A Neural Network Approach to Color Reproduction in Color Printers

Shoji Tominaga
Osaka Electro-Communication University, Osaka, Japan

Abstract

This paper describes a method for colorimetric color reproduction on a dye sublimation printer by means of neural networks. A multilayer feed forward network is regarded as a nonlinear transformer for color coordinate transformation between the printer coordinates and the color stimulus values. The network is trained to learn a mapping to determine the required CMY (RGB) values of printer primaries for producing a given XYZ color stimulus. We adopt the Back-Propagation learning rule far the training. The mapping is then realized in a simple network architecture in which nonlinear units are linked in parallel and in layers. The measured data of many color patches are used for training the network and testing the mapping accuracy. The accuracy is evaluated on the CIE-L*a*b* color difference between the reproduced color from the network output and the original color. We determine an effective network method based on experiments under different conditions.

Introduction

With the advent of desk-top publishing systems, the color reproduction of images has become more widespread and more easily accessible than ever before.[1-3] Advances in color printer technology have been made not only for the lithographic offset printing, but also for providing various new printing devices such as thermal transfer dye sublimation and ink jet printers to produce single color prints in a desk-top computer system. The pigments in printing inks absorb certain wavelengths from the incident light and so constitute a subtractive system.

The subtractive color reproduction of print is much more complicated than the additive system of displays which is characterized almost completely in terms of only a few parameters.[4] It is difficult to predict from a knowledge of the individual ink amounts what color stimulus will be generated on paper. In the case of color offset printing, there are several methods for estimating the values of three primary inks of cyan (C), magenta (M), and yellow (Y) required to produce a given XYZ color stimulus. These are (1) a classical analysis using the Neugebauer equation, (2) use of a look-up table and interpolation, and (3) a neural network method. It has been pointed out that the first analytical method often has an inevitable discrepancy between the estimates and the reproduced results. The second method has the difficulty in computational speed of the three-dimensional interpolation. The third method is novel. Use of a three layered network is proposed.[5] However there is still no useful method for accurate color reproduction on recent easy printers such as dye sublimation and ink jet printers.

The present paper proposes a method for colorimetric color reproduction on a dye sublimation printer by means of neural networks. A neural network is regarded as a nonlinear transformer for color coordinate transformation between the printer coordinates and the corresponding color stimulus values. The network can learn adaptively an inverse mapping that determines the required CMY values of printer primaries to reproduce a given XYZ color stimulus on the printer. We adopt the Back-Propagation learning rule for training the network. The mapping is then realized in a simple network architecture in which nonlinear units are linked in parallel and in layers. We determine an effective network method for the mapping from the CIE-XYZ color space to the printer CMY color space, based on experiments under different conditions.

Color Printer

The dye sublimation printer used in this study is a Shinko Electric model CHC-S443 printer. It prints three (CMY) or four (CMYK) colors at the resolution of 300 dots per inch. Each color printer primary is expressed in 256 levels (8-bits). Color images are usually expressed in additive mixture of R, G, and B in a computer. In the present system the two sets of signals, CMY and RGB can be converted each other with a simple relation for opponent color as

$$R=255-C,\ G=255-M,\ B=255-Y. \quad (1)$$

First, we investigated the stability of the printed colors. To do this, a uniform color was produced on the full size of a paper with the same color value, and the color variation was measured at different parts on the paper. This color variation was evaluated by the standard deviation of the CIE L*a*b* color difference. The standard deviations for three colors R=255, G=255, and B-255 are, respectively, 0.9, l.0 and 1.3 ΔE^*_{ab} units.

Next, the printer gamut were measured from color charts. The color charts consist of many color patches which are arranged in even steps in C(R), M(G), and Y(B) values. These object colors were measured by a colorimeter under D65. Figure 1 shows the printer gamut on the CIE chromaticity diagram. The filled diamond at the center indicates the white point of D65. Furthermore Figure 2 shows the relationship between the luminance of the reproduced color on a paper and the input digital value for each primary. We have no linear relationship.

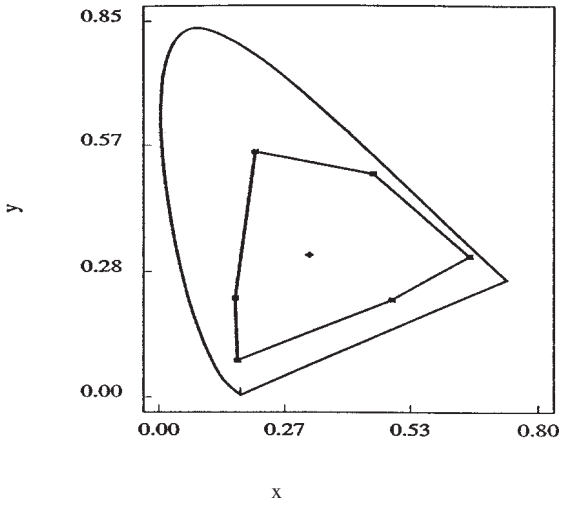

Figure 1. Gamut of a dye sublimation printer

Color Conversion

Network Structure

A color printer accepts the CMY digital value as the input, and generates the corresponding color stimulus as the output. We need to solve the inverse mapping, that of finding the CMY values to produce a desired color stimulus. This mapping is a conversion from the CIE-XYZ color space onto the printer CMY color space. This is also equivalent to the conversion from XYZ onto RGB through the relation (1). For the convenience of color image representation, we consider the mapping from XYZ to RGB by means of neural networks.

Figure 3 depicts the structure of a multilayer feedforward network used in this study. The network consists of an input layer, two hidden layers, and an output layer. The circles at nodes indicate processing units. Every unit sends its output to higher layers than its own, and receive its input from lower layers than its own.

Let o_i be the output of the unit i in the prior layer, w_{ji} be the weighting coefficient of connection from unit i to the target unit j, and b_j be the bias term of the unit. The input to unit j is then described as the sum of the weighted outputs from the prior layer

$$net_j = \sum_i w_{ji} o_i + b_j \qquad (2)$$

The nonlinear output of unit j is

$$o_j = f(net_j) , \qquad (3)$$

where f is an activation function. In this study we use the sigmoidal activation function

$$f(net_{j)} = 1/\{1 + \exp(-4\alpha\ net_j)\}. \qquad (4)$$

This function takes any real number in the interval [0, 1], and the positive constant a represents the slope of f at net=0. It is noted that this nonlinear operation is not applied to the units in the output layer. The final output is the weighted sum of $o_j = net_j$.

The number of two hidden layers is determined empirically. The most effective number of units in the hidden layers will be discussed later. The knowledge of mapping from XYZ to RGB color space is stored in the network in the form of weights in all the connecting links.

For simplicity we normalize all the input/output signals of the network. Concerning the input signals, the tristimulus values are positive, and usually larger than one. The minimum tristimulus values are fixed at 0.0. So finding the maxima in the data set, we calculate the tristimulus values normalized into the interval [0, 1] as

$$X_n = X/X_{max} \qquad Y_n = Y/Y_{max} \qquad Z_n = Z/Z_{max} \qquad (5)$$

On the other hand, the RGB values, which lie in the range 0 to 255, are scaled with $R_n = R/(1.1*255)$, $G_n = G/(1.1*255)$, and $B_n = B/(1.1*255)$ to fall in [0, 1].

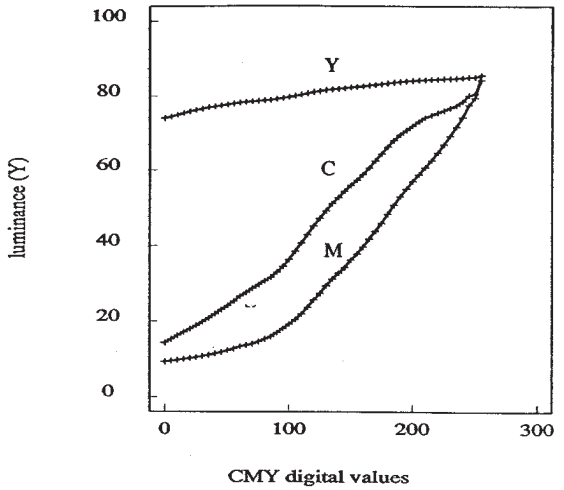

Figure 2. Relationship between the luminance and the input digital value for each primary.

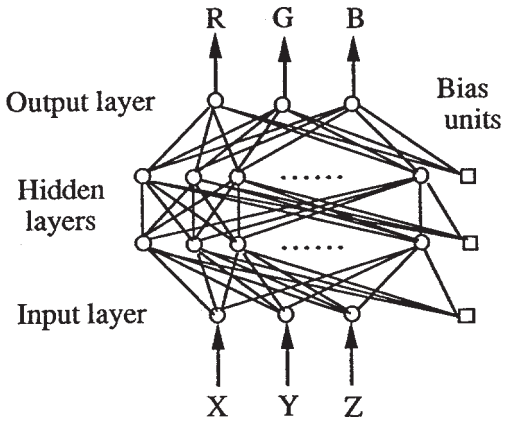

Figure 3. Structure of the neural network

Learning Procedure

Learning is the process of determining a set of weights and biases that produces a desired response to an

input color stimulus. We adopt the learning rule of Error Back-Propagation proposed by Rumelhart, et al.[6] The normalized tristimulus values and the corresponding normalized printer RGB values are presented as the pairs of input and output pattern vectors. First the input vector produces an output vector for the network. This output is compared with the desired output vector. Next the squared error is calculated between the actual and desired output. The weights and biases are then adjusted to reduce the error based on a gradient method. In the present learning rule, the error is propagated backward through the network from the output layer to the input layer, in order to minimize the overall error. For example in the case of a 3-10-10-3 type network, all the parameters to be adjusted are composed of 160 weights and 23 biases.

Now let us define the pth pair of the normalized vectors for tristimulus and printer primary values in the training data set as

$$\mathbf{i}_p \equiv [i_{p1}, i_{p2}, i_{p3}] = [X_{np}, Y_{np}, Z_{np}] \quad (6)$$

$$\mathbf{t}_p \equiv [t_{p1}, t_{p2}, t_{p3}] = [R_{np}, G_{np}, B_{np}] \quad (7)$$

Then the output of each unit in the network is described as follows:

$$o_{pj} = f(net_{pj}) \ (j = 1, 2, ..., J) \quad (8)$$

$$net_{pj} = \sum_{i=1}^{I} w_{ji} o_{pi} + b_j, \quad (9)$$

where w_{ji} represents a weight from unit i to unit j, and $o_{pj} = i_{pj}$ ($_j$=1,2,3) if the target unit j is at the input layer. The numbers I and J depend on the layer. The measure of the error is the squared error between the output vector $\mathbf{o}_p \equiv [o_{p1}, o_{p2}, o_{p3}]$ and the target vector.

The learning rule is based on the steepest gradient-descent method to minimize the error. The rule for changing the weights is given in the form

$$\Delta_p W_{ji} = \eta \delta_{pj} o_{pi}, \quad (10)$$

where η is a positive constant called the learning rate, and δ_{pj} is an error term of the jth component between the target and actual outputs. This error delta can be computed successively by the following two expressions:

$$\delta_{pj} = t_{pj} - o\delta_{pj}$$

for the output-layer units, (11)

$$\delta_{pj} = 4\alpha \, o_{pj} (1 - o_{pj}) \sum_k \delta_{pk} w_{kj}$$

for the hidden-layer units. (12)

The above learning procedure requires only that the change in weight be proportional to the error derivative. However this might also result in unstable oscillations. It is recommended to introduce a momentum term to increase the learning rate without leading to oscillation. The change of the weight is then given by the form

$$\Delta w_{ji}(n+1) = \eta \delta_{pj} o_{pi} + \beta \Delta w_{ji}(n), \quad (13)$$

where n indicates the nth step of learning, and β is a proportionality constant called the momentum constant.

Experiments

Training

First, we made color charts for the data set of training the network. Figure 4 shows the grid of the RGB color space, which is sampled in even steps to cover the entire range of the printer color space. In all 216 color patches were produced at every 51 step of each color scale. Next, the tristimulus values of the printed color patches were measured by a colorimeter. Certain variations in color occurs in printing. To remove this influence, color patches were printed four times in different arrangements on color charts, and the averages of four measurements were taken as the training data. Thus, we have a table of the training data set which consists of 216 pairs of the RGB and XYZ values.

The networks were simulated on a SUN SPARC Station. The initial values $w_{ij}(0)$ and $b_j(0)$ of the weights and biases are set to random numbers. Training is iterated for as many epochs as are necessary to decrease the mean squared error to an acceptable level.

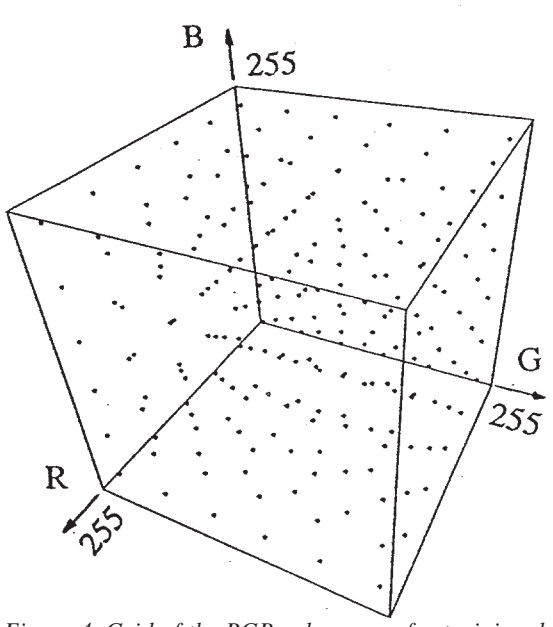

Figure 4. Grid of the RGB color space for training data

Testing

Test color charts were made for testing the mapping accuracy by the network. Each color component of RGB is sampled at 0, 64, 128, 192, and 255 in the color space. The number of color samples is 125. A color chart containing 125 color patches was printed. The measured XYZ values and the original RGB values constitute the test data set.

The network outputs the least-square estimate of the RGB values needed to produce a XYZ color stimulus. However the RGB values, which are the device dependent coordinates, seem to be not suitable for evaluating the performance of color reproduction. The performance should be evaluated on real measurement of the reproduced colors. By this reason, we reprinted color patches from the estimated RGB values, and compared the measured tristimulus values with the original XYZ values.

The discrepancy is then calculated using the CIE-L*a*b* color difference formula. For this calculation the tristimulus values are transformed into the perceptually uniform color space of L*a*b*, in which the color differences between the estimated values and the originals are calculated as the Euclidean distance. Finally the accuracy of color reproduction is represented by an average color difference over all the test data.

Determining the network structure

In order to determine an effective network structure and training strategy for the color mapping, we have carried out the training and testing under different conditions. In a previous paper,[7] the author proposed a neural network method for color notation conversion between the Munsell and CIE color systems. We discussed an effective network method from three points of view of (1) the network structure, (2) the learning constants, and (3) the data presentation. In the present study on color reproduction, we have obtained almost the same results on the second and third points. That is, the learning constants should be decreased in proper intervals of the iterative learning process, and the learning data should be presented in a random order among the given data set. Concerning the first point, the complexity of a network structure reflects the complexity of the mapping. In the following we summarize the results on the first point.

It should be noted that a large network with more layers and units is not necessarily effective for training and generalization. If the networks have similar complexity, the performance depends more heavily on the number of hidden layers than on the number of units. We have selected the structure of four layers, because no essential improvement is found for more than five layers. Next, we have determined the most effective number of units in the hidden layers. The learning behaviors of the networks were examined with different numbers of units.

Figure 5 depicts the learning error curves for 3, 5, 8, and 11 units, where each curve represents the rate of decrease of the mean squared error on R_n, G_n, and B_n, with the number of iterative presentations of the data set (epoches). In these experiments the constant of the sigmoidal function was set to $\alpha = 0.7$, and the training data are presented in a random order among the data set. The learning rate η and the momentum constant β were changed adaptively in the iterative learning process. In fact, in Figure 5 we decrease the coefficients η and β every 5000 epoches by 0.01 from 0.1 at start to 0.01 at 40000 iterations. It is seen that the learning behavior is improved with increasing the number of units, so that the structure of 11 units is the best in speed of convergence.

On the other hand, Figure 6 shows the test results. The two color differences are plotted as the function of the number of units in the hidden layers. The upper curve indicates the variation of the average. The lower is the variation of the average of the measured L*a*b* values. The detailed inspection of these curves shows that the color differences do not decrease monotonically with the number of units. It is found that both functions nearly reach the minimums at around 10 units. Thus the four-layer structure of 3-10-10-3 is chosen as the most effective system.

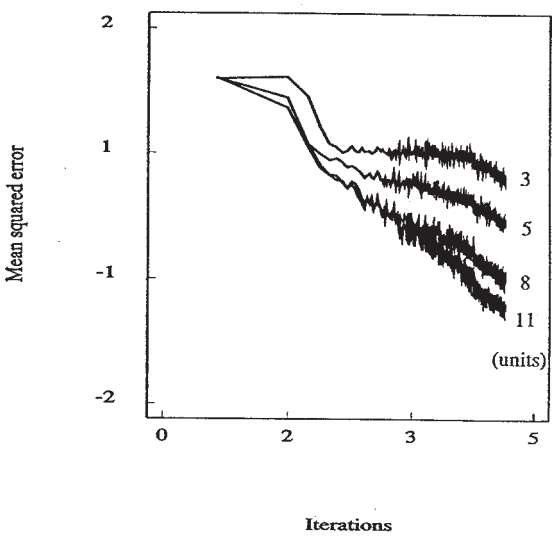

Figure 5 . Learning curves with different number of units

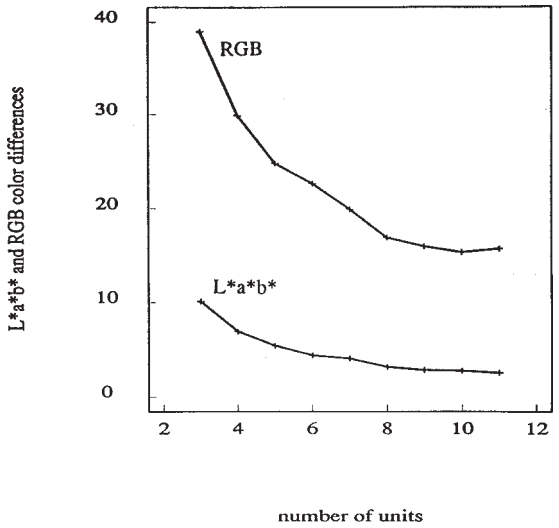

Figure 6. Color differences as a function of the number of units in the testing phase

Accuracy of color reproduction

We have trained a 3-10-10-3 type network by presenting the data in random order, and decreasing the coefficients at the proper interval, as shown in Figure 6. The test result provides the best performance ΔE^*_{ab}= 2.61 among the present experiments. For the respective quantities L*, a*, and b*, we have $\Delta L^* = 1.00$, $\Delta a^* = 1.35$. and $\Delta b^* = 1.52$. The reproduction accuracy is better on lightness. Figure 7 depicts the error distribution for all the test data in the three-dimensional L*a*b* color space. Each segment indicates a color difference vector which links the original coordinates to the estimated ones. In Figures 7 (a) and (b) the error distribution is projected on the (a*, b*) plane and the (L*, a*) plane, respectively.

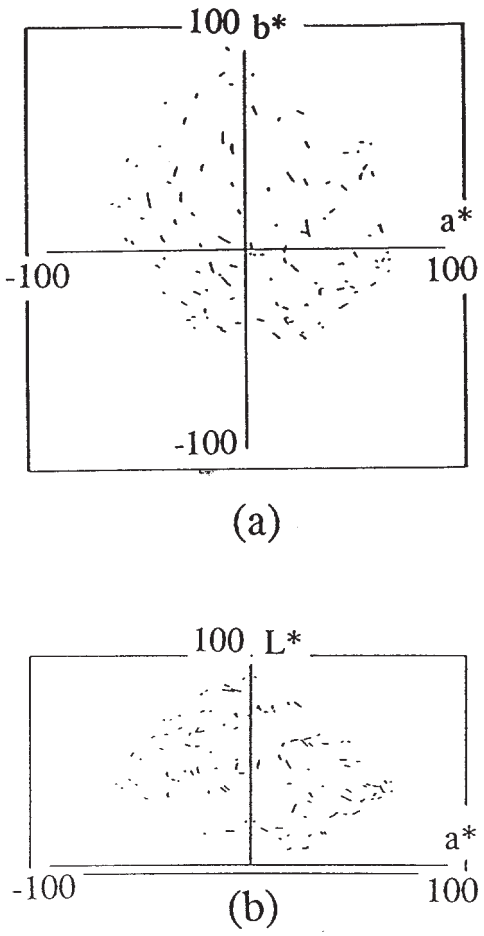

*Figure 7. Distribution of the errors in color reproduction. (*a*) (a*, b*) plane, (*b*) (L*, a*) plane*

Conclusion

This paper has proposed a method for colorimetric color reproduction on a dye sublimation printer by means of neural networks. A multilayer feedforward network is regarded as a nonlinear transformer for color coordinate transformation between the printer coordinates and the color stimulus values. The network is trained to learn a mapping to determine the required CMY (RGB) values of printer primaries for producing a given XYZ color stimulus.

Many color patches produced on a printer were measured as the data for training the network and testing the mapping accuracy. The accuracy is evaluated on the CIE-L*a*b* color difference between the reproduced color from the network output and the original color. Experiments were carried out under different conditions to determine an effective network method. The present results show that an effective network structure is a 3-10-10-3 type. We present the training data to this network in random order, and decrease the learning coefficients at proper intervals. This network achieves a good accuracy for color reproduction on our printer. Thus the complicated mapping from the XYZ to the printer color space is realized with a compact network with 183 weighting parameters.

References

1. M. C. Stone, W. B. Cowan, and J. C. Beatty, Color gamut mapping and the printing of digital color images, *ACM Trans. on Graphics*, **7**, 249-292 (1988).
2. L. W. MacDonald, Preserving the colour appearance of images across different media, *Proc. Image Processing Conf.*, London, October (1990).
3. R. S. Berns, Color WYSIWYG: A combination of device colorimetric characterisation and appearance modeling, *SID 92 Digest*, Boston, May, 540-552 (1992).
4. S. Tominaga, Production of realistic 3-D object color on a monitor, *SID 92 Digest*, Boston, May, 373-376 (1992).
5. Arai, et al., A method for transformation from CIE L*a*b* value to CMY value by a three-layered neural network, *IEICE Trans*. Japan, **76-D**: 967-975 (1993).
6. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, In D. E. Rumelhart and J. E. McClelland (Eds.), *Parallel distribute processing*, **1**, chap. 8, MIT Press (1986).
7. S. Tominaga, Color notation conversion by neural networks, *Color Research and Application*, **18**, 253-259 (1993).