Bootstrapping Color Constancy

Brian Funt and Vlad C. Cardei*

Simon Fraser University
Vancouver, Canada

ABSTRACT

Bootstrapping provides a novel approach to training a neural network to estimate the chromaticity of the illuminant in a scene given image data alone. For initial training, the network requires feedback about the accuracy of the network's current results. In the case of a network for color constancy, this feedback is the chromaticity of the incident scene illumination. In the past[1], perfect feedback has been used, but in the bootstrapping method feedback with a considerable degree of random error can be used to train the network instead. In particular, the grayworld algorithm[2], which only provides modest color constancy performance, is used to train a neural network which in the end performs better than the grayworld algorithm used to train it.

Keywords: color constancy, neural networks, color correction

1. ADDRESSING THE COLOR CONSTANCY PROBLEM

1.1. Why do we need color constancy?

The colors of surfaces in an image is determined in part by the color of the light source under which that image was taken. Thus, variations of the color of the illumination in a scene produce color shifts of the surfaces in that scene. Without color stability, most problems where color is taken into account (e.g. color based object recognition[3] systems and digital photography) will be adversely affected even by small changes in the scene's illumination. For a human observer, however, the perceived color shifts due to changes in illumination are relatively small; in other words, humans exhibit a relatively high degree of color constancy[4].

1.2. A computational approach

From a computational perspective, we would like to compensate for the effect that variations in the color of the incident illumination have on the colors in an image and create an image with the same colors as would be obtained for the same scene under a standard, 'canonical' illuminant.

In this paper, we will assume that the chromaticity of the scene illumination is constant spatially, although its brightness might vary across the image. The goal of a machine color constancy system will be taken to be the accurate estimation of the chromaticity of the scene illumination from a 3-band digital color image of the scene. After estimating the illuminant's chromaticity, the scene can then be color corrected[5] based on a diagonal, or coefficient-rule transformation.

In order to achieve color constancy, we proposed a multi-layer neural network[1]. Based only on the chromaticity histogram of the input image, the neural network computes an estimate of the scene's illumination. The network was trained on scenes that were synthesized from databases of surface reflectances and illuminant spectral power distributions. In addition to this data, the camera sensor sensitivity functions were also known. For each synthesized scene in the training set, the network was trained by feeding the binarized histogram of the scene's chromaticities to its input layer. The network outputs an estimate of the chromaticity of the scene illumination which is compared to the actual illumination used to generate that scene and the network's internal weights are then adjusted using the backpropagation method[6]. The network can also be trained using data from real images instead of synthesized ones.

Both methods have disadvantages. To synthesize scenes, the spectral sensitivity functions of the camera must be known and other artifacts[7] (e.g. noise, specularities, camera non-linearity, flare, etc.) that often appear in real images must also be

* Correspondence: Brian Funt. Other author information: Email: {funt, vcardei}@cs.sfu.ca; School of Computing Science, Simon Fraser University, 8888 University Drive, Burnaby, B.C., Canada, V5A 1S6.

taken into account. On the other hand, using real data requires a large set of images for which the actual illuminant has been measured. The bootstrapping approach that will be presented in this paper helps address both problems.

2. BOOTSTRAPPING THE NEURAL NETWORK TRAINING ALGORITHM

2.1. The bootstrapping algorithm

Consider a set of images taken under unknown illuminants with an uncalibrated camera (i.e. a camera with unknown sensor sensitivities). The only assumption we make, and which is easy to confirm given the actual data, is that the images have a relatively large number of colors.

The illumination chromaticity is estimated using the grayworld algorithm (GW) described below, which assumes that the average color of the scene is gray and that any departure from this average in the image is caused by the color of the illuminant. We generate a large training data set by sub-sampling the given set of images, which is then used for training a neural network. Instead of the actual illuminant, the network receives the grayworld algorithm's estimate of the illumination chromaticity. The same grayworld estimate is used for all sub-samplings of an image. Thus, we are able to train on a large data set derived from real images without knowing the actual illuminant of the images.

We tested this algorithm both on a very large number of artificial images generated from a database of 100 illuminants and 260 surface reflectances and on real images taken with a digital camera. Although trained with inexact target values, the neural network performs better than the GW algorithm that was initially used to train it, especially for scenes with a small number of surfaces.

2.2. The gray world algorithm

To illustrate the bootstrapping algorithm, we used a version of the gray world algorithm. The chromaticity of the illuminant is determined from a weighted average of all the pixels in an image. This average is computed relative to the gray world average of the colors in the database. This helps compensate for the fact that the distribution of surface colors is not necessarily precisely gray. For example, R_{illum}, the red component of the illuminant is computed as:

$$R_{illum} = \frac{\mu^R}{\mu^R_{DB}} R_{sensor} \tag{1}$$

where μ^R is the average red of the image, μ^R_{DB} is the average red of the database and R_{sensor} is the camera sensor response for white under the canonical illuminant. In our experiments, we took the canonical illuminant to be the one for which the camera is calibrated (i.e. a perfect white patch will generate equal responses for all three camera sensors). Consequently we have:

$$R_{sensor}=G_{sensor}=B_{sensor}=255 \tag{2}$$

For artificial images, the database average is computed from all the surface reflectances used to generate the data set, while for real images, the database average is computed as the average over a large number of images. The absolute RGB values of the illuminant are not important, since we are interested only in its chromaticity and not in its intensity. While there are a number of chromaticity spaces that could be used for our experiment, we used only the one described in the next section, to be consistent with previous neural network experiments.

2.3. The neural network architecture

The neural network we used is a Perceptron[6] with two hidden layers (*H-1* and *H-2*), depicted in Fig. 1. The input layer (*In*) consists of a large number of binary inputs representing the chromaticity of the RGBs in the scene. Each image RGB from a scene is transformed into the rg-chromaticity space:

$$r=R/(R+G+B) \text{ and } g=G/(R+G+B) \tag{3}$$

This space is uniformly sampled with a step *S*, so that all chromaticities within the same sampling square of size *S* are taken as equivalent. Each sampling square maps to a distinct network input neuron. This chromaticity representation disregards any spatial information in the original images and takes into consideration only the chromaticities present in the scene.

The input neurons are set either to 0 indicating that an RGB of chromaticity rg is not present in the scene, or 1 indicating that it is present. Fig. 2 shows the representation of an image taken under two different illuminants (a tungsten bulb and a bluish fluorescent tube). The dots in the figure represent chromaticities that are present the images and they correspond to the active input nodes of the neural network for these two images. This is the only information that the neural network receives as input.

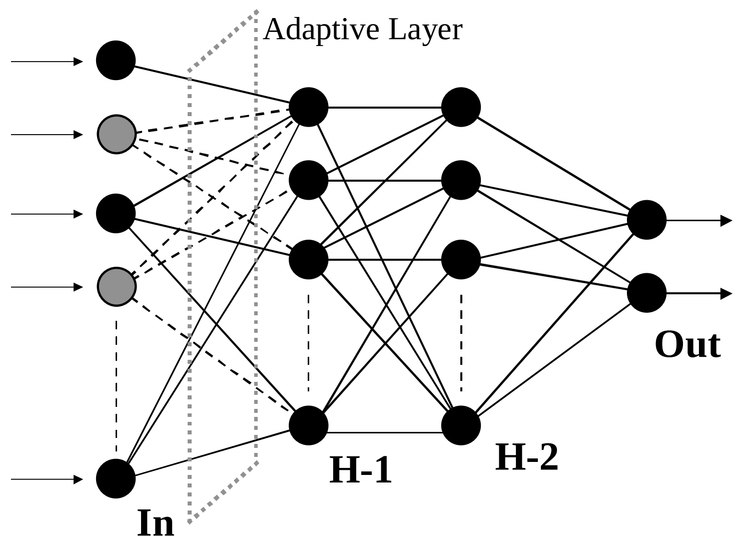

Figure 1. The Neural Network Architecture

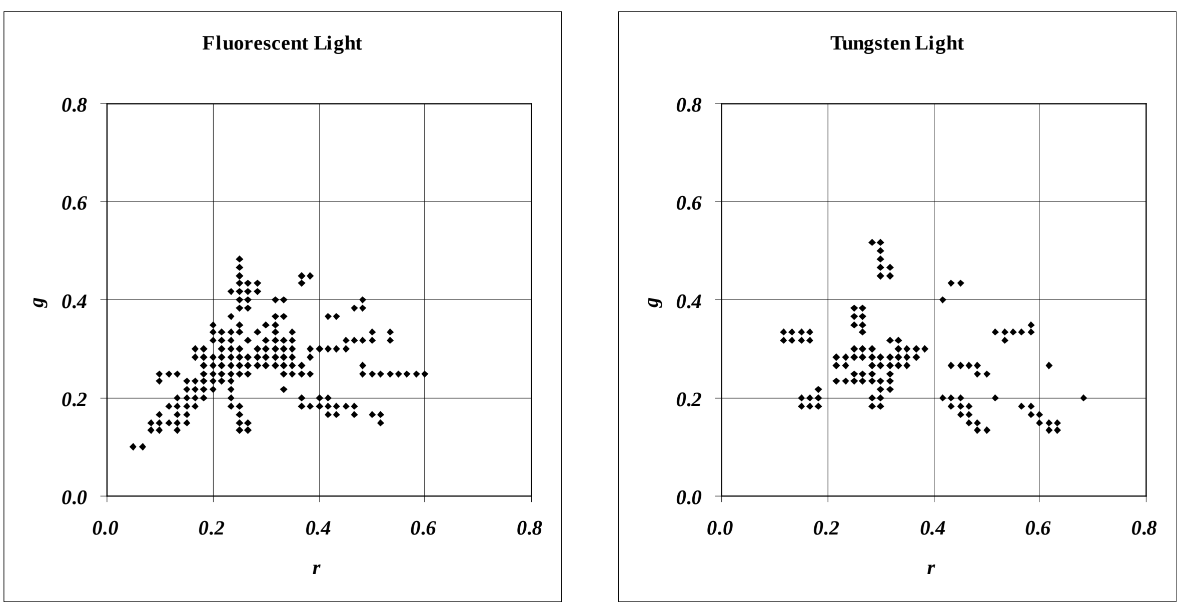

Figure 2. The chromaticity map of the same image taken under two different illuminants

We have made experiments with different sizes for the input layer and obtained comparable color constancy results in all cases. The networks used in this paper have 3600 input nodes. The first hidden layer (*H-1*) contains 400 neurons and the second layer (*H-2*), 40 neurons. The output layer (*Out*) consists of only two neurons, corresponding to the chromaticity values of the illuminant. Previous experiments[1,7] showed that the size of the hidden layers can vary without affecting the performance of the network. All neurons have a sigmoid activation function of the form:

$$y = \frac{1}{1 + e^{-(A-\theta)}} \tag{2}$$

where A is the activation (the weighted sum of the inputs) and θ is the threshold.

2.4. Neural network training

The neural network was trained using the backpropagation algorithm[6]—a gradient descent algorithm that minimizes the network's output errors. The training data is composed of a large set of input vectors (derived from synthesized or real images) and, for each input vector, a target output vector. This data set is presented a number of times (epochs) to the network during which time the network adjusts its weights and thresholds to minimize the output error (defined as the RMS error between the actual output values of the network and the target values).

Initial tests performed with a 'standard' neural network architecture described above showed that it took a large number of epochs to train the neural network. To overcome this problem, a series of improvements has been developed and implemented[7, 8]:

The gamut of the chromaticities encountered during training and testing is much smaller than the whole, theoretical, chromaticity space. Thus, we modified the neural network's architecture, such that the first hidden layer (*H-1*) receives input only from the active nodes (the input nodes that were activated at least once during training). The inactive nodes (those nodes that were not activated at any time) are purged from the neural network, together with their links to the first hidden layer. The network's architecture (see Fig. 1) is actually modified only during the first training epoch. The links from the first hidden layer are redirected only towards the neurons in the input layer that are active, i.e. those that correspond to existing chromaticities while links to inactive nodes are eliminated. Due to the fact that the sizes of the layers are so different, we used different learning rates for each layer, proportional to the fan-in of the neurons in that layer[9]. These optimizations shortened the training time by a factor of more than 10, to about 5-6 epochs.

The error function that we used for training and testing the network is the Euclidean distance in the rg-chromaticity space between the target and the estimated illuminant.

3. EXPERIMENTS DONE ON SYNTHESIZED IMAGES

3.1. Generating the synthetic images

Experimenting with synthetic images has the advantage that the whole environment is carefully controlled. Moreover, we can generate an arbitrarily large number of images, such that we obtain stable results when testing the algorithms.
For synthesized images, the user can set the number of the patches comprising a scene. A patch is generated for each chromaticity present in the image.

The RGB color of a patch is computed from its randomly selected surface reflectance S^j, the spectral distribution of the illuminant E^k (selected at random, but the same for all patches in a scene) and by the spectral sensitivities of camera sensors ρ according to:

$$R = \sum_i E_i^k \cdot S_i^j \cdot \rho_i^R,\ \ G = \sum_i E_i^k \cdot S_i^j \cdot \rho_i^G \text{ and } B = \sum_i E_i^k \cdot S_i^j \cdot \rho_i^B \tag{3}$$

The index i is over the wavelength domain, corresponding to wavelengths in the range of 380nm to 780nm. The sensor sensitivities are those of a SONY DXC-930 CCD video camera, with gamma correction turned off and calibrated for tungsten light.

The database of surface reflectances contains 260 measured reflectances. The database of illuminant power spectra is composed of 100 illuminants. All illuminants were either measured directly or generated as a linear combination of two measured illuminants, in order to provide a more uniform chromatic distribution. The color of the illuminants ranges from fluorescent lights with blue filters to reddish incandescent lights, covering a wide range of naturally occurring illuminants. Fig. 3 shows the distribution in the rg-chromaticity space of the illuminants and surfaces. The chromaticity of the surfaces is shown as viewed under a perfect white illuminant.

The data used for training the neural network was generated as follows. First we generated 10000 synthetic images (100 images per illuminant) composed of 35 different surfaces each, and in a later experiment, we used scenes with 50 surfaces instead of 35. Then instead of providing the neural network with the exact chromaticities of the illuminants used for generating these scenes, we used the illuminant estimates given by the GW algorithm. It must be noted that since the GW algorithm is calibrated for the surface database that we used, it yields very accurate estimates of the illuminant if all (or most) of the surfaces from the database are present in a scene.

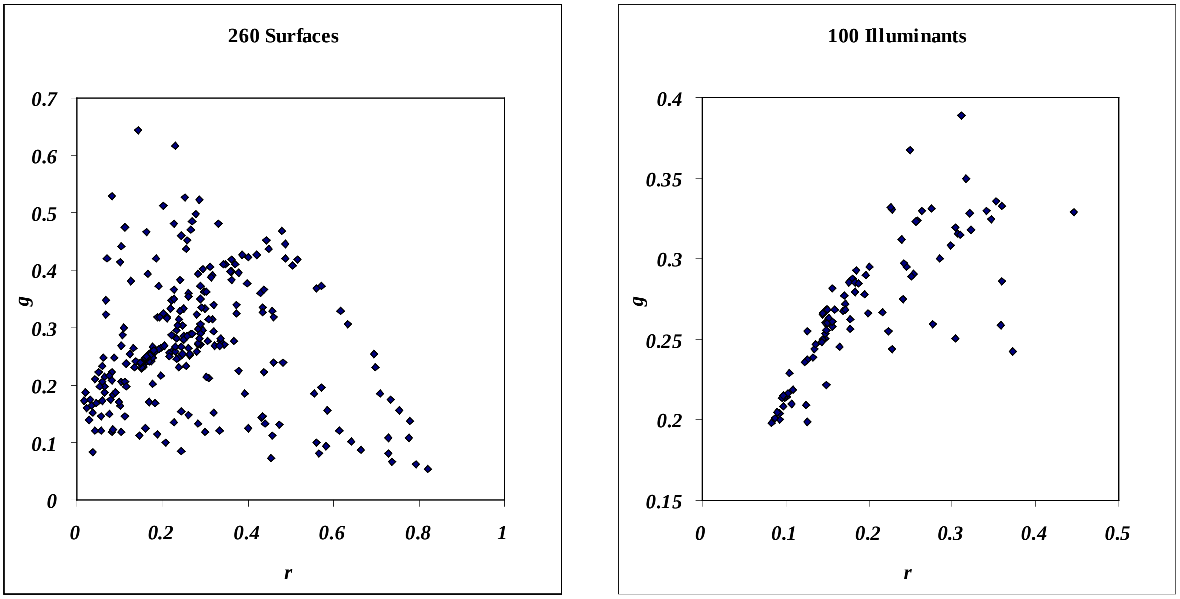

Figure 3. Database illuminants and surfaces

3.2. Training the neural network on synthetic images

The input data to the neural network is a training set composed of 10,000 images and the estimates of the corresponding illuminants provided by the GW algorithm. The network sub-samples these images into a larger set of 100,000 scenes. Each scene is generated by choosing a random number of surfaces from one of the input images. The minimum number of surfaces per scene was set to 10, while the maximum was given by the number of surfaces in the input images. Thus, if the number of surfaces in the synthetic images is equal to 35, the sub-sampled scenes have from 10 to 35 surfaces. In the case of a scene composed of 35 surfaces, the whole image is passed to the network. In a second experiment, where we generated synthetic images composed of 50 surfaces per scene, the network was trained on scenes composed of 10 to 50 surfaces per scene.

The illuminant of each sub-sampled image is inherited from the synthetic image from which it was generated. Thus the grayworld algorithm bases its estimate on the full image, while only the sub-sampled image is passed to the network. As a result, the grayworld estimate is more accurate and more stable than it would be if it were computed on only the sub-sampled data. The networks are trained for ten epochs on a set of scenes for which the illuminant is not known exactly. Even so, the average error rates drop fast to around 0.015, which is a satisfactory value.

In order to compare the bootstrapping algorithm to previous experiments[1, 7, 8], where the network was trained with exact illuminant values, we also generated a separate training set of 100,000 scenes, composed of 10 to 50 surfaces each, for which the exact illuminant values were provided to the network. In all other respects, the network was trained in the same way as before.

3.3. Results of experiments

All networks were tested on the same data. The test set is composed of 50,000 scenes generated from the databases described above. Each scene contains from 3 to 60 randomly selected surfaces. We compare the estimates of the neural networks (two trained using inexact illuminant estimates and one trained using exact values) against two other algorithms: the gray world algorithm (GW), described above, and the white patch algorithm (WP). The WP algorithm is a version of the retinex algorithm[10] that independently scales each channel of the image (R,G,B) by the maximum pixel value found in each channel. This is equivalent to estimating the color of the illuminant as being the color given by the maximum pixel value on the R,G and B channels. Since in synthesized scenes there are no ‘clipped’ pixels (i.e. pixels for which the sensor response on a channel is saturated), the WP algorithm performs much better then when tested on real world images where clipping usually occurs. The results are shown in Fig. 4. The error is computed as the Euclidean distance in the rg-chromaticity space between the actual illuminant and the estimate given by an algorithm. The errors of the neural networks and of the two other algorithms (GW and WP) are plotted against the number of patches in the scene.

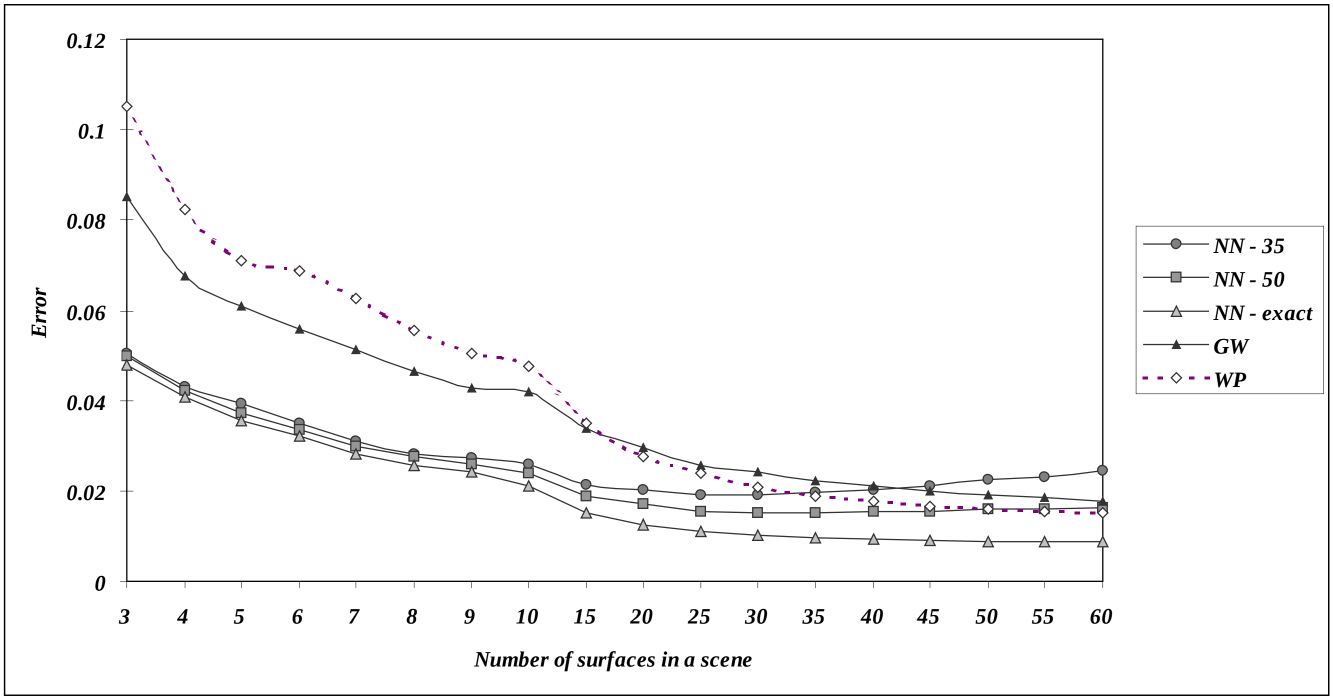

Figure 4. Results of the estimates of different color constancy algorithms on synthetic data

Even when trained with inexact illuminant chromaticities in the training set, the neural network is able to make quite good estimates of the illumination's chromaticity. Even more interesting, it exhibits a 'bootstrapping' capability, yielding better results than the grayworld algorithm that trained it, especially on images with few colors. For 35 or fewer surfaces in a scene, all neural networks yield better results than the GW and WP algorithms.

The neural network that was trained on exact data (NN-exact shown in Fig. 4) performs consistently better than GW and WP even on larger numbers of surfaces per scene. The neural network (NN-35) that was trained on scenes composed of maximum 35 surfaces and on illuminant estimates provided by GW is surpassed by the GW and WP algorithms for scenes with more than 35 surfaces. This happens for two reasons. One, GW and WP converge to almost zero estimation error as the number of surfaces in the scene approaches the number of surfaces in the database. Two, the neural network performs worse on scenes containing more surfaces than it ever encountered during training (35 in this case).

Similar results, although not as distinct, occur in the case of the neural network (NN-50) trained on scenes with a maximum 50 surfaces and on illuminant estimates provided by GW. This network is also surpassed by both GW and WP algorithms for scenes with more than 50 surfaces.

4. EXPERIMENTS DONE ON REAL IMAGES

4.1. Using real images for training and testing

Working with real images imposes some constraints on the whole experimental setup. First, the number of real images available is limited. Second, the number of illuminants is also limited. However, if both training and testing are done on data derived from real images, the camera sensitivity functions need not to be known. Moreover, artifacts that can not be avoided in real images (e.g. noise and flare) are implicitly modeled into the training set and thus the neural network can yield good results when tested on images that contain the same type of artifacts.

For the experiments done on real images we used images of natural scenes taken with a Kodak DCS460 digital camera. The original resolution of 2,000-by-3,000 pixels was reduced to around 1,000-by-600 for all images. We divided the images into two sets, one to be used for training and the other for testing. The images were linearized to compensate for the built-in gamma correction, but otherwise we did not make any other assumptions regarding the data. To remove part of the noise and to reduce the resolution, the images were also resampled. The chromaticities of the illuminants were measured by taking images of a standard white patch when taking images of the scenes. This white patch is not present in the main images, however, since it would otherwise bias the WP results, which partially rely on the presence of a white patch in the image. The database average used to compensate the GW algorithm was computed by averaging all surface RGBs in the images. Although it does not provide a perfect compensation, as it does with the synthetic data where the surface distribution is known in advance, it does improve the GW illumination estimates. The results of experiments done on real images show the positive effect of the database compensation.

4.2. Training the neural networks on data derived from real images

We used 47 images for training. Each image was sub-sampled into a number of scenes containing from 10 to 300 randomly selected pixels from the original images. The illuminant corresponding to these scenes was inherited from the estimation provided by the GW algorithm which computed the illuminant based on the entire image. A second neural network was trained on exact illuminant feedback. A total of 47,000 scenes were generated (1,000 scenes from each image) for the training set. Both networks were trained separately for 10 epochs.

4.3. Testing on real images

In a first experiment on real images, we tested both neural networks on 39 images. We compared the ‘bootstrapped’ neural network (i.e. trained on illuminant estimates provided by the GW algorithm) to a neural network trained on exact feedback. Comparisons are also made to the WP and GW algorithms and the ‘Illumination Chromaticity Variation’ (ICV). ICV is not a color constancy algorithm *per se*, rather it is simply a measure of the average shift in the rg-chromaticity space between a chosen canonical illuminant and the correct illuminants. This can be considered as a ‘worse case’ estimation, where we simply pick some illuminant and take it as the illuminant estimate for all input images. In our experiments, the canonical illuminant was selected to be the one for which the CCD camera was color balanced; in other words, the illuminant for which the image of a standard white patch records identical values on all three color channels.

The results are shown in Table 1. The mean error represents the average estimation error over all images. The standard deviation is also shown. Both neural networks perform much better than the other color constancy algorithms. The GW algorithm with database compensation has a small average error, too. However, in the general case, where the statistics of the surfaces are not known *a priori* (see ‘GW without database compensation’ in Table 1), the results of the GW algorithm are worse, comparable to those of the WP algorithm.

Table 1. Results of color constancy algorithms on real images

Color Constancy Algorithm	Mean Error	Std. Dev.
Illumination Chromaticity Variation (ICV):	0.1239	0.0632
Grey World (GW without database compensation):	0.0862	0.0440
Grey World (GW with database compensation):	0.0471	0.0340
White Patch (WP):	0.0847	0.0483
Neural Network trained on GW illuminant estimates:	0.0389	0.0179
Neural Network trained on exact illuminant data	0.0222	0.0293

4.4. Testing on data derived from real images

A second experiment was done on a large number of scenes, sub-sampled from the 39 test images. A total of 78,000 scenes were generated (2,000 from each image), each scene containing 5 to 80 distinct surfaces.

The estimate of the algorithms was compared against the illuminant corresponding to these scenes. The illuminants were inherited from the actual illuminants of the test images. The results of the test are shown in Fig. 5. The ‘bootstrapped’ neural network (NN-80) has better results than the GW algorithm; however, the network trained on exact data (NN-exact) has the best results.

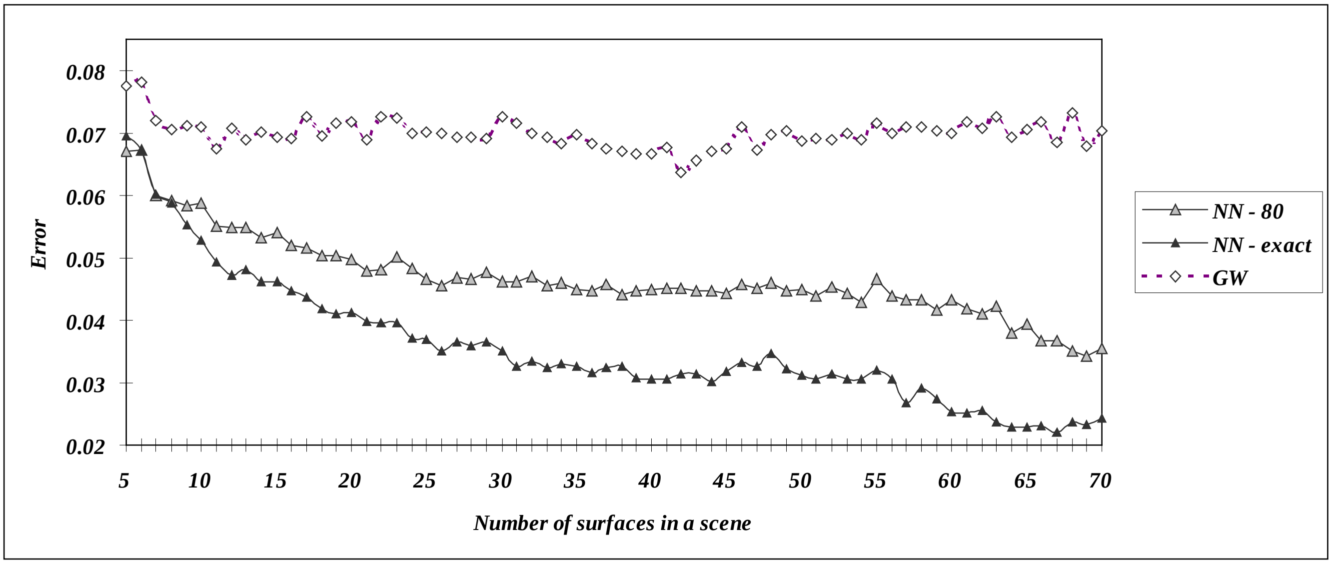

Figure 5. Results of the estimates of different color constancy algorithms on data derived from real images

5. CONCLUSION

A neural network for color constancy can be trained to estimate the chromaticity of the incident scene illumination without having exact knowledge of the illumination chromaticity in the training set. The network 'learns' to make a better estimate than the simple grayworld algorithm used in initially training it. This substantially simplifies the effort required to obtain or synthesize the training set. This approach works even if the camera sensors are unknown, thus providing an easy way for color correcting images taken with an uncalibrated camera.

6. REFERENCES

1. B. Funt, V. Cardei, and K. Barnard, "Learning Color Constancy," *Proc IS&T/SID Fourth Color Imaging Conf.*, pp. 58-60, Scottsdale, Arizona, November 1996.
2. G. Buchsbaum, "A Spatial Processor Model for Object Colour Perception," *J. Franklin Institute*, 310 (1), pp. 1-26, 1980
3. M. Swain and D. Ballard, "Color Indexing," *Int. J. of Computer Vision*, 7:1, pp. 11-32, 1991.
4. D.H. Brainard and W.T. Freeman, "Bayesian Color Constancy," *J. Opt. Soc. Am. A*, 14(7), 1393-1411, 1997.
5. G. Finlayson, M. Drew, and B. Funt, "Color Constancy: Generalized Diagonal Transforms Suffice," *J. Opt. Soc. Am. A*, 11(11), pp. 3011-3020, 1994.
6. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume I: Foundations*, D.E. Rumelhart, J.L. McClelland and the PDP Research Group, eds., pp. 318-362, MIT Press, Cambridge, MA, 1986.
7. B. Funt, V. Cardei, K. Barnard, "Neural Network Color Constancy and Specular Reflecting Surfaces," *AIC Color '97*, Kyoto, Japan, May 25-30, 1997.
8. V. Cardei, B. Funt, and K. Barnard, "Adaptive Illuminant Estimation Using Neural Networks," *Proc. of the 8th Int. Conf. on Artificial Neural Networks*, pp. 749-754, Skövde, Sweden, 1998.
9. D. Plaut, S. Nowlan, and G. Hinton, "Experiments on Learning by Back Propagation," Technical report, CMU-CS-86-126, Carnegie-Mellon University, Pittsburgh, USA, 1986.
10. Land E.H. The Retinex Theory of Color Vision. *Scientific American*, pp 108-129, 1977.