ELSEVIER

Image and Vision Computing 18 (2000) 213–222

image AND vision COMPUTING

www.elsevier.com/locate/imavis

Multithresholding of color and gray-level images through a neural network technique

N. Papamarkos*, C. Strouthopoulos, I. Andreadis

Electric Circuits Analysis Laboratory, Department of Electrical and Computer Engineering, Democritus University of Thrace, 67100 Xanthi, Greece

Received 17 November 1998; received in revised form 17 March 1999; accepted 23 June 1999

Abstract

One of the most frequently used methods in image processing is thresholding. This can be a highly efficient means of aiding the interpretation of images. A new technique suitable for segmenting both gray-level and color images is presented in this paper. The proposed approach is a multithresholding technique implemented by a Principal Component Analyzer (PCA) and a Kohonen Self-Organized Feature Map (SOFM) neural network. To speedup the entire multithresholding algorithm and reduce the memory requirements, a sub-sampling technique can be used. Several experimental and comparative results exhibiting the performance of the proposed technique are presented.

Keywords: Color image segmentation; Thresholding; Neural networks; Principal Component Analyzer; Self-Organized Feature Map

1. Introduction

Multithresholding is one of the most powerful techniques for image segmentation. The application of multithresholding techniques is based on the assumption that object and background pixels in a digital image can be distinguished by their gray-level or color values. The proposed multithresholding technique can be applied to gray-level images as well as to color images. Color is one of the most important properties which humans use for object discrimination. Color is a component, which adds a new dimension to machine vision, and it has been overlooked in the past. Limitations of possible applications of color machine vision have been associated with high cost and low processing of added information. Recent progress, however, in the microelectronics industry resulted in tackling, partially, some of these difficulties, and a limited, but increasing, number of systems that utilize color information have been reported [1–8]. Researchers have proposed various color image segmentation techniques. Ohta et al. [1] applied the Karnhunen–Loeve transform to the color images to derive color features with large discriminating power. Connah and Fishbourne [2] reported an algorithm based on the histogram analysis of color gray-level distribution. Huntsberger and Delxazi [3] reported a color edge detector, which uses fuzzy set theory. Barth et al. [4] presented a microcomputer color-based vision system to inspect integrated circuits. Tominaga [5] described a color classification method based on the Lab uniform color space. Representative colors are classified according to a color distance criterion. Jordan and Bovik [6] described an investigation of the utilization of chromatic information in solving stereo vision problems. Valchos and Constantinides [7] proposed an algorithm based on color and graph-theory, most suitable for applications such as broadcasting and pictorial information retrieval. Schettini [8] presented a color image segmentation based on recursive 1D histogram analysis. The resulting regions are merged on the basis of a criterion, which takes into account color similarity and spatial proximity.

Segmentation by multithresholding started many years ago from simple beginnings, and in recent years has been refined into a set of mature procedures [9–17]. The outstanding problem is how to devise an automatic procedure for determining the optimal thresholds. The various techniques can be classified into three categories. Methods based on edge matching and classification belong to the first category [10,11]. In the second category belong the histogram based multithreshold techniques [12–15]. These techniques are based on different criteria, such as the minimum entropy, the interclass variance between dark and bright regions, and changes of zero-crossing and curve-fitting. Finally, in the third category belong all the other techniques, which are usually characterized as hybrid. Extension of the

* Corresponding author. Tel.: +30-541-79566; fax: +30-541-26947.
E-mail address: papamark@voreas.ee.duth.gr (N. Papamarkos).

0262-8856/00/$ - see front matter
PII: S0262-8856(99)00015-3

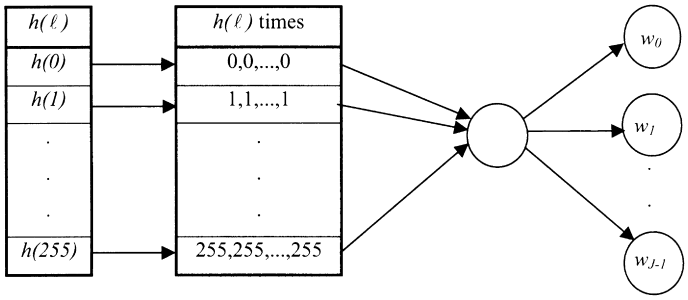

Fig. 1. Training the SOFM neural network.

aforementioned techniques to multispectral images and, in particular, to color images is not a trivial task. The increasing complexity and the structure of natural color scenes are the main reasons. Furthermore, a threshold technique must be fast and suitable for real-time applications since the basic concepts were, after all, to save computation time and memory.

This paper proposes a new algorithm suitable for gray-level and color image multithresholding. The new technique treats image multithresholding as a feature clustering approach. For gray-level images, a Kohonen Self-Organized Feature Map (SOFM) neural network [18,19] is used which self-organizes the input feature space (the image gray-level values) to a limited number of gray-level classes. For color images, a combination of Principal Component Analyzer (PCA) and a Kohonen SOFM neural network is used to perform the color clustering. The PCA is trained with the Generalized Hebbian Algorithm (GHA) [18]. Through the PCA the input vectors are transformed in order to increase the total variance in the feature space. The inputs to the SOFM are the outputs of the PCA. After the training stage, the output neurons of the SOFM define the optimal centers of the color clusters. In both gray-level and color images, the training set can be a representative sample of the image pixels. The sub-sampling is performed via histogram normalization or especially in color images, via a fractal scanning technique based on the Hilbert's space-filling curve [20]. The proposed multithresholding technique is fast, applicable to any type of image and offers substantial compression of the image space without significant loss of the color information contained in it. It should be noted that the optimal number of thresholds could be obtained by best fitting the image histogram through a linear combination of probability density functions. In comparison with other existing methods, the proposed technique offers two significant advantages that are related with the applicability of the method to color images as well as the self-organized operation. The method has been implemented in C++ and tested with a variety of images. Furthermore, characteristic examples for both gray-level and color images are presented. The computation times given in the experimental results are referred to a Pentium 233 MHz computer. The experimental and comparative results confirm the effectiveness of the proposed method.

2. The multithresholding technique

2.1. Gray-level images

The multilevel threshold selection can be considered as the problem of finding a set T_k, $(k = 1, 2, \ldots, J-1)$ of threshold values, in order that the original gray-level image $I(x,y)$ would be transformed to a new one with only J levels. More specifically, if T_k, $(k = 1, 2, \ldots, J-1)$ are the threshold values with $T_1 < T_2 < \ldots < T_{J-1}$, then the output image can be defined as

$$I_T(x,y) = \begin{cases} m_0, & \text{if } I(x,y) \le T_1 \\ m_1, & \text{if } T_1 < I(x,y) \le T_2 \\ \vdots & \\ m_{J-1}, & \text{if } I(x,y) > T_{J-1} \end{cases} . \quad (1)$$

where m_k represents the mean histogram values in the range $(T_k, T_{k+1}]$ with $T_0 = 0$ and $T_J = 255$.

The proposed method considers multithresholding as a clustering problem and adopts a Bayesian approach, which assumes Gaussian distributions. However, it is well known that a suitable neural network can effectively solve clustering problems. Thus, for gray-level images, we developed a non-parametric multithresholding technique, which is based on a Kohonen SOFM neural network [18]. It is well known than the main goal of a SOFM neural network is the representation of a large set of input vectors with a smaller set of "prototype" vectors so that a good approximation of the original input space would be obtained. In other words, a SOFM neural network decreases optimally the input space to a smaller one. The resultant space can be viewed as a representative of the original space and, therefore, it satisfies the main statistical characteristics of the input space. The Kohonen SOFM neural network combines its input layer with a competitive layer of J neurons and is trained by unsupervised learning. The competition operation of the output neurons is the main characteristic of the learning process. For every input vector, only one output neuron is activated which is called the "winner" neuron.

In the beginning it is assumed that the desired number of the histogram clusters is J. A procedure for the estimation of

Table 1
Application results

Number of thresholds	Thresholds	Gray-level values	Fitting error $\lvert e_{J_0}\rvert$
1	117	58,176	0.048182
2	56,135	22,91,184	0.038630
3	53,121,197	21,86,155,233	0.034450
4	36,77,126,197	14,57,95,157,233	0.032666
5	32,70,106,143,199	13,51,89,123,164,233	0.032246
6	24,54,82,111,145,200	10,38,69,94,127,164,234,	0.030339
7	20,44,68,88,113,146,200	8,31,56,78,97,128,164,234	0.031234
8	19,43,67,88,112,143,173,212	8,30,55,78,97,126,159,191,238	0.027881
9	16,37,59,79,95,116,145,175,215	6,25,47,69,87,102,129,160,194,239	0.029808

the optimal value of J for gray-level images is described below. The first stage of the multithresholding technique is the determination of the training set S for the neural network. As it is depicted in Fig. 1, the training set consists simply of the gray-level values of the pixels. More specifically, if $h(\ell)$ is the image histogram, then for each ℓ gray-level, the value ℓ is contained $h(\ell)$ times in the training set. To speed up the clustering procedure, the histogram is normalized in such a way as

$$h_{\text{norm}}(\ell) = 1000 \times h(\ell) / \sum_{\ell=0}^{L-1} h(\ell) \tag{2}$$

The Kohonen SOFM neural network is unsupervised trained with this set. Specifically, the Kohonen SOFM output competition layer consists of J neurons arranged in a 1D grid such that each neuron j to have one coefficient w_j (Fig. 1). After training, each coefficient w_j represents one class. Therefore, each pixel (x,y) is assigned to the class, which corresponds to the winner neuron of the output layer according to the relation:

$$|w_c - I(x,y)| = \text{minimum}\{|w_j - I(x,y)|\}, \quad j = 0, 1, \ldots, J-1 \tag{3}$$

where c is the winner neuron. Thus, for J classes, $J-1$ threshold values T_k, $(k = 1, 2, \ldots, J-1)$ are obtained from the relation

$$T_k = \frac{w'_{k-1} + w'_k}{2} \tag{4}$$

where the coefficients w'_k are the coefficients w_k sorted in ascending order. Using these thresholds, the gray-levels of the image can be decreased to only J values. It should be noted that instead of the use of the mean histogram values m_k (Eq. (1)) we can use the values of the coefficients of the output neurons directly. However, we prefer to use the mean values as a better representative of colors of each class.

2.2. Estimation of the optimal number of thresholds

An important, but difficult, issue in multithreshold selection techniques is the estimation of the optimal number of thresholds. If this number is smaller than the optimal, then multithresholding fails and object overlapping appears in the final output image. In our approach, the estimation of the optimal threshold values is based on the fitting of the normalized histogram function $H(\ell)$ by a function $H_f(\ell)$, which is a linear combination of the probability density functions for J classes. The C_j, $(j = 0, \ldots, J-1)$ classes are those that were obtained by the use of the SOFM neural network. The optimal value of J, from a desired set $\{2, 3, \ldots, J_{\max}\}$ of values, minimizes the fitting error between the $H(\ell)$ and $H_f(\ell)$ functions. The function $H(\ell)$ is defined by the relation:

$$H(\ell) = h(\ell) / \sum_{\ell=0}^{L-1} h(\ell) \tag{5}$$

Also, $H(\ell)$ is a probability density function since

$$\sum_{\ell=0}^{L-1} H(\ell) = 1.$$

The probability f_j of class C_j occurrence, the class mean μ_j and its variance σ_j^2 are given by the relations:

$$f_j = P_r(C_j) = \sum_{C_j} H(\ell) \tag{6}$$

$$\mu_j = \sum_{C_j} \ell H(\ell) / \sum_{C_j} H(\ell) \tag{7}$$

$$\sigma_j^2 = \sum_{C_j} (\ell - \mu_j)^2 H(\ell) / \sum_{C_j} H(\ell) \tag{8}$$

The fitting function $H_f(\ell)$ is the sum

$$H_f(\ell) = \sum_{j=0}^{J-1} f_j H_j(\ell) \tag{9}$$

where

$$H_j(\ell) = \frac{1}{\sigma_j \sqrt{2\pi}} \exp\left\{ -\frac{(\ell - \mu_j)^2}{2\sigma_j^2} \right\} \tag{10}$$

It can be easily shown that $H_f(\ell)$ is a probability density function too. The fitting error e_J for a number of J classes is

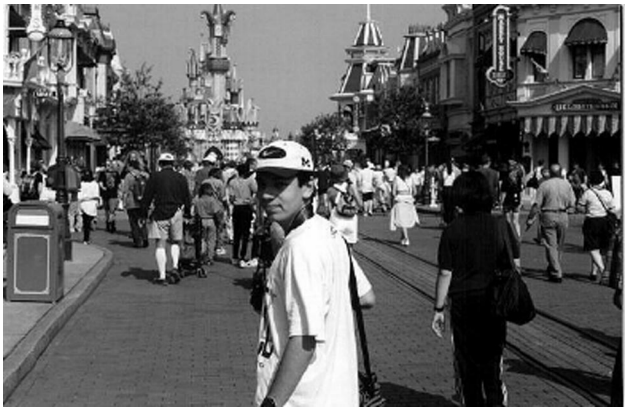

Fig. 2. Original image of 8-bit resolution.

Fig. 4. Image after multithresholding with only nine gray values.

defined as

$$e_J = \sum_{\ell=0}^{L-1} [H(\ell) - H_f(\ell)]^2 \tag{11}$$

and is computed for $J = 2,\ldots,J_{\max}$. The value J_0 is optimal when the corresponding fitting error $|e_{J_0}|$ satisfies the condition

$$|e_{J_0}| = \text{minimum}(|e_J|) \tag{12}$$

Table 1 summarizes the application of the method to the image of Fig. 2 for different number of thresholds. The size of the image is 400×259 pixels. As it can be noted, in this case, the optimal number of threshold is eight. Fig. 3 shows the histogram and the eight thresholds obtained. Fig. 4 depicts the final image after multithresholding, which has only nine gray-levels. For comparison reasons, Table 2 summarizes the application results of other existing gray-level multithresholding methods to the image of Fig. 2. This image is processed by applying the methods of Reddi et al. [12], Kapur et al. [13], and Papamarkos et al. [14]. These methods are selected because they are based on different clustering criteria. The methods of Reddi et al. determines the thresholds so that the interclass variance between dark and bright regions is maximized. It is a popular multithresholding technique, because it is an extension of the classical global thresholding algorithm of Otsu [15]. The method of Kapur et al. selects thresholds by using the maximum entropy criterion. Finally, the method proposed by Papamarkos et al. is based on a hill-clustering algorithm in combination with an appropriate linear programming approximation technique. As it can be observed from Table 2, for a small number of thresholds, the results obtained by the proposed method are close but not the same as the results given by the method of Reddi et al. This is only an observation and it is due to the fact that

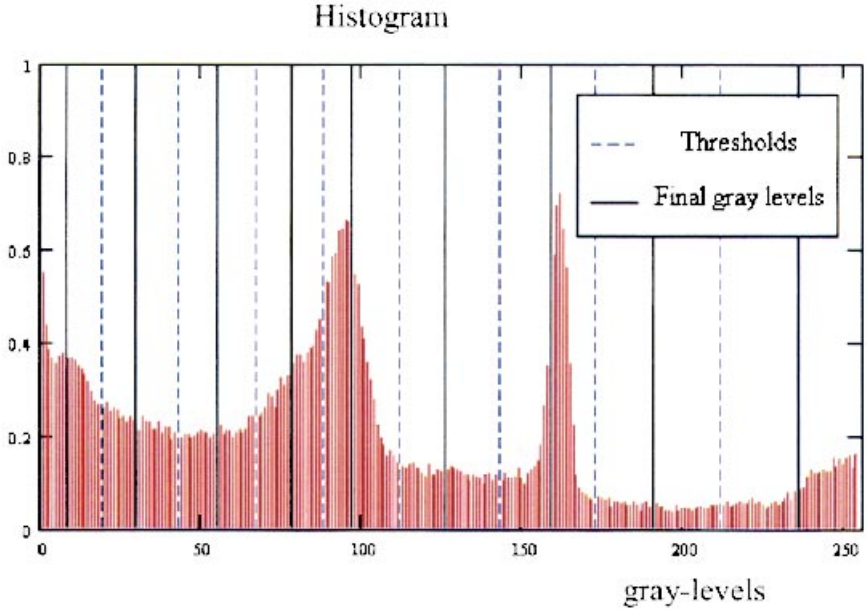

Fig. 3. Image histogram and the threshold values.

Table 2
Comparative results

Number of thresholds	Otsu [15]	Reddi et al. [12]	Kapur et al. [13]	Papamarkos et al. [14]
1	117	117	116	138
2		58,139	77,166	140,205
3		54,121,193	54,104,170	48,132,198
4		36,76,126,194	54,104,167,212	48,120,144,210
8		23,53,81,108,141,170,197,227	–	4,31,47,115,150,173,195,226

the nature of the proposed method and the Reddi et al. technique are both histogram clustering approaches. Table 3, provides some comparative computation time results of the application of the multithresholding techniques to the image of Fig. 2. It can be observed that for a number of thresholds greater than four the methods of Kapur et al. and Papamarkos et al. are time consuming. On the contrary, the proposed method and the method of Reddi et al. require approximately the same computation time.

3. Extension of color images

The technique described in the previous section can be extended in order to be able to accommodate color images. A color pixel is characterized by its three primary RGB values. The generalization of the method is implemented by increasing the number of the inputs of the neural network (from 1 to 3) and consequently a proportional increment of the output neurons and weight coefficients. In this case, the topology of the entire neural network has the structure shown in Fig. 5. As it can be observed, the first stage of the neural network is a PCA. PCA is useful because of the multi-dimensionality of the feature space. Through the PCA transform, the maximum variance in the input feature space is achieved and, hence, the discrimination ability of the SOFM is increased. The PCA is realized through a neural network, which is described later.

3.1. Sub-sampling

The proposed technique can be applied to the color images without any sub-sampling. However, in the case of large size color images, and in order to achieve reduction in computation time and memory size requirements, it is preferable to have a sub-sampled version of the original image. This sub-sampling can be made via histogram normalization as it happens in the case of the gray-level images. Now, the normalized image histogram can be defined as

$$h_{\text{norm}}(r,g,b) = 1000 \times \frac{h(r,g,b)}{\sum_r \sum_g \sum_b h(r,g,b)} \tag{13}$$

where $h(r,g,b)$ is the 3D image histogram. However, for the sub-sampling image to be a better representation of the original, we choose to use a fractal scanning process based on the well-known Hilbert's space-filling curve. A complete description of the Hilbert's curve is given in Ref. [20]. If k is the order of the Hilbert's curve then the ratio of the sub-sampled image pixels over the total number of pixels is

$$\frac{4^k}{\text{number of image pixels}} \tag{14}$$

As an example, Fig. 6 depicts a fractal image sub-sampling of a 200×200 pixel image with $k = 7$.

3.2. Multithresholding

In general, each neuron of the SOFM competition layer represents a class. In the case of color images, the center of each class C_j, $(j = 0,\ldots,J-1)$ is represented by a vector $\mathbf{c}_j = [R_j, G_j, B_j]^{\mathrm{T}}$. If $\mathbf{I}(x,y) = [R(x,y), G(x,y), B(x,y)]^{\mathrm{T}}$ represents the RGB colors of the pixel (x,y), then the output image with only J colors can be defined by the relation:

$$\mathbf{I}_T(x,y) = \begin{cases} \mathbf{m}_0, & \text{if } \mathbf{I}(x,y) \in C_0 \\ \mathbf{m}_1, & \text{if } \mathbf{I}(x,y) \in C_1 \\ \vdots & \\ \mathbf{m}_{J-1}, & \text{if } \mathbf{I}(x,y) \in C_{J-1} \end{cases} . \tag{15}$$

where $\mathbf{m}_j$ is the vector of the mean RGB color values of the pixels belonging to the class C_j. It should be remembered

Table 3
Comparative computation times (s)

Number of thresholds	Proposed	Reddi et al. [12]	Kapur et al. [13]	Papamarkos et al. [14]
1	0.35	0.32	1.48	0.42
2	0.65	0.68	2.10	1.34
4	1.12	1.32	58.04	2.23
8	2.50	2.02	–	8.25

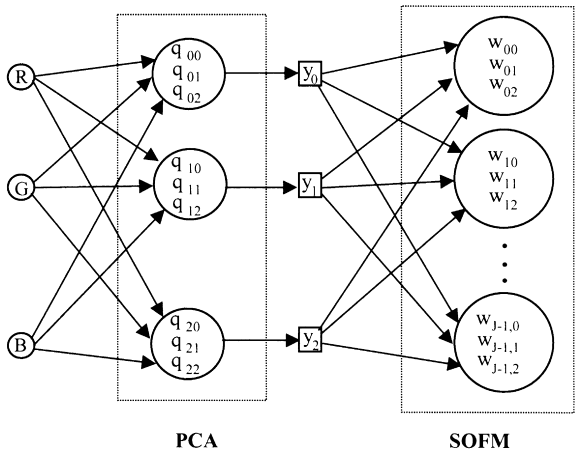

Fig. 5. Structure of the neural network for color multithresholding.

that each image pixel is classified to the class, which corresponds to the winner output neuron, or in other words to the class whose center $\mathbf{c}_j$ is the nearest to the RGB pixel vector according to the Euclidean distance.

3.3. Estimation of the optimal number of thresholds

The optimal number of threshold is estimated by an extension of the procedure used for the gray-level images. Specifically, if $\boldsymbol{\mu}_j = [\mu_r^j, \mu_g^j, \mu_b^j]^T$ is the mean value of class C_j and $\boldsymbol{\sigma}_j^2 = [\sigma_r^{j^2}, \sigma_g^{j^2}, \sigma_b^{j^2}]^{\mathrm{T}}$ is its variance, then

$$H_j(r,g,b) = \frac{1}{\sigma_r^j \sigma_g^j \sigma_b^j (2\pi)^{3/2}} \times \exp\left(-\frac{(r-\mu_r^j)^2}{2\sigma_r^{j^2}} - \frac{(g-\mu_g^j)^2}{2\sigma_g^{j^2}} - \frac{(b-\mu_b^j)^2}{2\sigma_b^{j^2}} \right) \tag{16}$$

Fig. 6. Image sub-sampling using fractal scanning.

The fitting function has now the following form

$$H_f(r,g,b) = \sum_{j=0}^{J-1} f_j H_j(r,g,b) \tag{17}$$

where $f_j = \Pr(C_j)$.

Finally, the fitting error e_J for a number of J classes is defined as

$$e_J = \sum_r \sum_g \sum_b [H(r,g,b) - H_f(r,g,b)]^2 \tag{18}$$

4. Structure and training of the neural network

4.1. The PCA neural network

As it was mentioned above, in the case of color images, the first stage of the neural network is a PCA. After training, the feature vector is optimally projected to a new axis set such that the maximum discrimination in the feature space would be obtained. Each different RGB color contributes proportionally to the training set. The transformation is designed in such a way that the original feature-set may be represented by a number of effective "features" and yet retains most of the intrinsic information contained in the data. In multivariate data analysis the Karhumen–Loeve Transformation (KLT) [18] is a widely used technique for PCA. It is a procedure for the estimation of the eigenvalues and eigenvectors requiring the computation of data convariance matrix. Here, a single-layer feed-forward neural network is used to perform a PCA. Using this technique, there is no need to compute the covariance matrix, as the eigenvectors are derived directly from the data. In our approach, the input vector is the 3-feature pixel vector $\mathbf{I} = [R,G,B]^{\mathrm{T}}$, whereas the output is a 3-feature vector $\mathbf{y} = [y_0, y_1, y_2]^{\mathrm{T}}$, which is computed by the relation $\mathbf{y} = \mathbf{QI}$. Specifically, the input of the PCA (input features) are the three primary values of each pixel of the color images and the output consists of three neurons. Matrix $\mathbf{Q}$ provides the PCA neural network coefficients. The net is trained by the GHA. This is an unsupervised learning algorithm based on a Hebbian learning rule. The Hebbian rule states that if two neurons on either side of a synapse are activated simultaneously, then the strength of that synapse is selectively increased, otherwise it is weakened or eliminated. The GHA is implemented in the training stage via the relation

$$\Delta q_{ik} = \gamma \left(\underbrace{y_i I_k}_{\text{(A)}} - \underbrace{y_i \sum_{n=0}^{i} q_{nk} y_n}_{\text{(B)}} \right), \qquad \text{with } i,k = 1,2,3,\ldots \tag{19}$$

Term (A) expresses the Hebbian rule and term (B) imposes a limit on the growth of synaptic weights. It should be noted that after training, $\mathbf{Q}$ approximates with probability one the matrix whose rows are the three eigenvectors of the

Fig. 7. (a) Original image and (b) image after color multithresholding.

covariance matrix, formed by decreasing eigenvalues. Also, for example, the weight coefficients q_{00}, q_{01} and q_{02} of the first neuron determine an eigenvector that exhibits the maximum discrimination power to its direction. The output values y_0, y_1 and y_2 of the PCA are the projections of the RGB vector to these eigenvectors. Thus, the training of SOFM is based on the projections of feature vectors.

4.2. The SOFM neural network

The SOFM has an input and an output layer. The input layer has one or three neurons depending on whether we have a gray-level or a color image, respectively. The output layer, which is called competition layer, consists of J neurons. The SOFM is competitively trained according to the following weight update functions:

$$\text{gray-level images} \rightarrow \Delta w_j = \begin{cases} a(y - w_j), & \text{if } |c - j| \le d \\ 0, & \text{otherwise} \end{cases}, \qquad j = 0, \ldots, J-1 \tag{20}$$

$$\text{color images} \rightarrow \Delta w_{ji} = \begin{cases} a(y_i - w_{ji}), & \text{if } |c - j| \le d \\ 0, & \text{otherwise} \end{cases}, \qquad \begin{matrix} i = 0, 1, 2 \text{ and} \\ j = 0, \ldots, J-1 \end{matrix} \tag{21}$$

where y_j are the input values, a the learning parameter, d the neighbor parameter (a typical initial value is less than 0.25) and c the winner neuron. The values of parameters a and d are reduced to zero during the learning process. This learning algorithm has been referred to as Kohonen's learning. As it is mentioned above, the output of SOME consists of J neurons, each of which is related, with the gray-level values or with the RGB components of a particular color. Thus, by adjusting the number of the output neurons we can define the number of colors in the final image.

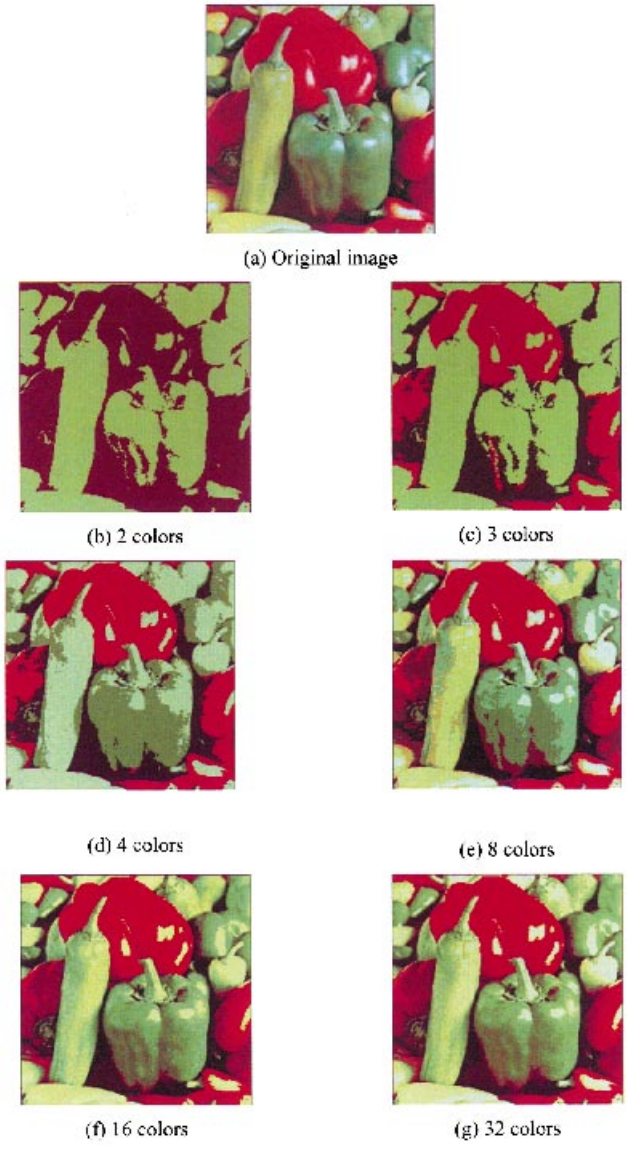

Fig. 8. Image representation using different number of thresholds.

5. The steps of the algorithm

According to the above analysis, the steps of the entire multithresholding algorithm for gray-level and color images are:

Gray-level images

Step 1. Determine the image histogram h and construct the training set S.
Step 2. Set the number of thresholds or determine the optimal number of thresholds.
Step 3. Train the SOFM without supervision. After training, the output neurons define the centers of the J classes.
Step 4. Sort the coefficients w_j, $j = 1, \ldots, J$ of the neural network in an ascending order. For every two

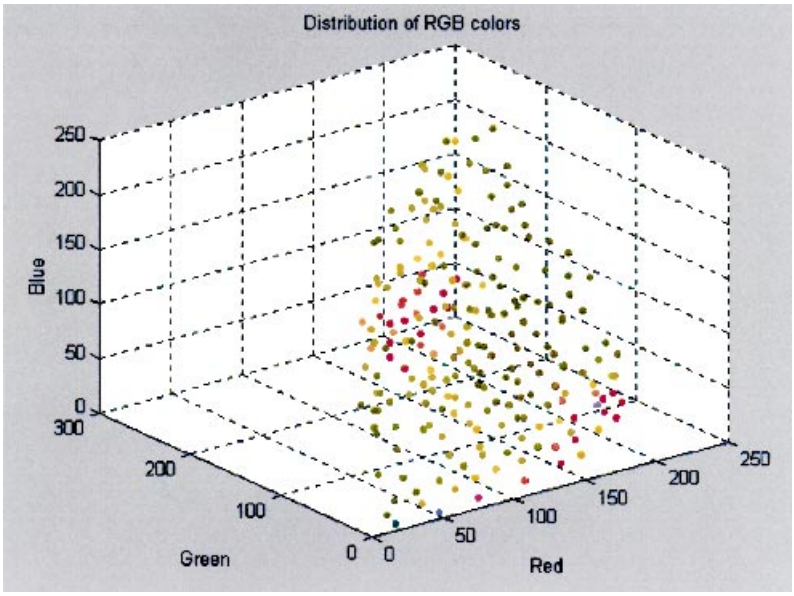

Fig. 9. Color distribution in 3D.

successive values w'_{k-1}, w'_k $(k = 1,\ldots,J-1)$ determine from Eq. (4) the threshold values T_k.

Step 5. In every region (T_k, T_{k+1}), calculate the mean histogram value m_k.

Step 6. Using Eq. (1) convert the original image to the final multithresholding image $I_T(x,y)$ with only J gray-levels.

Color images

Step 1. If it is necessary sub-sample the color image and construct the training set S.

Step 2. Set the number of thresholds or determine the optimal number of thresholds.

Step 3. Train the PCA with the set S.

Step 4. Using the PCA, the set S produces the output vectors $\mathbf{y}$ which construct the training set S' for the SOFM.

Step 5. Train the SOFM with set S' and without supervision. After training, the neurons of the output competition layer define the J classes.

Step 6. Using the entire neural network, the vector $\mathbf{I}(x,y)$ of every image pixel of the original color image is classified to one of the J classes.

Table 4
Computation times

Number of colors	Computation time (s)
2	2.5
4	4
8	7
16	12
32	21

Step 7. Using the color of each class calculate the mean color vector $\mathbf{m}_j$ of class j.

Step 8. Using Eq. (15) convert the original image to the final color multithresholding image $\mathbf{I}_T(x,y)$ with only J colors.

6. Experimental results

6.1. Experiment 1

An illustrative example of reduction of the number of colors in an 8-bit per primary channel color image is shown in Fig. 7. The optimal number of thresholds is 12 and it can be seen that the resulting image exhibits no noticeable degradation. This can be useful for image compression and picture transmission applications.

6.2. Experiment 2

Fig. 8 shows a color image and its representation using different numbers of thresholds obtained by applying the proposed technique. In this example, the optimal number of thresholds is eight. In addition, the distribution of the image RGB colors before the application of the PCA is shown in Fig. 9. It can be observed form Fig. 8(c) that even in the marginal case of the use of only three thresholds the main color characteristics of the image remain. The computation time for each example is given in Table 4.

6.3. Experiment 3

Fig. 10 shows an image consisting of two objects on a white background. The color of the circle is described by the vector (100,100,100) and the color of the square by the

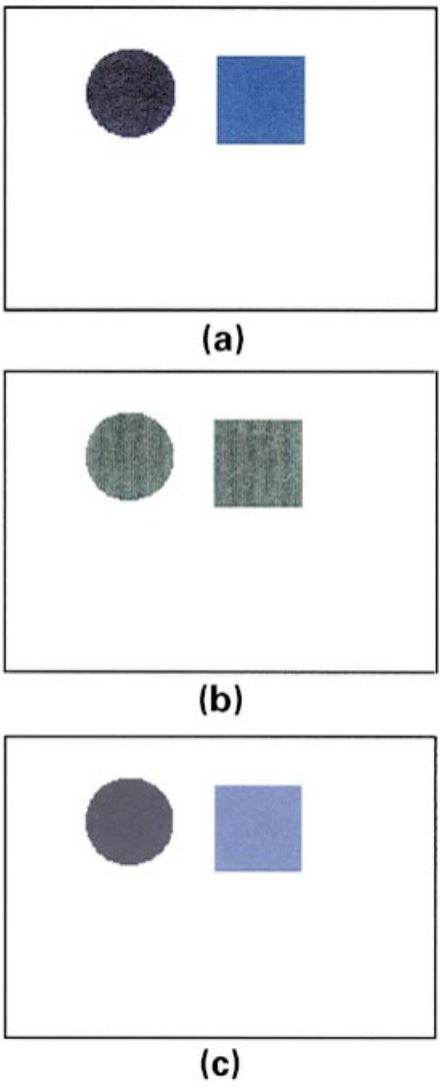

Fig. 10. (a) Original image, (b) corresponding gray-level image, and (c) image after color multithresholding.

Table 5
Application results for an IHS image

Two classes			
Class	I	H	S
1	155	242	153
2	221	7	107
Four classes			
1	109	231	175
2	131	236	155
3	169	244	132
4	215	7	102

vector (30,100,255). By considering the corresponding gray-level images it is impossible to discriminate the two objects as in both cases the same luminance value is computed. However, by applying the proposed technique and through the use of color information, both objects are successfully identified.

6.4. Experiment 4

The proposed technique can be applied directly to any color space. In this experiment, the multithreshold algorithm is applied to a Lenna picture in the Intensity, Hue and Saturation (IHS) color space. The IHS space is one of the most frequently used. This presentation of colors has the advantage of providing an interpretation of the visual result of a particular RGB stimulus. Table 5 and Fig. 11 summarize the results for two and four thresholds.

7. Conclusions

This paper proposes a general multithresholding technique, which is applicable both to gray-level and color images. The proposed technique is based on a neural network structure, which consists of a PCA and a SOFM. The training set of the neural network consists of the image

Fig. 11. Multithresholding of an IHS image.

gray-levels or color values or of a sub-sampling version of them. For gray-level images, the sub-sampling can be performed by using a histogram normalization technique. In the case of color images, the sub-sampling is based on a fractal scanning process. The output competition layer of the SOFM determines the centers of the color classes. In the case of multithresholding of gray-level images, the application of the PCA is optional. In addition, by using a fitting procedure, the optimal number of thresholds can be obtained.

In comparison to the gray-level images, the multithresholding of color images offers better segmentation results. Thus, objects that have different color distributions, but similar gray-level distributions can now be segmented.

The proposed method was successfully tested with a variety of images. The experimental results presented in this paper show the efficiency of the method for gray-level and color image segmentation, compression and interpretation.

References

[1] Y. Ohta, T. Kanade, T. Sakai, Color information for region segmentation, Computer Graphics and Image Processing 13 (1980) 222–241.
[2] O.M. Connah, C.A. Fishbourne, The use of color information in industrial scene analysis, Proceedings of the First International Conference on Robot Vision & Sensory Controls, Stratford-on Avon, UK, 1981, pp. 340–347.
[3] T. Huntsberger, M. Delxazi, Color edge detection, Pattern Recognition Letters 3 (1985) 205–209.
[4] M. Barth, S. Parthasanathy, S. Wang, J. Hu, E. Hackwood, G. Beni, A color vision system for microelectronics: application to oxide thickness measurement, Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, 1986, pp. 1242–1247.
[5] S. Tominaga, A color classification method for color images using a uniform color space, IEEE Int. Conf. Pattern Recognition, New Jersey, 1990, pp. 803–810.
[6] J. Jordan, A. Bovik, Conventional stereo vision using color, IEEE Control Systems Magazine 8 (1988) 31–36.
[7] T. Valchos, A.G. Constantinides, Graph-theoretical approach to color picture segmentation and contour classification, IEE Proc. Part I 140 (1993) 36–45.
[8] R. Schettini, A segmentation algorithm for color images, Pattern Recognition Letters 14 (1994) 499–506.
[9] P.K. Sahoo, S. Soltani, A.K.C. Wong, A survey of thresholding techniques, Computer Vision, Graphics and Image Processing 41 (1988) 233–260.
[10] S. Wang, R.M. Haralick, Automatic multithreshold selection, Computer Vision Graphics and Image Processing 25 (1984) 46–67.
[11] L. Hertz, R.W. Schafer, Multilevel thresholding using edge matching, Computer Vision Graphics and Image Processing 44 (1988) 279–295.
[12] S.S. Reddi, S.F. Rudin, H.R. Keshavan, An optimal multiple threshold scheme for image segmentation, IEEE Trans. on System, Man and Cybernetics 14 (4) (1984) 661–665.
[13] J.N. Kapur, P.K. Sahoo, A.K. Wong, A new method for gray-level picture thresholding using the entropy of the histogram, Compute Vision Graphics and Image Processing 29 (1985) 273–285.
[14] N. Papmarkos, B. Gatos, A new approach for multithreshold selection, Computer Vision Graphics and Image Processing-Graphical Models and Image Processing 56 (5) (1994) 357–370.
[15] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. on System, Man and Cybernetics 9 (1) (1979) 62–69.
[16] C. Strouthopoulos, N. Papamarkos, Text identification for image analysis using a neural network, Image and Vision Computing-Special Issue on Image Processing and Multimedia Environments 16 (1998) 879–896.
[17] P. Sahoo, C. Wilkin, J. Yeager, Threshold selection using Reny's entropy, Pattern Recognition 30 (1) (1997) 71–84.
[18] S. Haykin, Neural Networks: A Comprehensive Foundation, MacMillan College Publishing Company, New York, 1994.
[19] T. Kohonen, Self-organizing Maps, Springer, Berlin, 1997.
[20] H. Sagan, Space-filling Curves, Springer, Berlin, 1994.