Evolving Color Constancy

Marc Ebner
Universität Würzburg, Lehrstuhl für Informatik II
Am Hubland, 97074 Würzburg, Germany
ebner@informatik.uni-wuerzburg.de
http://www2.informatik.uni-wuerzburg.de/staff/ebner/welcome.html

26th July 2005

Abstract

The ability to compute color constant descriptors of objects in view irrespective of the light illuminating the scene is called color constancy. We have used genetic programming to evolve an algorithm for color constancy. The algorithm runs on a grid of processing elements. Each processing element is connected to neighboring processing elements. Information exchange can therefore only occur locally. Randomly generated color Mondrians were used as test cases. The evolved individual was tested on synthetic as well as real input images. Encouraged by these results we developed a parallel algorithm for color constancy. This algorithm is based on the computation of local space average color. Local space average color is used to estimate the illuminant locally for each image pixel. Given an estimate of the illuminant, we can compute the reflectances of the corresponding object points. The algorithm can be easily mapped to a neural architecture and could be implemented directly in CCD or CMOS chips used in todays cameras.

Color Constancy, Genetic Programming, Local Space Average Color

1 Motivation

The human visual system is able to determine the color of objects irrespective of the color of the illuminant [39, 40, 57]. If a room with a white wall is illuminated with a yellow light, i.e. caused by a yellow lamp shade, the wall nevertheless appears white to a human observer. However, if the observer takes a photograph of the room then the wall will appear yellow in the photograph. The human visual system is somehow able to compute color constant descriptors which do not vary with the color of the illuminant. This remarkable ability is called color constancy. Our goal is to develop algorithms with similar capabilities. There is already a vast body of research in the area of color constancy.

Land and McCann have developed the retinex theory [42, 40, 41]. Others have proposed variants of the retinex theory [6, 27, 33, 34, 45, 49]. Algorithms for color constancy include gamut-constraint methods [2, 23, 24], perspective color constancy [16], color by correlation [3, 20], the gray world assumption [7, 29], recovery of basis function coefficients [28, 32, 43], mechanisms of light adaptation coupled with eye movements [11], neural networks [9, 10, 26, 31, 47, 56], comprehensive color normalization [22], committee-based methods which combine the output of several different color constancy algorithms [8], algorithms based on the dichromatic color model [21, 50] or computation of intrinsic images [19, 55]. Learning color constancy was addressed by Hurlbert and Poggio [35, 36].

Accurate color reproduction is highly important for many computer vision tasks. Often algorithms are developed which work in one particular setting but no longer work when the lighting conditions change. Many algorithms in the area of object recognition assume that object colors obtained from a digital camera remain constant. If object recognition is done on a service robot which has to work under different lighting conditions then this assumption is not valid. What one really should be doing is to compute the reflectance characteristics, i.e. the material properties of the object, and perform object recognition based on these reflectance characteristics. Accurate color reproduction is of course also very important for consumer photography. What we want is that the colors of a photo are exactly the same as they appeared to the photographer who took the image. Color constancy algorithms may of course also be applied to images taken with an analog camera. Such images can be scanned using a film scanner and then processed with a computer.

Another goal would be to find out what algorithm is actually used by the human visual system. We have turned to artificial evolution for an answer to this question. Knowing the algorithm used by the human visual system would be a great advance for consumer photography. Since the algorithm used by the visual system is a product of evolution it is of considerably interest to see what type of algorithm artificial evolution would find.

2 Theory of Color Image Formation

We now briefly review some background material on color image formation. For this, we consider a planar patch located at some distance in front of a camera or measuring device. We assume that the measuring device has a number of different sensors which respond to light in different parts of the spectrum. The human eye contains three types of cones which respond mainly to the light in the red, green and blue part of the spectrum. Similarly, a digital camera contains light sensors with red, green, and blue filters fitted in front of the sensors. Let $S_i(\lambda)$ be the response characteristic of the sensor i with $i \in \{r, g, b\}$ for wavelength λ. Let $R(\lambda, \mathbf{x}_{\text{obj}})$ be the amount of light reflected for each point $\mathbf{x}_{\text{obj}}$ on the planar patch. Let $L(\lambda, \mathbf{x}_{\text{obj}})$ be light falling on the planar patch at position $\mathbf{x}_{\text{obj}}$. A single sensor of the measuring device located at position $\mathbf{x}_I$ sees a corresponding object patch at position $\mathbf{x}_{\text{obj}}$. Light from the light source will fall on the patch. Some light will be absorbed, the remainder will be reflected. In case of a matte surface, the light will be reflected equally in all directions. Some light will enter the lens of the camera where it is measured by the sensor located at position $\mathbf{x}_I$. In order to calculate the output of the sensor one has to integrate over the entire spectrum. Let $\mathbf{I}(\mathbf{x}_I)$ be the output of the sensor at position $\mathbf{x}_I$. Then the output of the sensor is given by

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

This is the standard equation used by many algorithm for color constancy [7, 16, 18, 19, 21, 22, 23, 24, 47].

If we assume that the sensors response characteristics can be described by delta functions, i.e. they are very narrow shaped,

$$S_i(\lambda) = \delta(\lambda - \lambda_i)$$

with $i \in \{r, g, b\}$ then the above equation simplifies to

$$\begin{aligned} I_i(\mathbf{x}_I) &= \int R(\lambda, \mathbf{x}_{\text{obj}}) L(\lambda, \mathbf{x}_{\text{obj}}) \delta(\lambda - \lambda_i) d\lambda \\ &= R(\lambda_i, \mathbf{x}_{\text{obj}}) L(\lambda_i, \mathbf{x}_{\text{obj}}). \end{aligned}$$

If the response characteristics are not narrow shaped, they may be sharpened [17, 18]. In other words, the intensity measured by the sensor is proportional to the reflectance times the amount of light falling onto the patch for the given wavelength λ_i.

In the following we will only be working in image space. Therefore, we use the coordinates (x, y) to reference an image pixel. Images are usually viewed on a CRT monitor or flat panel display. The light given off by the phosphor of a cathode ray tube does not vary linearly with the applied voltage. The relation can be described by

$$I \approx A(k_1 U + k_2)^\gamma$$

where I is the luminance of the pixel on the computer screen, U is the intensity of the color channel, A is the maximum intensity and k_1, k_2 and γ are constants [44, 53]. For $k_1 = 1$ and $k_2 = 0$ the relationship is simply

$$I \approx A \cdot U^\gamma.$$

Gamma values γ for computer monitors are usually in the range 2.3 to 2.6. In order to counter this effect, a gamma correction is applied such that a linear relationship is maintained when images are viewed on a monitor. Images are often stored with a gamma correction already applied and not in a linear format. The sRGB standard assumes a gamma factor of 2.2 [53]. If we are processing such images we have to transform them to a linear space first before any algorithms on color constancy can be applied. It will now be assumed that the pixel values $\mathbf{c}(x, y)$ are related linearly to the intensities measured by the sensor, i.e.

$$\mathbf{c}(x, y) = \mathbf{I}(x, y).$$

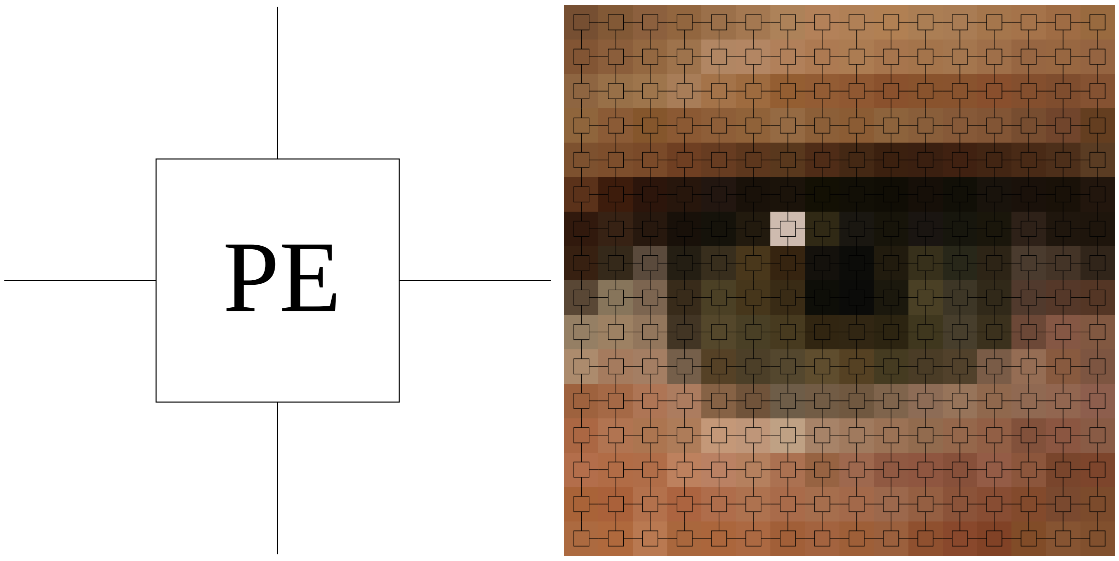


Figure 1: Rectangular grid of processing elements. A single element is shown on the left. A matrix of 16 $\times$ 16 elements is shown on the right. We use one processing element per image pixel.

Given the pixel colors $c_i(x, y)$ our goal is to compute the corresponding reflectances $R_i(x, y)$ for every image pixel. If the illuminant $L_i(x, y)$ where known for every image pixel, the reflectance could be computed as

$$R_i(x, y) = \frac{c_i(x, y)}{L_i(x, y)}.$$

where $c_i(x, y)$ is the intensity of channel i at position (x, y) in the image. The problem is that we only have three measurements but six unknowns. Only the color of the image pixel c_i is known. The reflectance R_i as well as the L_i with $i \in \{r, g, b\}$ is not known. Obviously, this problem cannot be solved without making some assumptions. One such assumption is that, on average the world is gray [7, 29]. Algorithms based on the gray world assumption assume that reflectances are uniformly distributed and that a large number of different colors occur in the viewed scene. We will discuss the gray world assumption in detail in Section 4. Another popular assumption is that somewhere in the image there is a white patch which reflects all of the incident light. This algorithm is referred to as white patch retinex [8, 25, 26, 42]. Note that we do not try to find the entire illuminant spectrum. We will only estimate the illuminant at three wavelengths using image statistics. As such, it is only an approximation to the actual illuminant.

3 Evolving an Algorithm for Color Constancy

Our goal was to find a biologically plausible algorithm, i.e. an algorithm which could be mapped to a neural architecture, for color constancy. Note that most existing color constancy algorithms, apart from the approaches using neural nets [8, 9, 10, 26, 48, 56], are quite complicated. Since we are turning to artificial evolution for an answer to the problem color constancy, we started out with a very simple architecture, a rectangular grid of processing elements [12]. The processing elements themselves are quite simple and can carry out only simple local computations. Each processing element has only access to the data stored at neighboring processing elements. Such an architecture is shown on the right of Figure 1. A single processing element is shown on the left.

It is assumed that we have one processing element per image pixel and that we have three layers of processing elements carrying out the same computations on the three image bands red, green, and blue. Each processing element will compute an estimate of the local illumination given the color of its input pixel and the data available from other neighboring processing elements. That is, we only allow a local exchange of information. Each processing element then uses the estimate of the illuminant and the color of the input pixel to compute the reflectance as described above. Since only local connections are allowed, the algorithm could be implemented directly on the imaging chip.

Let $L_i(x, y)$ be the current estimate of the illuminant computed by the processing elements. Initially, we set this estimate to the color of the current pixel, i.e. $L_i(x, y) = c_i(x, y)$. The task is to find a function which updates this estimate of the illuminant for each processing element. In other words, we are looking for a program or function which computes

$$L_i(x, y) = \text{program}(L_i(x, y), L_i(x-1, y), L_i(x+1, y), L_i(x, y-1), L_i(x-1, y+1), \mathbf{c}).$$

Genetic programming [1, 37, 38] was used to search for such a program. Table 1 shows the set of elementary functions and Table 2 shows the set of terminal symbols which were used. At this point, we do not know what type of algorithm evolution will chose. It is possible to solve the problem using either the gray world assumption or the assumption that there is a white patch in the image.

Individuals were evaluated by presenting them randomly generated color Mondrians. Each generation we created three Mondrians by drawing randomly colored boxes on top of each other. A size of 64×64 was used for the Mondrians. The width and height of the boxes were drawn randomly from the range $[8, 24]$. Positions of the boxes were chosen at random. This is our reflectance image, our ground truth. We then illuminated each image by randomly choosing a color for the illuminant and multiplying the reflectance component stored in the Mondrian image with the illuminant color.

Each individual was tested on three fitness cases. To evaluate the fitness, we iterated the update equation 100 times. We then calculated the difference between the output of the evolved individual and the known reflectance image. Let $\mathbf{c}_o(x, y)$ be the output of the evolved individual and let $\mathbf{c}_r(x, y)$ be the corresponding pixel value of the reflectance

Table 1: Elementary functions used for our experiments.

Elementary Function	Arity	Symbol
addition	2	`+`
subtraction	2	`-`
multiplication	2	`*`
protected division	2	`/`
multiply by 2	1	`mul2`
divide by 2	1	`div2`

Table 2: Set of terminal symbols.

Terminal Symbol	Symbol
constant one	`1`
red input band $c_r(x, y)$	`red`
green input band $c_g(x, y)$	`green`
blue input band $c_b(x, y)$	`blue`
current band $c_i(x, y)$	`band`
estimate from current element $L_i(x, y)$	`center`
estimate from left element $L_i(x-1, y)$	`left`
estimate from right element $L_i(x+1, y)$	`right`
estimate from element above $L_i(x, y-1)$	`up`
estimate from element below $L_i(x, y+1)$	`down`

image, then the difference d_i for test case k is calculated as

$$d_k = \frac{1}{3N} \sum_{x,y} \sqrt{(\mathbf{c}_{\mathrm{o}}(x, y) - \mathbf{c}_{\mathrm{r}}(x, y))^2}$$

where N is the number of image pixels. Fitness of an individual is defined based on the maximum difference over all test cases

$$\text{fitness} = \frac{1}{1 + max_k\{d_k\}}$$

with $k \in \{1, 2, 3\}$. An individual has to perform well on all three test cases to receive a high fitness. A similar approach was used by Harvey et al. [30] to evolve robust control algorithms for a simulated robot. The task of an individual is to accurately estimate the reflectances of the input image. The actual reflectances are only used in order to evaluate the individual. Since we randomly create the test cases, an individual cannot "learn" the reflectances from the training set. A successful individual must find a general algorithm which will estimate the reflectances using only the pixels of the input image.

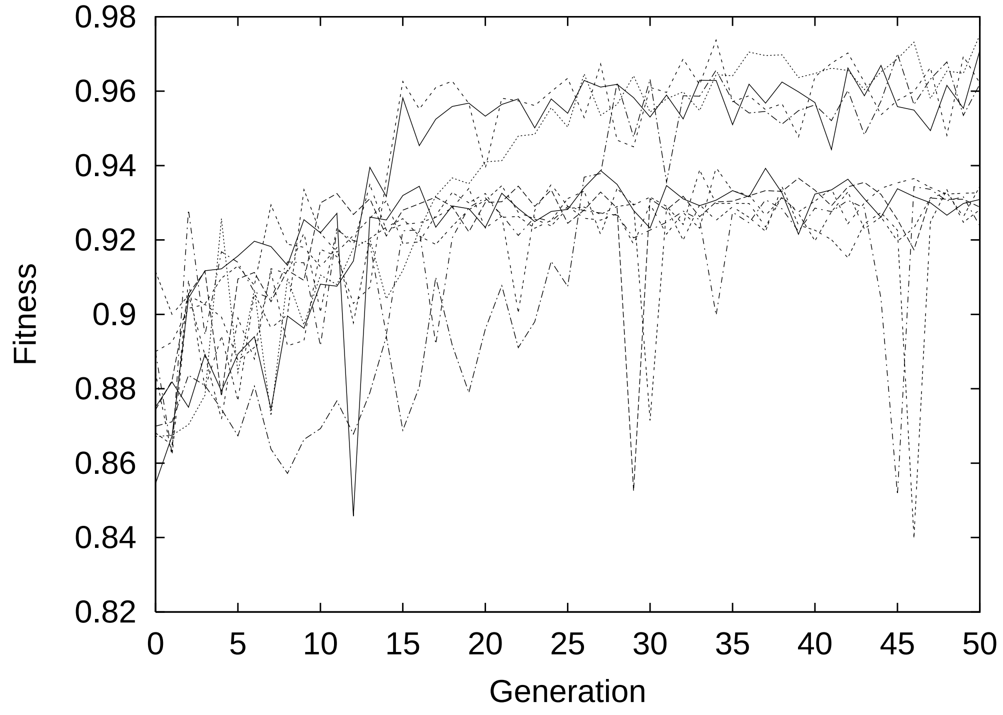


Figure 2: Fitness statistics for ten runs with different seeds for the number generator (from [12]).

A population of 1000 individuals was used. Ten experiments with different initial seeds for the random number generator were run for 50 generations each. A single run took approximately 10 days to complete as the evaluation was done on sequential hardware. The best individual of each generation was allowed to reproduce 10 times, filling one percent of the next generation. The remaining 99% of the population was filled using reproduction (9%), mutation (20%) and crossover (70%). Individuals were chosen using tournament selection of size 7.

Figure 2 shows the maximum fitness over 50 generations for all 10 experiments. At the end of 50 generations, 6 of the runs were stuck in a local optimum. The best individual of all 10 runs achieved a fitness of 0.9751. The performance of this individual is shown in Figure 3 for three color Mondrians. The images can be viewed in color on the author's web page[1] and are also included in color in the electronic version of the journal. We also tested the individual on real, i.e. not simulated input images. Figure 4 shows an image taken from a database created by Barnard et al. [4][2]. This database is used to test color constancy algorithms. The image on the left shows the input image, a Macbeth color checker illuminated with a Solux 4700K and a Roscolux 3202 full blue filter. The original

[1]http://www2.informatik.uni-wuerzburg.de/staff/ebner/research/evoColor/color.html
[2]http://www.cs.sfu.ca/~colour/data/index.html

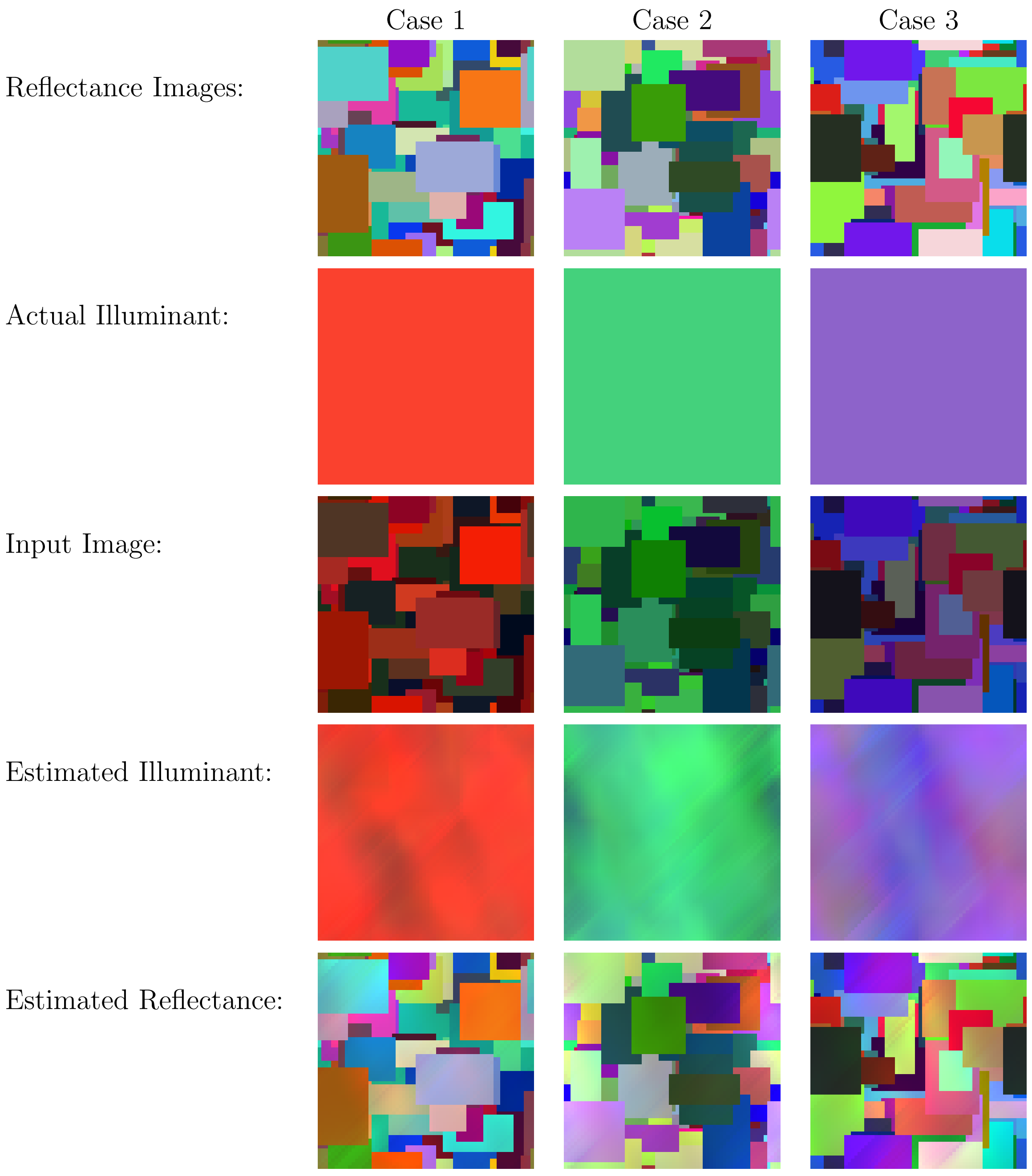


Figure 3: Performance of best individual on three different fitness cases. The images in the first row show three randomly created color Mondrians. The images in the second row show the actual illuminant. The images in the third row show what the individual received as input. This is the output a measuring device would give. The images in the fourth row show the estimated illuminant. Finally, in the last row the estimated reflectance is shown (from [12]).

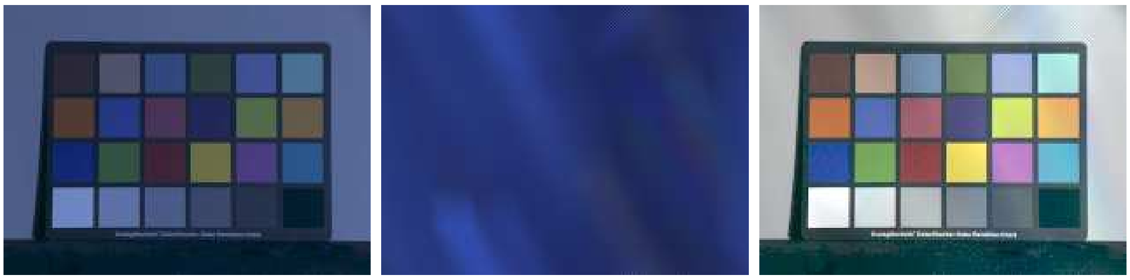


Figure 4: Results for a real image. The first image shows a Macbeth color checker taken from a database used to test color constancy algorithms. The database was put together by Barnard et al. [4]. The second image shows the estimated illuminant. The third image shows the output of the evolved individual.

image was down-sampled to a size of 210×154. The estimated illuminant is shown in the center. On the right the estimated reflectances are shown. Because the input image is larger than the images which were used to evolve the individual we have used 300 iterations to compute the output image. The evolved individual is able to remove the influence of the blue illuminant from the image.

The program code of the best individual is as follows: (div2 (+ (+ (div2 (+ (div2 (+ (div2 down) (div2 (+ (div2 (+ (div2 (+ band (div2 down))) (div2 (+ band (div2 (+ band (div2 center))))))) (div2 down))))) (* (div2 down) (/ right (div2 (+ down (+ (div2 down) (div2 (+ (div2 (+ band (div2 down))) (div2 (+ band (div2 (+ band (div2 center)))))))))))))) (div2 (+ (div2 (+ (div2 right) (div2 (+ band (div2 down))))) (* (div2 down) (/ right (div2 (+ down (+ (div2 down) (div2 (+ (div2 (+ band (div2 center))) (div2 (+ band band)))))))))))) down)).

Only the current color channel (`band`) is used. The individual does not use information from any of the other channels. In hindsight this is not surprising as the three color channels contain independent information and therefore may be processed independently. Such independent processing was assumed by Land [39] in his retinex theory. Also, the individual mainly uses addition and division by 2 to solve the problem of color constancy. The evolved algorithm essentially averages data from neighboring elements.

4 A Parallel Algorithm for Color Constancy

Encouraged by the above results, we developed a parallel algorithm for color constancy [13, 15]. As above, we assume that we have a grid of processing elements, one processing element per pixel. The algorithm is based on the computation of local space average color. Let us assume that the reflectances of the objects are evenly distributed in the range $[0, 1]$ and that a single uniform illuminant is used. Let us also assume that the viewed scene is sufficiently diverse, i.e. there are a large number of different surfaces in the scene. If we

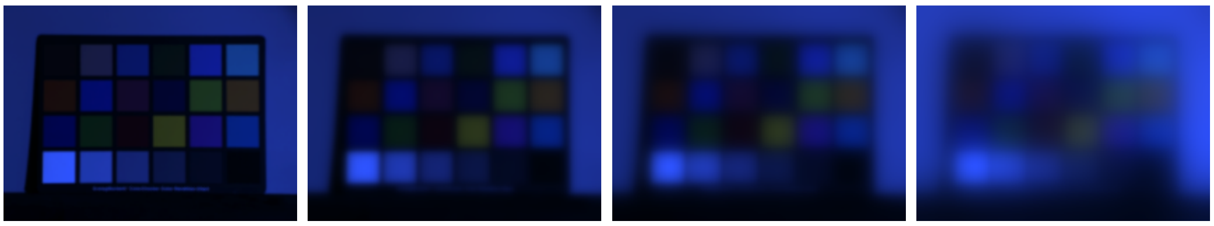


Figure 5: Computation of local space average color after 1, 50, 200 and 1000 iterations.

now average the observed pixel values, then we obtain

$$\frac{1}{N}\sum_{x,y} c_i(x,y) = \frac{1}{N}\sum_{x,y} R_i(x,y)L_i = L_i\frac{1}{N}\sum_{x,y} R_i(x,y) \approx \frac{1}{2}L_i$$

where N is the number of image pixels. The last approximation holds because we have assumed that the reflectances are uniformly distributed over the range $[0,1]$. If we randomly draw reflectances from this range, the average will be $\frac{1}{2}$ for a sufficiently large sample size. If we solve the above equation for L_i, we see that the illuminant can be estimates as twice the value of space average color.

$$L_i \approx \frac{2}{N}\sum_{x,y} c_i(x,y)$$

This is just the gray world assumption [7, 29]. Note that because we are averaging pixel data, it is very important that we are working in a linear space.

We apply the gray world assumption locally to each image pixel. Averaging is done in parallel by iteratively executing the following update equations. Let $\mathbf{a}(x,y)$ be an estimate of local space average color for each image pixel. Let $N(x,y)$ be a set of neighboring elements

$$N(x,y) = \{(x',y') | (x',y') \text{ is a neighbor of processing element } (x,y)\}.$$

We then average the estimate of local space average color from neighboring elements. Note that the current element can also be included in $N(x,y)$. This helps to eliminate oscillations which may otherwise occur. Finally, using a small percentage p we add a tiny amount of the color of the input pixel to the current average.

$$\begin{aligned} &1.) \qquad a'_i(x,y) = \frac{1}{|N(x,y)|}\sum_{(x',y')\in N(x,y)} a_i(x',y') \\ &2.) \qquad a_i(x,y) = c_i(x,y)\cdot p + a'_i(x,y)\cdot(1-p) \end{aligned}$$

The first equation averages the data, the second slowly adds the color of the current pixel. These two computations are carried out until convergence. Note that the initialization

Figure 6: Results for the parallel algorithm on the image of the Macbeth color checker. The first image shows the input image. The second image shows local space average color. The third image shows the output image.

of local space average color can be arbitrary as the initial value slowly fades away. For a technical implementation the two computations can be run indefinitely. Therefore one does not have to determine when convergence will be reached. Figure 5 shows how local space average color is computed iteratively for the Macbeth color checker.

The parameter p determines the extent of the averaging. When we compare the local averaging operation with the evolved program we see that the evolved program uses a number of constants to scale the input from neighboring elements. The evolved program cannot be run on arbitrary image sizes. Even though we made sure that the input/output relationship could not be learnt from example, we have used a single image size for training. Thus, the evolved algorithm depends on the size of the image. Here, the parameter p has to be set depending on the size of the image. In practice we have found a value of 0.0005 suitable for an image of size 256×256. For arbitrary sized images we use

$$p = 1 - (1 - 0.0005)^{256/\mathrm{s}}$$

where s is the longer of the width or height of the image. Figure 6 shows the output of this algorithm on the image of the Macbeth color checker. The first image is the input image. The second image shows the computed local space average color. The image on the right is the computed output image.

Barnard et al. [4], in creating their dataset for color constancy research, carefully measured the illuminant spectra and the response of the camera to the color of the illuminant using a reference white. Since the ground truth data is available, we can compare the estimated chromaticities with the actual chromaticities of the illuminant. Figure 7 shows the results of this comparison. The first image shows the estimated chromaticities. The graph on the right shows the comparison between the actual chromaticities and the estimated chromaticities for a single line of the image. While the match between the actual illuminant and the estimated illuminant is not perfect, the algorithm correctly estimates that the color of the illuminant to be blue. Note that human color constancy is not perfect either [46].

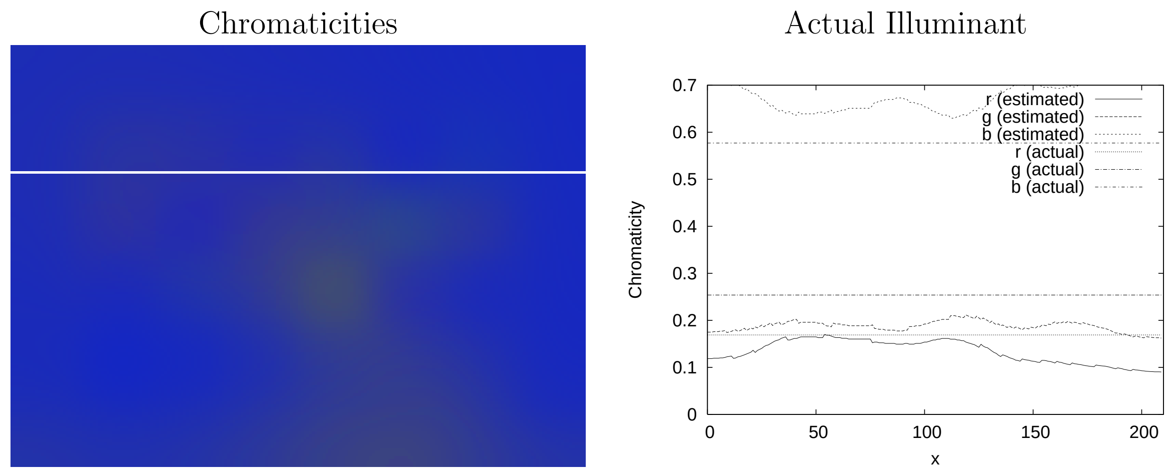


Figure 7: The image on the left shows the estimated chromaticity of the illuminant. Note that this image is best viewed in the electronic version of the journal. The graph on the right compares the actual with the estimated chromaticities for a single line of the image. The line where the chromaticities were extracted is marked by the white line.

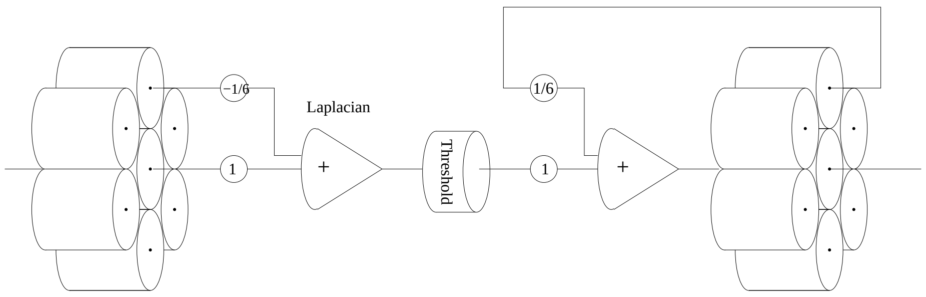
5 Comparison with Other Algorithms for Color Constancy

Numerous algorithms for color constancy have been developed. However, most algorithms would be hard to implement using a neural architecture. They are not biologically plausible. A notable exception is the retinex algorithm developed by Land and McCann [42] which was later extended to two dimensions by Horn [33] and refined by Blake [5]. Horn suggests to first take the logarithm of the input intensity. This separates the product of reflectance and illumination into a sum.

$$\log c_i = \log R_i + \log L_i$$

Next, edges are detected using a Laplacian. Note, that the response of the Laplacian will be large at points where the reflectance changes and almost zero everywhere else. The term $\log L_i$ cancels itself for adjacent points in the image. A threshold operation is applied to locate positions where the reflectance changes, eliminating all changes of the illuminant. Finally the output of the thresholding operation is re-integrated to compute the reflectances. The whole algorithm can be described by the sequence of operations shown in Figure 8. The first stage computes the Laplacian. Then a threshold operation is applied. The last stage re-integrates the result in parallel to compute the reflectances. In practice it is difficult to determine the best value for the threshold operation. If the threshold is set too low, then edges due to a change of the illuminant will not be removed. If the threshold is set too high, then some edges where the reflectance changes will also be removed.

For comparison, our own algorithm is shown on the bottom of Figure 8 using the same



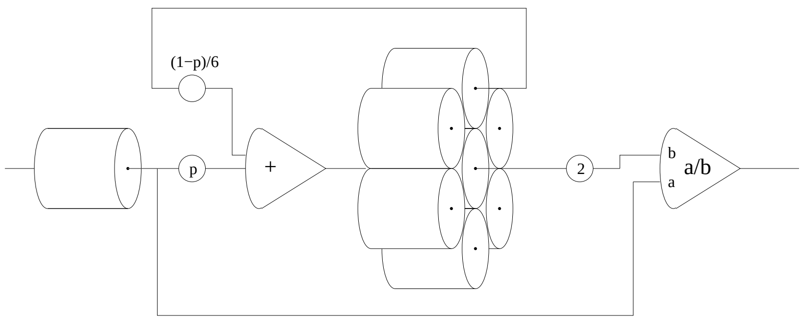


Figure 8: Two dimensional variant of the retinex algorithm shown on the top (redrawn from Horn [33]). On the bottom, our algorithm is shown in a similar notation.

notation. The integration step has been moved to the front. Note that the algorithm is now much simpler. It consists of an integration stage followed by a computation of the output using the color of the input pixel. The advantage of this algorithm is that it does not require a threshold operation. Our algorithm can be implemented easily using neural hardware. Should a division operation be difficult to implement, then this step can be replaced using addition and subtraction. This leads to local color shifts [14]. A discussion of local color shifts is, unfortunately, beyond the scope of this contribution.

Moore et al. [45] implemented a variant of the retinex algorithm [41] in VLSI. They have used a resistive grid to blur the input image. The output color of a pixel o'_i with $i \in \{r, g, b\}$ is given by

$$o'_i(x, y) = \log(c_i(x, y)) - \log(c_i) \otimes e^{-\frac{|r|}{\sigma}}$$

where $\otimes$ denotes convolution, r is the distance from the current pixel and σ is a scaling factor which defines the extent of the blurring. Since the logarithm has been applied the output colors must be transformed to the range $[0, 1]$ for display. This is done using a global operation over all pixels. Moore et al. determine the minimum and maximum value over all pixels. Then they subtract the minimum value from the output color o' and rescale the result to the full range, i.e. they compute

$$o_i(x, y) = \frac{o'_i(x, y) - \min}{\max - \min}$$

where

$$\begin{aligned}\max &= \max\{o'_i(x,y)| \text{ for all } (x,y) \text{ and } i \in \{r,g,b\}\} \\ \min &= \min\{o'_i(x,y)| \text{ for all } (x,y) \text{ and } i \in \{r,g,b\}\}.\end{aligned}$$

Note that due to the use of a global operation in the algorithm it is not clear how this algorithm could be mapped to a completely parallel algorithm.

6 Evaluation of Algorithms on an Object Recognition Task

We have used color based object recognition to evaluate the performance of the algorithms on the datasets of Barnard et al. [4]. Five different sets were used from this database. The images were down-sampled to 50% of the original size in order to speed up the evaluation. Image set 1 contains mainly Lambertian reflectors. Image set 2 contains objects with metallic specularities. Image set 3 contains objects with non-negligible dielectric specularities. Image set 4 contains objects with at least one fluorescent surface. Each image set shows a number of different scenes under up to 11 different illuminants. Image set 5 differs from the previous sets in that the object was placed in a random position whenever the illuminant was changed. A given color constancy algorithm is evaluated by applying it to every image of the image set. We then selected two color corrected images from each scene of single database. This gave us one test and one model image per scene. A match between test and model image was established using histogram based object recognition. Histogram based object recognition was originally introduced by Swain and Ballard [54]. A color histogram is created for each image and the χ^2 divergence measure between test and model histogram is computed. The χ^2 divergence measure was proposed by Schiele and Crowley [51, 52]. The model images are matched to the test images by choosing the lowest divergence measure. This process is repeated 50 times by randomly selecting pairs of images for each scene. A perfect color constancy algorithm which correctly estimates the reflectances of the viewed objects would result in a recognition rate of 1.0. In this case, if we also assume that the camera was not moved between images, the color distribution of both the test and model images would be equivalent.

Table 3 shows the results for several color constancy algorithms. The random recognition rate is shown at the top of the table. Results are shown for the white patch retinex algorithm, the gray world assumption, a simplified version of Horn's algorithm, the algorithm of Blake [5] and the retinex variant described by Moore et al. [45]. In order to make the white patch retinex algorithm more robust, the color of the illuminant was estimated by histogramming each color band. Instead of picking the maximum of each channel, the illuminant was estimated by setting the white point at the highest percentile. The simplified version of Horn's algorithm [33, 34] omits the application of the Laplacian operator, the application of the threshold operation and also the re-integration step. This essentially amounts to assuming that the illuminant is constant across the entire image. The threshold

Table 3: Results for image sets 1 through 5. Histograms were computed in RGB space. For each image set random performance is also shown. Best performance is marked in bold.

Histogram-based Object Recognition, RGB color space

Algorithm	1	2	3	4	5
Random Recognition Rate	0.045	0.071	0.111	0.167	0.050
White Patch Retinex	0.614	0.574	0.918	0.893	0.416
Gray World Assumption	0.752	0.536	0.953	0.950	0.311
Simplified Horn	0.442	0.443	0.620	0.593	0.198
Horn (1974)/Blake (1985)	0.465	0.394	0.560	0.607	0.232
Moore et al. (1991) Retinex	0.850	0.799	0.756	0.780	0.513
Local Space Average Color	**0.937**	**0.874**	**1.000**	**1.000**	**0.538**

used by the algorithm of Blake was set to 10/256. Local space average color for both the algorithm of Moore et al. [45] and our algorithm is computed using a convolution with $e^{|r|/\sigma|}$ with $r = |x| + |y|$ and $\sigma = \sqrt{\frac{1-p}{4p}}$. Among the different algorithms, the algorithm based on local space average color resulted in the highest recognition rate across all image sets.

7 Conclusion

Color constancy is an important problem in many areas of computer vision research. In particular, color constancy is required to develop robust algorithms for service robots which have to work under changing lighting conditions. It is equally important for consumer photography. In order to develop algorithms which mimic the algorithms used by the human visual system we also have to look at the neural architecture of the brain. The whole visual system is a product of natural evolution and we have turned to artificial evolution to search for an algorithm for color constancy.

Genetic programming was used to evolve an algorithm for color constancy for a parallel architecture. The architecture was designed such that only local exchange of information is possible. Only simple arithmetic operators were used as elementary functions. The evolved solution concentrated only on a single color channel. Information from the other color channels was not used. The evolved individual was tested on artificial as well as real input images. We then developed a simple algorithm for color constancy based on the computation of local space average color. It is biologically plausible and much simpler than existing algorithms. In fact, it may even be integrated directly into CCD or CMOS chips used in todays digital cameras.

Acknowledgments

We have used the lilgp Programming System Vers. 1.1 [58] for our experiments.

References

[1] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone. *Genetic Programming - An Introduction: On The Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[2] K. Barnard, G. Finlayson, and B. Funt. Color constancy for scenes with varying illumination. *Computer Vision and Image Understanding*, 65(2):311–321, 1997.

[3] K. Barnard, L. Martin, and B. Funt. Colour by correlation in a three dimensional colour space. In David Vernon, editor, *Proc. of the 6th Europ. Conf. on Computer Vision, Dublin, Ireland*, pp. 375–389, Berlin, 2000. Springer-Verlag.

[4] K. Barnard, L. Martin, B. Funt, and A. Coath. A data set for color research. *Color Research and Application*, 27(3):147–151, 2002.

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

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

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

[8] V. C. Cardei and B. Funt. Committee-based color constancy. In *Proc. of the IS&T/SID 7th Color Imaging Conference: Color Science, Systems and Applications, Scottsdale, AZ*, pp. 311–313, 1999.

[9] S. M. Courtney, L. H. Finkel, and G. Buchsbaum. A multistage neural network for color constancy and color induction. *IEEE Transactions on Neural Networks*, 6(4):972–985, 1995.

[10] P. A. Dufort and C. J. Lumsden. Color categorization and color constancy in a neural network model of V4. *Biological Cybernetics*, 65:293–303, 1991.

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

[12] M. Ebner. Evolving color constancy for an artificial retina. In J. Miller, M. Tomassini, P. Luca Lanzi, C. Ryan, A. G. B. Tettamanzi, and W. B. Langdon, eds., *Genetic Programming: Proc. of the 4th European Conference, EuroGP 2001, Lake Como, Italy*, pp. 11–22, Berlin, 2001. Springer-Verlag.

[13] M. Ebner. A parallel algorithm for color constancy. Tech. Report 296, Universität Würzburg, Lehrstuhl für Informatik II, Am Hubland, 97074 Würzburg, Germany, 2002.

[14] M. Ebner. Color constancy using local color shifts. In T. Pajdla and J. Matas, eds., *Proc. of the 8th Europ. Conference on Computer Vision, Part III, Prague, Czech Republic, 2004*, pp. 276–287, Berlin, 2004. Springer-Verlag.

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

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

[17] G. D. Finlayson, M. S. Drew, and B. V. Funt. Color constancy: generalized diagonal transforms suffice. *Journal of the optical society of america A*, 11(11):3011–3019, 1994.

[18] 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*, 11(4):1553–1563, 1994.

[19] G. D. Finlayson, M. S. Drew, and C. Lu. Intrinsic images by entropy minimization. In T. Pajdla and J. Matas, eds., *Proc. of the 8th Europ. Conference on Computer Vision, Part III,Prague, Czech Republic, 2004*, pp. 582–595, Berlin, 2004. Springer-Verlag.

[20] G. D. Finlayson, P. M. Hubel, and S. Hordley. Color by correlation. In *Proc. of IS&T/SID. The 5th Color Imaging Conference: Color Science, Systems, and Applications, The Radisson Resort, Scottsdale, AZ*, pp. 6–11, 1997.

[21] G. D. Finlayson and G. Schaefer. Solving for colour constancy using a constrained dichromatic reflection model. *Int. Journal of Comp. Vision*, 42(3):127–144, 2001.

[22] G. D. Finlayson, B. Schiele, and J. L. Crowley. Comprehensive colour image normalization. In H. Burkhardt and B. Neumann, eds., *5th Europ. Conf. on Computer Vision (ECCV '98), Freiburg, Germany*, pp. 475–490, Berlin, 1998. Springer-Verlag.

[23] D. A. Forsyth. A novel approach to colour constancy. In *2nd Int. Conf. on Computer Vision (Tampa, FL)*, pp. 9–18. IEEE Press, 1988.

[24] D. A. Forsyth. A novel algorithm for color constancy. In G. E. Healey, S. A. Shafer, and L. B. Wolff, eds., *Color*, pp. 241–271, Boston, 1992. Jones and Bartlett Publishers.

[25] B. Funt, K. Barnard, and L. Martin. Is machine colour constancy good enough? In H. Burkhardt and B. Neumann, editors, *5th Europ. Conf. on Computer Vision (ECCV '98), Freiburg, Germany*, pp. 445–459, Berlin, 1998. Springer-Verlag.

[26] B. Funt, V. Cardei, and K. Barnard. Learning color constancy. In *Proc. of the IS&T/SID 4th Color Imaging Conference*, pp. 58–60, Scottsdale, 1996.

[27] B. V. Funt and M. S. Drew. Color constancy computation in near-mondrian scenes using a finite dimensional linear model. In R. Jain and L. Davis, eds., *Proc. of the Comp. Society Conf. on Computer Vision and Pattern Recognition, Ann Arbor, MI*, pp. 544–549. Computer Society Press, 1988.

[28] B. V. Funt, M. S. Drew, and J. Ho. Color constancy from mutual reflection. *Int. Journal of Computer Vision*, 6(1):5–24, 1991.

[29] R. Gershon, A. D. Jepson, and J. K. Tsotsos. From [R,G,B] to surface reflectance: Computing color constant descriptors in images. In J. P. McDermott, ed., *Proc. of the 10th Int. Joint Conf. on Artificial Intelligence, Milan, Italy*, volume 2, pp. 755–758. Morgan Kaufmann, 1987.

[30] I. Harvey, P. Husbands, and D. Cliff. Issues in evolutionary robotics. In J.-A. Meyer, H. L. Roitblat, and S. W. Wilson, editors, *From animals to animats 2: Proc. of the 2nd Int. Conf. on Simulation of Adaptive Behavior, Honolulu, Hawaii, 1992*, pp. 364–373. The MIT Press, 1993.

[31] J. Herault. A model of colour processing in the retina of vertebrates: From photoreceptors to colour opposition and colour constancy phenomena. *Neurocomputing*, 12:113–129, 1996.

[32] J. Ho, B. V. Funt, and M. S. Drew. Separating a color signal into illumination and surface reflectance components: Theory and applications. In G. E. Healey, S. A. Shafer, and L. B. Wolff, eds., *Color*, pp. 272–283, Boston, 1992. Jones and Bartlett Publishers.

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

[34] B. K. P. Horn. *Robot Vision*. The MIT Press, Cambridge, MA, 1986.

[35] A. C. Hurlbert and T. A. Poggio. Learning a color algorithm from examples. Tech. Report A.I. Memo No. 909, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1987.

[36] A. C. Hurlbert and T. A. Poggio. Synthesizing a color algorithm from examples. *Science*, 239:482–483, 1988.

[37] J. R. Koza. *Genetic Programming. On the Programming of Computers by Means of Natural Selection.* The MIT Press, Cambridge, MA, 1992.

[38] J. R. Koza. *Genetic Programming II. Automatic Discovery of Reusable Programs.* The MIT Press, Cambridge, MA, 1994.

[39] E. H. Land. The retinex. *American Scientist*, 52:247–264, 1964.

[40] E. H. Land. The retinex theory of colour vision. *Proc. Royal Inst. Great Britain*, 47:23–58, 1974.

[41] E. 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, 1986.

[42] E. H. Land and J. J. McCann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61(1):1–11, 1974.

[43] L. T. Maloney and B. A. Wandell. Color constancy: a method for recovering surface spectral reflectance. *Journal of the Optical Society of America A3*, 3(1):29–33, 1986.

[44] T. Möller and E. Haines. *Real-Time Rendering.* A K Peters, Natick, MA, 1999.

[45] A. Moore, J. Allman, and R. M. Goodman. A real-time neural system for color constancy. *IEEE Transactions on Neural Networks*, 2(2):237–247, 1991.

[46] S. M. C. Nascimento and D. H. Foster. Dependence of colour constancy on the time-course of illuminant changes. In C. Dickinson, I. Murray, and D. Carden, eds., *John Dalton's Colour Vision Legacy. Selected Proc. of the Int. Conf.*, pp. 491–499, London, 1997. Taylor & Francis.

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

[48] T. Pomierski and H. M. Groß. Verfahren zur empfindungsgemäßen Farbumstimmung. In G. Sagerer, S. Posch, and F. Kummert, eds., *Mustererkennung 1995, Verstehen akustischer und visueller Informationen, 17. DAGM-Symposium, Bielefeld*, pp. 473–480, Berlin, 1995. Springer-Verlag.

[49] Z.-u. Rahman, D. J. Jobson, and G. A. Woodell. Method of improving a digital image. *United States Patent No. 5,991,456*, 1999.

[50] 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*, 2003.

[51] B. Schiele and J. L. Crowley. Object recognition using multidimensional receptive field histograms. In B. Buxton and R. Cipolla, eds., *4th Europ. Conf. On Computer Vision, Cambridge, UK*, pp. 610–619, Berlin, 1996. Springer-Verlag.

[52] B. Schiele and J. L. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *Int. Journal of Computer Vision*, 36(1):31–52, 2000.

[53] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta. A standard default color space for the internet - sRGB. Tech. Report, V. 1.10, 1996.

[54] M. J. Swain and D. H. Ballard. Color indexing. *Int. Journal of Computer Vision*, 7:11–32, 1991.

[55] M. F. Tappen, W. T. Freeman, and E. H. Adelson. Recovering intrinsic images from a single image. Tech. Report AI Memo 2002-015, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 2002.

[56] S. Usui and S. Nakauchi. A neurocomputational model for colour constancy. In C. Dickinson, I. Murray, and D. Carden, eds., *John Dalton's Colour Vision Legacy. Selected Proc. of the Int. Conf.*, pp. 475–482, London, 1997. Taylor & Francis.

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

[58] D. Zongker and B. Punch. *lil-gp 1.01 User's Manual (support and enhancements B. Rand).* Michigan State University, 1996.