A Look Up Table Based Real Time Color Classification System

Lih-Jen Kau

Dept. of Electrical and Control Engineering
National Chiao-Tung University
ADDR: 1001 Ta Hsueh Road, Hsin-Chu
Phone: 886-3-5712121 ext. 54308, Fax: 886-3-8236990
Email: u8912801@cc.nctu.edu.tw

Abstract

An algorithm based on neural networks applied for real time color classification system is proposed to recognize different objects using colors. The structure is mainly an image processing system with a variation in the look up table applied to function as a band pass filter. Simulation results show that object with multiple colors can be recognized successfully, justifying the structure and algorithm proposed. Parameters that will affect the recognition results including different color models used are also investigated. Success of this system can improve the deficiencies of gray images and can be used for automation applications doing real time object recognitions.

Keywords Color Classification, Neural Networks, Look Up Table, Object Recognition

1. Introduction

Human eyes can recognize different objects using colors. Based on this consideration, a real time color classification structure shown in Fig.1 is proposed to recognize targets using colors [1-2]. The structure is mainly an image processing system [3] with a variation in the LUT (Look Up Table) applied. Image pixels captured by camera are represented by 3-dimensional vectors at first, and then mapped to scalars through the LUT before stored in the video memory. Since a true color image system can have 16,777,216 (2^{24}) different colors, the LUT can be viewed as a band pass filter or a score table, mapping or storing how a [r, g, b] value is similar to the colors of desired target. Therefore, a color image is transformed to a gray image with each transformed gray value describes the extent of similarity to the targets.

This paper is mainly concentrated on the development of algorithm generating contents of the LUT described. Based on neural networks with hidden layers, target with multiple colors can be successfully recognized from the background. Parameters that can affect the performance of color classification including different color models used are also examined. Success of this research can be applied extensively to automation applications performing object recognitions such as fruits classification in agricultural automation, quality control in production lines, etc.

The rest of this paper is organized as follows. Section II is an overview of the system. The use of back propagation neural network on color classification is addressed in section III. Experiments of this research are covered in section IV, and section V concludes this paper.

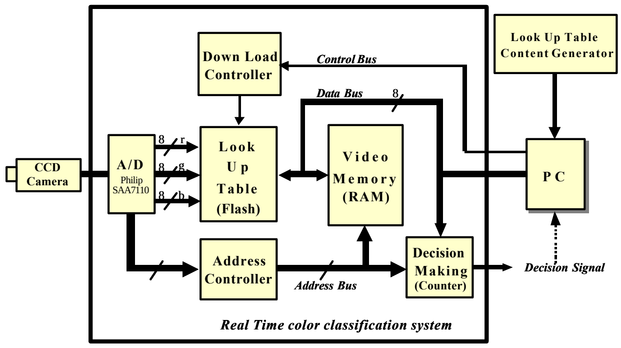

Fig.1 Real time color classification system.

2. Color Classification System

Generally speaking, the colors of chosen target always appear to be a solid in a three-dimensional color model [4]. The main work of this research is to give a score between 0~255 to each color element in the RGB color model and then fill it in the LUT on the real time color classification system shown in Fig.1. Any [r, g, b] value close to the colors of target will get a higher score. After the content of LUT is filled, the system should be capable of doing target recognition in real time.

2.1 System description

Image captured by CCD camera is converted to digital signal by the A/D converter with each pixel represented by a 3-dimensional vector [r, g, b]. The vectors are then used to address the LUT and mapped to scalars between 0~255 through the table before stored in the video memory, resulting in a gray image. To decide whether the captured image is the target or not, a threshold value for decision mechanism should be preset. Pixels in the converted image having value greater than or equal to the threshold value would be considered a candidate of the target. After that, the decision-making mechanism counts the number of pixels having value greater than or equal to the threshold described above and decides whether it's the targets or not. It should be noted that, both the mapping and decision mechanism are implemented on board, the recognition process can be finished real time.

The work of giving each [r, g, b] value a score is done offline; we care more about how well the LUT works than how fast it is created. Besides, the target vary with different applications, the LUT should be regenerated accordingly. It should be mentioned that a large memory of size 16Mbyte (2^{24}) is hardly found while implementing the LUT, means that the bus width to address the LUT should be less than 24bits. This is simply done using vector quantization by truncating the least significant bits of vectors r, g, and b. In addition to the reason mentioned above, human eyes can discern at most thousands of color shades and intensities, justifying the use of a smaller memory to implement the LUT.

2.2 Color models

Since color is a powerful descriptor that simplifies object recognition, to facilitate the specification of colors, we use HSI (Hue, Saturation, Intensity) [4] and RGB (Red, Green, Blue) [2] color models in this research. Though RGB model is commonly used as a color descriptor, the hue and saturation components in HSI model are intimately related to the way in which human beings perceive color. Furthermore, the intensity component, I, is decoupled from the color information in the image. These features make the HSI model an ideal tool for developing image processing algorithms based on some of the color sensing properties of the human visual system [4]. The applications of HSI model range from the design of imaging systems for automatically determining the ripeness of fruits and vegetables, to systems for matching color samples or inspecting the quality of finished color goods [4].

Though HSI model is closely related to the perception of human eye, the output of front-end image system or the model used for storing image pictures is usually RGB or YUV format. Conversion between color models is necessary while using HSI as the color descriptor. The formulas for converting RGB to HSI model are listed in equations (1)~(3) with all the variables normalized to unit interval [4].

$$I = \frac{1}{3}(R + G + B) \tag{1}$$

$$S = 1 - \frac{3}{(R + G + B)}\left[\min(R, G, B)\right] \tag{2}$$

$$H = \begin{cases} \dfrac{H'}{360^\circ} & \text{for } \left(\dfrac{B}{I}\right) < \left(\dfrac{G}{I}\right) \\ \dfrac{360^\circ - H'}{360^\circ} & \text{otherwise} \end{cases} \tag{3}$$

where

$$H' = \cos^{-1}\frac{\frac{1}{2}\left[(R - G) + (R - B)\right]}{\left[(R - G)^2 + (R - B)(G - B)\right]^{1/2}}$$

3. Proposed Algorithm

Since the true color system can have 16Million colors, the critical problem comes to be how we give each color a score. Back propagation neural network [5-7] together with HSI or RGB color model is proposed to fill up the contents of LUT. The network for color classification using single hidden layer with 3 inputs and only 1 output is shown in Fig.2. Though the use of HSI color model would get better results in most cases, either RGB or HSI values of the corresponding pixel can be used as inputs for the network shown below in this system.

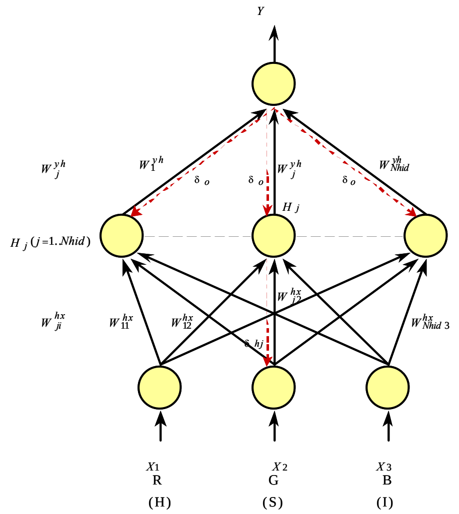

Fig.2 Proposed color classification system with Single hidden layer.

To make the network capable of doing color classification, representative image pictures are selected as templates to generate the training patterns for the network. The input space is partitioned into two groups while sampling critical colors as training patterns among templates, one for target and the other for background respectively. Each sampled pattern is then assigned a value 1 or 0 manually according to whether it is the colors of target or not. These critical colors and graded values are then used as the training patterns and fed into the network repeatedly. Finally the well-trained network should be able to recognize the colors of target and background. When a novel color is fed into the network, the network will output a value, which indicates to what extent this color is close to the target.

3.1 Neuron model and connectivity

Since we are using back propagation neural network in this research, the network would be fully connected as shown in Fig.2 with 3 neurons describing the pixel color in the input layer, and 1 neuron which represents the similarity to the target in the output layer. The network can have double or single hidden layer with the number of hidden neurons decided by user. As the neuron model is considered, they are made up of perceptrons each with the same type as shown in Fig.3.

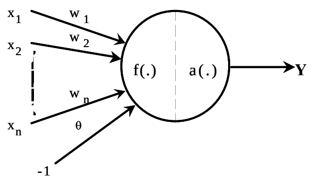

Fig.3 Neuron model

The function denoted by f(.) in Fig.3 is the integration function, which is a weighted summation given by EQ. (4) in this research.

$$f = net\ input = \sum_{j=1}^{n} w_j x_j - \theta \qquad (4)$$

Parameter θ, which is sometimes expressed implicitly in the weights is the threshold of the neuron, x_j is the input connected to this neuron, and w_j is the weight corresponding to input x_j. The unipolar sigmoid function given in EQ. (5) is used as the activation function, and is denoted by a(.).

$$a(f) = \frac{1}{1+e^{\lambda f}} \quad , where\ \lambda > 0 \qquad (5)$$

where f in the above equation is the net input of the neuron given by EQ (4). For simplification, we let $\lambda = 1$ throughout this paper.

3.2 Weight updating method

Connection weights of the network are updated using gradient descent method during training phase. The backward direction dotted lines shown in Fig.2 means error signal δ_o of output layer are back propagated to hidden and input layer for updating the current weights. Since we are using unipolar sigmoid function as the activation function, and we have only one neuron in the output layer, the error signals of output layer and hidden layer are thus calculated by EQ. (6) and EQ. (7) respectively.

$$\delta_o = Y(1-Y)(T-Y) \qquad (6)$$

$$\delta_{hj} = H_j(1-H_j)W_j^{yh}\delta_o \qquad (7)$$

Parameter T in EQ. (6) is the desired output of training pattern, while Y is the real output. δ_{hj} in EQ. (7) denotes the error signal of *j*th neuron in the hidden layer, H_j is the output of *j*th neuron in the hidden layer, and W_j^{yh} is the weight connecting *j*th neuron in the hidden layer to the output neuron. Having defined error signals, the weight increment ΔW_j^{yh} connecting *j*th hidden neuron to output neuron, and the increment ΔW_{ji}^{hx} connecting *i*th input neuron to *j*th hidden neuron are calculated according to EQ. (8) and EQ. (9), and then the new weights are updated according to EQ. (10) and EQ. (11) respectively.

$$\Delta W_j^{yh} = \eta\delta_o H_j + \alpha W_j^{yh} \qquad (8)$$

$$\Delta W_{ji}^{hx} = \eta\delta_{hj} X_i + \alpha W_{ji}^{hx} \qquad (9)$$

$$W_j^{yh} = W_j^{yh} + \Delta W_j^{yh} \qquad (10)$$

$$W_{ji}^{hx} = W_{ji}^{hx} + \Delta W_{ji}^{hx} \qquad (11)$$

where X_i $, i = 1..3$ is the input to the *i*th neuron in the input layer, η is the learning rate, and α is a momentum term to accelerate convergence speed.

To avoid being trapped into saturation region in the first few steps while training, input values are normalized between [0,1] before fed into the network. On the other hand, output value is scaled by a factor of 255 before stored in the LUT since the use of unipolar sigmoid function. There is no single learning rate suitable for different training cases. Larger values of learning rate speed up the convergence but might cause overshooting problem, while a smaller one has the complementary effects. Basing on this consideration, adaptive learning rate is used in proposed network during training period. The increment of learning rate is calculated according to

$$\Delta\eta = \begin{cases} +a & if\ \Delta E < 0\ \ for\ \ 20\ steps \\ -b\eta & if\ \Delta E > 0 \\ 0 & otherwise \end{cases} \qquad (12)$$

where we let *a*=0.01, *b*=0.1 in this research.

4. Experiment Results

Both the work to grade each critical colors a score 0 or 1 and fill out the contents of LUT are tedious. For this reason, an integrated color classification system working under 32-bit operating systems with graphic user interface had been developed successfully in this research to handle these tasks.

To verify the usefulness of the proposed system, Fig.4 is chosen as one of test images in our experiments. Yellow hat and green hat in Fig.4 are selected as the target in the simulation process. 500 pixels in Fig.4 are sampled to be training patterns and fed into a three-layered network with five hidden neurons. After 5000 training cycles with adaptive learning rate and $\alpha = 0.5$, a LUT of size 4Kbytes or each component of color pixel is represented by 4bits equivalently is generated. Fig.5 is the gray image showing the target can be recognized successfully when the LUT described above is used. Besides, success of the recognition can also be found in Fig.6, the histogram of Fig.5. As we can see from Fig.6, the target and background are clearly separated. Success of the experiment also justifies the use of a memory with smaller size. It should be mentioned again that even we are using HSI model to generate the contents of LUT, the final LUT should be addressed by the elements of RGB. This is because most of the front-end chips produce their output in the format of RGB or YUV, not HSI.

Fig.4 Five hats on the wall.

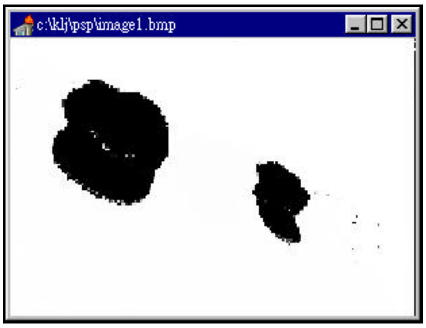

Fig.5 Yellow and green hats are recognized successfully.

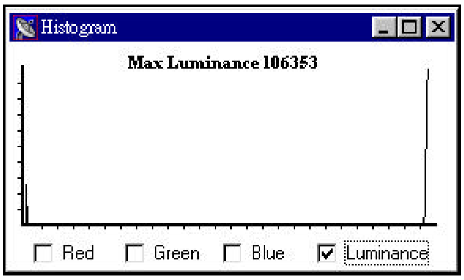

Fig.6 Histogram of Fig.5.

5. Conclusions

We proposed a structure for object recognition using colors. Furthermore, back propagation neural network had been applied successfully as an offline algorithm to generate the contents of LUT in this structure. Target with multiple colors can be successfully recognized from the background as can be seen in experiments. Investigations also show that the use of HSI model in LUT generating process would get better performance than the use of RGB model in general cases. Success of this research has justified the use of color in object recognition, and the results can be applied extensively for automation applications.

References

[1] Y. Deng, B. S. Manjunath, H. Shin, “Color Image Segmentation”, Proc. Of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1999.

[2] T. Uchiyama, M. A. Arbib, “Color Image Segmentation Using Competitive Learning“, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 16, no. 12, pp.1197-1206, DEC. 1994.

[3] Charles A. Poynton, “A Technical Introduction to Digital Video”, John Wiley & Sons, New York, 1996.

[4] Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing”, Addison-Wesley, pp.221-249, 1996.

[5] Chin-Teng Lin, C.S. George Lee, “A Neural-Fuzzy Synergism to Intelligent Systems”, Prentice Hall, 1996.

[6] Simon Haykin, “Neural Networks A Comprehensive Foundation“, Macmillan, 1994.

[7] S. C. Lee, E. T. Lee, “Fuzzy Sets and Neural Networks “, Journal of Cybernetics, Vol. 4, pp.83-103, 1974.