Learning invariants to illumination changes typical of indoor environments: application to image color correction

B. Bascle, O. Bernier, V. Lemaire

Orange Labs / France Telecom R & D

Abstract. This paper presents a new approach for automatic image color correction, based on statistical learning. The method both parameterizes color independently of illumination and corrects color for changes of illumination. This is useful in many image processing applications, such as image segmentation or background subtraction. The motivation for using a learning approach is to deal with changes of lighting typical of indoor environments such as home and office. The method is based on learning color invariants using a modified multi-layer perceptron (MLP). The MLP is odd-layered. The middle layer includes two neurons which estimate two color invariants and one input neuron which takes in the luminance desired in output of the MLP. The advantage of the modified MLP over a classical MLP is better performance and the estimation of invariants to illumination. The trained modified MLP can be applied using look-up tables (LUTs), yielding very fast processing. Results illustrate the approach and compare it with other color correction approaches from the literature.

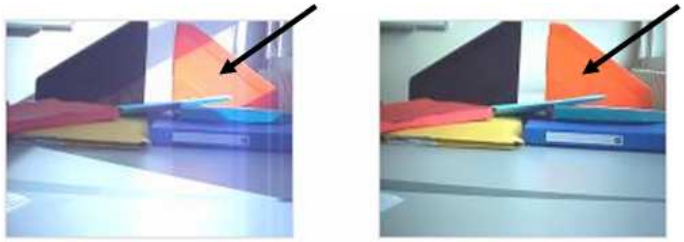

Fig. 1. Illumination changes can change the appearance of colors.

1 Illumination correction: problem and prior art

The apparent color of objects in images depends on the color of the light source(s) illuminating the scene (see example on fi g. 1). Because of this color constancy problem, image processing algorithms using color, such as color image segmentation or object recognition algorithms, tend to lack robustness to illumination changes. Such changes occur frequently in images, due to shadows, switching lights on or off, and/or the variation of sunlight during the day. To deal with this, a color correction scheme that can compensate for illumination changes is needed.

Color in images is usually represented by a triband signal, for instance Red-Green-Blue (RGB) or Cyan-Magenta-Yellow (CMY). As discussed above, such a triband signal is sensitive to changes in illumination. However, image processing techniques need to be made robust to such changes. This can be done by reparameterizing color (with one or two parameters) independently of illumination or by correcting the triband color signal. A number of color parametrization and color correction schemes have been described in the literature (Barnard, Martin, Coath, & Funt, 2002). This section describes a number of approaches that work on a single image. Table 1 summarizes their pros and cons.

Table 1. Comparison of color correction approaches that work on a single image.

approach	principle of the approach	local / global	cons	pros
estimation of illuminant color (Funt, Cardei, & Barnard, 1997)	neural network estimates illuminant chromaticity from image uv histogram	global	same illuminant for whole image, further processing for image correction	illuminant explicitly identified
ratio-based color invariants (Gevers & Smeulders, 1997)	analytic color invariants	local / pixel-wise	original image can't be reconstructed from invariant images	fast
luminance correction in HSV space (Gonzalez & Woods, 2002)	simple analytic color correction	local / pixel-wise	completely local, relatively sensitive to illumination changes	very fast using LUTs
color transfer (Reinhard *et al*, 01)	normalization by mean and variance in $l\alpha\beta$ color space	global	limited to global changes in illumination	fast
intrinsic image by entropy minimization (Finlayson, Drew, & Lu, 2004)	finds an axis invariant to illuminant color by entropy minimization, then projects image perpendicularly to axis	global	need for few colors and many illuminations in image to find invariant axis, not fast	works for any illuminants
diagonal color correction	linear	global	restricting assumptions, no non-linearities	very fast
non-diagonal color correction (Funt & Jiang, 2003)	PCA-based linear correction	pixel	illuminants must be known	fast (using LUTs)
enhancement of dark images using modified multi-scale retinex (Tao & Asari, 2003)	multi-scale convolution (linear)	local areas	color correction for visual effect, performance for background subtraction unknown	fairly fast (3 fps for 640x480 images), any lighting (blueish, etc ...)
color correction using a "classic" MLP (Yin & Cooperstock, 2004)	statistical learning of non-linear color correction transform by MLP	pixel with learnt global a priori for lighting	trained for given rear projection setup & lighting conditions, does not estimate color invariants	could be very fast (using LUTs)
color correction using a trained modified MLP (this paper)	statistical learning of non-linear color correction transform by MLP + statistical learning of 2 color invariants	pixel with learnt global a priori about type of lighting	trained for range of lightings (e.g. customary in home and office e.g. whitish or yellowish)	very fast (LUTs, 3.75 ms per frame or 266 fps for 320x240 images), trained for range of illuminations

Examples of directly correcting the triband signal are diagonal color correction (such as gray world and white patch (Rizzi, Gatta, & Marini, 2002)) and non-diagonal color correction (Funt & Jiang, 2003). They are both linear, and cannot model non-linearities. They also rely on limiting assumptions (known image mean for gray world, known maximum value for each channel for white patch, known illuminants for (Funt & Jiang, 2003)). They are very fast and can be implemented using LUTs for even greater speed.

Another approach which directly corrects the triband signal is (Yin & Cooperstock, 2004). A neural network is used to learn the color correction needed in a specifi c rear projection environment. It does not estimate color invariants. It also is trained for specifi c and unique lighting conditions.

An example of mono-band parametrization of color is hue (from hue-saturation-value, a.k.a. HSV) (Gonzalez & Woods, 2002). Examples of bi-band color parameterization are chrominances uv (from the YUV color space) (Gonzalez & Woods, 2002) and the ab values from the CIE Lab color space (Gonzalez & Woods, 2002). These three color representations (H, uv or ab) are analytical and thus do not require learning. They are fast pixel-wise methods, and have a certain robustness to illumination changes, but this robustness is limited. Color transfer (Reinhard, Ashikhmin, Gooch, & Shirley, 2001) is a method with a similar philosophy, normalizing color by its mean and variance in $l\alpha\beta$ space. It is global and fast, but limited to global changes in illumination.

An approach for estimating color invariants from images consists in calculating ratios of RGB components at a given pixel (R/B) or between neighboring pixels (such as $(R_{x_1}G_{x_2})/(G_{x_1}R_{x_2})$) (Gevers & Smeulders, 1997). This method is also pixel-wise and

thus fast. These invariants are also very robust to illumination changes. However, a lot of information about the original signal is lost and reconstructing it from the invariants is difficult.

A more sophisticated method has been proposed by (Finlayson et al., 2004). It estimates a mono-band invariant and is based on a physical model of image formation. It works globally from the whole image. In $(log(R/B), log(G/B))$ color space, an axis invariant to illuminant color is determined by entropy minimisation. Projecting the image perpendicularly to the axis gives corrected colors. The approach does not require learning and applies to any type of illuminant, but is relatively slow. It also requires that the image contains relatively few different colors and many changes of illumination for each color.

Yet another approach consists in explicitly estimating the color of the illuminant (Funt et al., 1997). A neural network estimates the chromaticity of the illuminant from the histogram of chromaticity of the whole image. The method works globally from the whole image and supposes there is only one illuminant for the entire image.

Another method is (Tao & Asari, 2003). It is a bit out of the scope of this paper, since it aims at the enhancement of dark images for visual effect, and does not give information about performance for color correction. However, it gives a benchmark about speed, since the authors aimed at fast processing. This will be discussed in section 3.5.

This brief description of the single image color correction approaches in the literature shows that there are a variety of approaches, that go from very general to specific and from slow to fast. In fact, each method corresponds to a different compromise be-

tween correction quality / generality and speed, with more general approaches usually being slower. The choice of a correction method depends on the context, e.g. the application, the possible customization of the system, how controlled the environment is, the CPU power and the processing time constraints. For instance, in a controlled environment with precise and known settings, such as known light sources (Yin & Cooperstock, 2004), color correction can be customized and done faster. However, it cannot be applied to another environment. In a totally uncontrolled and unknown environment with unknown light source(s), color correction is time consuming and can be done only for certain types of images: for instance images containing few colors if there are many different illuminations (Finlayson et al., 2004), or images with only one global illuminant (Funt et al., 1997). Very general approaches also tend to lose a lot of information about the original color signal, for example when calculating invariants such as in (Gevers & Smeulders, 1997). Thus their discrimination between different colors is not always very good. This is a problem if corrected colors are then used for image segmentation or object detection. Articles in the literature tend toward the ends of the speed/generality spectrum: methods are either very general or customised to a specific environment. This paper proposes a color correction method that tries to find a medium point on the spectrum. It is fast but makes only limited hypotheses about the environment (namely that it is an indoor environment such as home or office). The details of the approach and the motivation are described in section 2.

Fig. 2. Our main idea is to learn (using statistical learning techniques) the influence on colors of lighting changes typical of indoor environments such as home and office.

2 A statistical approach to measure color invariants

2.1 A modified multi-layer perceptron: motivation

The motivation of this work is twofold: (1) to parameterize color compactly and independently of illumination by two invariants (2) to do it in real-time. Firstly, two parameters are needed to parameterize color with enough degrees of freedom to reconstruct a triband signal, given a luminance (or a gray level signal). Secondly, real-time processing (25/30 images per second for video) is also necessary for some applications. For this, slow methods such as (Funt et al., 1997) and (Finlayson et al., 2004) are unsuitable. Pixel-wise approaches are more suited. Among those, Hue-Saturation, uv (from YUV) and ab (from the CIE Lab color space) lack robustness to illuminations changes. (Gevers & Smeulders, 1997) is robust to these, but reconstructing an image from the invariant(s) is difficult. A new fast approach is needed.

In practice, a limited range of illuminants are available in indoor environments. It is therefore interesting to use learning methods to find a color parameterization invariant to the "usual" illumination changes. This also provides a priori information about the

illuminants, making the color correction global, which is, as Land showed (Land & McCann, 1971)), necessary to perform correct illuminant correction. In practice, the lighting usually found in home and offi ces comes from fluorescent lights, incandescent light bulbs and natural sunlight from windows (see fi g. 2). They tend towards the whitish and yellowish areas of the spectrum (very few bluish or reddish lights). These are the illuminants that our approach deals with.

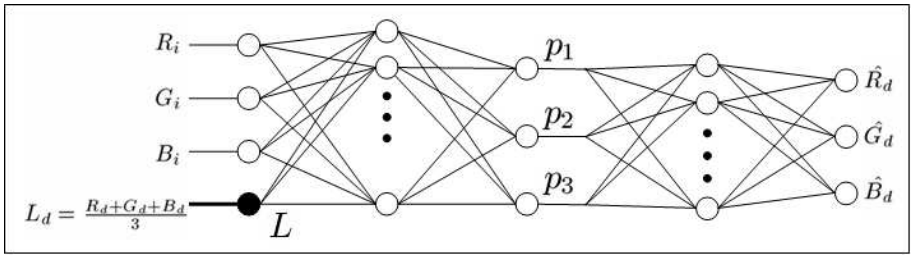

Fig. 3. A classical MLP with 4 inputs can be used to perform color correction. (R_i, G_i, B_i) is the input color. (R_d, G_d, B_d) is the desired output color, corresponding to the same color seen under a different illumination. $L_d = \frac{R_d+G_d+B_d}{3}$ is the luminance of the desired output and is a direct function of the illumination. Bias neurons are omitted from this fi gure.

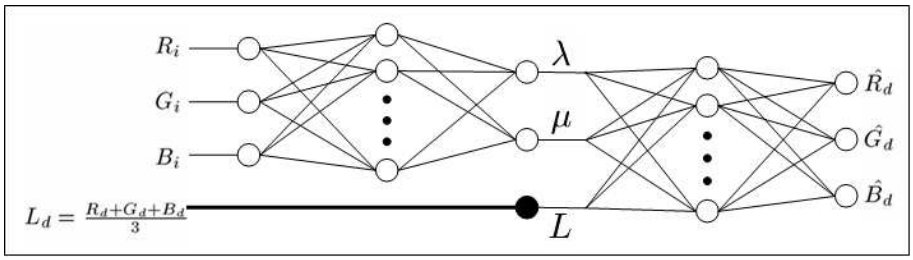

Fig. 4. A modified MLP is proposed for color correction and color invariant learning. λ and μ are the color parameters invariant to illumination that the MLP is trained to estimate. $(\hat{R}_d, \hat{G}_d, \hat{B}_d)$ are the actual outputs of the network. Bias neurons are omitted from this fi gure.

Our learning method of choice is neural networks and more specifically multi-layer perceptrons (MLPs) for their ease of use and adaptability (see appendix A for more details about MLPs, their training and use).

A classic MLP with 4 input neurons and 3 output neurons can be used for color correction under varying illuminations (see fig. 3). The three first inputs give the input color. The fourth input, a context input, is the luminance L of the expected output and is a direct function of the illumination. This fourth input neuron prevents the mapping to be learnt by the MLP from including one-to-many correspondences (the different corrected colors corresponding to the same input color with different illuminations) and thus makes it solvable. If in addition the MLP contains a bottleneck layer with 3 neurons, then these perform a re-parameterization of RGB space. However the three color parameters estimated by the 3 neurons (called here $p_1 p_2 p_3$) have no reason to be invariant to illumination.

To force the MLP to code color independently of illumination, the architecture of the traditional MLP is modified (see fig. 4). The entry point L of the MLP (fourth input neuron) is moved to the bottleneck layer of the network so that it becomes the third and last neuron of this layer. This displaced entry makes our MLP different from a trivial compression network. The two other neurons of the bottleneck layer have outputs (λ, μ). During training, the network learns to reconstruct the corrected color (R_d, G_d, B_d) from (λ, μ) and the desired output luminance $L_d = \frac{R_d+G_d+B_d}{3}$. Thus it learns to ignore the luminance of the input (R_i, G_i, B_i) and learns to estimate two color characteristics (λ, μ) that are invariant to illumination.

The approach does not require any camera calibration or knowledge about the image. However, it supposes that the illuminants available in images on which we want to perform color correction in the future are of the type commonly found in indoor environments (and that the MLP-based color correction is trained for). It also supposes that the basis of images used for learning is representative of the illuminants commonly available in indoor environments.

2.2 Training the modified multi-layer perceptron

As shown in fig. 4, the modified MLP includes 5 layers. This could be generalized to any odd number of layers, however using too many hidden layers could lead the neural network to overfit the data and to have bad generalization. This means that learning would become too specific to the learning data and lose the ability to successfully deal with test sets that are independent to the training data .

Experiments showed that 5 layers provided enough generalization without overfitting. The input and output layers have 3 neurons each (plus an additional bias), for RGB inputs and outputs. The middle layer includes 3 neurons (excluding bias): their outputs are called λ, μ and L. The second and fourth layers have arbitrary numbers of neurons, typically between 3 and 10 in our experiments. Cross-validation (Kohavi, 1995), which is a model evaluation method that estimates generalization error by partitioning the data into subsets, showed 8 neurons to give the best results. The links between neurons are associated to weights. Neurons have sigmoid activation functions. The network includes biases and moments (Bishop, 1996).

A database of images showing the same scenes under different illuminations is used to train the modified MLP. The illuminations are typical of indoor environments such as home and office.

A classic MLP training scheme based on backpropagation is applied. A pixel is randomly sampled at each iteration from the training set. Its RGB values before and after an illumination change (from real images) are used as input (R_i, G_i, B_i) and desired output (R_d, G_d, B_d) to the network. Propagation and back-propagation are then performed, with one modification: as mentioned above, the output L of the third neuron of the third layer is forced to the value of the luminance corresponding to the desired output color.

2.3 Use of the modified multi-layer perceptron

The trained modified MLP can be used to correct color images. Each image pixel is propagated through the first half of the trained network to find the invariants λ and μ. An arbitrary luminance L is imposed on the pixel by forcing the output of the third neuron of the third layer to L. The output of the trained network then gives the corrected color. If a constant luminance L is used for all pixels in the image, an image corrected for shadows and for variations of illumination across the image and between images is obtained. The color correction can be tabulated for fast implementation. Note that the approach could be easily extended to a greater number of inputs and outputs or different inputs/outputs than RGB. For instance, YUV or HSV, or redundant characteristics such as RGBYUVLab could be used as inputs and outputs.

3 Image correction results

3.1 Experimental conditions and database

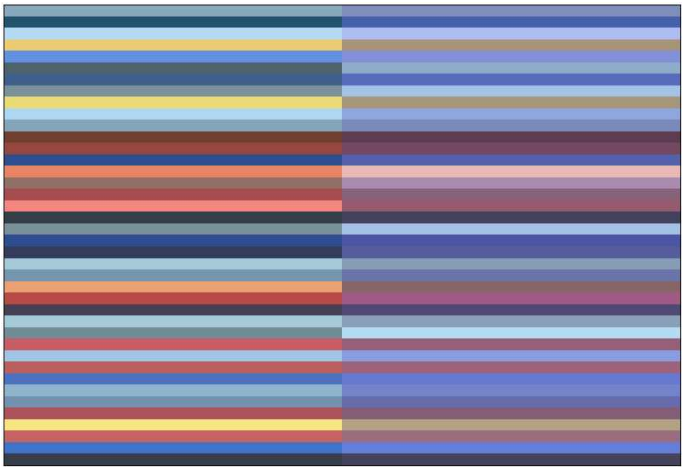

Fig. 5. Examples of pixel pairs in the training set.

The network was trained using 546000 pixels, randomly sampled from 91 training images, taken by 2 webcams (Philips ToUCam Pro Camera and Logitech QuickCam Zoom). The images were taken at home and in the office (containing images of desks, sofas, papers, colored file holders, clothing, etc...). The images included a large range of colors (including but not limited to the range of colors available in the Gretag Macbeth color checker), which is critical to get good generalisation results. Indeed the learning literature shows that having maximum variety in a training set is critical to correctly modeling data without modeling the noise and artifacts specific to a given training set. Only 6000 randomly selected pixels per image were used for training and testing to limit training time. Some examples of training pixels (before and after illumination changes)

are shown in fig. 5. The training images are of indoor scenes viewed under different illuminations typical of home and office environments (fluorescent lights, incandescent light bulbs and natural sunlight from windows). Many illumination conditions were filmed (light bulbs and fluorescent lights on and off at different times of the day). Testing was performed on other images taken by the 2 webcams used for training and by a third webcam, not used for training, a Logitech QuickCam for Notebooks Pro. Some results are also shown on images taken using a Canon Ixus camera.

In practice, using 8 neurons in the second and fourth layers of the MLP gives good performance. A gain of 1.0 was used, with a momentum factor of 0.01 and a learning rate of 0.001. Pixels that were too dark (luminance ≤ 20) or too bright / saturated (luminance ≥ 250) were not used for training, as the effect of illumination is slightly different toward the ends of the saturation spectrum.

3.2 Comparison with a "classical" multi-layer perceptron

Table 2. Mean error between reconstructed and target images for a "classical" MLP and the modified MLP presented in this article. The mean error was calculated using 748 320x240 test images (not in the training set). The error is averaged over the three color components (R,G,B).

	for a classical MLP	for the modified MLP
mean error (in pixel values $\in [0, 255]$)	10.47	5.54
relative mean error	4.11%	2.17 %

Table 2 shows that the modified MLP (fig. 4) performs better in reconstructing target images than a classic MLP (fig. 3). The reconstruction is done given the expected luminances L_d of the pixels of the desired target image.

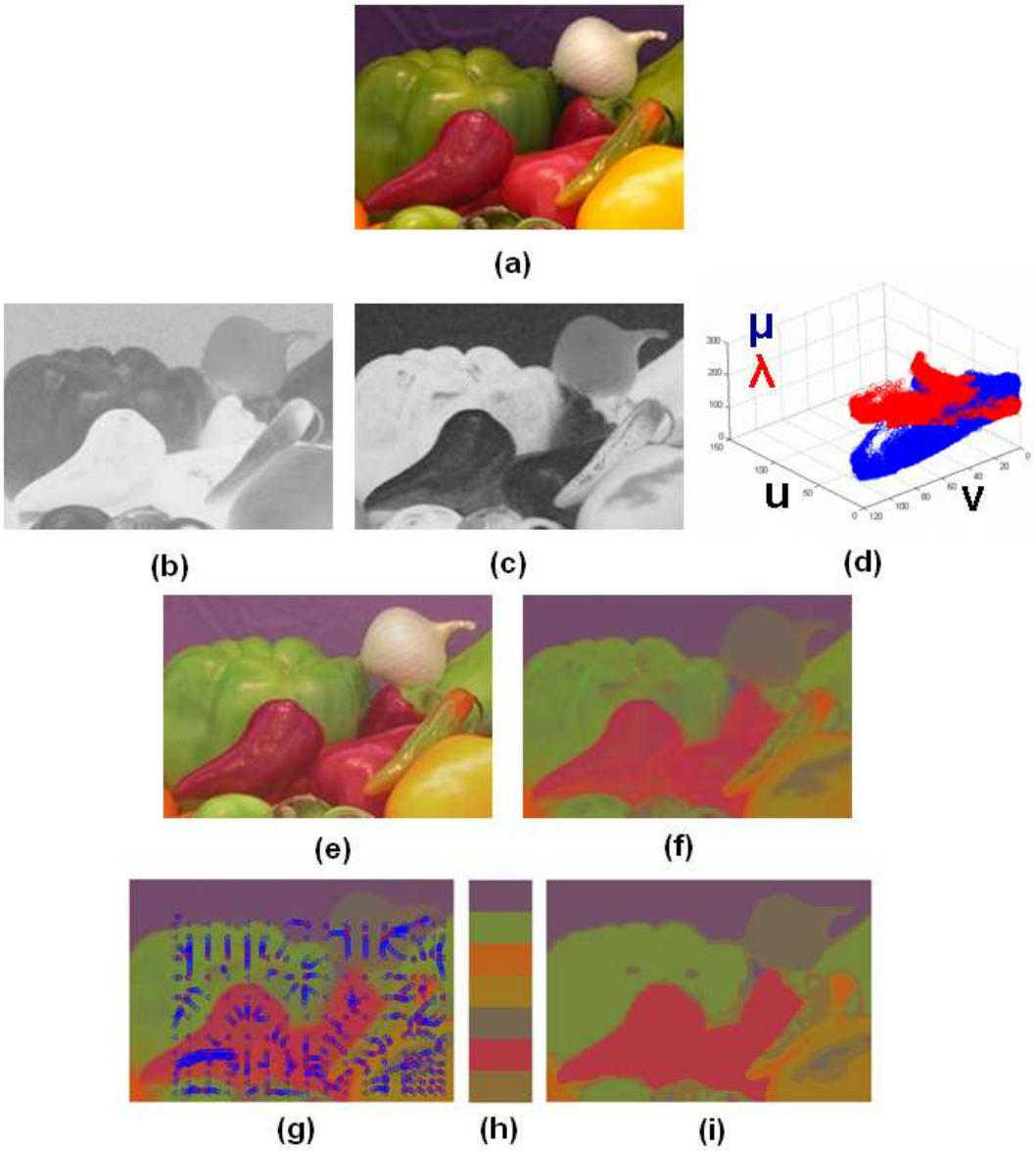

Fig. 6. Example of color correction learnt by the modified MLP. (a) original image (unknown illumination). (b) and (c) invariants λ and μ estimated by the MLP. (d) locus of the invariants in the uv space. (e) corrected image with pixel luminance inputs set to values proportional to pixel luminances in the original image (plus a constant). (f) corrected image with the pixel luminance inputs set to a constant value for all pixels. (h) 7 color peaks found by mean shift (g) in the corrected image (f). (i) resulting image segmentation.

3.3 Invariant estimation by the modified MLP

Figure 6 shows the two invariants (λ, μ) learnt by the modified MLP and calculated on an image (see part (a) of fig 6) of unknown illumination. The two invariants are seen in parts (b) and (c) of the figure. Objects of similar color to the human eye have similar values of λ and μ. Part (d) of fig. 6 shows the locus of the invariant values (λ, μ) in the image as a fonction of the chrominance values (u, v) (from YUV color space) of the image pixels. The locii of the two invariants are not identical, and thus we have two invariants and not only one. Part (f) of figure 6 shows the corrected image estimated for a constant luminance input over the image. Much of the influence of shading and variations of illumination across the image is removed, apart from specularities (white saturated areas) which the network is not trained to correct and which happen to be mapped to gray by the learnt color correction scheme. Areas of similar color in the original image (despite shading and illumination) have much more homogeneous color in the corrected image. This is further shown by performing mean-shift based color segmentation (Comaniciu, Ramesh, & Meer, 2000) (see (g)) on the corrected image. Seven areas of uniform color are readily identified and segmented (see part (h) and (i) of fig. 6) in the corrected image. They correspond roughly to what is expected by a human observer. This example illustrates that our modified MLP successfully learns a parameterization of color by two parameters that are invariant to illumination.

3.4 Comparison with other color correction methods from the literature

Figures 7, 8 and 9 compare our color correction approach with other color correction approaches.

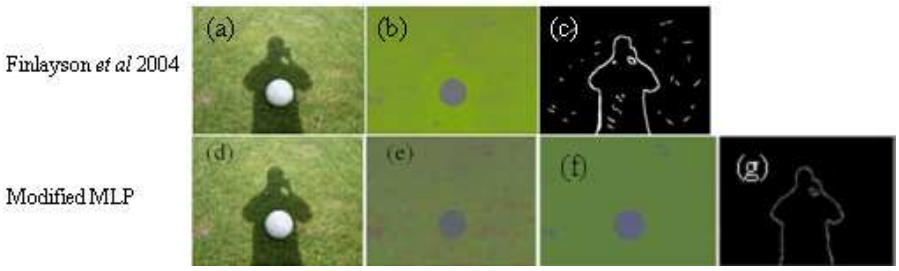

Fig. 7. Comparison of the pixel-wise color correction by the modified MLP presented in this paper and the whole-image color correction method of Finlayson *et al* 2004. Application to shadow detection. Example I. (a) and (d) show the original image. (b) is the invariant image obtained using the method of Finlayson *et al* 2004 and (c) shows the shadow edges estimated from (b). (e) shows the corrected image estimated using the modified MLP, (f) and (g) the results of mean shift color segmentation from (e) and (h) the shadow edges estimated from (g).

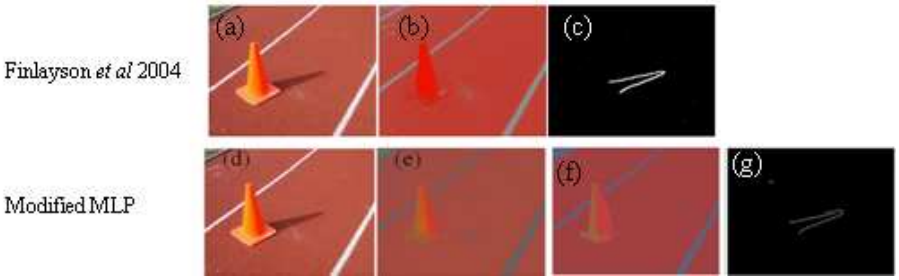

Fig. 8. Comparison of the pixel-wise color correction by the modified MLP presented in this paper and the whole-image color correction method of Finlayson *et al* 2004. Application to shadow detection. Example II. (a), (b), (c), (d), (e), (f), (g) and (h) illustrate the same steps as in fig. 7.

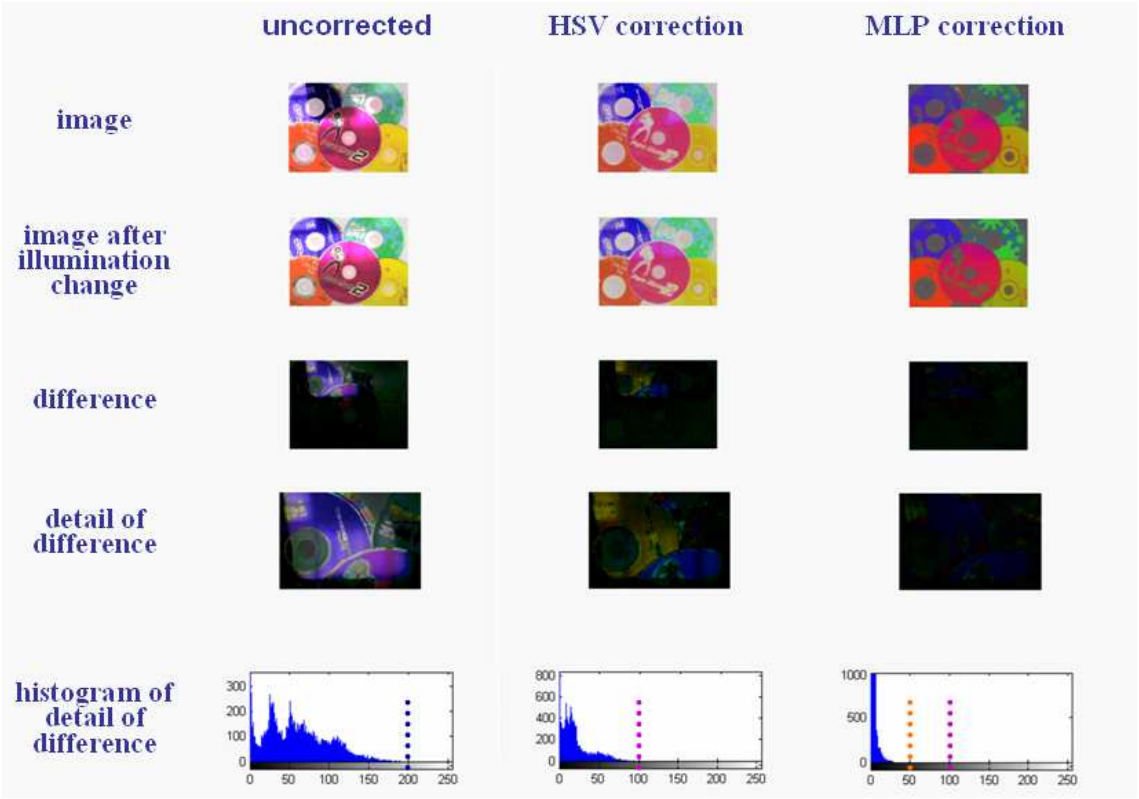

Fig. 9. Comparison of the pixel-wise color correction by the modified MLP presented in this paper and pixel-wise HSV-based color correction, HSV being the well known hue-saturation-value color space.

Figures 7 and 8 illustrate that our correction is of similar quality to that of Finlayson et al (Finlayson et al., 2004) (briefly described in the introduction of this paper). The application of color correction is the detection of shadow contours (which can be used for shadow removal, as shown in (Finlayson et al., 2004)). Even though it might be less robust to large light changes or unusual light changes (such as turning on a blue or red light), our method is faster, being pixel-wise.

Figure 9 compares our approach to HSV-based color correction and applies it to color-based background subtraction. The two first images of the first and third columns of the figure show that our color correction scheme is indeed robust to changes in illumination, since there is much less difference between the images after correction than before. The last row of figure 9 also shows that the correction performed in this paper compares favorably with an HSV-based color correction (which consists in taking an RGB color to hue-saturation-value space, setting its value/luminance to a constant, then going back to RGB space to get the corrected color).

3.5 Performance of a LUT implementation of the trained modified MLP

Color correction by the modified MLP can be tabulated, making it one the fastest possible color correction approaches. Execution time using LUTs is 3.75 ms for an entire 320x240 image, on a Pentium4 3GHz. This way, color correction can be used as a first step in video-rate image processing, without using a large part of the frame processing time (40ms). This LUT implementation is possible because the approach is pixel-wise.

An HSV correction scheme could be as fast (using LUTs), but it would be less performant, as illustrated by fig. 9. A color correction scheme based on (Finlayson et

al., 2004) would be of equal performance, as illustrated on examples by fig. 7 and 8. It could deal with more changes of illumination, since our approach is limited to the type of frequently found indoor lighting the modified MLP was trained for. However, working globally on the image, it could not be implemented as a LUT, and would thus be slower. Another basis of comparison for processing time is the approach of (Tao & Asari, 2003) (briefly described in section 1), which performs good-quality color enhancement at good speed, and which is slower than our approach (3 frames per second on a Pentium4 2.26GHz for a 640x480 image). Both approaches scale linearly in terms of the number of pixels to process. Our algorithm would take approx. 3.75x4=15ms to process a 640x480 image on a P4 3GHz, compared to approx. 330ms per image for (Tao & Asari, 2003)'s approach on a P4 2.26GHz (equivalent to 250 ms per image for a P4 3 GHz if we scale linearly).

4 Examples of applications

The approach can be used for a variety of applications. One application is color-based segmentation (see fig. 6), as shown in a previous section of this paper. Another is background subtraction (see fig. 10). Our color correction scheme makes background subtraction robust to illumination changes. Yet another application is the removal of shadow contours from contour maps. This makes them less noisy and more easily usable for such applications as object recognition. This is illustrated by 3 examples. Fig. 11 shows how color correction simplifies the contour map and makes it less dependent on the current illumination. Fig. 12 also illustrates this, but in a setting where the color correction becomes crucial. In this example, reflections on the water create many small

contours and make the contour of interest (that of the fi sh) very diffi cult to detect. The color correction removes the spurious contours and leave the fi sh contour easily detectable. The last fi gure (fi g. 13) again illustrates how our approach can remove shadow contours (as can be seen on the blue boat), but also some limitations of the approach. Contours between regions of the same color (here the white snoopy and the white background) are also detected as shadows and removed.

5 Conclusion

This paper presents a new neural network-based approach to estimating image color independently of illumination. A modifi ed multi-layer perceptron is trained to estimate two color invariants and an illumination- corrected color for each input color. The network is trained for typical indoor home and offi ce lighting (fluorescents and light bulbs) and outdoor natural light, using two webcams. Such statistical training gives the approach a good compromise between generality (being able to handle different types of illuminants) and discrimination power (being able to discriminate between different colors). Experiments with lighting changes and another webcam show that the training seems to have good generalization properties. Once learning has been achieved, color correction is very fast using look-up tables, so that color correction can be performed as a part of image pre-processing before applying other image processing algorithms (such as background subtraction or color-based image segmentation).

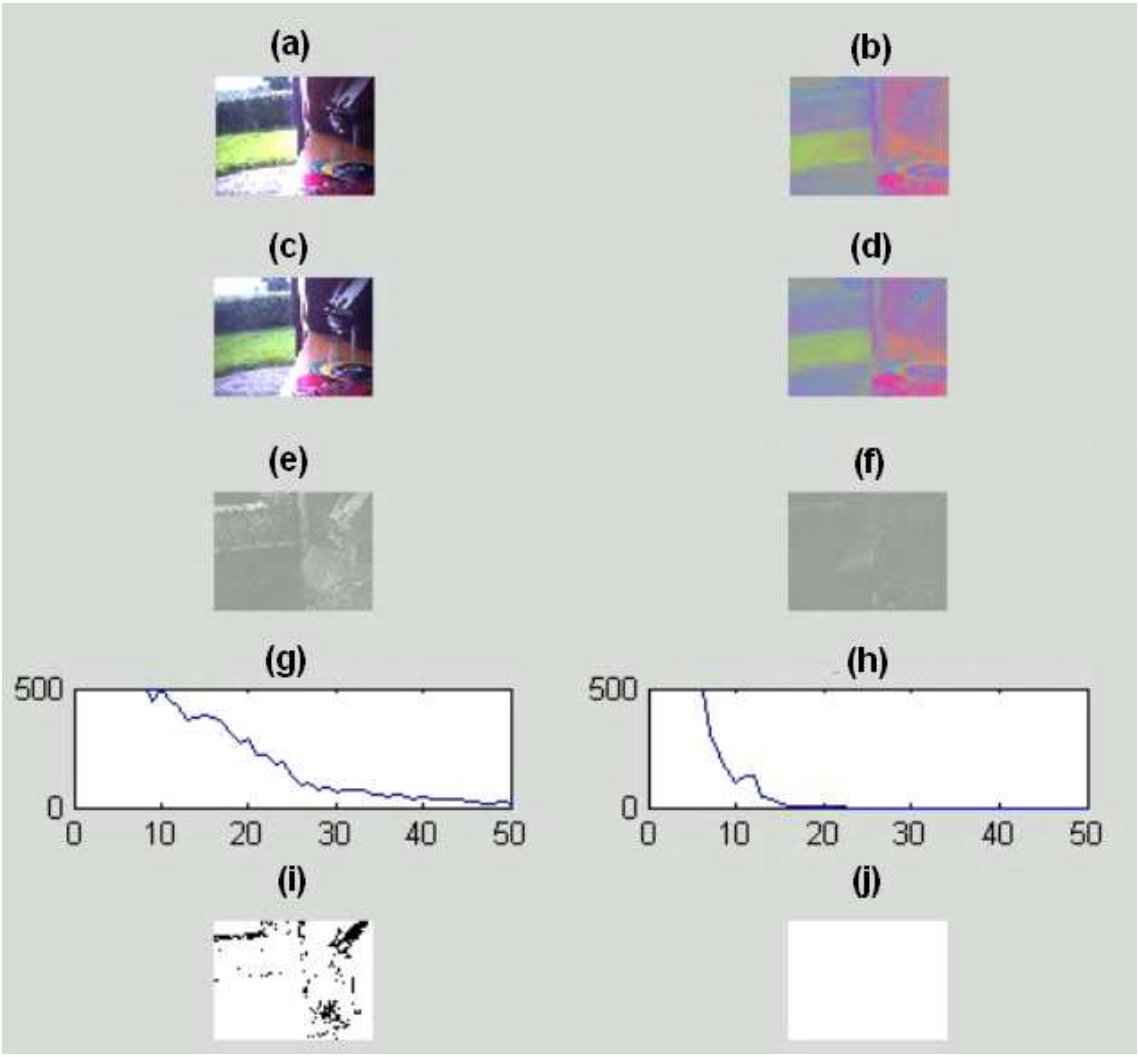

Fig. 10. Application of the approach to background subtraction. (a) and (c) show two images of the same scene under different illuminations (due to a change in sunlight, as can be seen on the grass). (b) and (d) show the color-corrected images. (e) shows the differences between the original images and (f) the difference between the corrected images (gray means that the difference is zero, and white that the difference is high). (g) and (h) show the histograms of the difference images (using absolute values). (i) and (j) show the thresholded difference images (white means that the difference is zero, and black that the difference is high). The color correction makes the background subtraction robust to the illumination change.

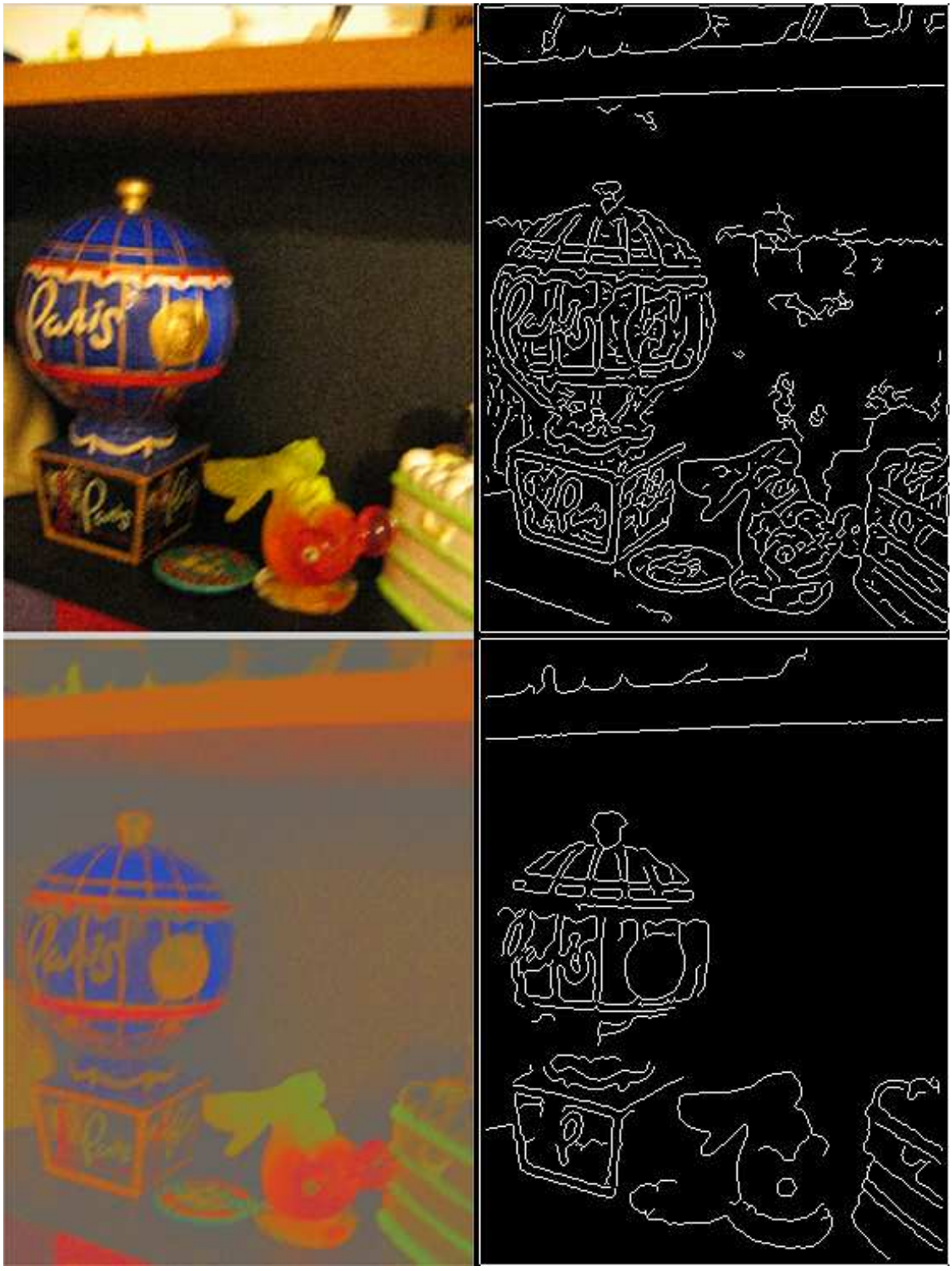

Fig. 11. Color correction for shadow contour removal I. Upper left is the original image. Upper right shows the edges of the original image. Lower left is the color corrected image and lower right shows the associated edges. The color correction simplifi es the contour map by removing shadow contours. It also makes the contour map less dependent on illumination, which would be useful in such applications as object recognition.

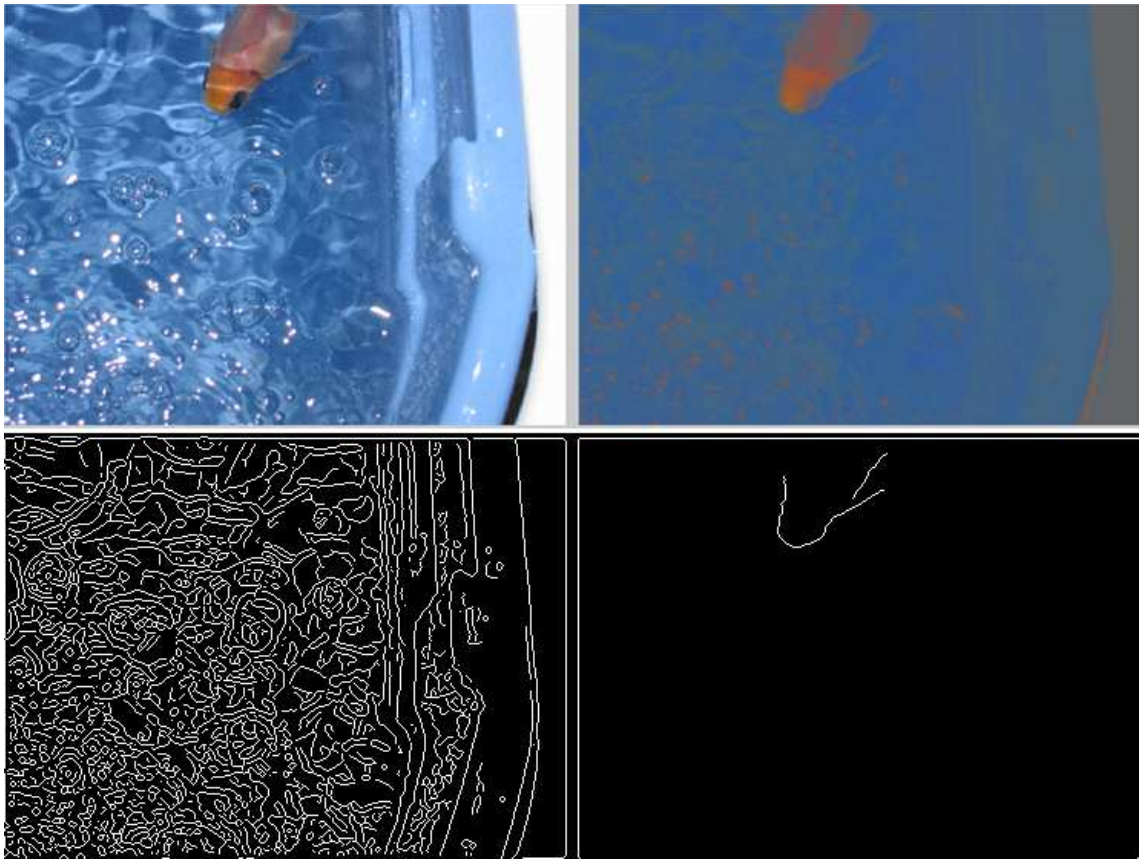

Fig. 12. Color correction for shadow contour removal II. Upper left is the original image. Lower left shows the edges of the original image. Upper right is the color corrected image and lower right shows the associated edges. The color correction simplifi es the contour map by removing shadow contours. In this example, it is crucial to the detection of signifi cant contours (that of the fi sh) among all the contours due to highlights.

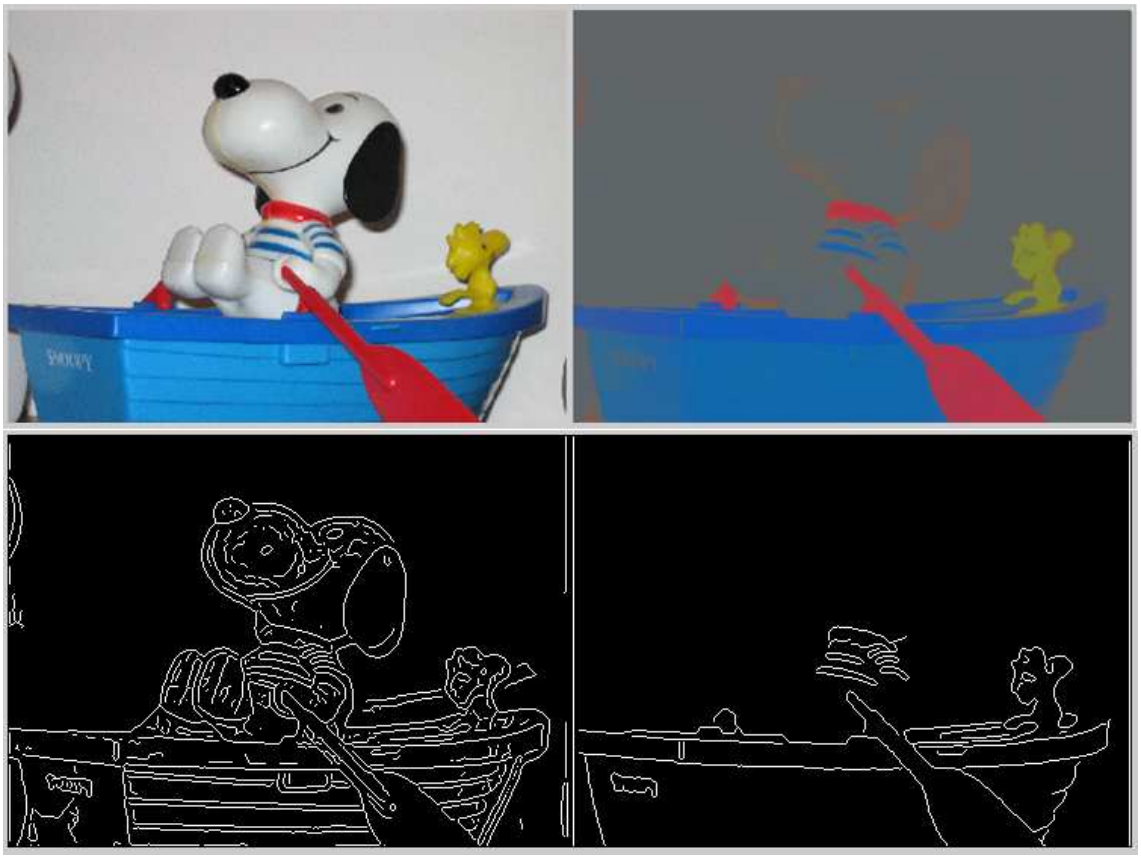

Fig. 13. Color correction for shadow contour removal III. Upper left is the original image. Lower left shows the edges of the original image. Upper right is the color corrected image and lower right shows the associated edges. The color correction simplifi es the contour map by removing shadow contours, as can be seen on the boat. On the other hand, it also removes signifi cant contours between regions of similar colors (the white snoopy and the white background). This is a limitation of the current approach.

A Appendix on neural networks and MLPs, their training and use

Neural networks are a wide class of non linear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to fi nd patterns in data. Examples of applications are flexible nonlinear regression and discriminant models, data reduction models and nonlinear dynamical systems. Artifi cal neural networks (ANNs or NNs) consist of an often large number of "neurons", i.e. simple linear or nonlinear computing elements, interconnected in often complex ways and often organised into layers.

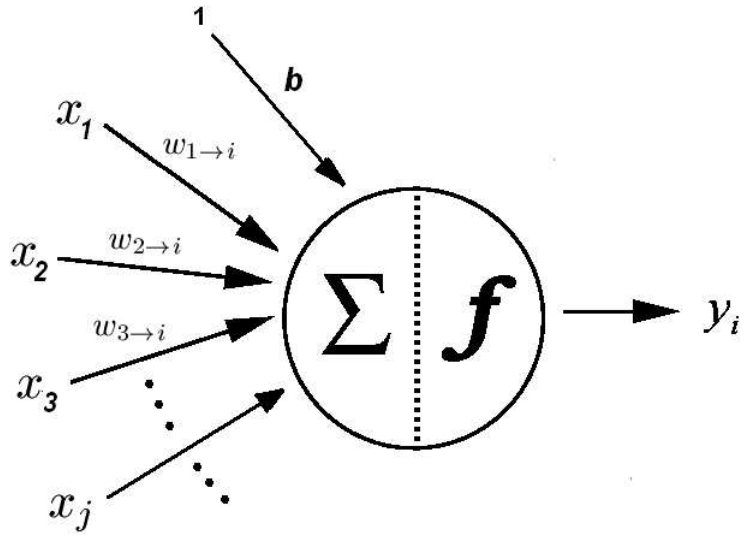

Fig. 14. A single neuron with multiple inputs $x_j, j \in \{1, .., R\}$ and output $y_i = f(\sum w_{j\to i} x_j + b_i)$, with b_i the bias, $w_{j\to i}$ the weights and f the activation function.

The commonly used McCullogh-Pitts neuron model is shown by fi g. 14. A neuron i applies weights $w_{j\to i}$ to its inputs x_j, sums them with a bias b_i, then passes them to

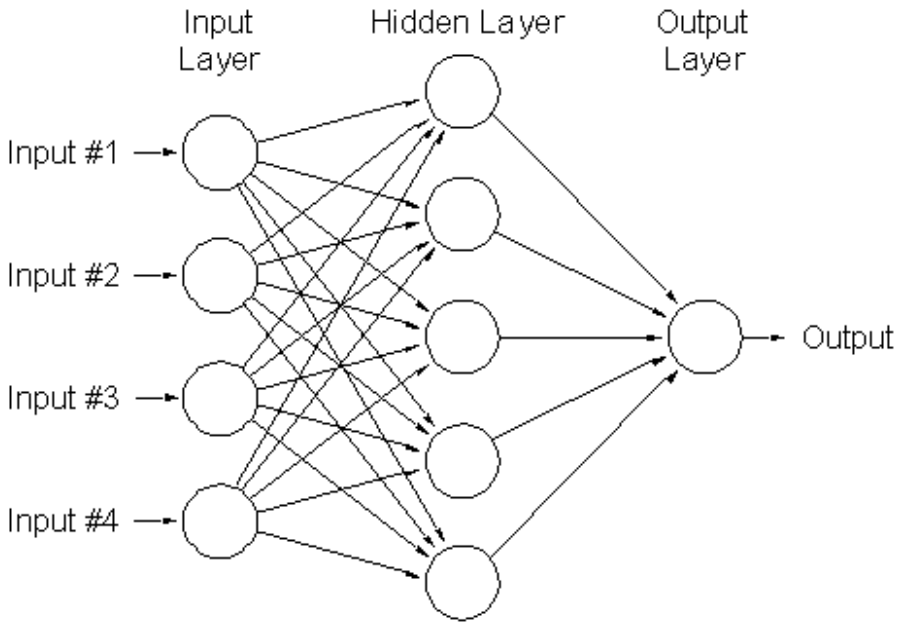

Fig. 15. Example of layered neural network or multi-layer perceptron (here with 3 layers).

a function f which produces the neuron output $y_i = f(\sum w_{j \to i} x_j + b_i)$. A commonly used non linear activation function $f(x)$ is the logistic or sigmoid function $1/(1+e^{-x})$.

Neural networks are displayed as network diagrams, showing neurons represented by circles and boxes, and their connections. One of the most commonly used artificial neural networks are called multilayer perceptrons (MLPs). They are nothing more than nonlinear regression and discriminant models. As their name indicates, they consist in multiple layers of neurons. A typical MLP network diagram is shown by fig. 15.

Neural networks and MLPs can be trained and their weights optimised for a task, for instance regression in our case.

The training procedure for MLPs is commonly referred to as backpropagation of error, or backpropagation (backprop) for short. Backpropagation is any technique for

adapting the weights of parameters of a nonlinear system by somehow using such derivatives or the equivalent. One of the most commonly used technique is the delta rule, developed by Widrow and Hoff for one layer only, and also called the Least Mean Square (LMS) method. For a given input vector, the output vector is compared to the correct answer. If the difference / error is zero, no learning takes place; otherwise, the weights are adjusted to reduce this difference. The weight change is derived from gradient descent (with derivatives being computed by simple application of the chain rule). The change in weight $w_{j \to i}$ from neuron j to i is given by: $dw_{j \to i} = -r * y_j * e_i * f'(a_i)$, where r is the learning rate, y_j represents the output of neuron j and $e_i = y_i - y_i^d$ is the difference between the desired output y_i^d and the actual output y_i of neuron i. f' is the derivative of the activation function f and $a_i = \sum_j w_{j \to i} y_j + b_i$ is the activation. A generalized form of the delta rule, developed by D.E. Rumelhart, G.E. Hinton, and R.J. Williams, is used for networks with hidden layers. The errors are backpropagated from the output layer backwards with $e_j = y_j - y_j^d$ for the output layer and $e_j = \sum_i w_{j \to i} e_i f'(a_i)$ for the hidden layers. The weights of each layer are updated during the backpropagation using $dw_{j \to i} = -r * y_j * e_i * f'(a_i)$. In practice, the activation function f is often a sigmoid function $f(x) = 1/(1 + e^{-Gx})$ with gain G.

References

Barnard, K., Cardei, V., & Funt, B. (2002). A comparison of computational color constancy algorithms—part 1: Methodology and experiments with synthesized data. *IEEE Trans. on Image Processing, 11*(9), 972–983.

Barnard, K., Martin, L., Coath, A., & Funt, B. (2002). A comparison of computational color constancy algorithms, part 2: Experiments with images. *IEEE Trans. on Image Processing*, *11*(9), 985–996.

Bishop, C. (1996). *Neural networks for pattern recognition*. Oxford University Press.

Comaniciu, D., Ramesh, V., & Meer, P. (2000). Real-Time Tracking of Non-Rigid Objects using Mean Shift. In *Proc. of IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR 2000)* (pp. 142–151).

Finlayson, G. (2006). Three-, two-, one-, and six-dimensional color constancy. *"Color Image Processing: Methods and Applications ", (eds.) R. Lukac and K.N. Plataniotis, CRC Press / Taylor & Francis"*, 55–74.

Finlayson, G., Drew, M., & Lu, C. (2004). Intrinsic Images by Entropy Minimization. In *Proc. 8th European Conf. on Computer Vision (ECCV'04), Prague, pp 582-595* .

Funt, B., Cardei, V., & Barnard, K. (1997). Neural Network Colour Constancy and Specularly Reflecting Surfaces. In *Proc. of 8th Congress of the Internal Colour Association (AIC Colour 97), Kyoto, Japan.*

Funt, B., & Jiang, H. (2003). Non-diagonal colour correction. *Proc. International Conference on Image Processing (ICIP 2003), Barcelona*.

Gevers, T., & Smeulders, A. W. M. (1997). Color Based Object Recognition. In *ICIAP '97: Proceedings of the 9th International Conference on Image Analysis and Processing-Volume I* (pp. 319–326).

Gonzalez, R., & Woods, R. (2002). *Digital image processing*. Prentice Hall.

Gu, I.-H., & Gui, V. (2001). Colour image segmentation using adaptive mean shift filters. In *Proc. of . Int. Conference on Image Processing (ICIP'01)* (Vol. 1, pp. 726–729).

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proc. of the Fourteenth International Joint Conference on Artificial Intelligence 2 (12): 1137–1143*.

Land, E., & McCann, J. (1971). Lightness and retinex theory. *Journal of the Optical Society of America, 61*(1), 1.

Lukac, R. (2007). Refined automated white balancing. *IET Electronics Letters, 43*(8), 445–446.

Luo, Q., & Khoshgoftaar, T. (2004). Efficient Image Segmentation by Mean Shift Clustering and MDL-Guided Region Merging. In *Proc. of 16th IEEE Int. Conf. on Tools with Artificial Intelligence (ICTAI'04)* (pp. 337– 343).

Reinhard, E., Ashikhmin, M., Gooch, B., & Shirley, P. (2001). Color transfer between images. *IEEE Computer Graphics and Applications, 21*(5), 34–41.

Rizzi, A., Gatta, C., & Marini, D. (2002, June). Color correction between gray world and white patch. In *Proc. SPIE Vol. 4662, p. 367-375, Human Vision and Electronic Imaging VII* (p. 367-375).

Rosenberg, C., Minka, T., & Ladsariya, A. (2003). Bayesian color constancy with non-gaussian models. *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003]*.

Tao, L., & Asari, V. (2003). Modified Luminance Based MSR for Fast and Efficient Image Enhancement. *Proc. of 32nd Applied Imagery Pattern Recognition Workshop*

(AIPR'03), 174–179.

Tappen, M., Freeman, W., & Adelson, E. (2005, sept). Recovering intrinsic images from a single image. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (Vol. 27, pp. 1459–1472).

Tominaga, S. (1998). Color coordinate conversion via neural networks. *Proc. of International Conf. on Colour Imaging in Multimedia (CIM'98)*, 87–96.

Vrhel, M., & Trussell, H. (1999). Color scanner calibration via a neural network. *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 99)*, *6*, 3465–3468.

Weng, C., Chen, H., & Fuh, C. (2005). A novel automatic white balance method for digital still cameras. *Proc. IEEE Int. Symposium on Circuits and Systems*, *4*, 3801–3804.

Yin, J., & Cooperstock, J. (2004). Color Correction Methods with Application to Digital Projection Environments. In *12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG'04)* (p. 499-506).