Integrating Color Constancy into JPEG2000

Marc Ebner, German Tischler and Jürgen Albert *Member, IEEE*

***Abstract*—The human visual system is able to perceive colors as approximately constant. This ability is known as color constancy. In contrast, the colors measured by a sensor vary with the type of illuminant used. Color constancy is very important for digital photography and automatic color based object recognition. In digital photography this ability is known under the name automatic white balance. A number of algorithms have been developed for color constancy. We review two well known color constancy algorithms, the gray world assumption and the Retinex algorithm and show how a color constancy algorithm may be integrated into the JPEG2000 framework. Since computer images are usually stored in compressed form anyway, little overhead is required to add color constancy into the processing pipeline.**

***Index Terms*—Color Constancy, JPEG2000, wavelet transform**

I. Motivation

The human visual system is able to determine the color of objects from the spectral power distribution entering the eye. This ability to compute color constant or approximately color constant descriptors is called color constancy [1], [2]. Even though a number of theories exist, it is not known exactly how the human brain computes color constant descriptors. Color constancy is very important for many different areas such as consumer photography or automatic color based object recognition. Many different algorithms have been proposed in order to solve the problem of color constancy. One of the early algorithms is Land and McCann's original Retinex algorithm [3]. Other algorithms include gamut constraint methods [4]–[7], color by correlation [8], [9], color cluster rotation [10], or use of neural networks [11]. Whereas most color constancy algorithms assume that the objects shown in the image can be modeled as diffuse reflectors, some algorithms also take specular reflections into account [12]–[16]. Color constancy for images of unknown origin is discussed and evaluated by Cardei et al. [17], [18].

Today, most computer images are stored in a compressed form. The JPEG file format is ubiquitous. JPEG compression is achieved by segmenting the image into 8×8 blocks. These blocks are then subjected to a discrete cosine transform and only the most significant components are actually saved. In contrast, the JPEG2000 format, which is in many ways superior to the baseline JPEG format, is based on a transform of the entire image. We will show how a simple color constancy algorithm may be integrated directly into the JPEG2000 transform. All that needs to be done is to modify the decoding pipeline slightly. Alternatively one can also modify the encoding pipeline. If the decoding pipeline is modified, then the original image does not have to be color corrected permanently. The color correction can be applied optionally whenever the image is decoded.

In order to describe how the method works, we will first need to review relevant color constancy algorithms known from the literature.

The authors are with the Universität Würzburg, Lehrstuhl für Informatik II, Am Hubland, 97074 Würzburg, Germany. E-mail: {ebner,tischler,albert}@informatik.uni-wuerzburg.de Phone: (+49)931/888-6612, Fax: (+49)931/888-6603, http://wwwi2.informatik.uni-wuerzburg.de/staff/ebner/welcome.html

II. The Gray World Assumption

The problem of color constancy can only be solved if some assumptions are made. A frequently made assumption is that the illuminant is uniform across the image. An additional assumption which may be made is that the response curves of the camera's sensor are very narrow band. Another frequently made assumption which was introduced by Buchsbaum [19] is that, on average, the world is gray. We briefly review this algorithm because it is important to understand how color constancy may be added to the JPEG2000 format.

Suppose that we have a single light source illuminating the scene. Light falls onto an object patch and is reflected into the lens of the camera. Let $L(\lambda, \mathbf{x}_{\text{obj}})$ be the radiance given off by the light source for wavelength λ which is falling onto the object at position $\mathbf{x}_{\text{obj}}$. Some light is absorbed, the remainder is reflected. Let $\rho(\lambda, \mathbf{x}_{\text{obj}})$ be the reflectance function at object position $\mathbf{x}_{\text{obj}}$ for wavelength λ. We assume that the sensor located at position $\mathbf{x}_I$ measures the light being reflected from a corresponding object position $\mathbf{x}_{\text{obj}}$. The energy $\mathbf{I}(\mathbf{x}_I)$ measured by the sensor is then given as

$$\mathbf{I}(\mathbf{x}_I) = G(\mathbf{x}_{\text{obj}}) \int \rho(\lambda, \mathbf{x}_{\text{obj}}) L(\lambda, \mathbf{x}_{\text{obj}}) \mathbf{S}(\lambda) d\lambda \qquad (1)$$

where $G(\mathbf{x}_{\text{obj}})$ is a geometry factor which describes the local geometry at position $\mathbf{x}_{\text{obj}}$ and $\mathbf{S}(\lambda)$ is a vector which describes the sensor's response characteristics. The integration is done over all wavelengths λ. Usually, three sensors which respond to light in the red, green and blue part of the spectrum are used. In this case, $\mathbf{S}(\lambda)$ is a three element vector where $S_i(\lambda)$ describes the response function of the i-th sensor. Assuming that the object surface can be modeled as a Lambertian reflector, we have $G(\mathbf{x}_{\text{obj}}) = \mathbf{N}_L \cdot \mathbf{N}_{\text{obj}}$ where $\mathbf{N}_L$ is the unit vector pointing from the surface of the object into the direction of the light source and $\mathbf{N}_{\text{obj}}$ is the unit vector describing the surface normal. This model of color image formation is used by many algorithms for color constancy, e.g. [5]–[7], [14], [19]–[22].

The sensor's response functions are modeled as being very narrow band. If they are not narrow band, they may be sharpened [23]–[26]. We will be modeling the sensor's response functions as delta functions. In this case, we have $S_i(\lambda) = \delta(\lambda - \lambda_i)$ with $i \in \{r, g, b\}$. In other words, each

sensor responds to a single wavelength in the red, green and blue part of the spectrum. We now obtain

$$I_i(\mathbf{x}_I) = G(\mathbf{x}_{\text{obj}})\rho(\lambda_i, \mathbf{x}_{\text{obj}})L(\lambda_i, \mathbf{x}_{\text{obj}}). \quad (2)$$

Let us now move to image coordinates. Assuming a linear relationship between the measurements made by the sensor and image data, we obtain

$$c_i(x, y) = I_i(x, y) = G(x, y)\rho_i(x, y)L_i(x, y) \quad (3)$$

for the color $c_i(x, y)$ stored in color channel i at position (x, y) of the image. Note that we no longer use object position $\mathbf{x}_{\text{obj}}$ and the wavelength λ_i. Instead, we refer to the corresponding reflectance ρ and the illuminant L using the index of the color band i. If the illuminant is constant over the entire image, i.e. $L_i(x, y) = L_i$, we have

$$c_i(x, y) = G(x, y)\rho_i(x, y)L_i. \quad (4)$$

Thus, we now see that a simple diagonal transform suffices in order to obtain a color descriptor which is independent of the illuminant. If the illuminant $\mathbf{L} = [L_r, L_g, L_b]$ were known, we could simply divide each color channel by L_i. The result will be independent of the illuminant.

The illuminant can be estimated from the image data. We only have three measurements $c_i(x, y)$ available for each pixel. We do not know the reflectance $\rho(x, y)$ nor do we know the color of the illuminant $\mathbf{L}$. We also don't know anything about the geometry $G(x, y)$. In order to estimate the color of the illuminant some assumptions have to be made. Buchsbaum [19] suggested the gray world assumption. According to the gray world assumption, on average, the world is gray. To see how the gray world assumption works, we compute the average color over all pixels. Let N be the number of image pixels and let a_i be the computed average.

$$a_i = \frac{1}{N}\sum_{x,y} c_i(x, y) \quad (5)$$

Using the results from above, we obtain

$$a_i = \frac{1}{N}\sum_{x,y} G(x, y)\rho_i(x, y)L_i. \quad (6)$$

We now assume that there are many differently colored objects located in the image. We don't know which colors will be present. Therefore, we will assume that the reflectances are uniformly distributed over the range $[0, 1]$ and that geometry information is independent of reflectance. We then obtain

$$a_i = L_i\frac{1}{N}\sum_{x,y} G(x, y)\rho_i(x, y) \approx L_i E(G)E(\rho) \quad (7)$$

where $E(G)$ is the expected value of the geometry factor and $E(\rho)$ is the expected value of the reflectance. Since we have assumed a uniform distribution of the reflectances over the range $[0, 1]$, we have $E(\rho) = \frac{1}{2}$. Therefore, we obtain

$$a_i \approx L_i E(G)\frac{1}{2} \quad (8)$$

which can be solved for the illuminant L_i

$$L_i \approx \frac{2}{E(G)}a_i = f a_i \quad (9)$$

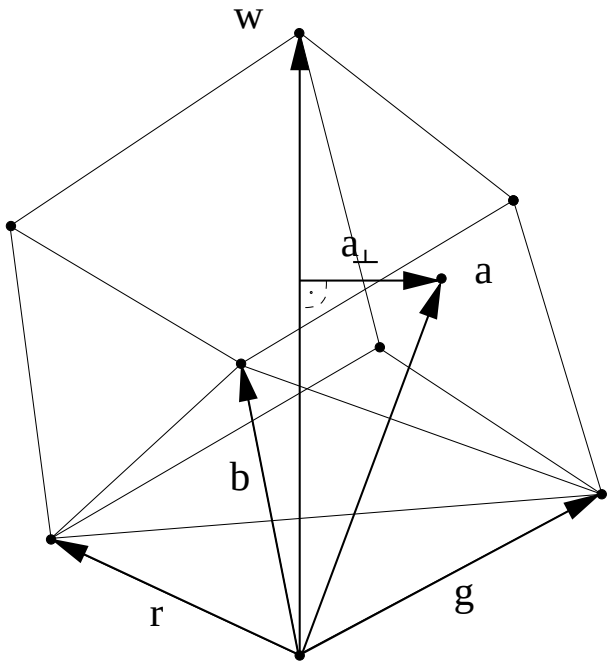


Figure 1. The RGB color cube is spanned by the three color vectors red (**r**), green (**g**) and blue (**b**). The gray vector (**w**) runs from black to white through the middle of the cube. A color correction may be achieved by moving local space average color **a** onto the gray vector **w**.

with $f = \frac{2}{E(G)}$. The expected value of the geometry factor, of course, depends on the image content. We now see that the color of the illuminant can be estimated from the average color of the image pixels. Since we now have an estimate of the illuminant, a color constant descriptor o_i may be computed as

$$o_i(x, y) = \frac{c_i(x, y)}{f a_i} \approx \frac{G(x, y)\rho_i(x, y)L_i}{L_i} = G(x, y)\rho_i(x, y). \quad (10)$$

After we divide each channel by twice the space average color, the global average will be at $[0.5, 0.5, 0.5]$. Ebner [27] has shown that the same principle may also be applied locally. By computing local space average color, we obtain an estimate of the illuminant locally for each image pixel. Another way to perform a color correction is the use of color shifts [28]. Local space average color can be used to perform a horizontal shift in the direction of the gray vector. This process is illustrated in Figure 1. First, local space average color $\mathbf{a}$ is computed. We then subtract the component $\mathbf{a}_\perp$ which is perpendicular to the gray vector $\mathbf{w}$ from the color of each pixel $\mathbf{c}$. This effectively moves the new average onto the gray vector.

III. The Retinex Algorithm

Another color constancy algorithm was suggested by Horn [29] and refined by Blake [30]. Horn formulated a two-dimensional variant of Land and McCann's Retinex algorithm [3]. He suggested to first separate the product between reflectance and the illuminant into a sum by applying the logarithm. We have already seen that the energy measured by the camera's sensor is proportional to the reflectance times the illuminant. If we assume a planar surface which is viewed

at a right angle, we have

$$c_i(x,y) = \rho_i(x,y) L_i(x,y). \tag{11}$$

Applying the logarithm, we obtain

$$\log c_i(x,y) = \log \rho_i(x,y) + \log L_i(x,y). \tag{12}$$

If we now assume that the illuminant varies smoothly across the image, we have sharp discontinuities at positions where the reflectance changes. Horn suggested to first apply the Laplacian to the logarithm of the input image. A 3×3 Laplacian computes the difference between adjacent pixels in all four directions and adds the results. Since we have assumed that the illuminant varies smoothly across the image, the second component will be almost identical for neighboring pixels. Thus, in computing the Laplacian, the second component will nearly cancel. Strong spikes will be located wherever the reflectance changes. Horn suggests to apply a threshold operation to the output of the Laplacian removing any small values. This sequence of operations can be summarized as

$$\Delta \log \rho_i(x,y) = \Theta(\Delta \log c_i(x,y)) \tag{13}$$

where Δ denotes the Laplacian and Θ denotes the thresholding operation. Next, the Laplacian is inverted and the result is transformed back to the range $[0,1]$. The result will be independent of the illuminant. Blake [30] extended the algorithm by splitting the computation of the Laplacian into two steps. First the derivative is computed, then the threshold operation is applied and finally another derivative is computed.

Suppose that the illuminant is constant across the image. In this case, we do not need the thresholding operation nor do we need the application of the Laplacian and subsequent integration. A color constant descriptor may be computed by simply applying the logarithm and then transforming the result to the range $[0,1]$. If we apply the logarithm, we obtain

$$\log c_i(x,y) = \log \rho_i(x,y) + \log L_i. \tag{14}$$

Since the second term is a constant, it will automatically be removed by this transformation.

Land [31] suggested an algorithm where the logarithm of the average color of points surrounding the given point is subtracted from the logarithm of the color of the given point. In other words, one computes

$$\log c_i - \log a_i = \log \frac{c_i}{a_i}. \tag{15}$$

We have seen above that a linear scaling of color channels is required in order to compute a color constant descriptor. However, due to the application of the logarithm, we now need to subtract $\log a_i$ from $\log c_i$. A variant of this method was implemented by Moore et al. [32] in VLSI.

A color constant descriptor can also be obtained if one computes global space average color after the logarithm has been applied and subtracts this value from the logarithm of the color at the given point. In this case, each pixel is divided by the geometric mean of the color channel [33], [34]. If we apply the logarithm first, and then compute space average color $\mathbf{a}'$, we obtain

$$\begin{aligned} a_i' &= \frac{1}{N} \sum_{x,y} \log c_i(x,y) && (16)\\ &= \frac{1}{N} \sum_{x,y} (\log \rho_i(x,y) + \log L_i) && (17)\\ &= \log L_i + \frac{1}{N} \sum_{x,y} \log \rho_i(x,y) && (18)\\ &= \log L_i + \log \left(\prod_{x,y} \rho_i(x,y) \right)^{\frac{1}{N}}. && (19) \end{aligned}$$

If we now subtract this value from the logarithm of the color of each pixel, we obtain

$$\begin{aligned} o_i(x,y) &= \log c_i(x,y) - a_i' && (20)\\ &= \log \left(\frac{c_i(x,y)}{\left(\prod_{x,y} c_i(x,y) \right)^{\frac{1}{n}}} \right) && (21)\\ &= \log \rho_i(x,y) + \log L_i \\ &\quad - \log L_i - \log \left(\prod_{x,y} \rho_i(x,y) \right)^{\frac{1}{N}} && (22)\\ &= \log \rho_i(x,y) - \log \left(\prod_{x,y} \rho_i(x,y) \right)^{\frac{1}{N}} && (23) \end{aligned}$$

which is again a color constant descriptor. The second term can be simplified if we assume that the reflectances are uniformly distributed over the range $[0,1]$. In this case, we can write the second term as

$$\frac{1}{n} \sum_{i=1}^{n} \log \left(\frac{i}{n} \right) = \frac{1}{n} \log \prod_{i=1}^{n} \frac{i}{n} = \frac{1}{n} \log \frac{n!}{n^n} = \log \frac{n!^{\frac{1}{n}}}{n}. \tag{24}$$

Assuming that n is sufficiently large, i.e. $n \to \infty$ and using Stirling's formula

$$\lim_{n \to \infty} \frac{(n!)^{\frac{1}{n}}}{n} = \frac{1}{e}, \tag{25}$$

we obtain

$$o_i(x,y) = \log \rho_i(x,y) + 1. \tag{26}$$

IV. Homomorphic Filtering

Stockham [35] suggested to perform image processing within a framework of the human visual system. Stockham originally worked with gray scale images. He noted that once the logarithm is applied to the image data, all operators which are subsequently applied, operate linearly on the illuminant and reflectance components of the original data. Faugeras [36] extended the approach to color image processing. He suggested homomorphic filtering for color image enhancement (see also Parker [37]). In homomorphic filtering, the image is transformed into a color space where the desired operation is easier to perform. If one assumes that the illuminant varies smoothly over the image and that sudden changes in the data are due to a change of reflectance, the change of the illuminant

is located in the low frequency components of the image and the reflectance changes are located in the high frequency components. Thus, it makes sense to transform the image into frequency space where low frequency changes can be attenuated and high frequency changes can be emphasized.

We will now see how a color constancy algorithm may be integrated into the JPEG2000 pipeline.

V. Image Compression using JPEG2000

A color image when encoded by JPEG2000 is basically fed through a color transform, a discrete wavelet transform and a quantizer [38], [39]. The resulting bit stream is encoded using an arithmetic coder. The two dimensional discrete wavelet transform is tree-structured in that a two dimensional sub-band transform is applied recursively to a low-pass filtered version of the image. This is illustrated for a sample image in Figure 2(a) which shows the transformed original image. The LL band (low pass filtered sub-band) is located in the upper left corner of the transformed image. The HL band (high pass filtered in the horizontal direction and low pass filtered in the vertical direction) is located in the upper right corner of the transformed image. The LH band (low pass filtered in the horizontal direction and high pass filtered in the vertical direction) is located in the lower left corner of the transformed image. The HH band (high pass filtered in both directions) is located in the lower right corner. This recursive sub-division is continued for D levels. The image shown in Figure 2(b) shows the result for $D = 5$ levels.

After this transformation, a recursively low pass filtered image resides in the upper left corner of the image. If we continue this process until we have only one pixel left, then this pixel would be the global space average color of the image. If we stop at an intermediate level, we would obtain local space average for the pixels which have been averaged so far. Figure 3 shows what happens if we take the contents of the LL band for $D = 7$ and rescale it to the size of the original image. We now have local space average color for each image pixel. Thus, the JPEG2000 compression algorithm essentially computes local space average color when the image is transformed. In order to produce a color constant or color adjusted output, we only need to modify the data of the LL sub-band of the highest level.

Figure 4 shows how a color constancy algorithm can be embedded into the JPEG2000 transform. The algorithm can either be included into the encoding or into the decoding pipeline. Inside the encoder, one can compute an arithmetic average from the data which is obtained from the LL-Band of the highest level. This arithmetic average can then be subtracted from the computed coefficients of the LL-Band of the highest level. The resulting effect is that the global space average color is pushed onto the gray vector. Alternatively, it is also possible to move the global space average color into the direction of the gray vector by a certain percentage. It is also possible to integrate this method into the decoder. In this case, the global space average color would be adjusted before the inverse wavelet transform is applied. This has the advantage that the image data can either be decoded exactly the same as it was encoded or it can be decoded with a color corrective step applied.

VI. Integrating Color Constancy into JPEG2000

An ideal encoding/decoding pipeline is shown in Figure 6(a). Encoding is done best in a uniform perceptual space [36]. We start off with linear RGB values. First, a color transform $\mathbf{P}_1$ is applied and we obtain values inside the CIE XYZ color space. Then the color space is made linear by applying a cube root function [40] and a second transform is applied. The result is a transformation to Lab color space. This is the ideal color space for encoding. The decoding reverses these steps. For display purposes a gamma correction has to be applied to obtain non-linear $R'G'B'$ values. The gamma correction has the following form

$$\text{gamma}(x) = x^{\gamma} \tag{27}$$

with $\gamma = 1/2.2$. The sRGB standard [41] uses a slightly different transform which also contains a linear section for small intensities.

Figure 6(b) shows the JPEG2000 encoding/decoding pipeline. The JPEG2000 pipeline deviates from the ideal encoding/decoding pipeline slightly. First a gamma correction is applied, next a color transform $\mathbf{P}_2$ is applied. Encoding is done inside the color space $Y'C_BC_R$. The data is encoded using a wavelet transform. After decoding, the inverse wavelet transform is used, the inverse color space transform is applied. The resulting non-linear R'G'B' values can then be shown directly on a display device. We see that the cube-root function has been moved outward compared to Figure 6(a). By moving the cube-root function outward, the function cancels nicely with the gamma correction. The result is a simplified pipeline and a small deviation from an end-to-end gamma factor of one [41].

The color transformation which is usually used with the JPEG2000 pipeline is given by

$$\begin{aligned} Y' &= 0.299R' + 0.587G' + 0.114B' \\ C_R &= 0.713(R' - Y') \\ C_B &= 0.564(B' - Y'). \end{aligned} \tag{28}$$

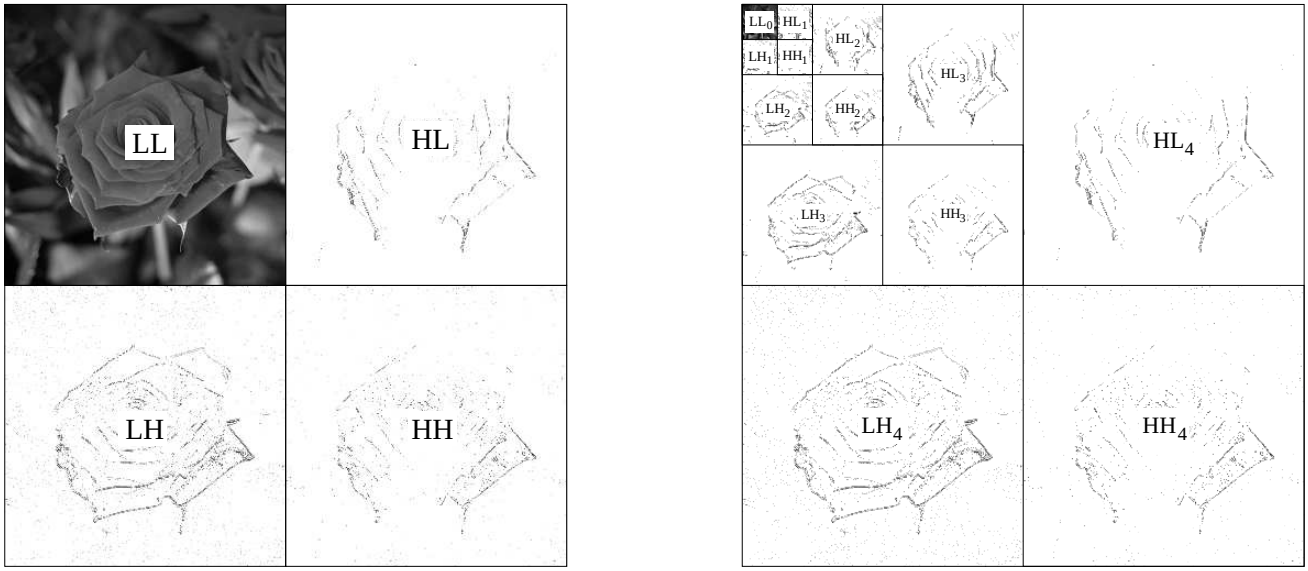
This transformation can also be written as

$$\begin{bmatrix} Y' \\ C_R \\ C_B \end{bmatrix} = \mathbf{P}_2 \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{pmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}.$$

This transformation cannot be perfectly inverted. That is why the following transformation is used by the JPEG2000 lossless compression.

$$\begin{aligned} Y' &= \lfloor \tfrac{R' + 2G' + B'}{4} \rfloor \\ C_R &= R' - G' \\ C_B &= B' - G' \end{aligned} \tag{29}$$

Application of this color transform is optional. It is not necessary for the encoder to make this transformation [39]. However, in practice, it is always applied. The color space now consists of the luma channel Y' and two color channels C_B and C_R. Luma Y' is computed from non-linear $R'G'B'$ signals [41]. It is not equivalent to luminance. Similarly, the



(a) (b)

Figure 2. (a) A sample image which is transformed using a two-dimensional wavelet transform. The image is decomposed into four sub-bands LL, HL, LH, HH which are low pass filtered, high pass filtered in the horizontal direction and low pass filtered in the vertical direction, low pass filtered in the horizontal direction and high pass filtered in the vertical direction and high pass filtered, respectively. (b) The same image transformed recursively for 5 levels.

two signals C_B and C_R are also computed from non-linear data.

In the derivation of the gray world assumption, we have assumed that we were working with linear RGB data, i.e.

$$c_i = I_i. \tag{30}$$

To use the gray world hypothesis, we would first have to restore this linearity by applying a gamma transform should this be necessary. Then we would compute global space average color and then divide each color channel by the computed global space average color. Finally, a gamma correction would again have to be applied in order to display the data. This is shown in Figure 6(c).

But what about the Retinex algorithm? If we want to use the Retinex algorithm, we would have to take the logarithm first, then compute space average color and then subtract the result from each channel. This is illustrated in Figure 6(d). If one applies the logarithm first, one needs to subtract space average color. As a result, the log-reflectances $\log \rho_i$ with $i \in \{r, g, b\}$ are obtained. Since the result has to be displayed, the application of the logarithm has to be undone and then a gamma correction has to be applied. A gamma factor which may have been applied originally is not relevant. This is, because after the logarithm has been applied, the gamma factor simply becomes a scaling factor.

Let us now have a look at Figure 5. The graph shows the gamma correction used by the sRGB standard. A fit of the function $a \log x + b$ to this data is also shown. The best fit was obtained using $a = 59.28$ and $b = -94.79$. Although the fit between the two curves is not exact, we can see that the general behavior of the two curves is similar. The similarity between a logarithmic function and a power-law function is also noted by Wyszecki and Stiles [42]. Thus, we can regard the gamma correction as an approximation to the logarithmic function. We

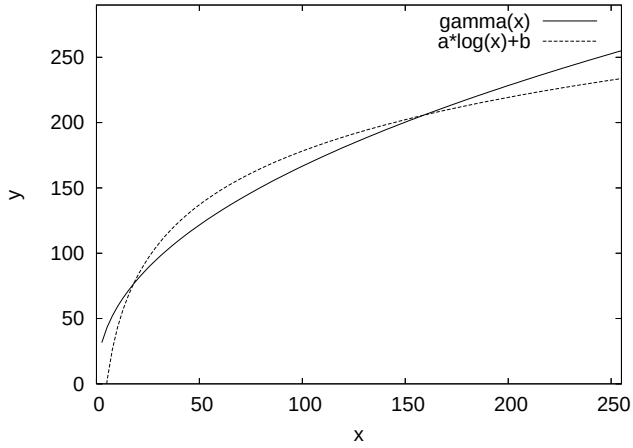


Figure 5. Gamma function which is used by the sRGB standard. A fit of $a \log x + b$ to the curve is also shown.

now see that the exponentiation required for display and the subsequent gamma correction cancel approximately and we can omit these two operations from the pipeline.

Figure 6(e) shows the JPEG2000 pipeline with integrated color constancy algorithm. In compressing an image using JPEG2000, a color transformation is applied to the non-linear $R'G'B'$ values. This color transformation performs a change of coordinate axes where Y' (luma) describes the brightness of the pixel and the two components C_B and C_R describe the color of the pixel. After this color transform is applied, the image pixels of an image taken under a white illuminant will be distributed around the luma axis. For a non-white illuminant, the main axis of the distribution will not be aligned with the luma axis. A color shift can be used to push the average onto the luma axis. We only have to subtract space

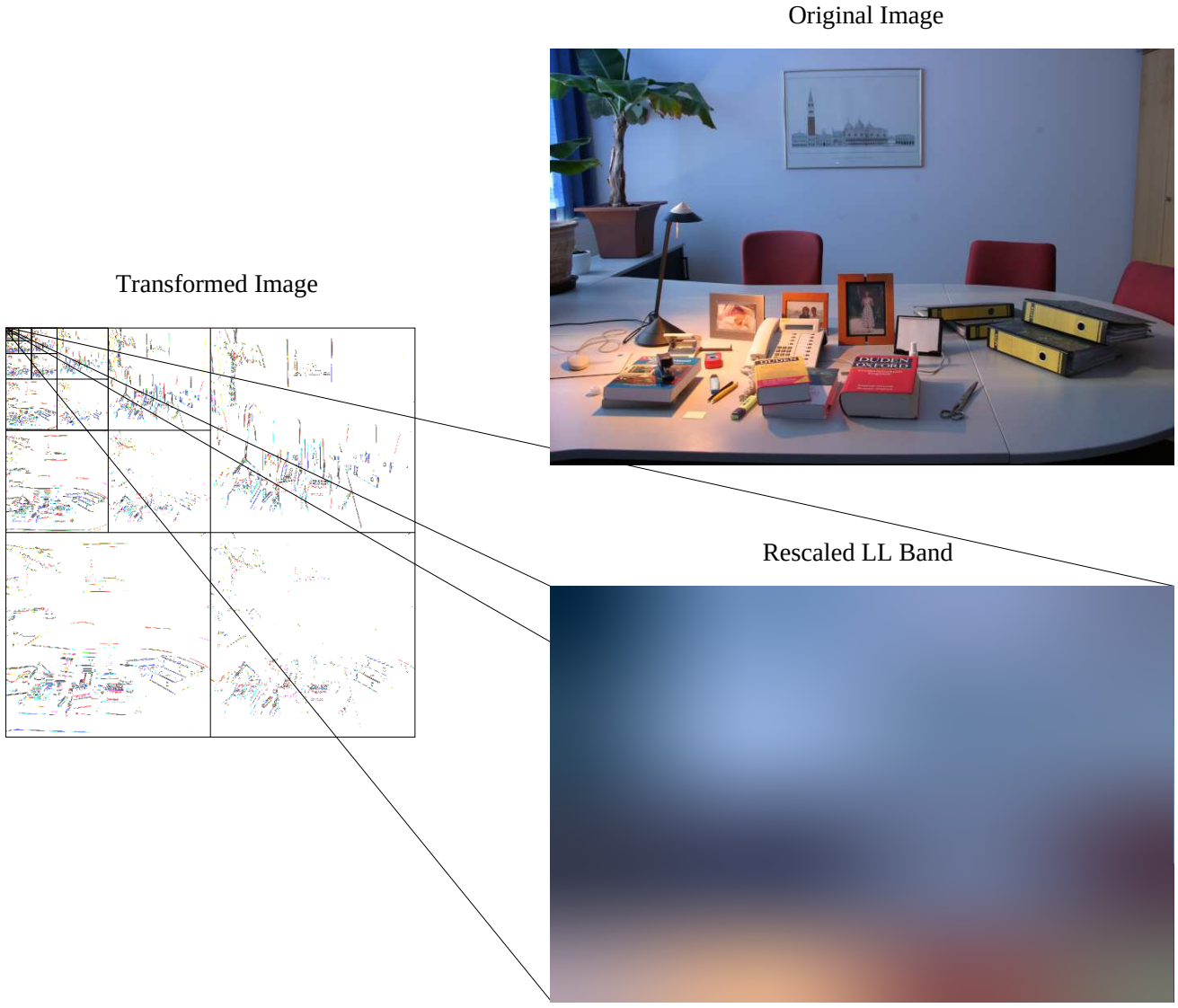


Figure 3. The original image is shown on the top right. The transformed image is shown on the left. Below the original image we see the contents of the LL band rescaled to the size of the original image.

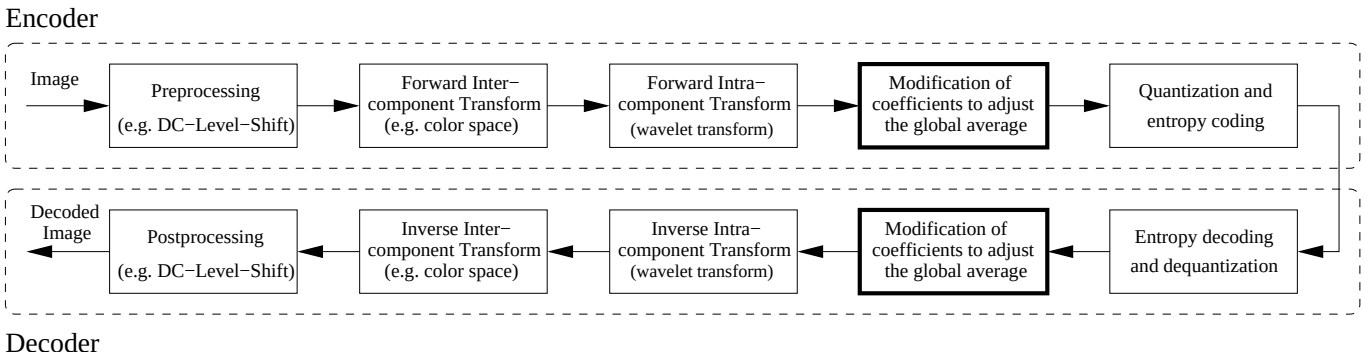


Figure 4. JPEG2000 encoding/decoding pipeline. The color constancy algorithm can either be included into the encoding or into the decoding pipeline.

average color from the color of each pixel. We have seen above that the gamma correction can be approximated by the function $a \log x + b$. Let $\tilde{c}_i$ be the data point inside the $Y'C_BC_R$ color space, then we obtain

$$o'_i = a \log(\tilde{c}_i) - \frac{1}{n} \sum_{x,y} a \log(\tilde{c}_i) \tag{31}$$

after we subtract the average of the log image pixels. This is

a color constant descriptor, as we have already derived above.

A JPEG2000 compressed image has space average color computed in the rotated coordinate system at its disposal. We now need to subtract space average color from the color of each pixel. If we do not want to adjust the brightness of the image, the luma channel is left untouched. We just need to subtract the average from the C_B and C_R channels. This essentially zeros the C_B and C_R components of the LL_0 sub-band. This operation does not change the brightness of the pixels as only the two components C_B and C_R are modified. Other methods of color corrections such as applying the gray world assumption locally [27] may also change the brightness of an image pixel. Alternatively, we could subtract the average scaled by a certain percentage. This would allow the user to vary the amount of attenuation of a colored illuminant.

This subtraction could be carried out before the image is encoded. If this is done then the image can no longer be viewed in its original form including the color cast. Information about such a change could be included inside an ICC color profile [43] in which case the change would be reversible. However, it makes more sense to apply the subtractive step during decoding. This allows the viewer to decide whether the corrective step is applied or not. The user could also specify the extent of the color correction by setting a percentage which scales the computed average before subtraction.

After the image is decoded, we end up with an (approximately) color constant descriptor o_i'

$$o_i' = a\log(\rho_i) + c \tag{32}$$

for some constant c. In order to obtain the reflectance information we have to undo this transformation. We could compute linear reflectances o_i using

$$o_i = e^{\frac{o_i'}{a}} = c'\rho_i \tag{33}$$

where $c' = e^{\frac{c}{a}}$. Now we have linear RGB values at our disposal. Since the response function of the display device is non-linear we have to again apply a gamma correction. The same holds if we want to store the linear reflectance values using the sRGB standard. We have already shown above that the application of the logarithm was approximated by the gamma correction using a gamma factor of $1/2.2$. Similarly, we can approximate the gamma correction with a gamma factor of 2.2 by $e^{\frac{x-b}{a}}$. If we make this approximation, the two operations cancel and can be dropped from the pipeline.

We have modified the JasPer Software [44] to automatically subtract the average of the C_B and C_R components from the corresponding channels of the LL_0 sub-band at the request of the user. The interested reader is referred to Tischler et al. [45] which describes the details of this implementation. Figure 7 shows the resulting output for two sample images. The two input images are shown on top. The output images are shown below. The color cast is quite nicely removed or attenuated in the output images.

A database to be used for color constancy research was created by Barnard et al. [46]. The database consists of 5 different sets of images showing different setups. Each setup was illuminated by several different illuminants. Set 1 contains only objects with minimal specularities, i.e. 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 to be used for object recognition. It contains objects where the object was moved whenever the illuminant was changed. The images from the database are stored in a linear color space. Therefore, we first transformed the images to the sRGB color space.

We have applied the color constancy algorithm integrated into JPEG2000 to each of the images and then compared the computed colors in Lab color space with two other color constancy algorithms. Algorithm 1 is the gray world assumption. For this algorithm, the color space was linearized by applying a gamma correction using the sRGB standard. Then global space average color is computed and each pixel is divided by global space average color. After this, all channels are rescaled such that the luminance is identical to the original image. Algorithm 2 works as follows. First we transform the pixels to the $Y'C_RC_B$ color space. We then compute global space average color for channels C_R and C_B and subtract this average from the channels C_R and C_B. We then transform the color space back to sRGB. Table I shows the average difference of the computed colors in Lab color space between algorithm 1 and the JPEG2000 algorithm and between algorithm 2 and the JPEG2000 algorithm. It is clear that the algorithm basically realized by the JPEG2000 algorithm as described above is actually algorithm 2. We see that the difference between the exact application of the gray world assumption and the JPEG2000 algorithm is not very great.

VII. Conclusion

We have shown how color constancy may be integrated into the JPEG2000 framework with little additional costs. Local space average color is computed by the discrete wavelet transform used by JPEG2000. Since local space average color may be used as an estimate of the illuminant, we only need to change the LL band of the highest level in order to obtain a color adjusted image. The nice thing about this method is that it may be integrated very easily into decoding devices. Decoding devices could modify the LL_0 sub-bands at the discretion of the user. This would not modify the contents of the original image only the displayed output would be altered.

References

[1] S. Zeki, *A Vision of the Brain*. Oxford: Blackwell Science, 1993.

[2] M. Ebner, *Color Constancy*. England: John Wiley & Sons, 2007.

[3] E. H. Land and J. J. McCann, "Lightness and retinex theory," *Journal of the Optical Society of America*, vol. 61, no. 1, pp. 1–11, Jan. 1971.

[4] K. Barnard, G. Finlayson, and B. Funt, "Color constancy for scenes with varying illumination," *Computer Vision and Image Understanding*, vol. 65, no. 2, pp. 311–321, Feb. 1997.

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

[6] ——, "A novel algorithm for color constancy," in *Color*, G. E. Healey, S. A. Shafer, and L. B. Wolff, Eds. Boston: Jones and Bartlett Publishers, 1992, pp. 241–271.

[7] G. D. Finlayson, "Color in perspective," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1034–1038, Oct. 1996.

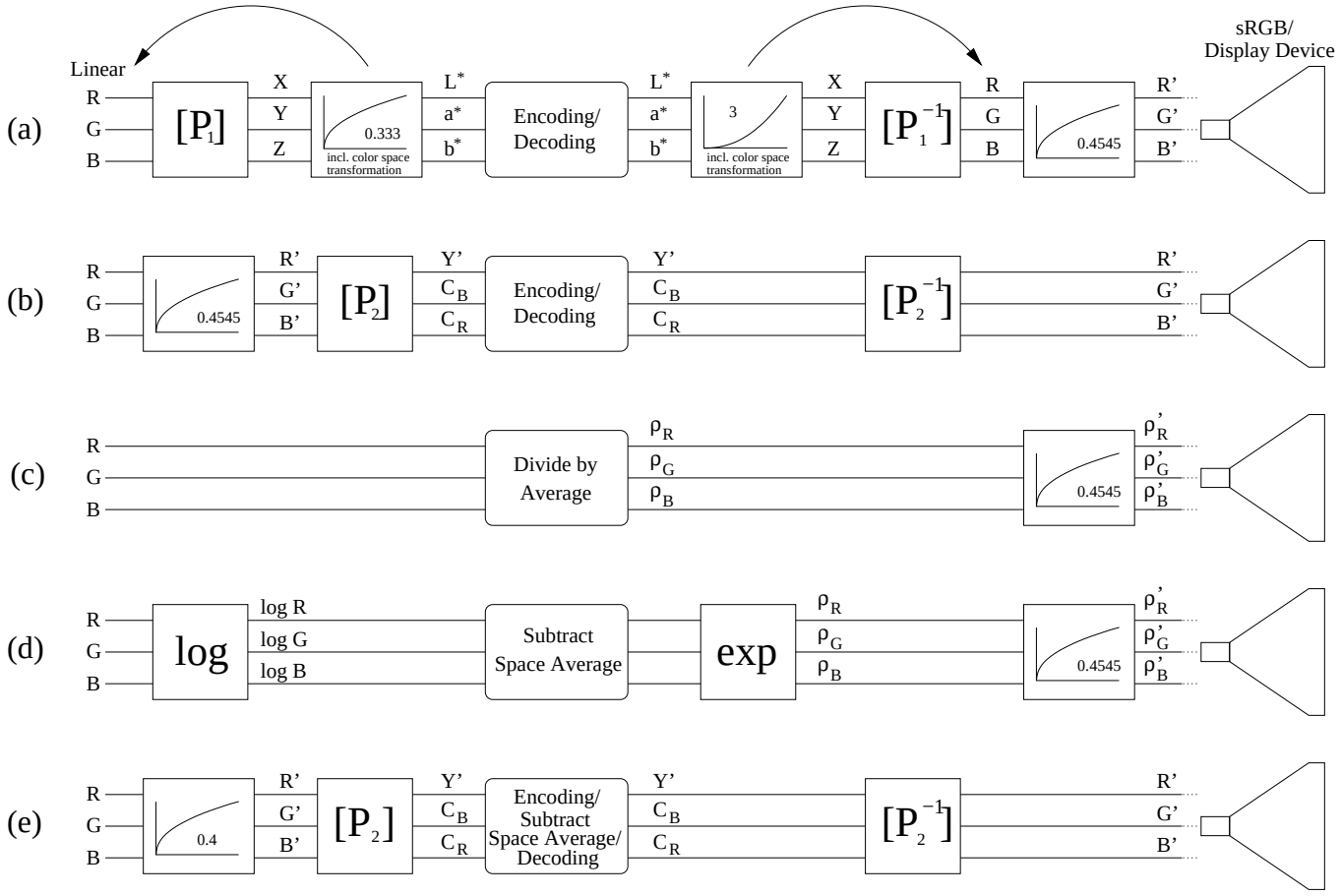


Figure 6. (a) Ideal encoding/decoding pipeline. (b) JPEG2000 encoding/decoding pipeline. (c) Gray world assumption color constancy pipeline. (d) Retinex color constancy pipeline. (e) Integrated JPEG2000/color constancy encoding/decoding pipeline.

Table I
IMAGE SETS FROM A DATABASE CREATED BY BARNARD ET AL. [46]. THE TABLE SHOWS THE AVERAGE EUCLIDEAN DIFFERENCE IN LAB COLOR SPACE BETWEEN ALGORITHM 1 AND THE JPEG2000 ALGORITHM AND ALSO THE DIFFERENCE BETWEEN ALGORITHM 2 AND THE JPEG2000 ALGORITHM.

No.	Name	No. of Objects	Scenes per Obj.	No. of Images	Alg. 1 vs JPEG2000	Alg. 2 vs JPEG2000
1	Lambertian Objects	21	2-11	223	5.25	2.69
2	Metallic Objects	14	9-11	149	4.68	2.05
3	Specular Objects	9	10-11	98	4.76	2.23
4	Fluorescent Objects	6	9-11	59	5.29	2.27
5	Different Objects	20	11	220	4.93	1.75

[8] K. Barnard, L. Martin, and B. Funt, "Colour by correlation in a three dimensional colour space," in *Proceedings of the 6th European Conference on Computer Vision, Dublin, Ireland*, D. Vernon, Ed. Berlin: Springer-Verlag, 2000, pp. 375–389.

[9] G. D. Finlayson, P. M. Hubel, and S. 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*, 1997, pp. 6–11.

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

[11] B. Funt, V. Cardei, and K. Barnard, "Learning color constancy," in *Proceedings of the IS&T/SID Fourth Color Imaging Conference*, Scottsdale, 19-22 Nov. 1996, pp. 58–60.

[12] S. Tominaga, "Surface identification using the dichromatic reflection model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 658–670, July 1991.

[13] M. D'Zmura and P. Lennie, "Mechanisms of color constancy," in *Color*, G. E. Healey, S. A. Shafer, and L. B. Wolff, Eds. Boston: Jones and Bartlett Publishers, 1992, pp. 224–234.

[14] G. D. Finlayson and G. Schaefer, "Solving for colour constancy using a constrained dichromatic reflection model," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 127–144, 2001.

[15] V. 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.

[16] M. Ebner and C. Herrmann, "On determining the color of the illuminant using the dichromatic reflection model," in *Pattern Recognition, Proceedings of the 27th DAGM Symposium, Vienna, Austria*, W. Kropatsch, R. Sablatnig, and A. Hanbury, Eds. Berlin: Springer-Verlag, 2005, pp. 1–8.

[17] V. C. Cardei, B. Funt, and K. Barnard, "White point estimation for uncalibrated images," in *Proceedings of the IS&T/SID Seventh Color Imaging Conference: Color Science, Systems and Applications*, 1999, pp. 97–100.

[18] ——, "White point estimation for uncalibrated images," in *12th International Conference on Control Systems and Computer Science CSCS-12*, Bucharest, May 1999.

[19] G. Buchsbaum, "A spatial processor model for object colour perception," *Journal of the Franklin Institute*, vol. 310, no. 1, pp. 337–350, July 1980.

[20] G. D. Finlayson, B. Schiele, and J. L. Crowley, "Comprehensive colour image normalization," in *Fifth European Conference on Computer Vision*

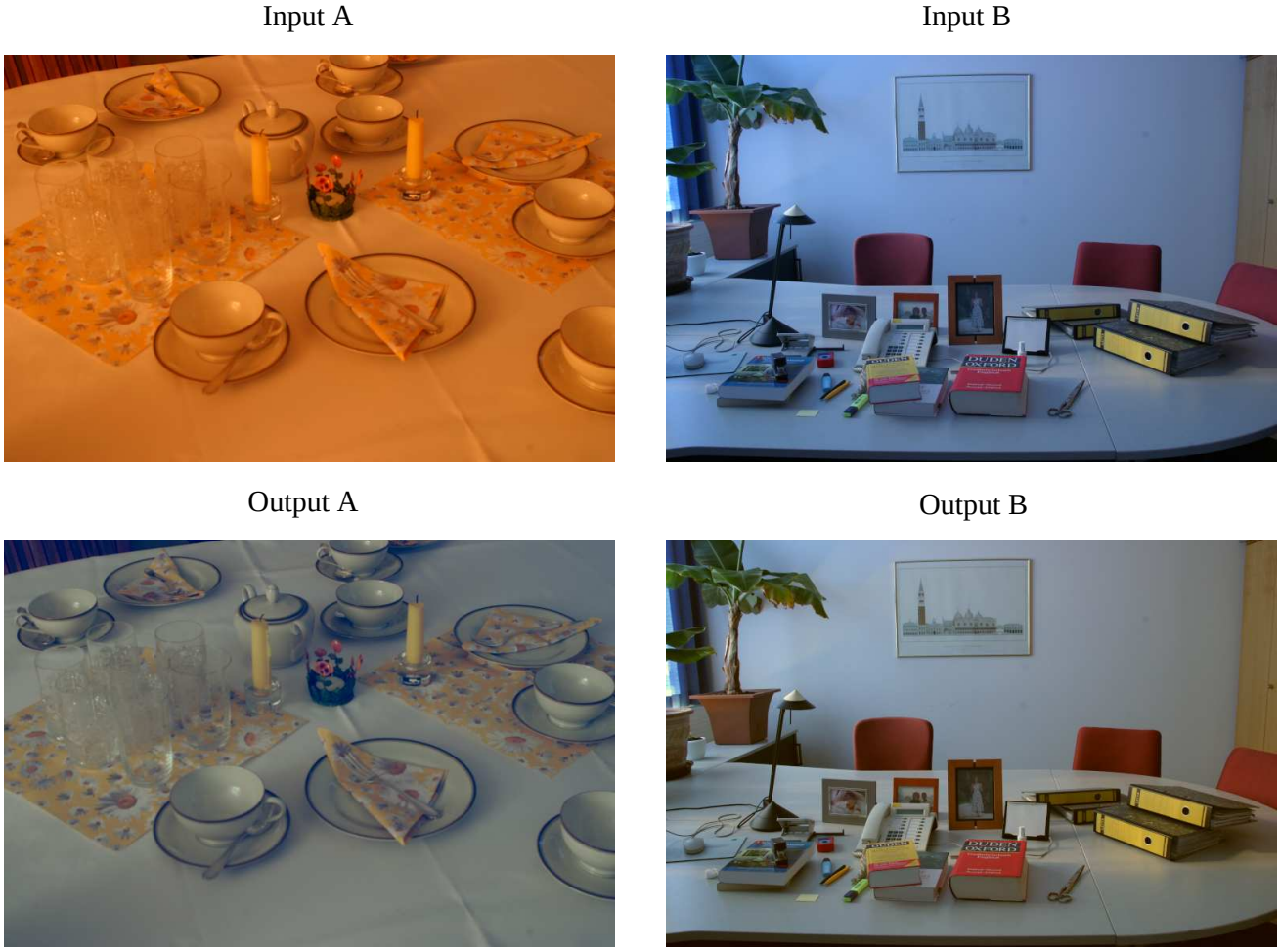


Figure 7. Results for two sample images. The images were color corrected by subtracting the average of the C_B and C_R components from the corresponding channels of the LL_0 sub-band.

(ECCV '98), Freiburg, Germany, H. Burkhardt and B. Neumann, Eds. Berlin: Springer-Verlag, 1998, pp. 475–490.

[21] G. D. Finlayson, M. S. Drew, and C. Lu, "Intrinsic images by entropy minimization," in *Proceedings of the 8th European Conference on Computer Vision, Part III,Prague, Czech Republic, May, 2004*, T. Pajdla and J. Matas, Eds. Berlin: Springer-Verlag, 2004, pp. 582–595.

[22] C. L. Novak and S. A. Shafer, "Supervised color constancy for machine vision," in *Color*, G. E. Healey, S. A. Shafer, and L. B. Wolff, Eds. Boston: Jones and Bartlett Publishers, 1992, pp. 284–299.

[23] G. D. Finlayson, M. S. Drew, and B. V. Funt, "Spectral sharpening: sensor transformations for improved color constancy," *Journal of the Optical Society of America A*, vol. 11, no. 4, pp. 1553–1563, Apr. 1994.

[24] ——, "Color constancy: generalized diagonal transforms suffice," *Journal of the Optical Society of America A*, vol. 11, no. 11, pp. 3011–3019, Nov. 1994.

[25] G. D. Finlayson and B. V. Funt, "Coefficient channels: Derivation and relationship to other theoretical studies," *COLOR research and application*, vol. 21, no. 2, pp. 87–96, Apr. 1996.

[26] K. Barnard, F. Ciurea, and B. Funt, "Sensor sharpening for computational color constancy," *Journal of the Optical Society of America A*, vol. 18, no. 11, pp. 2728–2743, Nov. 2001.

[27] M. Ebner, "A parallel algorithm for color constancy," *Journal of Parallel and Distributed Computing*, vol. 64, no. 1, pp. 79–88, 2004.

[28] ——, "Color constancy using local color shifts," in *Proceedings of the 8th European Conference on Computer Vision, Part III,Prague, Czech Republic, May, 2004*, T. Pajdla and J. Matas, Eds. Berlin: Springer-Verlag, 2004, pp. 276–287.

[29] B. K. P. Horn, "Determining lightness from an image," *Computer Graphics and Image Processing*, vol. 3, pp. 277–299, 1974.

[30] A. Blake, "Boundary conditions for lightness computation in mondrian world," *Computer Vision, Graphics, and Image Processing*, vol. 32, pp. 314–327, 1985.

[31] E. H. Land, "An alternative technique for the computation of the designator in the retinex theory of color vision," *Proc. Natl. Acad. Sci. USA*, vol. 83, pp. 3078–3080, May 1986.

[32] A. Moore, J. Allman, and R. M. Goodman, "A real-time neural system for color constancy," *IEEE Transactions on Neural Networks*, vol. 2, no. 2, pp. 237–247, Mar. 1991.

[33] A. Hurlbert, "Formal connections between lightness algorithms," *J. Opt. Soc. Am. A*, vol. 3, no. 10, pp. 1684–1693, Oct. 1986.

[34] D. H. Brainard and B. A. Wandell, "Analysis of the retinex theory of color vision," in *Color*, G. E. Healey, S. A. Shafer, and L. B. Wolff, Eds. Boston: Jones and Bartlett Publishers, 1992, pp. 208–218.

[35] T. G. Stockham, Jr., "Image processing in the context of a visual model," *Proceedings of the IEEE*, vol. 60, no. 7, pp. 828–842, July 1972.

[36] O. D. Faugeras, "Digital color image processing within the framework of a human visual model," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-27, no. 4, pp. 380–393, 1979.

[37] J. R. Parker, *Algorithms for Image Processing and Computer Vision*. New York: John Wiley & Sons, Inc., 1997.

[38] D. S. Taubman and M. W. Marcellin, *JPEG2000. Image Compression Fundamentals, Standards and Practice*. Norwell, Massachusetts: Kluwer Academic Publishers, 2002.

[39] M. D. Adams, "The jpeg-2000 still image compression standard," ISO/IEC JTC 1/SC 29/WG 1, N 2412, 2002.

[40] L. G. Glasser, A. H. McKinney, C. D. Reilly, and P. D. Schnelle, "Cube-root color coordinate system," *Journal of the Optical Society of America*, vol. 48, no. 10, pp. 736–740, Oct. 1958.

[41] C. Poynton, *Digital Video and HDTV. Algorithms and Interfaces*. San Francisco, CA: Morgan Kaufmann Publishers, 2003.

[42] G. Wyszecki and W. S. Stiles, *Color Science. Concepts and Methods*,

Quantitative Data and Formulae, 2nd ed. New York: John Wiley & Sons, Inc., 2000.
[43] International Color Consortium, "File format for color profiles (version 4.1.0)," Specification ICC.1:2003-09, 2003.
[44] M. D. Adams, "Jasper software reference manual (version 1.700.0)," ISO/IEC JTC 1/SC 29/WG 1, N 2415, 2003.
[45] G. Tischler, M. Ebner, and J. Albert, "Implementing color constancy within JPEG and JPEG2000," *to be submitted*, 2007.
[46] K. Barnard, L. Martin, B. Funt, and A. Coath, "A data set for color research," *Color Research and Application*, vol. 27, no. 3, pp. 147–151, 2002.

Marc Ebner received 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 35 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, Image and Vision Computing, 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.

German Tischler German Tischler studied computer science and physics at the Julius Maximilians Universität in Würzburg, Germany, where he also received the Dipl.-Inform. degree in 2002. He is currently pursuing his Ph.D. degree at the Universität Würzburg. His research interests include automata theory, fractal imaging, interval arithmetic and wavelets. He is author of 8 peer-reviewed publications on these subjects.

Jürgen Albert has been a member of IEEE and an affiliate of the Computer Society since 1986. He studied mathematics and computer science at the Technical University in Karlsruhe, Germany, where he also received his Dipl-Math. degree (1974), Ph.D. (1976) and habilitation (inaugural dissertation, 1983) degrees.

After serving as a lecturer at the military university in Munich, he was appointed a professorship in computer science at the University Würzburg, Germany. There he holds the Chair for Programming Languages and Programming Methodology since 1984. His current research interests include data-compression methods and their applications in digital libraries and information retrieval systems, especially for online-journals in medicine and pharmacology.

Prof. Albert is a member of the German Gesellschaft für Informatik and its SIGs on Parallel Algorithms and Artificial Intelligence. He is an editor of the Journal of Universal Computer Science and frequently serves as a referee for papers on automata-based methods for data-compression.