Characteristic Colors for Texture Image Analysis: A Parallel Distributed Approach *

J. Scharcanski †
H.C. Shen †
A.P. Alves da Silva ‡
† Department of Systems Design Engineering
‡ Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada, N2L 3G1

Abstract

This paper analyzes the problem of describing texture colors in a concise way. This problem is formulated as a color quantization problem, where the texture image is quantized using the smallest number k of representative colors *given some criteria. These colors are the texture* characteristic colors, *and an algorithm is proposed to obtain them. It is shown that a network of simple processing elements implements a generalization of the algorithm that outperforms it in many cases.*

1 Introduction

Real visual textures are in fact color textures, but in the last decade most of the literature about texture analysis concentrates on the *achromatic* (gray-scale) aspect. Various topics in texture analysis have been intensively discussed, and a great deal of work has been put into selecting features that efficiently represent visual textures [4] [19] [15], but it is still an active research area.

Color textures may present the same colors and different structural patterns, the same structural patterns in different colors, or yet, different colors and structural patterns. Therefore, a texture description should include the *color* and the *structural* aspects. Surprisingly, the literature introduced various methods to represent the *structural aspect* of visual textures, but not much work has been done about the *color aspect.*

In this paper, *characteristic colors* are proposed as a *feature set* to represent the *color aspect* of visual textures. This set contains the smallest number of colors satisfying some criteria, and an algorithm is proposed to obtain it. We also discuss a parallel distributed approach to obtain this feature set. A network is presented that outperforms the algorithm in terms of the color set conciseness, without an appreciable increase in *total quantization error.* This improved performance is attributed to the fact that the network implements a *generalization* of the algorithm, as we detail later.

*This work is supported by the Brasilian National Council of Research and Development (CNPq)

2 A Representation for the Color Aspect of a Texture

The input color texture image itself may be regarded as the *best* representation. Unfortunately it contains excessive data, much of it is often irrelevant to the problem. Usually, colors are *fuzzy* in color space, causing texture images to have ambiguous representations for computer texture classification and discrimination purposes. Some authors proposed to deal with the *color fuzziness* problem using *fuzzy sets* [7] [13], but for practical color texture analysis situations (that may involve hundreds or even thousands of colors) this approach may turn to be of a prohibitive computational cost. Other authors have proposed to deal with uncertainty in color texture images using a *feature filling-in* approach [12] [2]. However, further developments are still necessary for this approach be used in practical problems (e.g. resolution of conflicting color edge information in the *filling-in* process).

In order to minimize the ambiguity in color texture representation, a *representative colors* approach is pursued. The problem of selecting a set of colors to represent the color gamut of an image and compute the mapping from color space to representative colors is known as *color image quantization*, and many approaches have been proposed in the literature [5]. Usually, a number k of representative colors is chosen, and then used for realistic reproductions of color images in *computer graphics.* In order to achieve realistic results in color image exhibition (e.g. smooth contours), some regions of the color space are oversampled compared to the others (i.e. *adaptive quantization*).

For color texture representation purposes, the approaches proposed for color image quantization are not appropriate. In this case, the interest is shifted from display quality to representational aspects (e.g. conciseness). Usually, the number k of characteristic (or representative) colors that produces a concise and informative texture description is not known in advance. It depends of the input pattern and its colors.

In this paper, the smallest representative color set (according to an established criteria) is proposed as a *feature set* in computer color texture representation. The elements of this *feature set* are the *characteristic colors* of the texture. To each one of these colors we associate a *measure of importance*. An algorithm to obtain these *characteristic colors* is presented, and a network is shown to outperform and generalise this algorithm is many situations.

2.1 A Color Texture Image Quantization Problem

The way we approach the problem of selecting a set of k *characteristic colors* to represent the texture colors of a texture pattern is similar to a *color image quantization* problem, with the difference that we do not know in advance the number k of colors to select. But we wish this number to be the smallest possible (given a criterion and constraints). In a later stage, this set of k colors is used to quantise the original texture image into a final image where some texture properties are more easily measured (e.g. the texture elements are more clearly defined when described by fewer colors).

Following a similar notation to [5], let x be an M-dimensional input point (a 3-D color for our purposes). We represent colors in terms of the three *primaries*: red (R), green (G) and blue (B). Therefore, x corresponds to a point in the RGB color space. Let $c_{i,j}$ be the color of the pixel in the original image at row i, column j, where $0 \leq i < NI$ and $0 \leq j < NJ$, NI and NJ are the number of image rows and columns respectively. The color at row i, column j of the final (quantized) image is denoted by $f_{i,j}$.

The color quantisation procedure that we describe consists of:

1. a set of k representative or output points: $Y = \{y_l, l = 1, 2, ..., k\}$;
2. the quantisation function that maps input points into output points: $f_{i,j} = p(c_{i,j})$ $(f_{i,j} \epsilon Y)$.

In our *color selection* problem, Y is the set of *characteristic colors*, k is the number of *characteristic colors* in Y, and p is a mapping from the colors in the original texture image to the colors in the quantised image. Each color x in the quantised image is a *characteristic color* y_l. Notice that the input image is the *true* image, and the image quantised using the *characteristic colors* closely resembles it (based on an established criterion). The input and the final images have the same resolution.

To measure the difference between the original and the quantized images (the total quantisation error), the following formula is used [5] :

$$D = \sum_{i,j} d(c_{i,j}, f_{i,j}) \qquad (1)$$

where $d(x, y)$ is a color metric which measures the difference between colors in the original and the final images. The square of the *euclidean* distance in RGB color space is used:

$$d(x, y) = (x_r - y_r)^2 + (x_g - y_g)^2 + (x_b - y_b)^2 \qquad (2)$$

where $x = (x_r, x_g, x_b)$ and $y = (y_r, y_g, y_b)$. This formula was chosen because is simple and fast to compute.

Given an input image and a set of k representatives (or *characteristic colors*) Y, the output image is obtained by choosing for each input point $c_{i,j}$ its nearest neighhbor in Y, i.e. y_n, to be the output point $f_{i,j}$.

2.2 An Algorithm to Select the Characteristic Colors

The *characteristic colors* are the *smallest set of representative texture colors* according to a specific criteria. We have adopted a criterion for the *representativity* of a color, and another criterion to decide when a certain set of representative colors is the smallest set obtainable.

We assume that two colors are *similar* when their *distance* in color space $d(x_m, x_n) \leq R$. It defines spheric *color tolerance volumes* of radius R (any colors belonging to the same volume are considered *similar*) [20]. A *color tolerance volume* of radius R defines the *vicinity* R of a color in color space.

The *representativity* of colors is decided locally in the color space, and the most important (or frequent) color within its vicinity R is taken as the *vicinity's representative*. If to each color is associated a *vicinity* R, depending on the color we start with, different representatives may be obtained. Unless we establish an *initialization criterion*, the obtained results are not reliable. Therefore, we priorise the densest regions in color space and choose the most frequent colors as the best candidates to start the quantisation process.

If to each *vicinity* corresponds a unique representative, the larger the *vicinity* R, the smaller number of representatives. But at the same time, the larger the *vicinity* R the larger the *total quantization error* D. We propose to use $R = 2^{b_{max}-b}$, where b_{max} is the *maximum number of bits* available for each *channel* (i.e. red, green and blue), and b is the *number of bits* effectively used. Usually, b_{max} is pre-defined. Most equipments use 24 *bits/color*, or 8 *bits/channel*. However, b depends on the application. Some applications require less than b_{max} *bits/channel* per *pixel* in terms of *color resolution*. This criterion says explicity to adopt the *smallest* meaningful R value, given that only b *bits/channel* are being used in each *pixel* (i.e. the least significant $(b_{max} - b)$ *bits/channel* are set *null*).

Therefore, the smallest set of representative colors, given the above criteria and constraints, are the texture *characteristic colors*. Next, an algorithm is described to select these colors.

The proposed algorithm is inspired by the *popularity algorithm* for color image quantisation [5]. The *popularity algorithm* chooses the k' most frequent (or *popular*) colors in the image as the set of representatives, where k' is a number defined in advance. Our algorithm uses the concept of *color tolerance volumes* to find the smallest number k of representative colors.

This algorithm chooses the smallest number k of the most frequent colors that satisfy simultaneously the following criteria: *a)* they are at distance more than R from each other in the *RGB space*; *b)* they correspond to *centers* of a set of *color tolerance volumes* of radius R, covering the entire color gamut presented by the input texture image.

1. for each color x_n compute its *relative frequency* $\overline{f}(x_n)$:

$$\overline{f}(x_n) = \frac{f(x_n)}{\sum_{m=1}^{N} f(x_m)} \quad (3)$$

where $n = 1, ..., N$ and N is the number of colors in the input texture image;

2. sort the colors in decreasing order based on $\overline{f}(x_n)$;

3. for $n = 1, ..., N$
for $m = 1, ..., N$
if $(m \neq n)$ and $(d(x_n, x_m) \leq R)$
then *discard* color x_m $(\overline{f}(x_m) = 0)$;

4. for $n = 1, ..., N$
if $\overline{f}(x_n) > 0$ then *include* x_n in the set of *characteristic colors*;

5. quantize the input image using the *nearest neighbour* criterion;

2.3 Measure of Importance of the Characteristic Colors

The *measure of importance* is a feature that complements the *characteristic colors* in the texture representation.

The *relative frequency* of a *characteristic color* $\overline{f}(y_i)$ in the final image is taken as its *measure of importance* $MI(y_i)$. Therefore *characteristic colors* that are more frequent, are more important in the texture *color aspect*.

3 A Parallel Distributed Approach

Our proposed parallel distributed approach consists of a network of competitive linear units. This network operates continuously in time, and its organisation and dynamics are described next.

The human color vision system has been subject of intensive studies, and several researchers contributed to improve our understanding of its intricate mechanisms [3] [14] [16] [20]. Zeki [21] has provided evidence that true colors perceived in visual stimuli have a corresponding code in the visual cortex (i.e. cells *tuned* to specific wavelength compositions). Therefore, it looks sensible to conceive a collection of *units* corresponding each one to a color (i.e. an (r, g, b) triplet). These units are *topographically* organised according to the colors they represent [8]. For simplicity, we suppose an organization where an unit u_n is an immediatelly neighbor of an unit u_m if they correspond to colors that are immediatelly neighbors in the *RGB* color space.

The role of the network is to determine a reduced set of colors to describe the *color aspect* of the input pattern. The network units interact with their vicinity in a center-surround fashion. The *initial state* of this network depends of the input color distribution, and the obtained *characteristic colors* are associated with the k *winning units* when the network achieves steady state.

3.1 Unit Activation

An unit u_n (*coding* a color x_n) has its activation given by I_n, whose intensity depends on the presence of the color x_n within its *receptive field* [10] (the texture image).

$$I_n = \frac{f(x_n)}{\sum_{p=1}^{N} f(x_p)} \quad (4)$$

where $f(x_n)$ denotes the frequency of occurrence of x_n in the *unit receptive field*. The unit u_n is *activated* if $I_n > 0$.

Let us assume that for each unit u_n we have an input matrix ψ_n, whose elements $\psi_n(i, j)$ are binary-valued:

$$\psi_n(i,j) = \begin{cases} 1 \text{ if } c_{i,j} = x_n \\ 0 \text{ otherwise} \end{cases} \quad (5)$$

In the case of a single *Hebbian* linear unit, the unit activation I_n^h is [6]:

$$I_n^h = \eta \sum_{i,j} w_n(i,j)\psi_n(i,j) \quad (6)$$

This network operates in continuous time. We adopt the simplifying assumption that changes in connection weights are negligible enough, with respect to the change in unit excitation values, that weights can be considered constant [1].

Therefore, the unit activation I_n (Equation 4) is a particular case of the *Hebbian* linear rule where $w_n(i, j) = 1$ and it is used a multiplicative *normalization constant* $\eta = \frac{1}{NI \times NJ}$ (i.e. $\eta = \frac{1}{\sum_p f(x_p)}$).

3.2 Unit Lateral Interactions

In this network, each active unit interacts with its neighborhood by *exciting* the immediately neighboring units with *strength* e and *inhibiting* the next-to-immediate neighbor units with *strength* h. Both, excitatory and inhibitory connection strengths diminish as a function of the distance to unit u_n. These units keep interacting until convergence is achieved, and the network settles down in a stable state. They are named *lateral* (or *on-center off-surround*) interactions, and have been studied before within various contexts [9] [10] [1] [6].

Without loss of generality, let us analyse the interaction of a single unit u_n (with activation I_n) laterally with its neighborhood. And, let us define the neighborhood (or vicinity) of unit u_n as all the units spatially located within a *distance* R. These *interactions* are described by:

$$I_n' = \begin{cases} I_n - \sum_{p=1}^{U} I_p r_{np} & \text{if } I_n' \geq \epsilon_n \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

where I_n' denotes the new unit activation level, and the coefficient r_{np} equals to r in the vicinity of radius R of the unit u_n (i.e. within the vicinity R the neighbouring active units *inhibit* u_n, and the units outside the vicinity of radius R have no influence over the unit u_n). The threshold ϵ determines when u_n is in steady state. We adopted $\epsilon_n = \epsilon$ $(\forall n,\ n = 1, 2, ..., U)$, where ϵ is in $(0, 1]$ and U is the number of network units $(U = 2^{3b_{max}})$.

It is a special case (*lateral inhibition* [6]) of the general *on-center off-surround* interactions described in [10] (i.e. in the equation: $I_n' = \sum_{p \neq n} I_p s_{np} - \sum_p I_p r_{np} + \sum_p I_p z_{np}$, we have assumed the coefficients $s_{np} = 0$, $z_{np} = 1$ when $n = p$ and $z_{np} = 0$ anywhere else).

3.3 Effective Vicinity of an Unit

The activation of an unit u_n is influenced by units located even beyond its *vicinity* R. In a certain moment in time, the unit activation I_n depends of the unit u_n and the activations I_p of the *vicinity* R units (u_p). Without loss of generality, let us consider a small time increment Δt. The unit activation $I_n(t)$ may be described by:

$$I_n(t) = I_n(t - \Delta t) - r \sum_{p=1}^{Q} I_p(t - \Delta t) \qquad (8)$$

where Q is the number of units in the *vicinity* R of an unit u_n. However, the activation of each neighboring unit u_p depends on itself and also on its own *vicinity* R units (u_q):

$$I_p(t - \Delta t) = I_p(t - 2\Delta t) - r \sum_{q=1}^{Q} I_q(t - 2\Delta t) \qquad (9)$$

Substituting the above expressions and changing subscripts, we obtain the following expression for the activation of u_n:

$$I_n(t) = I_n(t - \Delta t) - r \sum_{p_1=1}^{Q} (I_{p_1}(t - 2\Delta t)$$

$$- r \sum_{p_2=1}^{Q} (I_{p_2}(t - 3\Delta t) - ... - r \sum_{p_U=1}^{Q} I_{p_U}(t - (U+1)\Delta t)...) \qquad (10)$$

where the u_{p_1} units are in the *vicinity* R of u_n, an u_{p_2} unit is in the *vicinity* R of an u_{p_1}, and so on. It means that, in a certain moment in time, the activation I_n depends on the unit u_n and on the previous activations of large number of units. The unit lateral interactions eventually involve all units, but the further away an unit is, the smaller its influence. In practice, we may consider an *effective vicinity* which influences an unit u_n. It is larger than *vicinity* R, and its size depends on r. Notice that the network behavior described above holds in continuous time (i.e. in the limit $\Delta t \to 0$).

Therefore, the network *changes* (or *generalizes*) the concept of a well defined *vicinity* R around an unit u_n, to a larger *effective vicinity* not so well defined. As a consequence, the results obtained with the network differ from those obtained by the algorithm, where the *vicinity* R is used strictly.

3.4 Network Stability and Convergence

We would like to make some comments about the network *global stability* (i.e. the eventual stabilization of all unit activations from any initial input) and *convergence* (i.e. the eventual minimization of error between the desired and the computed unit outputs) [18].

Let us assume that inputs are stable patterns, then the dynamic behavior of a unit u_n in this network is described by:

$$\Delta I_n = I_n' - I_n = -\sum_{p=1}^{U} I_p r_{np} \qquad (11)$$

Since r_{np} is an element of a symmetric matrix with nonnegative values, and I_p only assumes positive values, the activation of a single unit decreases assymptotically as the *lateral interactions* progress in time. Therefore, for any arbitrary input pattern the network is *stable* (i.e. it eventually achieves steady state with a negligible deviation ϵ, see Equation 7).

It is quite intuitive that the computation done by this network *converges* to the high activation units within a vicinity R. When $I_n' \geq \epsilon$, Equation 7 may be written as:

$$I_n' = I_n - r \sum_{p=1}^{Q} I_p \qquad (12)$$

and the computation converges to the unit with higher activation level (i.e. the unit corresponding to the most frequent color within the vicinity R). Notice that the convergence may terminate, and instead of a single unit, a set of units with high activation get *selected*. The smaller the value of r, the higher the chance of a unique state termination (i.e. the smaller r, the smaller differences in activation levels among units that may be discriminated), on the other hand the convergence time gets longer.

After the network has achieved *steady state*, a set of representative colors have been selected (given a certain input pattern and vicinity R). Each color is associated to one of the remaining active units v_m $(m = 1, 2, ..., V$ and $V \leq U)$. These are the texture *characteristic colors*. Given this color set, the input pattern (color texture image) is *recoded* (or *quantized*), and a final pattern is obtained.

3.5 Pattern Recoding

Each color $c_{i,j}$ in the input spatial pattern is *quantized* by a color v_m using the *nearest neighbor* criterion, i.e. the color v_m that is located at the smaller distance from $c_{i,j}$ $(d(c_{i,j}, v_m))$ in the RGB space is its *nearest neighbor*. This computation may be carried out by a network that at each point $c_{i,j}$ of the input pattern minimizes the energy function:

$$E(m) = d(c_{i,j}, v_m) \ , \ m = 1, 2, ..., V \qquad (13)$$

At the end of this computation it is obtained a *final image* (i.e. the quantised version of the input image). The *measure of importance* of each *characteristic color* corresponds to the *relative frequency* of this color in the *final image* (it is analogous to Equation 4).

4 Experimental Results

A texture image with resolution 256×256 is used to demonstrate the effectiveness of our approach for texture colors representation. In these experiments it was used $b = 4$ (i.e. a pallette with 4,096 colors). For the network simulation the following parameter setting was used: $r = 10^{-3}$ and $\epsilon = 10^{-6}$.

The original image presented 162 colors (i.e. (r, g, b) triplets). The algorithm described in Section 2 selected 66 *characteristic colors*, and produced a *total quantization error* $D_{alg} = 8.40$. The network selected 24 *characteristic colors*, and produced a *total quantization error* $D_{nn} = 10.21$. Figure 1 shows the original and the quantised images. Notice that the quantised images closely resemble the original. The network selects a more concise set of *characteristic colors* without an appreciable increase in quantisation error.

The results obtained by the network and a conventional unsupervised clustering method (*k-means* algorithm) were compared. The *k-means* represents the occurring colors in each image by k *cluster centers* (prototypes), where k was made equal to the number of *characteristic colors* selected by the network. The *k-means* data produced an *average quantization error* $\overline{D}_{km} = 12.65$. Therefore, the network performed better than a conventional unsupervised clustering method (*k-means*) under the same circunstances, presenting the smaller *average quantization error*.

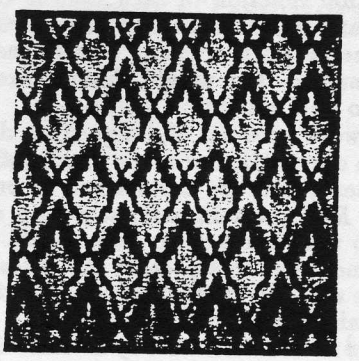
In our experiments we found a large variation in execution time depending on the image of interest. The algorithm described in Section 2 executed from 6.6 to 43.2 times faster than the sequential simulation of the network. However, a parallel distributed implementation of this network is expected to reduce its execution time drastically.

5 Concluding Remarks

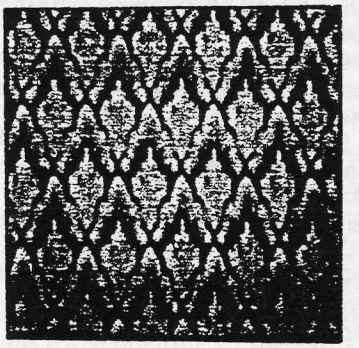
It has been proposed a *feature set* to represent concisely the *color aspect* of visual textures. This *feature set* is constituted by the texture *characteristic colors* and their respective *measure of importance*.

In order to obtain these *characteristic colors*, an algorithm inspired by the *popularity algorithm* for color image quantization [5] was proposed. This algorithm uses the concept of *color tolerance volumes* to find the smallest number k of representative colors that satisfy simultaneously the following criteria: a) they are at distance of at least R from each other in *RGB space*; b) they correspond to *centers* of a set of *color tolerance volumes* of radius R, covering the entire color gamut presented by the input texture image.

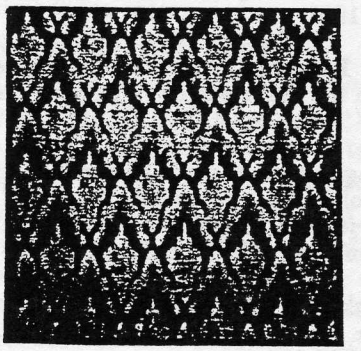
The proposed parallel distributed approach consists of a network of competitive linear units. Each color triplet (r, g, b) in *RGB* space corresponds to one unit.



(a) Original image



(b) Quantised image from the algorithm



(c) Quantised image from the network

Figure 1: Original and Quantized images

The activation function of each unit is the *Hebbian linear rule*, and the units interact in a *lateral inhibition* fashion. The network operates continuously in time, and it is assumed that weight changes are neglegible compared to unit activation values.

This network executes the algorithm described in Section 2 in one particular mode of operation [17]: *a)* the unit interactions are constrained to one *spot* (i.e. the *vicinity R* of a certain unit) at a time, and outside this *spot* no unit interactions take place; *b)* the *spot* moves in time from the *highest* to the *lowest* activation units; *c)* the *spot* only moves from one point to another when the unit interactions stabilise.

When the network units freely interact using *lateral inhibition*, the concept of a well defined *vicinity R* is changed (or generalised) to a larger *effective vicinity* whose boundaries are not precisely defined. Therefore, we can say that the network generalises the algorithm in the sense that it enables units to interact in a larger *vicinity*, and it allows overlapping *vicinities* to interact with each other.

The experimental results showed that the network outperforms the algorithm in terms of conciseness of the *characteristic colors* set, without an appreciable increase in the total quantisation error. Visually, the images *quantized* using the *characteristic colors* closely resemble their originals.

An extension of this work is to consider the weight changes in the *Hebbian units*. The application of other parallel distributed methods to this problem should also be investigated. These issues will be addressed in a future paper.

References

[1] M.A. Cohen and S. Grossberg. Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks. *IEEE SMC.* **13**, 1983, 815-825.

[2] J. Davidoff. *Cognition Through Color.* Ed. MIT Press,1991.

[3] O. D. Faugeras. Digital Color Image Processing within the Framework of a Human Visual System. *IEEE Trans. on ASSP*, **27**, 1979, 380-393.

[4] A. Gagalowicz. Visual Discrimination of Stochastic Color Texture Fields in *Signal Processing: Theories and Applications* (M. Kunt and F. de Coulon, Ed.), North-Holland, 1980.

[5] P. Heckbert. Color Image Quantisation for Frame Buffer Display. *Computer Graphics.* **16**, no. 3, 1982, 297-307.

[6] J. Hertz, A. Krogh and R.G. Palmer. *Introduction to the Theory of Neural Computation.* Ed. Addison-Wesley, 1991.

[7] J. Hsu and T. Wei. Classification of Colors through Fuzzy Statistical Experiment. *Color Research and Application.* **14**, no. 2, April 1989, 64-68.

[8] T. Kohonen. The Self-organising Map. *Proceedings of the IEEE.* **78**, 1990, 1464-1480.

[9] S. Grossberg. On the Development of Feature Detectors in the Visual Cortex with Applications to Learning and Reaction-Diffusion Systems. *Biological Cybernetics.* **21**, 1976, 145-159.

[10] S. Grossberg. Adaptive Pattern Classification and Universal Recoding: I. Parallel Development and Coding of Neural Feature Detectors. *Biological Cybernetics.* **23**, 1976, 121-134.

[11] S. Grossberg. Adaptive Pattern Classification and Universal Recoding: II. Feedback, Expectation, Olfaction, Illusions. *Biological Cybernetics.* **23**. 1976, 187-202.

[12] S. Grossberg. *Cortical Dynamics of Three-Dimensional Form, Color and Brightness Perception, I: Monocular Theory* in **Neural Networks and Natural Intelligence**, by Stephen Grossberg (ed.). Ed. MIT Press, 1988.

[13] Andrew E. K. Pienkovsky. *Artificial Colour Perception Using Fuzzy Techniques in Digital Image Processing.* Ed. Verlag TUV Rheinland, 1989.

[14] M. Livingstone and D. Hubel. Segregation of Form, Color, Movement, and Depth: Anatomy, Physiology, and Perception. *Science*, **240**, May 1988, 740-749.

[15] A. R. Rao. A taxonomy for Texture Description and Identification. Ed. Springer-Verlag, New York, 1990.

[16] D. Rose and V.G. Dobson. *Models of the Visual Cortex.* John Wiley and Sons, 1985.

[17] J. Scharcanski, H.C. Shen and A.P. Alves da Silva. Color Quantisation for Color Texture Analysis. Technical Report PAMI-TR-004-91. University of Waterloo, Waterloo, Canada.

[18] P. K. Simpson. *Artificial Neural Systems.* Ed. Pergamon Press, 1990.

[19] F. Tomita and S. Tsuji. *Computer Analysis of Visual Textures.* Kluwer Academic Publishers, 1990.

[20] G. Wyszecki and W.S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae,* John Wiley and Sons, 1982.

[21] S. Zeki. The Representation of Colours in the Cerebral Cortex.*Nature.* **284**, 1980, 412-418.