixel. Figure 6 shows how the time until convergence depends on the number of iterations. If we set $p = 0.0004$, we need approximately 27000 iterations until convergence. This is quite a large number and will take a prohibitive length of time to run for large images on a sequential computer. If the algorithm is run on parallel hardware, then this is not a constraint. In fact, as we will see in the next section, the algorithm can be implemented using a resistive grid. A significant speed-up can also be achieved by using successive over-relaxation (Bronstein et al., 2001; Demmel, 1996). Successive over-relaxation is based on the difference between the old and the new estimate computed for the current processing element. This difference tells us the direction in which to move, i.e. the corrective step. Instead of adjusting the current estimate by this difference, the current estimate is adjusted using a slightly larger step.

An additional parameter w is used to adjust the step size. If we apply successive over-relaxation, then local space average color $\tilde{a}$ is computed as

$$\mathbf{a}'(x,y) := \frac{1}{|N(x,y)|} \sum_{(x',y')\in N(x,y)} \tilde{\mathbf{a}}(x',y') \quad (32)$$

$$\mathbf{a}(x,y) := \mathbf{a}'(x,y) \cdot (1-p) + \mathbf{c}(x,y) \cdot p \quad (33)$$

$$\tilde{\mathbf{a}}(x,y) := \tilde{\mathbf{a}}(x,y) \cdot (1-w) + \mathbf{a}(x,y) \cdot w \quad (34)$$

where the center element is included in the neighborhood N. The time until convergence is considerably faster compared to the standard method. Convergence times for $w = 1.999$ are shown in Figure 6. An additional speed-up can be achieved if we use a resistive grid to estimate local space average color. Neighboring points are weighted exponentially in an exponential grid. Therefore, the computation which is carried out by a resistive grid is almost equivalent to the convolution of the input with an exponential kernel $g(x,y) = e^{\frac{-|r|}{\sigma}}$ where σ denotes the area of support and $r = |x| + |y|$. If we use a resistive grid to compute local space average color the result will be available almost instantaneously.

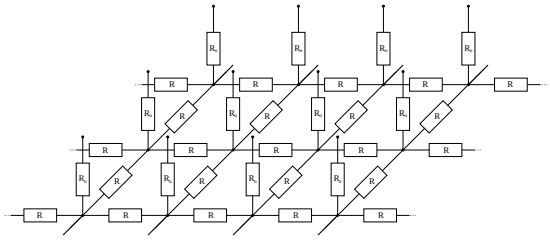

Fig. 7 A two-dimensional resistive grid.

5 Implementation using a Resistive Grid

Figure 7 shows a two-dimensional resistive grid which can be used to compute local space average color. Input is fed into the network from above. Each input resistor has the resistance R_o. All other resistors have the resistance R. Kirchhoff's law (Orear, 1982) states that the current flowing into the node at position (x,y) must be equivalent to the current flowing out of the node. Let $I_o(x,y)$ be the current flowing into the network at position (x,y) and let I_{left}, I_{right}, I_{up}, and I_{down} be the current flowing to the left, right, up and down. Thus, we have,

$$\begin{aligned} I_o(x,y) =& I_{\text{left}}(x,y) + I_{\text{right}}(x,y) + \\ & I_{\text{up}}(x,y) + I_{\text{down}}(x,y). \end{aligned} \quad (35)$$

We obtain the voltage across the resistor using Ohm's law. Let U be the voltage across a resistor R, then the current flowing through the

resistor is given by $I = \frac{U}{R}$. Let $U_o(x, y)$ be the applied voltage and let $U(x, y)$ be the voltage at node (x, y) then we obtain

$$\frac{1}{R_o}(U(x,y) - U_o(x,y)) = \frac{1}{R}(U(x-1,y) + U(x+1,y) + U(x,y-1) + U(x,y+1)) - \frac{4}{R}U(x,y). \quad (36)$$

We can solve this equation for $U(x, y)$.

$$U(x,y) = \frac{R_o}{4R_o + R}(U(x-1,y) + U(x+1,y) + U(x,y-1) + U(x,y+1)) + \frac{R}{4R_o + R}U_o(x,y) \quad (37)$$

This expression is equivalent to the update equations as described above. We need three resistive grids, one for each color band to estimate local space average color. We now see that local space average color $a(x, y)$ corresponds to the voltage $U(x, y)$ at points (x, y) of the grid. The color of the input pixel corresponds to the voltage $U_o(x, y)$ applied at grid point (x, y). We can compute the relationship between the percentage p and the resistances R_o and R of the resistive grid.

$$p = \frac{R}{4R_o + R} \quad (38)$$

If we solve this for $\frac{R_o}{R}$, we obtain

$$\frac{R_o}{R} = \frac{1-p}{4p}. \quad (39)$$

In other words, the resistance R_o has to be larger than the resistance R by a factor of $\frac{1-p}{4p}$.

We now look at the function which is computed by the resistive grid, i.e.

$$\frac{1}{R_o}(U(x,y) - U_o(x,y)) = \frac{1}{R}(U(x-1,y) + U(x+1,y) + U(x,y-1) + U(x,y+1)) - \frac{4}{R}U(x,y). \quad (40)$$

The right hand side is the discrete version of the Laplace operator. If we go to the continuous domain, we obtain

$$\frac{1}{R_o}(U(x,y) - U_o(x,y)) = \frac{1}{R}\nabla^2 U(x,y). \quad (41)$$

Let us now represent both $U(x, y)$ and $U_o(x, y)$ by its Fourier transform $U(w_x, w_y)$ and $U_o(w_x, w_y)$.

$$U(x,y) = \int\int U(w_x, w_y) e^{i(w_x x + w_y y)} dw_x dw_y \quad (42)$$

$$U_o(x,y) = \int\int U_o(w_x, w_y) e^{i(w_x x + w_y y)} dw_x dw_y \quad (43)$$

Let $\sigma = \sqrt{\frac{R_o}{R}}$, this gives us

$$U(x,y) - U_o(x,y) = \sigma^2 \nabla^2 U(x,y). \quad (44)$$

If we now substitute $U(x, y)$ and $U_o(x, y)$ both written in terms of their Fourier transform, we obtain

$$\int\int (U(w_x, w_y) - U_o(w_x, w_y))\, e^{i(w_x x + w_y y)} dw_x dw_y = -\sigma^2 \int\int (w_x^2 + w_y^2) U(w_x, w_y) e^{i(w_x x + w_y y)} dw_x dw_y \quad (45)$$

Thus, it must be that

$$U(w_x, w_y) = \frac{U_o(w_x, w_y)}{1 + \sigma^2(w_x^2 + w_y^2)}. \quad (46)$$

Therefore, the output computed by the resistive grid is

$$U(x,y) = \int\int \frac{U_o(w_x, w_y)}{1 + \sigma^2(w_x^2 + w_y^2)} e^{i(w_x x + w_y y)} dw_x dw_y. \quad (47)$$

In simulation we can compute the output by applying a Fourier transform and then multiplying the result by

$$\frac{1}{1 + \sigma^2(w_x^2 + w_y^2)}, \quad (48)$$

after which the result is transformed back into the spatial domain. A serious drawback of this

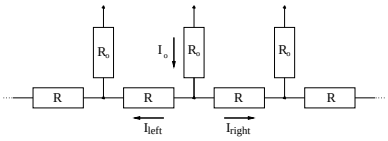

Fig. 8 A one-dimensional resistive grid.

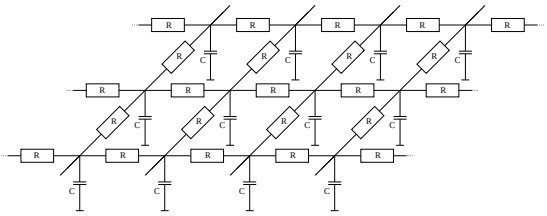

Fig. 9 A resistive grid where each node point is connected to ground via a capacitor.

method is that toroidal boundary conditions are assumed. Therefore, local space average color computed for pixels at the lower boundary of the image will be a mixture of the color of the pixels located at the lower and upper border of the image. If we assume that there is an illumination gradient across the image, the estimate will most likely be incorrect.

Since a one-dimensional network is easier to handle, let us consider the one-dimensional case for a moment. A one-dimensional network of resistors is shown in Figure 8. When we do the same derivation as above, except for a one-dimensional resistive grid, we arrive at the following equation (Jähne, 2002).

$$\frac{1}{R_o}(U(x) - U_o(x)) = \frac{1}{R}\frac{\partial^2}{\partial x^2}U(x) \tag{49}$$

Its solution is

$$U(x) = \int \frac{U_o(w)}{1+\sigma^2 w^2} e^{iwx} dw \tag{50}$$

where $U_o(w)$ is the one-dimensional Fourier transform of $U_o(x)$. Thus, for the one-dimensional case, we simply multiply the Fourier transform by $\frac{1}{1+\sigma^2 w^2}$ in the frequency domain. A multiplication in the frequency domain is equivalent to a convolution in the spatial domain (Gonzalez and Woods, 1992; Horn, 1986; Jähne, 2002; Jain et al., 1995; Parker, 1997; Shapiro and Stockman, 2001). Therefore, we can achieve the same result by convolving $U_o(x)$ with the inverse Fourier transform of $\frac{1}{1+\sigma^2 w^2}$. The Fourier transform of $e^{-\frac{|x|}{\sigma}}$ is $\frac{2\sigma}{1+\sigma^2 w^2}$ (Bronstein et al., 2001). The function computed by a one-dimensional resistive grid is therefore simply a convolution of the input with $\frac{1}{2\sigma} e^{-\frac{|x|}{\sigma}}$. The correspondence between the parameter σ, the resistors of the resistive grid and the parameter p is given by

$$\sigma = \sqrt{\frac{R_o}{R}} = \sqrt{\frac{1-p}{4p}}. \tag{51}$$

When we solve this equation for p, we obtain

$$p = \frac{1}{4\sigma^2 + 1} \tag{52}$$

where σ is the scaling factor of the exponential kernel. This allows us to tune the parameter to the size of the image. In practice, we first compute the scaling factor of the exponential kernel as described in Section 4 and then compute the factor p.

Going back to the two-dimensional case, we have to find the inverse Fourier transform of

$$\frac{1}{1+\sigma^2(w_x^2 + w_y^2)} \tag{53}$$

in order to find the function computed by the two-dimensional resistive grid. The Fourier transform of $e^{-\frac{r}{\sigma}}$ with $r = \sqrt{x^2+y^2}$ is $\frac{2\pi\sigma^2}{(1+\sigma^2(w_x^2+w_y^2))^{\frac{3}{2}}}$ (Weisstein, 1999). The Fourier transform of $e^{-\frac{r}{\sigma}}$ with $r = |x| + |y|$ is $\frac{4\sigma^2}{(1+\sigma^2 w_x^2)(1+\sigma^2 w_y^2)}$. In other words, the output computed by the resistive grid is not quite described by either a convolution with $e^{-\frac{r}{\sigma}}$ with $r = \sqrt{x^2+y^2}$ or $e^{-\frac{r}{\sigma}}$ with $r = |x|+|y|$. In practice, however, we can approximate the output computed by the resistive grid with a convolution using either one of the two kernels. Note that the kernel $e^{-\frac{|x|+|y|}{\sigma}}$ is separable which is an advantage because it can be applied in both the x and y direction independently. Of course, we can also use the kernel $e^{-\frac{x^2+y^2}{2\sigma^2}}$ which is also separable. Both the output from an exponential kernel, as well as from a Gaussian kernel, closely matches the output computed by a resistive grid. The root mean squared error between the output computed by the resistive grid and the output obtained from an exponential kernel is 0.009289 for the sample image shown in Figure 10.

We have already noted above, that the computation of local space average color using the

parallel algorithm, which is described above, can be viewed as a diffusion process where color flows form one processing element to the next. A similar analogy can be made with the resistive grid (Jähne, 2002). In order to see this, let us remove the input resistors and connect each grid point to ground via a capacitor. This gives us the circuit shown in Figure 9. Let $I_C(x,y)$ be the current flowing through the capacitor at position (x,y). The current flowing through the capacitor is given by $I_C(x,y) = \frac{1}{C}\frac{\partial}{\partial t}U(x,y)$. This gives us the following equation which describes the voltage $U(x,y)$ at each grid point (x,y)

$$\begin{aligned} 0 = &\frac{U(x-1,y)-U(x,y)}{R} \\ &+ \frac{U(x+1,y)-U(x,y)}{R} \\ &+ \frac{U(x,y-1)-U(x,y)}{R} \\ &+ \frac{U(x,y+1)-U(x,y)}{R} \\ &- C\frac{\partial}{\partial t}U(x,y) \end{aligned}$$

If we again go to the continuous domain, we obtain

$$0 = \frac{1}{R}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)U(x,y) - C\frac{\partial}{\partial t}U(x,y) \quad (54)$$

$$RC\ \frac{\partial}{\partial t}U(x,y) = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)U(x,y) \quad (55)$$

This equation describes a diffusion process (Bronstein et al., 2001; Jähne, 2002; Weickert, 1997). Let $\mathbf{j}$ be a flow which occurs in a direction opposite to the concentration gradient. Since the concentration is given by $U(x,y)$ and its gradient is $\nabla U(x,y)$, the flow $\mathbf{j}$ is given by

$$\mathbf{j} = -D\nabla U(x,y) \quad (56)$$

where D is the diffusion coefficient. The fact that the net charge of the system is constant, is described by the continuity equation (Jähne, 2002; Nolting, 1992).

$$\frac{\partial}{\partial t}U(x,y) + \nabla\mathbf{j} = 0 \quad (57)$$

Using the continuity equation, we again arrive at the above equation.

$$\frac{\partial}{\partial t}U(x,y) = \nabla(D\nabla U(x,y)) = D\nabla^2 U(x,y) \quad (58)$$

The general solution to the homogeneous diffusion equation with $U(x,y)|_{t=0} = f(x,y)$ is given by (Bronstein et al., 2001; Jähne, 2002; Weickert, 1997)

$$U(x,y) = g_{D\sqrt{2t}}(r) \otimes f(x,y) \quad (59)$$

where $\otimes$ denotes convolution and $g_\sigma(r)$ with $r = \sqrt{x^2+y^2}$ is a Gaussian kernel

$$g_\sigma(r) = \frac{1}{2\pi\sigma^2}e^{-\frac{r^2}{2\sigma^2}}. \quad (60)$$

6 Comprehensive Local Space Average Color

In Section 7, we will evaluate the performance of the algorithm which has just been described. We will compare its performance with other algorithms known from the literature. Some of the algorithms allow for a spatially varying illuminant while others do not. An algorithm called comprehensive normalization which was originally developed by Finlayson et al. (Finlayson et al., 1998) assumes that a single uniform illuminant is present. In this section, we describe how this algorithm may be extended to work also in the presence of a spatially varying illuminant.

Comprehensive normalization removes both dependencies due to lighting geometry as well as dependencies on the type of illuminant. The influence of scene geometry is removed by normalizing all pixel values. The influence of the illuminant is removed using the gray world assumption. These two operations are interleaved and iteratively applied until convergence is reached.

Let $\mathcal{A}_{\text{norm}}$ be the algorithm which removes the geometry information by normalizing all colors, i.e.

$$o_i(x,y) = \frac{c_i(x,y)}{\sum_{j\in\{r,g,b\}} c_j(x,y)}. \quad (61)$$

Let $\mathcal{A}_{\text{gray}}$ be the algorithm which applies the gray world assumption, i.e.

$$o_i(x,y) = \frac{1}{3}\frac{c_i(x,y)}{\frac{1}{n}\sum_{x',y'} c_i(x',y')}. \quad (62)$$

Here, a scaling factor of $\frac{1}{3}$ instead of $\frac{1}{2}$ is used. This scaling factor is required in order to reach a stable state. Note that after the first operation,

the average value of a single channel is $\frac{1}{3}$. Thus in the end, both requirements can be fulfilled simultaneously.

Let us denote the input image by $\mathcal{I}_0$. Comprehensive normalization iteratively applies the two algorithms $\mathcal{A}_{\text{norm}}$ and $\mathcal{A}_{\text{gray}}$ to a given image $\mathcal{I}_j$.

$$\mathcal{I}_{j+1} = \mathcal{A}_{\text{gray}}(\mathcal{A}_{\text{norm}}(\mathcal{I}_j)) \tag{63}$$

This process continues until convergence, i.e. $\mathcal{I}_{j+1} = \mathcal{I}_j$. Finlayson et al. (1998) have shown that by iteratively applying both normalization algorithms convergence is always reached. The algorithm may be stopped after the difference falls below a suitably defined threshold. Because of the normalization, all intensity information is lost. It may be added back to the image by scaling all color channels using the intensity of the original image. Let $L = w_r c_r + w_g c_g + w_b c_b$ be the lightness of the input color with weights $w_r = 0.2125$, $w_g = 0.7154$, $w_b = 0.0721$. Let $L' = w_r m_1 \frac{c_r}{c_b} + w_g m_2 \frac{c_g}{c_b} + w_b$ be the lightness of the output color then the adjusted output color is given by

$$\mathbf{o} = \frac{L}{L'} \left[m_1 \frac{c_r}{c_b}, m_2 \frac{c_g}{c_b}, 1 \right]. \tag{64}$$

Comprehensive normalization assumes that the illuminant is uniform over the entire image. A possible extension of this method is to use local space average color instead of global space average color. In other words, for each step of the comprehensive normalization procedure, we would first normalize the colors and then use local space average color instead of global space average color to do another normalization.

7 Experimental Results

Figure 10 shows results on the accuracy of estimating the illuminant using space average color. The input image, showing a number of randomly arranged matte surfaces, is taken from the database created by Barnard et al. (2002c). Barnard et al. created a set of test images in order to test color constancy algorithms[1]. The images are linear, i.e. no gamma correction has been applied, and are therefore very dark. They were also purposely under-exposed in order to avoid any clipped pixel values. Detailed information on the spectral power distribution of all illuminants used and the response functions of the camera sensor is also available. Barnard et al. measured the color of the illuminant through the sensor of the camera by taking an image of a white reference standard.

Given this ground truth we can compare local space average color to the color which is computed by the different methods as described above. The second row of Figure 10 shows local space average computed once using an exponential kernel, once using a Gaussian kernel and once using a grid of processing elements or a resistive grid. A comparison between the estimated and the actual color of the illuminant along a horizontal line of the image is shown in the graphs below. The chromaticities of the estimated color matches the chromaticities of the actual illuminant closely in the center of the image. The angular error (defined in Eqn 67) and the actual color of the illuminant gets larger as the border of the image is approached due to the presence of a black background. The Gaussian kernel produces a smoother approximation compared to the results when either the exponential kernel or the resistive grid is used.

8 Evaluation of Algorithms

Color based object recognition can be used to evaluate the performance of the different algorithms (Funt et al., 1998). Color is an important cue and may be used to locate an object in a scene cluttered with other objects (Swain and Ballard, 1991). Here, we are not concerned with locating an object but in determining how well the algorithms compute color constant descriptors from the input data. Hence, we will use image matching to evaluate the performance of the different algorithms.

We again use the extensive database of images created by Barnard et al. (2002c) to test color constancy algorithms. The database consists of five different sets of images. Each set shows a variety of different objects illuminated by different illuminants. We down-sampled the images to 50% of the original size in order to speed up the evaluation. We also aligned the images from the first four sets with sub-pixel accuracy. In total, 19 images had to be removed from the resulting sets because these images could not be perfectly aligned. For instance, a ball was

[1] http://www.cs.sfu.ca/∼colour/data/index.html

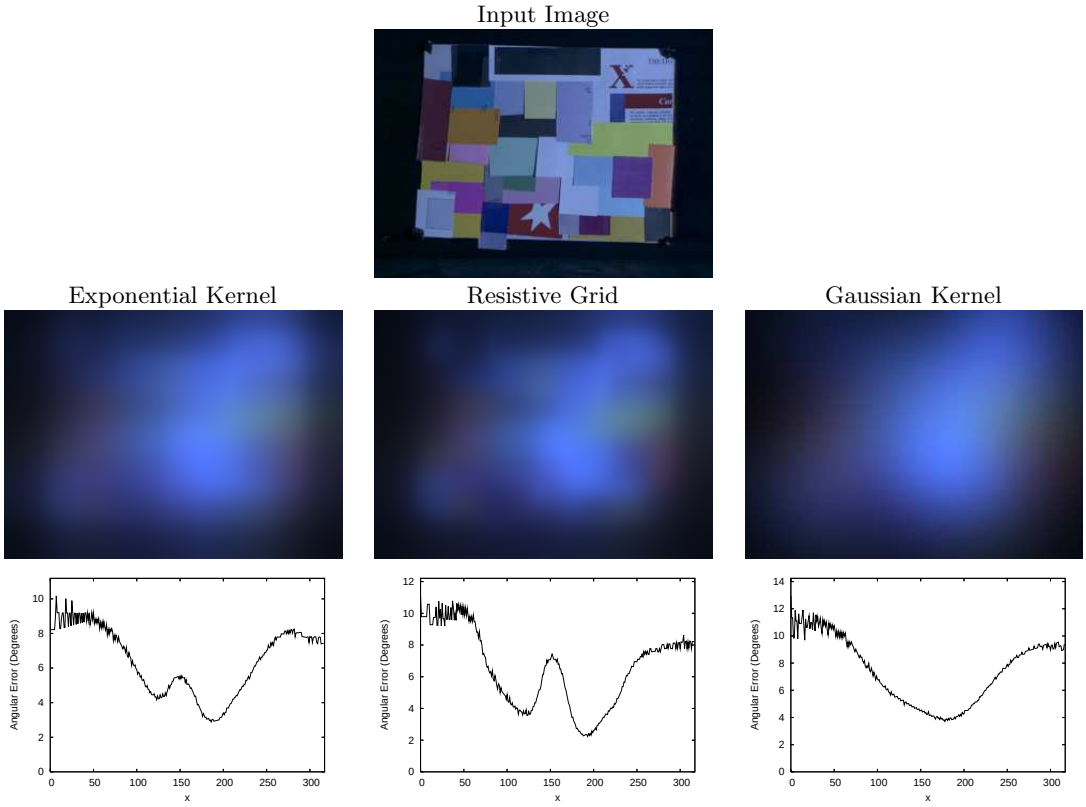

Fig. 10 Local space average color computed using an exponential kernel, a Gaussian kernel, and a resistive grid. The first image on the top is the input image which was taken from a database for color constancy research created by Barnard et al. (2002c). The graphs below show the angular error between the estimated and the actual color of the illuminant along a horizontal line of the image.

moved slighly by the experimenter whenever the light source was changed. Hence images of this ball were removed completely from the image sets. We confirmed that the alignment does not have a large influence on the recognition rate. In some cases, the recognition rate became slighly better, in other cases it became slightly worse. The reason why this alignment is necessary will become clear later on. Set 1 contains mainly Lambertian reflectors. Set 2 contains objects with metallic specularities. Set 3 contains objects with non-negligible dielectric specularities. Set 4 contains objects with at least one fluorescent surface. Set 5 is different from the previous sets. For set 5 the object was placed in a random position whenever the illuminant was changed.

The image sets of Barnard et al. show scenes which were illuminated by a single illuminant. In order to further test the performance of the different algorithms on scenes which were illuminated non-uniformly, we created additional image sets. Image sets 6 through 9 show natural scenes which were illuminated with a variety of different light sources including red, green, blue and yellow light bulbs. The light bulbs were just ordinary light bulbs with a color coating. We have placed the light sources to the left and to the right of the scene. Either one or both of the lights were turned on. In some scenes a desk lamp was present which was turned on for some of the images. Ceiling lamps and natural illumination was also present in some of the images. The images of sets 6, 7, and 8 were taken with an analog camera (a Canon EOS 5) and then digitized. Images of set 6 were digitized by ordering a CD-ROM of the developed film. Images of set 7 are from a Kodak Picture CD. Images of set 8 were scanned using a calibrated film scanner (a Canon FS4000US). Images of set 9 were taken with a digital camera (a Canon EOS 10D). The white balance was set to 6500K and the sRGB color space was used. The images

were also aligned with sub-pixel accuracy and down-sampled to a range of around 360×240 pixel for further processing. Table 1 describes all image sets used.

Let us assume that we are given a perfect color constancy algorithm which computes color constant descriptors which are independent of the illuminant. A perfect color constancy algorithm would compute the same output for each scene irrespective of the type of illuminant used. Note that in practice most color constancy algorithms are not perfect. For instance, many algorithms assume that the input image is free from shadows. If an image is fed into such an algorithm containing shadows then the shadows are still present in the output image. An algorithm which estimates reflectance or computes a descriptor based on reflectance (Finlayson et al., 2002; Finlayson and Hordley, 2001b; Geusebroek et al., 2001) should produce an output image free from shadows. Algorithms based on local space average color can only handle shadows which are large in extent and without any sharp boundaries. These algorithms only attenuate shadows but do not remove them entirely.

We run each input image through a color constancy algorithm and then perform image matching based on the color distribution of the output image. Use of the color distribution for image matching has the advantage that small shadow areas do not have a large impact on the color distribution of the image. The recognition rate will depend on the color constancy algorithm being able to compute a color constant output. Recognition rates will be lower for color constancy algorithms which compute color descriptors which are only approximately color constant. Of course, measuring recognition rates, only tests for illuminant invariance. It does not test for equivalence to a perceived color.

We evaluate each algorithm by applying it to every image of the image set. Recognition performance is evaluated by selecting two color corrected images for each scene. One is the test image the other one is the model image. A histogram is computed to establish a match between the test image and one of the possible model images. This method of histogram based object recognition was introduced by Swain and Ballard (1991). Swain and Ballard originally used histogram intersection in order to establish the best match. Schiele and Crowley (1996, 2000) proposed the χ^2 divergence measure. Let H_T be the color histogram of the test image and and let H_M be the color histogram of the model image. Let $H(\mathbf{c})$ be the probability that the color $\mathbf{c}$ occurs in the image. The χ^2 divergence measure is then computed as

$$\chi^2(H_T, H_M) = \sum_{\mathbf{c}} \frac{(H_T(\mathbf{c}) - H_M(\mathbf{c}))^2}{H_T(\mathbf{c}) + H_M(\mathbf{c})}. \qquad (65)$$

The lowest divergence measure tells us which model image best matches the data of the test image. This is done for all images of the test set. The selection of test and model images and subsequent matching is repeated 100 times to determine the recognition rate. Note that if we assume that we have n different illuminants per scene and m different scenes then we would have to carry out $\left(\frac{n(n-1)}{2}\right)^m$ experiments to cover all possible combinations of model and test images. With $n \approx 10$ and usually $m > 10$ this is infeasible. The recognition rate is reported as the number of times the correct model image is chosen. A recognition rate of 1.0 would signal that all model images are correctly matched to the test images all of the time.

Table 3 shows object recognition results for the white patch Retinex algorithm, scaling using global space average color, the gray edge hypothesis, the simplified version of Horn's algorithm (Horn, 1974, 1986), the algorithm of Blake (1985), the algorithm of Moore et al. (1991), the algorithm of Rahman et al. (1999), the gamut constraint method of Forsyth (1988, 1990), comprehensive normalization (Finlayson et al., 1998), scaling using local space average color and comprehensive normalization using local space average color. We also report results for a very simple algorithm which transforms each color band to the range $[0, 1]$ (Full Range Per Band). The parameters which were used for the different algorithms are summarized in Table 2. Note that Table 4 shows the performance on the same data sets except that the histograms were computed in RGB chromaticity space. Computation of color constant descriptors using local space average color gives best results for image sets 1 through 4 when histogram matching is done in RGB color space. A perfect recognition rate was achieved for sets 3 and 4. Computation of comprehensive local space average color gives best results on set 5. If the comparison is done in RGB chromaticity space, the algorithm of Moore et al. which also computes

Table 1 Image sets. Image sets 1 through 5 were created by Barnard et al. (2002c). Image sets 6 through 9 contain images showing natural scenes with spatially varying illuminants. Images of sets 1-4 and 6-8 were aligned with sub-pixel accuracy.

No.	Name	No. of Scenes	Images per Scene	No. of Images	Alignment
1	Lambertian Objects	21	2-11	212	aligned
2	Metallic Objects	14	9-11	146	aligned
3	Specular Objects	9	4-11	83	aligned
4	Fluorescent Objects	6	9-11	59	aligned
5	Different Objects	20	11	220	not
6	Natural Scenes (CD-ROM)	4	13-27	73	aligned
7	Natural Scenes (Kodak Photo CD)	5	5-27	54	aligned
8	Natural Scenes (Canon FS4000US)	9	4-28	130	aligned
9	Natural Scenes (Canon 10D)	10	12-20	137	not

Table 2 Parameters used for the color constancy algorithms.

Name of Algorithm	Parameters
Full Range Per Band	Transforms each color band to the range $[0, 1]$.
White Patch Retinex	Rescaling of each channel was done at 4%.
Gray World Assumption	The scaling factor was determined automatically from the image by scaling all pixels such that only 2% of the pixels are clipped.
Gray Edge Hypothesis	The parameters were set as follows: $n = 2$, $p = 7$, and $\sigma = 5$. The scaling factor was determined automatically from the image by scaling all pixels such that only 2% of the pixels are clipped.
Simplified Algorithm of Horn	No parameters required.
Horn (1974)/Blake (1985)	The threshold was set to 3.
Moore et al. (1991)	Local space average color was computed using an exponential kernel with a sigma of $\sigma = 0.17s$ where $\mathrm{s} = \max\{n_x, n_y\}/2$ and n_x is the width of the image and n_y is the height of the input image.
Rahman et al. (1999)	Three exponential kernels are applied with $\sigma_j \in \{1s, 0.17s, 0.05s\}$ where $\mathrm{s} = \max\{n_x, n_y\}/2$ and n_x is the width of the image and n_y is the height of the input image.
Gamut Constraint 3D	If an empty intersection is created, the convex hulls are increased by a small amount. This process is repeated until a non-empty intersection is achieved.
Comprehensive Normalization	The intensity of the original image was used to add back the original shading information.
Local Space Average Color	The scaling factor was set to 2. Local space average color was computed using an exponential kernel with a sigma of $\sigma = 0.17s$ where $\mathrm{s} = \max\{n_x, n_y\}/2$ and n_x is the width of the image and n_y is the height of the input image.
Comprehensive Local Space Average Color	Five levels were used for normalization. Local space average color was computed using an exponential kernel with a sigma of $\sigma = 0.17s$ where $\mathrm{s} = \max\{n_x, n_y\}/2$ and n_x is the width of the image and n_y is the height of the input image. The intensity of the original image was used to add back the original shading information.

local space average color performed best on set 5. The algorithm of Rahman performed best on sets 1, 3, and 4 with a perfect recognition rate on sets 3 and 4. A perfect recognition rate was also achieved using simple scaling by local space average color for set 4. The algorithm of Moore et al. performed best on set 5. Comprehensive local space average color performed best on set 2.

The recognition rates on image set 5 is much lower across all of the algorithms except for comprehensive local space average color, compared to the recognition rates for sets 1 through 4. This is due to the fact that object recognition becomes much harder once the object to

Table 3 Object recognition results for image sets 1 through 9. Histograms were computed in RGB space. Random performance is also shown. Best performance is marked in bold.

Algorithm	1	2	3	4	5	6	7	8	9
Random Recognition Rate	0.048	0.071	0.111	0.167	0.050	0.250	0.200	0.111	0.100
Full Range Per Band	0.562	0.524	0.764	0.702	0.483	0.400	0.418	0.346	0.265
White Patch Retinex	0.613	0.672	0.939	0.885	0.409	0.370	0.332	0.369	0.290
Gray World Assumption	0.789	0.664	0.960	0.872	0.428	0.380	0.332	0.363	0.315
Gray Edge Hypothesis	0.691	0.606	0.871	0.810	0.400	0.410	0.276	0.351	0.289
Simplified Horn	0.462	0.451	0.744	0.617	0.182	0.507	**0.468**	0.328	0.169
Horn (1974)/Blake (1985)	0.480	0.409	0.597	0.613	0.254	0.435	0.438	0.304	0.152
Moore et al. (1991) Retinex	0.913	0.805	0.796	0.825	0.500	0.525	0.356	0.414	0.418
Rahman et al. (1999)	0.877	0.832	0.787	0.845	0.496	0.410	0.328	0.376	0.388
Gamut Constraint 3D	0.555	0.513	0.741	0.707	0.468	0.398	0.370	0.340	0.263
Comprehensive Normalization	0.593	0.533	0.889	0.820	0.236	0.410	0.386	0.373	0.367
Local Space Average Color	**0.952**	**0.884**	**1.000**	**1.000**	0.537	0.545	0.432	**0.493**	0.417
Comprehensive L.S.A. Color	0.487	0.600	0.622	0.670	**0.567**	**0.565**	0.448	0.441	**0.491**

Table 4 Object recognition results for image sets 1 through 9. Histograms were computed in chromaticity space. Random performance is also shown. Best performance is marked in bold.

Algorithm	1	2	3	4	5	6	7	8	9
Random Recognition Rate	0.048	0.071	0.111	0.167	0.050	0.250	0.200	0.111	0.100
Full Range Per Band	0.574	0.482	0.723	0.767	0.357	0.495	0.412	0.303	0.172
White Patch Retinex	0.696	0.735	0.969	0.960	0.429	0.443	0.312	0.330	0.202
Gray World Assumption	0.820	0.704	0.978	0.930	0.441	0.497	0.346	0.328	0.222
Gray Edge Hypothesis	0.730	0.634	0.861	0.858	0.427	0.505	0.326	0.326	0.206
Simplified Horn	0.463	0.422	0.624	0.637	0.251	0.453	0.470	0.316	0.138
Horn (1974)/Blake (1985)	0.370	0.362	0.500	0.643	0.176	0.395	0.410	0.294	0.139
Moore et al. (1991) Retinex	0.940	0.918	0.991	0.985	**0.552**	0.565	0.440	0.449	0.335
Rahman et al. (1999)	**0.955**	0.904	**1.000**	**1.000**	0.485	0.485	0.412	0.404	0.316
Gamut Constraint 3D	0.571	0.478	0.696	0.730	0.354	0.507	0.340	0.300	0.159
Comprehensive Normalization	0.727	0.632	0.970	0.952	0.166	0.490	0.404	0.348	0.269
Local Space Average Color	0.945	0.912	0.998	**1.000**	0.528	0.492	0.414	0.442	0.313
Comprehensive L.S.A. Color	0.930	**0.922**	0.968	0.998	0.467	**0.652**	**0.516**	**0.519**	**0.358**

be recognized may be in arbitrary pose. Note that even though a single illuminant was used to illuminate the scenes of sets 1 through 5, the algorithms based on local space average color perform better than the standard gray world assumption on sets 1, 2 and 5. The algorithms based on local space average color are also able to compensate for any illumination intensity gradients which may have been present in the image. Note that once we move to chromaticities, the difference between the performance of the algorithms based on local space average color and the standard gray world assumption is reduced.

Results for image sets 6 through 9 are also shown in Table 3 and Table 4. Image matching in the presence of a non-uniform illumination is a much more difficult problem. Thus, the recognition rates are lower compared to the results on image sets 1 through 5. Another cause for lower performance of the algorithms on sets 6 through 9 is that the data is processed multiple times (first through the response function of the film and then again through the response function of the digitizer). The simplified algorithm of Horn performed best on set 7. Computation of color constant descriptors using local space average color gave best results on image set 8 if the comparison is done in RGB color space. Comprehensive local space average color performed best on sets 6 and 9. If the comparison is done in RGB chromaticity space then comprehensive local space average color gave best results for sets 6 through 9. The interested reader is referred to Ebner (2007a) for additional data and tests on the performance of the algorithms.

Evidently, algorithms using local space average color perform very well on image matching tasks. Why is this the case? In order to investigate the matter further, we computed the

average standard deviation σ_1 of the image pixels across the different scenes after a color constancy algorithm has been applied. This is the reason why we require aligned images. A perfect color constancy algorithm would have an average standard deviation of zero for all image pixels, i.e. each pixel would be mapped to the object reflectance or some other constant value. However, simply mapping the data to some constant value (whatever the input may be) is not good enough for object recognition. The result has to be non-trivial. Object recognition becomes easier the better the color space is used. Thus, let us also compute the standard deviation σ_2 of the image pixels within a single image. We do not want a constant output across all image pixels of a single image. The smaller the first measure (σ_1), the better the color constancy algorithm. The larger the second measure (σ_2), the larger the spread of the output pixels. In other words, algorithms with a large standard deviation σ_2 map the input colors to a large part of the available color space. Algorithms with a small standard deviation may only use a subspace of the available color space. Given the two standard deviations, we can compute a color constancy measure m.

$$m = \frac{\sigma_2}{\sigma_1} \tag{66}$$

When we look at the correlation between the color constancy measure and the performance on the image matching task as shown in Figure 11, we find that the measured data correlate very well. The higher the color constancy measure the higher the recognition rate. Algorithms with a low color constancy measure perform poorly on the image matching task. The algorithms based on local space average color and the algorithm which used local space average color to scale each channel by twice the local space average color in particular is very good at producing a color constant output but at the same time also uses a large part of the available color space.

Since the ground truth data is available for the illuminant used for data sets 1 through 5, we can compare the color of the illuminant which was estimated by the algorithms to the actual color of the illuminant. Barnard et al. (2002a,b) performed an in depth study on how accurately color constancy algorithms estimate the color of the illuminant. They conducted experiments using synthetic as well as real images. We will focus on real images here. Barnard et al. used the angular error between the actual and the estimated illuminant as a primary error measure. They also measured the distance between the actual and the estimated illuminant in RG chromaticity space. For each algorithm, Barnard et al. (2002a,b) report the root mean squared error of these error measures. For some algorithms they also report illuminant RGB error and the brightness error ($R+G+B$). Both are not considered here.

Let $\mathbf{L}_E = [L_{E,r}, L_{E,g}, L_{E,b}]$ be the estimated illuminant. Let $\mathbf{L}_T = [L_{T,r}, L_{T,g}, L_{T,b}]$ be the actual illuminant. Then the angular error E_α is given by

$$E_\alpha = \cos^{-1} \frac{\mathbf{L}_E \cdot \mathbf{L}_T}{|\mathbf{L}_E||\mathbf{L}_T|}. \tag{67}$$

The second error measure is based on chromaticities. Let $\hat{\mathbf{L}}_E = \frac{1}{L_{E,r}+L_{E,g}+L_{E,b}}[L_{E,r}, L_{E,g}, L_{E,b}]$ be the chromaticity of the estimated illuminant. Let $\hat{\mathbf{L}}_T = \frac{1}{L_{T,r}+L_{T,g}+L_{T,b}}[L_{T,r}, L_{T,g}, L_{T,b}]$ be the chromaticity of the actual illuminant. Then the distance E_d between the estimated and the actual illuminant is given by

$$E_d = \sqrt{(\hat{L}_{E,r} - \hat{L}_{T,r})^2 + (\hat{L}_{E,g} - \hat{L}_{T,g})^2}. \tag{68}$$

Barnard et al. (2002b) suggest to exclude pixels less than two because results for those pixels may be very noisy.

We computed the same error for the algorithms which are described above. The results are shown in Table 5. Only algorithms which estimate the color of the illuminant are included in the table. Computation of local space average color estimates the color of the illuminant locally for each image pixel. In order to compare the performance of this algorithm with the ground truth data, the average of the local estimates was computed and then compared to the ground truth data. None of the algorithms is able to estimate the color of the illuminant with an angular error less than 8 degrees. Since the data set of Barnard et al. do not contain any clipped pixels, simply transforming the given range of each band to the range $[0, 1]$ works quite well. Barnard et al. (2002b) report very good results for the 3D gamut constraint algorithm developed by Forsyth (1988, 1990). However, neither the gamut constraint algorithm nor the algorithm which transforms each band to the

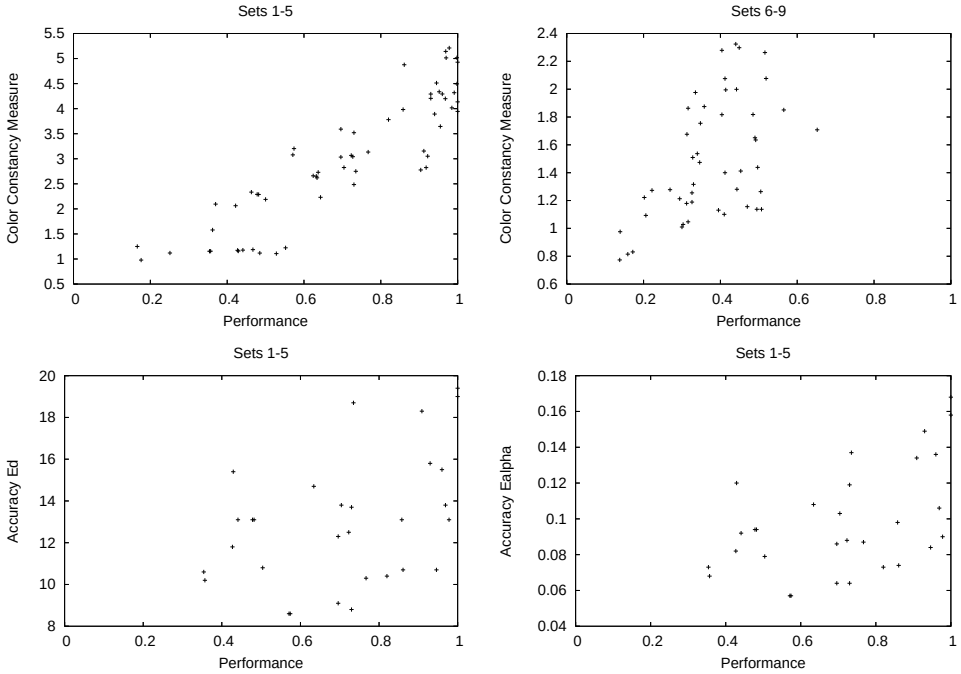

Fig. 11 The top two graphs show the correlation between the color constancy measure and image matching performance for sets 1 through 5 and sets 6 through 9. A very good correlation (0.895) is observed for sets 1 through 5. The correlation for sets 6 through 9 is not as good (0.539) which is not surprising given that the images from sets 6 through 9 are from a variety of sources. The correlation between the ability to accurately estimate the illuminant and the performance at the image matching task is rather low, it is 0.423 and 0.509 respectively. The correlation plot between the measure E_d and image matching performance is shown in the bottom left. The correlation plot for the measure E_α is shown on the bottom right.

Table 5 Comparison to ground truth data for image sets 1 through 5.

Algorithm	1		2		3		4		5	
	E_α	E_d	E_α	E_d	E_α	E_d	E_α	E_d	E_α	E_d
Full Range Per Band	**8.6**	**0.057**	**13.1**	**0.094**	12.5	0.088	**10.3**	**0.087**	**10.2**	**0.068**
White Patch Retinex	9.1	0.064	18.7	0.137	13.8	0.106	15.5	0.136	15.4	0.120
Gray World Assumption	10.4	0.073	13.8	0.103	13.1	0.090	15.8	0.149	13.1	0.092
Gray Edge Hypothesis	8.8	0.064	14.7	0.108	**10.7**	**0.074**	13.1	0.098	11.8	0.082
Gamut Constraint 3D	**8.6**	**0.057**	**13.1**	**0.094**	12.3	0.086	13.7	0.119	10.6	0.073
Local Space Average Color	10.7	0.084	18.3	0.134	19.4	0.158	19.0	0.168	10.8	0.079

range $[0, 1]$ performed very well on the image matching task. Even if an algorithm performs well on the image matching task it may not necessarily be the case that this algorithm also accurately estimates the illuminant. The good performance is also in part due to the use of the entire available color space.

9 Conclusion

Natural scenes frequently have multiple illuminants. A room may be illuminated by artificial light as well as reflected sunlight. Even if there is only a single illuminant, the intensity of the illuminant usually varies across the image. In order to compute color constant descriptors from the measured data, one has to estimate the illuminant locally for each image pixel. We have looked at several different methods on how to estimate the illuminant locally. A simple yet very effective method is the use of local space average color. We also described how such an algorithm can be integrated directly into the imaging device. Algorithms using local space average color were evaluated on several different image sets.

It was found that algorithms using local space average color are particularly suited for object recognition tasks.

References

Vivek Agarwal, Besma R. Abidi, Andreas Koschan, and Mongi A. Abidi. An overview of color constancy algorithms. *Journal of Pattern Recognition Research*, 1(1):42–54, 2006.

George S. Almasi and Allan Gottlieb. *Highly Parallel Computing*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, California, 1994. ISBN 0-8053-0443-6.

Kobus Barnard, Graham Finlayson, and Brian Funt. Color constancy for scenes with varying illumination. *Computer Vision and Image Understanding*, 65(2):311–321, February 1997.

Kobus Barnard, Lindsay Martin, and Brian Funt. Colour by correlation in a three dimensional colour space. In David Vernon, editor, *Proceedings of the 6th European Conference on Computer Vision, Dublin, Ireland*, pages 375–389, Berlin, 2000. Springer-Verlag.

Kobus Barnard, Florian Ciurea, and Brian Funt. Sensor sharpening for computational color constancy. *Journal of the Optical Society of America A*, 18(11):2728–2743, November 2001.

Kobus Barnard, Vlad Cardei, and Brian Funt. A comparison of computational color constancy algorithms – part I: Methodology and experiments with synthesized data. *IEEE Transactions on Image Processing*, 11(9):972–984, September 2002a.

Kobus Barnard, Lindsay Martin, Adam Coath, and Brian Funt. A comparison of computational color constancy algorithms – part II: Experiments with image data. *IEEE Transactions on Image Processing*, 11(9):985–996, September 2002b.

Kobus Barnard, Lindsay Martin, Brian Funt, and Adam Coath. A data set for color research. *Color Research and Application*, 27 (3):147–151, 2002c.

Andrew Blake. Boundary conditions for lightness computation in mondrian world. *Computer Vision, Graphics, and Image Processing*, 32:314–327, 1985.

David H. Brainard and William T. Freeman. Bayesian color constancy. *Journal of the Optical Society of America A*, 14(7):1393–1411, July 1997.

I. N. Bronstein, K. A. Semendjajew, G. Musiol, and H. Mühling. *Taschenbuch der Mathematik*. Verlag Harri Deutsch, Thun und Frankfurt/Main, 5. edition, 2001. ISBN 3-87171-2015-X.

G. Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin Institute*, 310(1):337–350, July 1980.

James Demmel. Cs267: Lectures 15 and 16. Solving the discrete Poisson equation using Jacobi, SOR, conjugate gradients, and the FFT. Computer Science Division, University of California at Berkeley, 1996.

M. D'Zmura and P. Lennie. Mechanisms of color constancy. In Glenn E. Healey, Steven A. Shafer, and Lawrence B. Wolff, editors, *Color*, pages 224–234, Boston, 1992. Jones and Bartlett Publishers.

Marc Ebner. Combining white-patch retinex and the gray world assumption to achieve color constancy for multiple illuminants. In Bernd Michaelis and Gerald Krell, editors, *Pattern Recognition, Proceedings of the 25th DAGM Symposium, Magdeburg, Germany*, pages 60–67, Berlin, 2003. Springer-Verlag. ISBN 3-540-40861-4.

Marc Ebner. A parallel algorithm for color constancy. *Journal of Parallel and Distributed Computing*, 64(1):79–88, 2004.

Marc Ebner. Evolving color constancy. *Special Issue on Evolutionary Computer Vision and Image Understanding of Pattern Recognition Letters*, 27(11):1220–1229, 2006.

Marc Ebner. *Color Constancy*. John Wiley & Sons, England, 2007a.

Marc Ebner. How does the brain arrive at a color constant descriptor? In Francesco Mele, Giuliana Ramella, Silvia Santillo, and Francesco Ventriglia, editors, *Proceedings of the 2nd International Symposium on Brain, Vision and Artificial Intelligence, 10-12 October, 2007, Naples, Italy*, pages 84–93, Berlin, 2007b. Springer.

Marc Ebner and Christan Herrmann. On determining the color of the illuminant using the dichromatic reflection model. In Walter Kropatsch, Robert Sablatnig, and Allan Hanbury, editors, *Pattern Recognition, Proceedings of the 27th DAGM Symposium, Vienna, Austria*, pages 1–8, Berlin, 2005. Springer-Verlag.

G. D. Finlayson. Color in perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1034–1038, October 1996.

G. D. Finlayson, S. S. Chatterjee, and B. V. Funt. Color angle invariants for object recognition. In *Proceedings of the Third IS&T/SID Color Imaging Conference: Color Science, Systems and Applications, Nov. 7-10, The Radisson Resort, Scottsdale, Arizona*, pages 44–47, 1995.

Graham Finlayson and Steven Hordley. Colour signal processing which removes illuminant colour temperature dependency. *UK Patent Application GB 2360660A*, January 2001a.

Graham D. Finlayson and B. V. Funt. Coefficient channels: Derivation and relationship to other theoretical studies. *COLOR research and application*, 21(2):87–96, April 1996.

Graham D. Finlayson and Steven D. Hordley. Color constancy at a pixel. *Journal of the Optical Society of America A*, 18(2):253–264, February 2001b.

Graham D. Finlayson and Gerald Schaefer. Solving for colour constancy using a constrained dichromatic reflection model. *International Journal of Computer Vision*, 42(3): 127–144, 2001.

Graham D. Finlayson and Elisabetta Trezzi. Shades of gray and colour constancy. In *Proceedings of the IS&T/SID Twelfth Color Imaging Conference*, pages 37–41, 2004.

Graham D. Finlayson, Mark S. Drew, and Brian V. Funt. Spectral sharpening: sensor transformations for improved color constancy. *Journal of the Optical Society of America A*, 11(4):1553–1563, April 1994a.

Graham D. Finlayson, Mark S. Drew, and Brian V. Funt. Color constancy: generalized diagonal transforms suffice. *Journal of the Optical Society of America A*, 11(11):3011–3019, November 1994b.

Graham D. Finlayson, Paul M. Hubel, and Steven Hordley. Color by correlation. In *Proceedings of IS&T/SID. The Fifth Color Imaging Conference: Color Science, Systems, and Applications, Nov 17-20, The Radisson Resort, Scottsdale, AZ*, pages 6–11, 1997.

Graham D. Finlayson, Bernt Schiele, and James L. Crowley. Comprehensive colour image normalization. In Hans Burkhardt and Bernd Neumann, editors, *Fifth European Conference on Computer Vision (ECCV '98), Freiburg, Germany*, pages 475–490, Berlin, 1998. Springer-Verlag.

Graham D. Finlayson, Steven Hordley, and Paul M. Pubel. Color by correlation: A simple, unifying framework for color constancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1209–1221, November 2001.

Graham D. Finlayson, Steven D. Hordley, and Mark S. Drew. Removing shadows from images. In *Proceedings of the European Conference on Computer Vision*, pages 823–836, Berlin, 2002. Springer-Verlag.

Graham D. Finlayson, Mark S. Drew, and Cheng Lu. Intrinsic images by entropy minimization. In Tomáš Pajdla and Jiří Matas, editors, *Proceedings of the 8th European Conference on Computer Vision, Part III,Prague, Czech Republic, May, 2004*, pages 582–595, Berlin, 2004. Springer-Verlag.

D. A. Forsyth. A novel approach to colour constancy. In *Second International Conference on Computer Vision (Tampa, FL, Dec. 5-8)*, pages 9–18. IEEE Press, 1988.

D. A. Forsyth. A novel algorithm for color constancy. *International Journal of Computer Vision*, 5(1):5–36, 1990.

D. A. Forsyth. A novel algorithm for color constancy. In Glenn E. Healey, Steven A. Shafer, and Lawrence B. Wolff, editors, *Color*, pages 241–271, Boston, 1992. Jones and Bartlett Publishers.

Brian Funt, Vlad Cardei, and Kobus Barnard. Learning color constancy. In *Proceedings of the IS&T/SID Fourth Color Imaging Conference*, pages 58–60, Scottsdale, 19-22 November 1996.

Brian Funt, Kobus Barnard, and Lindsay Martin. Is machine colour constancy good enough? In Hans Burkhardt and Bernd Neumann, editors, *Fifth European Conference on Computer Vision (ECCV '98), Freiburg, Germany*, pages 445–459, Berlin, 1998. Springer-Verlag.

Ron Gershon, Allan D. Jepson, and John K. Tsotsos. From [R,G,B] to surface reflectance: Computing color constant descriptors in images. In John P. McDermott, editor, *Proceedings of the Tenth International Joint Conference on Artificial Intelligence, Milan, Italy*, volume 2, pages 755–758. Morgan Kaufmann, 1987.

Jan-Mark Geusebroek, Rein van den Boomgaard, Arnold W. M. Smeulders, and Hugo Geerts. Color invariance. *IEEE Transac-*

tions on Pattern Analysis and Machine Intelligence, 23(12):1338–1350, December 2001.

Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1992. ISBN 0-201-60078-1.

Harry Helson. Fundamental problems in color vision. I. the principle governing changes in hue, saturation, and lightness of non-selective samples in chromatic illumination. *Journal of Experimental Psychology*, 23(5):439–476, November 1938.

Berthold K. P. Horn. Determining lightness from an image. *Computer Graphics and Image Processing*, 3:277–299, 1974.

Berthold Klaus Paul Horn. *Robot Vision*. The MIT Press, Cambridge, Massachusetts, 1986. ISBN 0-262-08159-8.

Bernd Jähne. *Digitale Bildverarbeitung*. Springer-Verlag, Berlin, fifth edition, 2002. ISBN 3-540-41260-3.

Ramesh Jain, Rangachar Kasturi, and Brian G. Schunck. *Machine Vision*. McGraw-Hill, Inc., New York, 1995. ISBN 0-07-113407-7.

Deane B. Judd. Appraisal of Land's work on two-primary color projections. *Journal of the Optical Society of America*, 50(3):254–268, 1960.

Edwin H. Land. An alternative technique for the computation of the designator in the retinex theory of color vision. *Proc. Natl. Acad. Sci. USA*, 83:3078–3080, May 1986.

Edwin H. Land and John J. McCann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61(1):1–11, January 1971.

John J. McCann, Suzanne P. McKee, and Thomas H. Taylor. Quantitative studies in retinex theory. *Vision Res.*, 16:445–458, 1976.

Andrew Moore, John Allman, and Rodney M. Goodman. A real-time neural system for color constancy. *IEEE Transactions on Neural Networks*, 2(2):237–247, March 1991.

Wolfgang Nolting. *Grundkurs: Theoretische Physik, Band 3 Elektrodynamik*. Verlag Zimmermann-Neufang, Ulmen, zweite edition, 1992. ISBN 3-922410-20-0.

Carol L. Novak and Steven A. Shafer. Supervised color constancy for machine vision. In Glenn E. Healey, Steven A. Shafer, and Lawrence B. Wolff, editors, *Color*, pages 284–299, Boston, 1992. Jones and Bartlett Publishers.

Jay Orear. *Physik*. Carl Hanser Verlag, München, Wien, 1982. ISBN 3-446-12977-4.

James R. Parker. *Algorithms for Image Processing and Computer Vision*. John Wiley & Sons, Inc., New York, 1997. ISBN 0-471-14056-2.

D. Paulus, L. Csink, and H. Niemann. Color cluster rotation. In *Proceedings of the International Conference on Image Processing (ICIP)*, pages 161–165. IEEE Computer Society Press, 1998.

T. Pomierski and H. M. Groß. Verfahren zur empfindungsgemäßen Farbumstimmung. In G. Sagerer, S. Posch, and F. Kummert, editors, *Mustererkennung 1995, Verstehen akustischer und visueller Informationen, 17. DAGM-Symposium, Bielefeld, 13.-15. September 1995*, pages 473–480, Berlin, 1995. Springer-Verlag.

Zia-ur Rahman, Daniel J. Jobson, and Glenn A. Woodell. Method of improving a digital image. *United States Patent No. 5,991,456*, November 1999.

Valery J. Risson. Determination of an illuminant of digital color image by segmentation and filtering. *United States Patent Application, Pub. No. US 2003/0095704 A1*, May 2003.

Bernt Schiele and James L. Crowley. Object recognition using multidimensional receptive field histograms. In Bernard Buxton and Roberto Cipolla, editors, *Fourth European Conference On Computer Vision, Cambridge, UK, April 14-18*, pages 610–619, Berlin, 1996. Springer-Verlag.

Bernt Schiele and James L. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision*, 36(1): 31–52, 2000.

Linda G. Shapiro and George C. Stockman. *Computer Vision*. Prentice Hall, New Jersey, 2001. ISBN 0-13-030796-3.

Michael Stokes, Matthew Anderson, Srinivasan Chandrasekar, and Ricardo Motta. A standard default color space for the internet - sRGB. Technical report, Version 1.10, 1996.

Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.

Marshall F. Tappen, William T. Freeman, and Edward H. Adelson. Recovering intrinsic images from a single image. Technical Report AI Memo 2002-015, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, September 2002.

Shoji Tominaga. Surface identification using the dichromatic reflection model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):658–670, July 1991.

J. van de Weijer and Th. Gevers. Color constancy based on the grey-edge hypothesis. In *Proc. ICIP 2005, Genua, Italy*, October 2005.

J. van de Weijer, Th. Gevers, and A. Gijsenij. Edge-based color constancy. *IEEE Transactions on Image Processing*, 2007.

Joachim Weickert. A review of nonlinear diffusion filtering. In B. ter Haar Romeny, L. Florack, J. Koenderink, and M. Viergever, editors, *Scale-Space Theory in Computer Vision*, pages 3–28, Berlin, 1997. Springer-Verlag.

Yair Weiss. Deriving intrinsic images from image sequences. In *Proceedings of the International Conference on Computer Vision, Vancouver, Canada, July 9-12, 2001*. IEEE, 2001.

Eric W. Weisstein. Hankel transform. From *MathWorld* – A Wolfram Web Resource. http://mathworld.wolfram.com/HankelTransform.html, 1999.

Semir Zeki. *A Vision of the Brain*. Blackwell Science, Oxford, 1993. ISBN 0-632-03054-2.

Marc Ebnerreceived the M.S. degree in computer science from New York University, NY, in 1994, the Dipl.-Inform. degree from the Universität Stuttgart, Germany, in 1996 and the Dr. rer. nat. degree from the Universität Tübingen, Germany, in 1999. He received the venia legendi in 2006. He is a lecturer at the Julius Maximilians Universität Würzburg, Germany, and teaches computer graphics, virtual reality and evolutionary algorithms. He is author of over 40 peer-reviewed publications and is a frequent speaker at international conferences; particularly on areas such as machine intelligence, computer vision, biologically inspired systems and genetic programming. Dr. Ebner is author of the first textbook on color constancy, i.e. algorithms for automatic white balance which try to mimic human color perception. He serves or has served as a reviewer for many different technical journals such as Journal of the Optical Society of America A, IEEE Transactions on Pattern Analysis and Machine Intelligence, Image and Vision Computing, Color Research and Application, Evolutionary Computation, IEEE Transactions on Evolutionary Computation, IEEE Transactions on Systems, Man and Cybernetics - Part B, Genetic Programming and Evolvable Machines, Artificial Life and Journal of the Royal Society Interface, and is a member of the program committee for the major conferences on evolutionary algorithms.