A comparative study of the characterisation of colour cameras by means of neural networks and polynomial transforms

Vien Cheung,[a] Stephen Westland,[a,†,*] David Connah[a] and Caterina Ripamonti[b]

[a]*Colour and Imaging Institute, Derby University, Kingsway House East, Derby DE22 3HL, UK*

[b]*Department of Psychology, University of Pennsylvania, 3815 Walnut Street, Philadelphia, PA 19104-6196, USA*

Received: 6 October 2003; Accepted: 5 November 2003

Coloration Technology

Society of Dyers and Colourists

The proliferation of low-cost colour imaging devices in the consumer market has led to a greater need to transfer images from one medium or device to another without loss of colour fidelity. A common solution is to characterise each device in terms of its CIE tristimulus values. In this paper two general techniques, artificial neural networks and polynomial transforms, are compared for their usefulness in characterising colour cameras. The neural and polynomial techniques are shown to give approximately similar performance once the parameters of the models are optimised. Since neural networks can be difficult and time-consuming to train, it is concluded that polynomial transforms offer the better alternative for camera characterisation.

Introduction

The proliferation of low-cost colour imaging devices in the consumer market has led to an increased need to be able to transfer images from one medium or device to another without loss of colour fidelity. A common solution is to characterise each device in terms of CIE tristimulus values. Supposing, for example, one wished to capture an image using a digital camera and then display the image on a computer monitor. It would be possible, with appropriate characterisation procedures, to convert the camera *RGB* values to CIE *XYZ* values and in turn convert these *XYZ* values to computer monitor *RGB* values. This paper presents experimental data derived from a camera system, to compare directly two mathematical techniques which can be used for characterisation. These are polynomial transforms and artificial neural networks.

Typical digital colour cameras capture device- and illuminant-dependent images. In other words, the *RGB* values which a camera measures are specific to that camera and the illuminating conditions under which it is used. The camera *RGB* values are illuminant-dependent, because they describe the colour of the image only when viewed under the light source that was used during capture. This device dependency means that in order to display the captured image on a display monitor, for example, it is necessary to convert the camera *RGB* values to monitor *RGB* values. Characterisation is the relationship between the device co-ordinates, usually *RGB* or *CMYK* and a device-independent colour space such as CIE *XYZ* [1,2]. Characterisation of devices into a standard colour space that is independent of the device reduces the number of transformations which may need to be performed.

Before a device can be characterised it must be calibrated. This is defined as the setting of the imaging device in an identified state to ensure that the device can produce consistent results. Green argues that there are three main methods for achieving characterisation mappings: physical models, look-up tables and numerical methods [3]. Most practical applications of characterisation for camera systems involve either look-up tables or numerical methods.

Look-up tables

In the use of look-up tables, a large number of examples of camera *RGB* values and corresponding CIE *XYZ* values are obtained and used to define the mapping. Interpolation is inevitably required to implement the mapping for samples not present in the look-up table. In numerical methods a series of coefficients are determined, usually based upon a set of measured samples, with only minimal prior assumptions about the physical behaviour of the device. Look-up tables are frequently encountered in the characterisation of colour printers.

Numerical methods

Polynomial transforms are widely used for camera and monitor characterisation. A key property of any transform is whether it can easily be inverted. The advantage of a linear transform is that it is a simple matter to invert a transform that computes *XYZ* values from *RGB* values to one that computes *RGB* values from *XYZ* values, whereas

† Prof. Westland has now taken up a new position as Professor of Colour Science & Technology, School of Design, University of Leeds, Leeds LS5 9JT, UK; email: s.westland@leeds.ac.uk

polynomials and many empirical models are not easily inverted [4]. If inversion is not possible then iteration may be required to perform the inverse mapping [5].

For many cameras the process of characterisation can be considered to consist of two stages. The first stage performs a linearisation, termed gamma correction for certain devices. The second stage transforms the linearised values into CIE *XYZ* tristimulus values. Practical device characterisation will almost certainly require that the spatial and temporal properties of the device are accounted for.

It is important to note that effective characterisation is usually only practicable if the camera does not perform automatic white-point balancing. This is a process in which the brightest patch in any captured image scene is denoted as white and the *RGB* values of each pixel are transformed accordingly [6].

Although the physical response of a charge-coupled device (CCD) material is linearly related to the intensity of the light falling on it, the *RGB* outputs of a digital camera are often not linearly related to the *XYZ* tristimulus values of the surfaces in the scene. The raw channel responses are invariably processed by on-board software in an attempt to generate *RGB* responses more closely related to the CIE tristimulus values. Furthermore, many manufacturers impose a nonlinearity during this 'matrix-mixing' stage to match approximately the inverse of the nonlinearity of display systems, or as part of the provision of high signal-to-noise ratios. It is therefore common to consider a correction for nonlinearity as the first stage of the camera characterisation process. It would be possible, for example, to consider a relation of the form shown in Eqn 1:

$$C_i = (R_i)^{p(i)} \tag{1}$$

where R_i is the raw response of the camera channel i, $p(i)$ is an exponent for the channel, and C_i is the output camera response for that channel. A set of grey scale samples is often used to determine empirically the exponent p. Thus the output camera responses are determined for a range of grey samples under an identified constant light source. The *XYZ* values can then easily be computed for the grey samples from their known spectral reflectance values, or measured directly using a colorimeter. Linearisation may be achieved by finding a value of the exponent p such that a linear relationship is shown between C_i and the CIE Y value for the set of grey samples. The grey samples of the Munsell ColorChecker provide a convenient grey scale for this purpose.

The second stage of camera characterisation requires that a mapping is found between the linearised camera *RGB* values and the CIE tristimulus values. An example of such mapping is shown as Eqn 2:

$$\begin{aligned} X &= a_{11}R + a_{12}G + a_{13}B \\ Y &= a_{21}R + a_{22}G + a_{23}B \\ Z &= a_{31}R + a_{32}G + a_{33}B \end{aligned} \tag{2}$$

where the coefficients a_{11}–a_{33} must be determined. If there are n examples of camera responses and their corresponding tristimulus values, then in terms of matrix algebra it is necessary to find the 3×3 standard matrix **A** (Eqn 3):

$$\mathbf{T} = \mathbf{CA} \tag{3}$$

where **T** is the $n \times 3$ matrix of tristimulus values, **C** is the $n \times 3$ matrix of camera values and **A** is a 3×3 matrix containing the coefficients a_{11}–a_{33}. The matrix **A** may be found directly (Eqn 4):

$$\mathbf{A} = \mathbf{C}^{-1}\mathbf{T} \tag{4}$$

where $\mathbf{C^{-1}}$ denotes the inverse of the matrix **C**. When **C** is a square matrix the system of equations has a single and unique solution. For over-determined systems, however, in which there are more equations than variables, a solution of Eqn 3 may be obtained by computing the psuedo-inverse of **C**, denoted $\mathbf{C^{+}}$ (Eqn 5).

$$\mathbf{A} = \mathbf{C}^{+}\mathbf{T} \tag{5}$$

Methods for computing the inverse and pseudo-inverse are widely discussed in the literature [5,7,8] and can easily be performed using a programming language such as MATLAB, which provides the function *pinv* to allow the psuedo-inverse to be computed. For trichromatic cameras the system is almost always overdetermined, since more than three coloured patches are usually used to define the mapping. The use of pseudo-inverse methods effects a least-squares fit solution to Eqn 3.

Once the entries of the system matrix **A** have been determined it is easy to compute the tristimulus values for any set of camera responses using Eqn 3. It is also a simple matter to invert Eqn 3 to allow the camera responses to be predicted from the tristimulus values.

The linear transform shown in Eqn 3 is a special case of the set of polynomial transforms. Practically, higher order (nonlinear) transforms are often used [9], for example that shown in Eqn 6.

$$\begin{aligned} X &= a_{11}R + a_{12}G + a_{13}B + a_{14}R^2 + a_{15}G^2 + a_{16}B^2 \\ Y &= a_{21}R + a_{22}G + a_{23}B + a_{24}R^2 + a_{25}G^2 + a_{26}B^2 \\ Z &= a_{31}R + a_{32}G + a_{33}B + a_{34}R^2 + a_{35}G^2 + a_{36}B^2 \end{aligned} \tag{6}$$

This can also be expressed in matrix form (Eqn 7):

$$\mathbf{T} = \mathbf{DA} \tag{7}$$

where **T** is the $n \times 3$ matrix of tristimulus values and **D** is the $n \times 6$ matrix of augmented monitor values, where each row contains six terms: $[R\ G\ B\ R^2\ G^2\ B^2]$. In order to define this transform it is necessary to find the 6×3 standard matrix **A**. The methods for solving Eqn 7 are exactly the same as those for solving Eqn 3.

Artificial neural networks

An alternative method to achieve a mapping between camera responses and tristimulus values is to use artificial neural networks (ANNs). There are many different types

of ANN, and extensive literature is available to explain the principles and algorithms of neural computing [10–13]. It is sufficient here to describe briefly one class of ANN known as a multilayer perceptron (MLP).

An MLP consists of layers of processing units known as neurones. Each unit receives input and performs some function upon this input to produce an output. The function between input and output for any unit is known as the activation or transfer function and is normally nonlinear. A typical nonlinear transfer function is the sigmoid function (Eqn 8):

$$f(x) = 1/(1+e^{-x}) \tag{8}$$

where the input to the unit is x and the output is $f(x)$. Linear transfer functions are sometimes used for the units in the output layer. The input for each unit is the weighted sum of the outputs from all of the units in the previous layer. The units in the first layer, known as the input layer, receive their input from an input vector. Those in the final layer, the output layer, generate an output vector. Each unit in the hidden and output layers also receives weighted input from a bias unit, whose output is fixed at unity. This is illustrated in Figure 1, in which the circles show the processing units, arranged in layers. Each unit in the hidden and output layers computes a weighted input of the outputs from those in the previous layer and the bias unit. It then computes an output value based on the weighted input and the activation function for that unit. Each line in Figure 1 represents a weighted connection, and the network is trained by finding the values of the weights for each connection.

The network as a whole can be regarded as a universal function approximator that attempts to find a mapping between input and output vectors. The network is trained by finding the set of weights that produce the smallest difference between the actual and target output vectors for a set of samples known as the training set.

The number of units in the input and output layers are determined by the nature of the problem being solved. If, for example, the network is being used to perform a mapping between one three-dimensional vector (*RGB*) and another (*XYZ*), there would be three units in each of the input and output layers, as shown in Figure 1. However, the number of hidden layers, and the number of units in each, must be determined empirically.

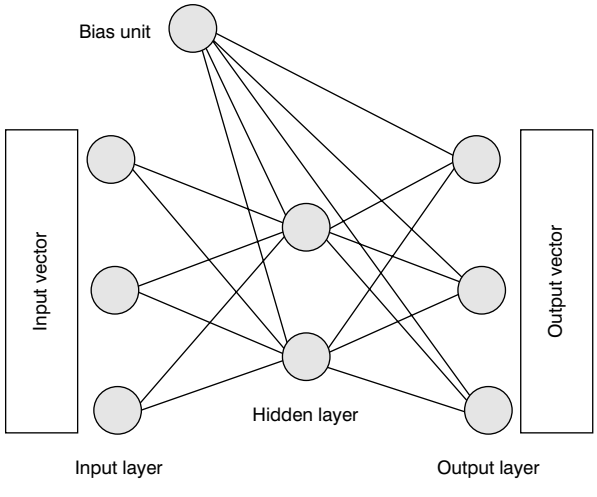

Figure 1 Schematic diagram showing how a neural network can be used to find a mapping between the *RGB* values (input vector) and the *XYZ* values (output vector)

For ANNs it is particularly important to distinguish between the training and the testing modes. However, the following discussion of memorisation and generalisation, and hence the need for separate training and test sets, applies equally to the use of polynomial transforms. During the training mode, examples of input–output pairs are presented to the network. The error between the target output and the actual output is computed using, for the first iteration, random values for the weights, these being modified to reduce the error. This process is repeated for each input–output pair in the training set and the presentation of the whole set in this way is known as a training epoch. Effective training may require thousands, or even hundreds of thousands, of epochs and typically the training procedure is very computationally intensive. However, at the end of the training period the values of the weights are fixed. During the testing mode, input vectors are presented to the network and output vectors are computed. The ability of the trained network to map the input training vector to the output training vector is known as the training error.

Generalisation is the ability of the network to operate using data which were not used during the training period, whereas memorisation is the ability of the network to predict the training set. A useful transform or mapping must have good generalisation properties, and a second data set, known as the testing data set, is therefore used to determine the testing error. A number of publications have described attempts to characterise imaging devices using ANNs [14,15].

Objective of the current study

The main purpose of the present work was to compare directly and quantitatively the use of nonlinear transforms and ANNs for the specific problem of colour camera characterisation. For this comparison to be valid it was necessary to use the same data and to explore fully the parameters used in each approach.

Experimental

An Agfa digital StudioCam camera, a three-chip CCD device with 8-bit resolution for each channel and 4500 × 3648 pixel spatial resolution, was used in this study. During the experiment the automatic white-balance setting was disabled.

Two imaging targets, 166 Macbeth ColorChecker DC (excluding the repeated grey-scale colours located around the boundary of the chart) and 50 Natural Color System (NCS) selected samples, were used for the characterisation. The spectral reflectance factors of the patches on the two charts were measured using an X-Rite 938 spectro-densitometer. The colour distributions of the selected ColorChecker DC and NCS samples are presented as a CIELAB a^*b^* diagram in Figure 2.

A Minolta CS1000 spectroradiometer was used for the measurement of the spectral power distribution of the

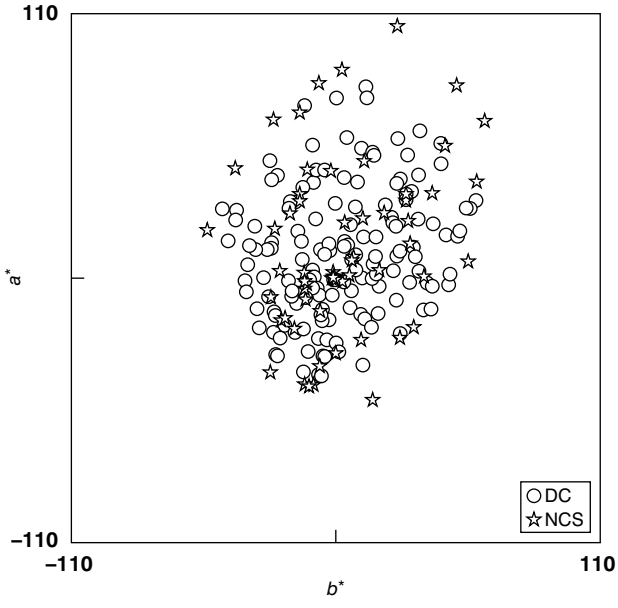

Figure 2 Colour distributions of 166 Macbeth ColorChecker DC and 50 NCS samples in CIELAB a^*b^* diagram

illuminating source. The lighting system consisted of two gas-filled tungsten lamps arranged approximately in a 0/45° illumination and viewing geometry.

Linearisation and spatial correction

The linearisation and spatial correction method was based on work by Sun and Fairchild [16]. The camera *RGB* responses were measured for a series of Munsell grey chips (N6/ to N9/ at 0.5 value intervals), an NCS uniform white paper, and the dark condition (with the camera lens cap in place). This allowed a gamma correction for the camera, so that the camera responses could be converted to values linearly related to the camera input. During the experiment the camera and lighting positions were fixed, and the *RGB* values of the Munsell grey chips were measured with each in the centre of the camera field of view. Each patch generated an image region of about 40×60 pixels, but the values of the central sub-region (11×11 pixels) were averaged to generate the mean *RGB* values for that patch. For each camera channel, the camera responses for each grey patch were plotted against the mean reflectance of each, and the relationship was fitted using a second-order polynomial. In this way a polynomial relationship was established for each channel, and all subsequent camera responses were linearised using these relationships before further processing.

Spatial correction was performed in order to minimise the effect of any lack of spatial uniformity in the intensity of the illumination or of the sensitivity of the camera CCD. For example, for the red channel, Eqn 9 was used to convert the linearised channel response R_i' to the spatially corrected value R_{Si} at each pixel position i. Thus:

$$R_{Si} = \frac{(R_W - R_B) \times (R_i' - R_{Bi})}{(R_{Wi} - R_{Bi})} \tag{9}$$

where R_W and R_B are the mean linearised channel values for the uniform white and black (dark) samples, respectively. R_{Wi} and R_{Bi} are the channel responses for the uniform white and black (dark) samples at each pixel location i, respectively. Similar equations were used to obtain the spatially corrected values for the green and blue channels.

Training/testing protocol

A selection of 166 patches from the Macbeth ColorChecker DC chart and 50 NCS chips were used as training and testing sets, respectively. These two characterisation stimuli were used for memorisation and generalisation tests. Smaller training sets were derived by randomly sub-sampling the 166 patches to generate training sets containing 120, 80 and 40 samples. The test set always consisted of the 50 NCS patches.

Linear and polynomial transforms

Polynomial functions were used to perform a mapping between a vector of camera responses **c** and a vector of tristimulus responses **t** (Eqn 10):

$$\mathbf{t} = \mathbf{A}\mathbf{c} \tag{10}$$

where **A** is the system matrix. For the linear transform, **A** is a 3×3 matrix and **c** is a 3×1 matrix. The values of the matrix **A** are easily determined using methods of linear algebra [8]. So, for example, the polynomial represented by Eqn 11 may be represented by Eqn 10 where **A** is a 3×8 matrix and **c** is a 8×1 matrix of augmented *RGB* values containing the terms [1 R G B R^2 G^2 B^2 RGB].

$$\begin{aligned} X &= a + bR + cG + dB + eR^2 + fG^2 + gB^2 + hRGB \\ Y &= i + jR + kG + lB + mR^2 + nG^2 + oB^2 + pRGB \\ Z &= q + rR + sG + tB + uR^2 + vG^2 + wB^2 + xRGB \end{aligned} \tag{11}$$

In this study a variety of linear and nonlinear transforms were evaluated and the augmented matrices **c** are listed for each in Table 1.

Artificial neural networks

Fully connected multilayer perceptron networks were used in this study to derive mappings between the camera responses and tristimulus values. The networks always contained three input units to receive the camera responses, three output units to output the tristimulus values, and a single hidden layer. The number of units N in the hidden layer was varied to be 3, 5, 10, 18, 27 or 40. The activation function of the units in the hidden and output layers was the sigmoid function. The sigmoid activation function can only output in the range [0,1] and therefore the output data (tristimulus values) were scaled to the range [0.1,0.9] before they were used for training. The outputs of the trained network were rescaled back to the original format before comparing them with the target *XYZ* values. The networks were trained using the Levenberg–Marquardt optimisation method for 100 epochs, since it was determined empirically that training for greater than this number of epochs did not improve the results. Each time the network was trained the weights were randomised to different starting values, and different

Table 1 Definition of polynomial transforms used

A	Augmented matrix
3×3	$[R\ G\ B]$
3×4^a	$[R\ G\ B\ 1]$
3×5	$[R\ G\ B\ RGB\ 1]$
3×10^b	$[R\ G\ B\ RG\ RB\ GB\ R^2\ G^2\ B^2\ 1]$
3×20^c	$[R\ G\ B\ RG\ RB\ GB\ R^2\ G^2\ B^2\ RGB\ R^2G\ G^2B\ B^2R\ R^2B\ G^2R\ B^2G\ R^3\ G^3\ B^3\ 1]$
3×35^d	$[R\ G\ B\ RG\ RB\ GB\ R^2\ G^2\ B^2\ RGB\ R^2G\ G^2B\ B^2R\ R^2B\ G^2R\ B^2G\ R^3\ G^3\ B^3\ R^3G\ R^3B\ G^3R\ G^3B\ B^3R\ B^3G\ R^2GB\ RG^2B\ RGB^2\ R^2G^2\ R^2B^2\ G^2B^2\ R^4\ G^4\ B^4\ 1]$

a First order
b Second order
c Third order
d Fourth order

results were usually obtained. Each network was therefore trained five times and the average performance calculated.

Evaluation of characterisation methods

CIE tristimulus values were computed for the patches using the 1964 CIE observer data and the spectral power of the light source used to illuminate the stimuli. Colour errors between measured and estimated tristimulus values were computed using the CIELAB colour difference formula. All computations were performed in the MATLAB programming environment.

Results and Discussion

Polynomial approach

Figure 3 shows the performance of the polynomial models for both training and testing sets, using the full 166 training set. In theory, as the complexity of the model increases it could be expected that the training error would decrease consistently. On the other hand the testing error should reach a minimum and subsequently increase as the model over-fits the training data. It was found that performance for the training set did in general decrease, whereas there was some evidence that the performance on the test set was approaching a minimum for the third-order polynomial transform.

The effect of reducing the number of training samples for polynomial models is illustrated in Figure 4 for the polynomial model with median and maximum CIELAB errors, for the matrix of size 3×20 (third order). Figure 4a shows that the memorisation error decreased as the size of the training set decreased, but as anticipated the reverse trend was seen for the generalisation performance. However, generalisation error was quite stable until the training set size fell to fewer than about 100 samples. In the opposite limit, as the number of training samples became large, the performance of the models was statistically indistinguishable. Figure 4b shows that the maximum test error for the polynomial model was generally stable except when the training sample size was very small.

Neural network approach

Figure 5 shows the performance of the neural network models for both training and testing sets using the full 166 training set. It is evident that, given a training set size of 166, a neural network with about 18 hidden units is optimum. Increasing the complexity of the network thereafter only leads to poorer generalisation performance as the network over-fits the training data.

The effect of reducing the number of training samples for neural networks is illustrated in Figure 6 for the neural network model with 18 hidden units. The results show that the memorisation error again decreased with the size of the training set, but the reverse trend was seen for the generalisation performance. The maximum test error for the neural network shown in Figure 6b was generally

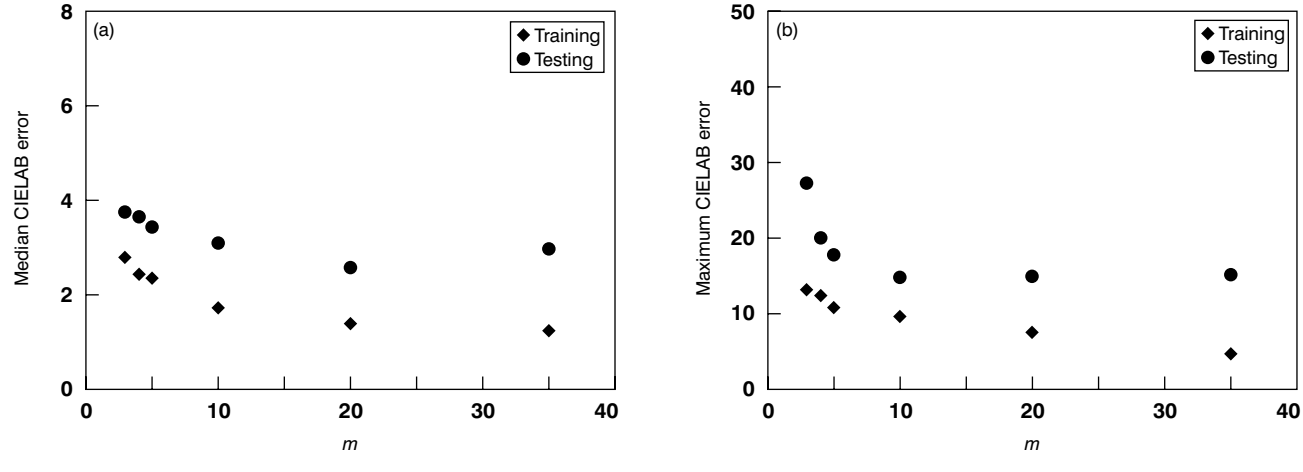

Figure 3 Effect of number of terms (m) in the polynomial model on training and testing performance: (a) median and (b) maximum colour difference

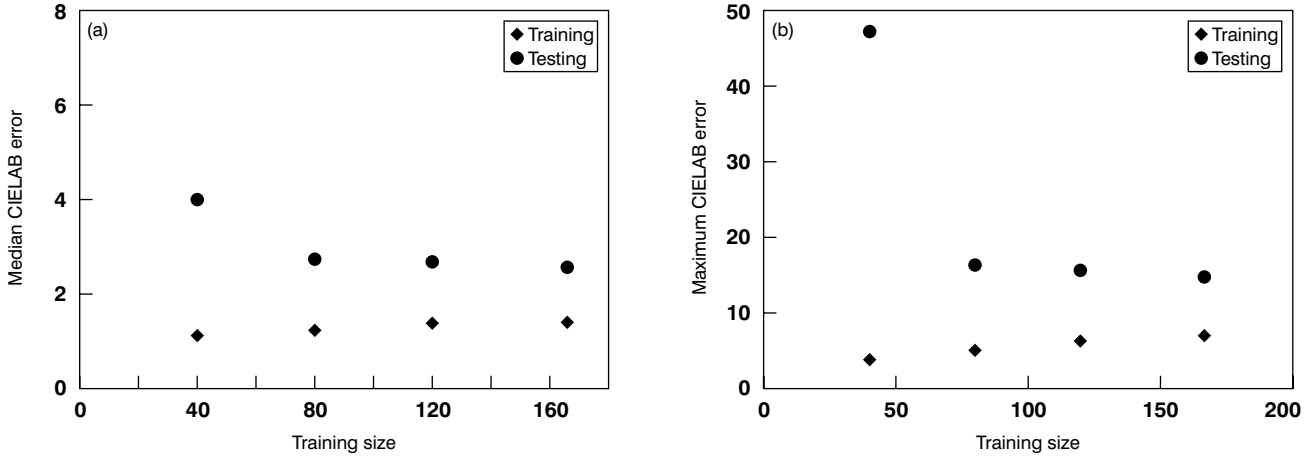

Figure 4 Effect of number of training samples on training and testing performance for the third-order polynomial model: (a) median and (b) maximum colour difference

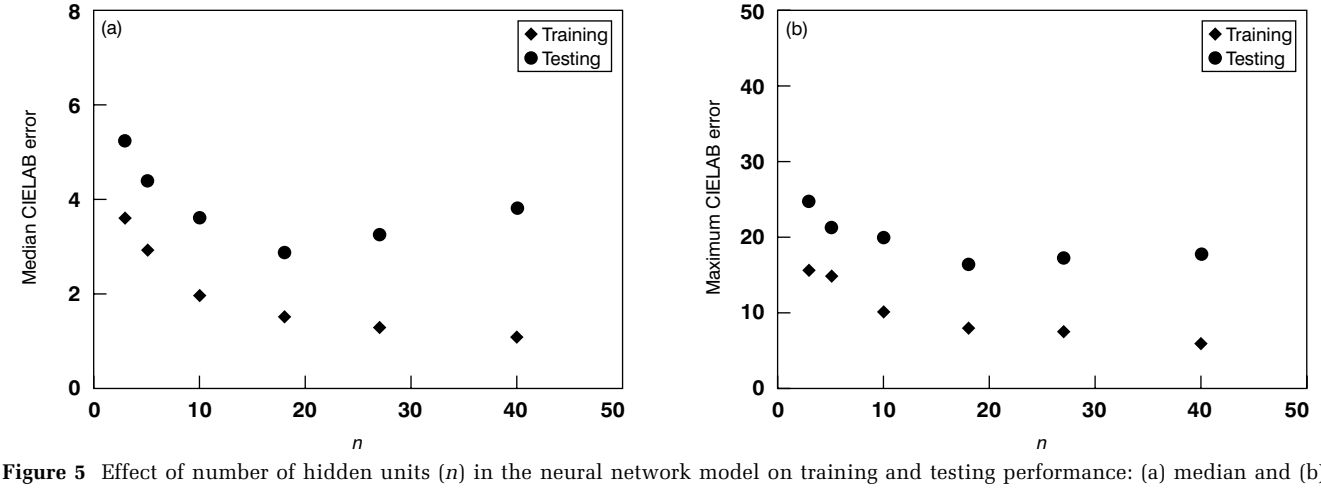

Figure 5 Effect of number of hidden units (n) in the neural network model on training and testing performance: (a) median and (b) maximum colour difference

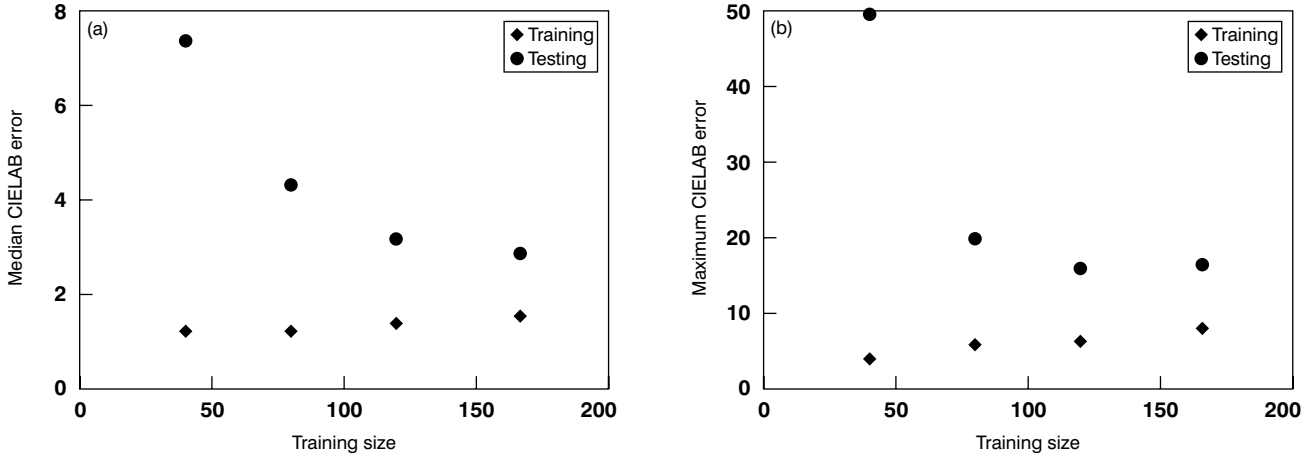

Figure 6 Effect of number of training samples on training and testing performance for the neural network with 18 hidden units: (a) median and (b) maximum colour difference

stable. When the number of training samples was very small the neural network maintained a reliable performance.

Comparison of the polynomial and neural network approaches

This study has shown that, when properly assessed, the abilities of camera characterisation models based on polynomials and neural networks are approximately the same. The median CIELAB colour differences are 2.57 and 2.89 for the generalisation performance of the best polynomial and neural network characterisation models, respectively. That these two performance figures are so close should not be surprising, since MLPs have often been described as being nonlinear curve fitters, or generalised polynomial models.

There seems to be little advantage in using a neural network model rather than a polynomial model for this type of problem. With polynomial models the user needs to ascertain the exact nature of the best polynomial and this

can only be achieved by rigorous experimentation, as was the case in the present study. However, with neural networks the user faces a similar problem since the number of hidden units in the network needs to be empirically determined.

The effect of the size of the training set on generalisation performance was similar for the two types of model, a large difference only becoming apparent for very small training sets. There was some evidence that the maximum colour difference error was greater for the neural network than for the polynomial. There are however other disadvantages to the neural approach. The training of the network can be very slow, sometimes taking more than an hour, whereas the polynomial model can be solved in fractions of a second. There is also evidence that the neural network models may not always converge to a global minimum, whereas for the polynomials implementation of training and testing is easy and reliable. It is concluded that polynomial transforms offer the better alternative for camera characterisation.

It should be noted that this study was concerned only with the colorimetric accuracy of the characterisation methods. It is possible that these methods introduce a spatial artefact, and this will be explored in further research.

Conclusions

The purpose of this study was to investigate whether there was any advantage to using artifical neural networks for camera characterisation compared with the more traditional technique of polynomial transforms. The study shows that the two techniques are capable of producing almost identical results when properly used. Given that neural networks can be difficult and time-consuming to train, we therefore recommend the use of polynomial transforms for camera characterisation. The study also investigated the number of training samples that are required for accurate characterisation. Further work may be needed to ascertain whether using more than 200 training examples would yield better results. However, in this study it was found that there was no large difference in the results obtained as the number of training examples was changed above a threshold value of about 100 training examples.

References

1. M D Fairchild, *Color Appearance Models* (London: Addison Wesley Longman, 1998).
2. A J Johnson, in *Colour Engineering*, Eds. P Green and L W MacDonald (New York: John Wiley and Sons, 2002) 165.
3. P Green, in *Colour Engineering*, Eds. P Green and L W MacDonald (New York: John Wiley and Sons, 2002) 127.
4. K Iino and R S Berns, *J. Imaging Sci. Technol.*, **42** (1998) 165.
5. J Y Hardeberg, *Acquisition and reproduction of colour images: colorimetric and multispectral approaches* (Florida: Universal Publishers, 2001).
6. J Holm, I Tastl, L Hanlon and P Hubel, in *Colour Engineering*, Eds. P Green and L W MacDonald (New York: John Wiley and Sons, 2002) 179.
7. H Anton, *Elementary linear algebra* (New York: John Wiley and Sons, 1994).
8. G J Borse, *Numerical methods with MATLAB: A resource for scientists and engineers* (London: PWS Publishing Company, 1997).
9. G Hong, M R Luo and P A Rhodes, *Color Res. Appl.*, **26** (2001) 76.
10. I Aleksander and H Morton, *An introduction to neural computing* (New York: Chapman and Hall, 1991).
11. S Haykin, *Neural Networks, a Comprehensive Foundation* (Basingstoke: Macmillan, 1994).
12. T Kohonen, *Self-organization and associative memory* (Berlin: Springer-Verlag, 1988).
13. D E Rumelhart and J L McClelland, *Parallel distributed processing* (Cambridge: MIT Press, 1986).
14. S Tominaga, in *Colour Imaging, Vision and Technology*, Eds. L W MacDonald and M R Luo (New York: John Wiley and Sons, 1999) 79.
15. H R Kang, *J. Electron. Imaging*, **1** (1992) 125.
16. Q Sun and M D Fairchild, *Proc. 9th Color Imaging Science Conf.*, Scottsdale, USA (2001) 73.