Texture Synthesis using Soft-Computing

Javier Ruiz-del-Solar, Patricio Parada

Dept. of Electrical Engineering
Universidad de Chile
Email: {jruizd,pparada}@cec.uchile.cl

Mario Köppen

Dept. of Pattern Recognition
Fraunhofer-Institut IPK Berlin
Email: mario.koeppen@ipk.fhg.de

Abstract

The TEXRET-System, a texture retrieval system based on soft-computing technologies is being developed. One of the main system features is synthesis of the requested textures when these are not found in the database, which allows a growing of the database. Missing textures are synthesized interactively using Markov Random Fields and interactive genetic algorithms. This article is centered on the texture synthesis of the textures

1. Introduction

Textures are homogeneous visual patterns that we perceive in natural and synthetic scenes. They are made of local micropatterns, repeated somehow, producing the sensation of uniformity. Texture perception plays an important role in human vision. It is used to detect and distinguish objects, to infer surface orientation and perspective, and to determine shape in 3D-scenes [9]. An interesting psychological observation is the fact that human beings are not able to describe textures clearly and objectively, but only subjectively by using a fuzzy characterization of them. On the other hand, with the new advances in communication and multimedia computing technologies, accessing mass amounts of digital information (image databases) is becoming a reality. In this context, textures, due to their esthetical properties, play today an important role in the consumer-oriented design, marketing, selling and exchange of products and/or product information. For this reason, systems that allow the search and retrieval of textures in image databases are of increasing interest [12].

This work is part of a main research effort, whose aim is the construction of the TEXRET-System, a texture retrieval system based on soft-computing technologies. The TEXRET-System has the following features: (i) direct access from the Internet, (ii) texture queries using human-like or fuzzy description of the textures, and (iii) synthesis or generation of the requested textures when these are not found in the database, which allows a growing of the database. This article is centered on this last feature of the system.

Texture Synthesis has been an increasingly active research field in computer graphics. Many different approaches have been used to generate textures, but till now no fully successful generation model has been found. Among the most interesting models we can distinguish structural models, reaction-diffusion like models and probabilistic models. In this last group, Markov random fields stands out for their large versatility and performance. For this reason, in this work Markov random fields are chosen to implement the generation of textures.

The article is structured as follows. The TEXRET-System is outlined in section 2. In section 3 and 4, the system modules dealing with the synthesis of textures are described. Finally, in section 5 and 6 some preliminary results and conclusions are given.

2. The TEXRET-System

The TEXRET-System, whose block diagram is shown in figure 1, is made of the *FI* (Fuzzy Interface), the Q^2TPT (Qualitative to Quantitative Textural Properties Transformation), the *TR* (Texture Retrieval), the *TG* (Texture Generation), and the *EPA* (Evolutionary Parameter Adjustment) modules.

The on-line phase of the texture retrieval process works as follows: A human user makes a query of a texture using a subjective, linguistic or human-like texture description. The FI module enters this description into the system using a fuzzy representation of it. The Q^2TPT module interprets the query and translates it into a quantitative texture description that is implemented using Tamura Descriptors [15]. This qualitative description is used by the TR module to search the texture in the database (see description in [12]).

In the case that the texture is not found in the database, the user can choose the automatic generation of it. The TG module generates the texture using Markov Random Fields (MRF) [1] [2] [3] [4] [5] [6] [16]. The parameters of the MRF are calculated from the Tamura descriptors and then the textures are generated. As a result of this generation process a set of textures is presented to the user. If the user considers that one of the generated textures satisfy his query, the process finishes here. If not, the user enters into an iterative process. The iterative generation of the textures is implemented using interactive evolutionary computation (EPA module).

It should be pointed out that the subjective or human-like texture description that the system accepts, was determined by a psychological study in texture perception, performed by co-workers of the authors, and that will be presented elsewhere.

In the next two sections the modules dealing with the generation of textures are presented.

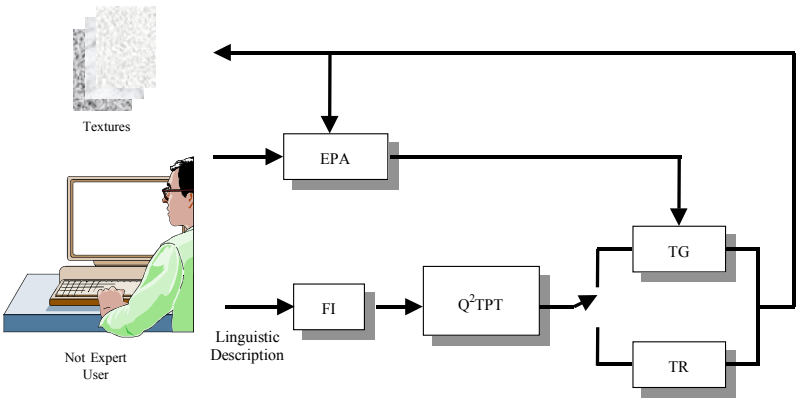

Figure 1. Block Diagram of the Proposed System.

3. The FI and Q^{2}TPT Modules

The function of the Q^{2}TPT module is to interpret subjective textural properties and to translate them into objective features. The FI module allows obtaining an internal representation of these subjective textural properties using fuzzy logic. The Q^{2}TPT is implemented by using a Neuro-Fuzzy architecture, whose training phase is shown in Figure 2.

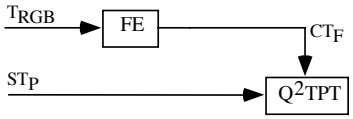

Figure 2. Training-phase block diagram of the Q^{2}TPT module.

3.1. Neuro-Fuzzy Network

The implemented architecture corresponds to a modified variant of the Khan network [8]. This network is based on a variation of the Multilayer Back-Propagation Network (MBPN). In the original description it consists of seven layers, which also do fuzzification and rule approximation. In the modified variant, the product-sums of the MBPN formulas are replaced by multiplications and the learning rule is modified appropriately (see a detailed description in [9]).

3.2. FE (Features Extraction)

As it can be seen in Figure 2, the extraction of Color-Textural Features (CT_F) from textures (textured images) is a fundamental step in the training of the Q^{2}TPT module, and also in the construction of the texture database. This process is performed by the FE module. The features extraction process works as following (see block diagram in figure 3):

- First, the input texture is transformed from its original RGB color space representation into a HSL (Hue-Saturation-Luminosity) color space representation by the **R2H** (RGB to HSL) module. This transformation is performed because the representation of the colors in the RGB space is quite adapted for monitors but not for human beings, like the HSL.
- Then, Color (C_F) and Textural Features (T_F) are extracted from the HSL images (T_H, T_S and T_L) by the **CFE** (Color Features Extraction) and **TFE** (Textural Features Extraction) modules.
- Finally, both of them, Color and Textural Features, are fused by the **FF** (Features Fusion) Module.

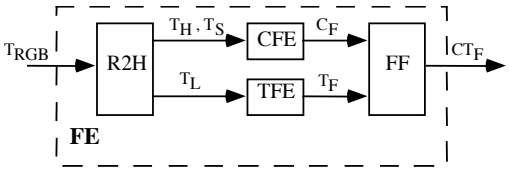

Figure 3. Block diagram of the FE.

3.2.1 CFE Module

This module is implemented by using color histograms of the Hue and Saturation components of the HSL texture representation (T_H and T_S). From each component histogram, the *mean* and the *standard deviation* values are taken as color features. Color histograms are chosen because they are invariant to translations and rotations about the viewing axis, and they change only slowly under modifications of angle of view, changes in scale, and occlusion [14].

3.2.2 TFE Module

This module is implemented by using modified Tamura Features [15], which correspond to standard features for texture description. We used all six Tamura Features, *coarseness*, *directionality*, *contrast, line-likeness*, *regularity* and *roughness*, but we modified the calculation process of first three ones.

The calculation of *coarseness* is performed by using the *Minimal Binary Representation* [10] and the

Morphological Distance Transformation. On the calculation of the *directionality* and *line-likeness* features, we used the original method of Tamura but we changed the gradient he used by a fuzzy definition of the gradient [11].

3.2.3 FF Module

While some color features are useful features for texture search, other color features influence the psychological evidence of textural features. For example, the subjective impression of contrast is influenced by background color. Contrasts on blue or green backgrounds appear to be different, even if the contrasts measured from the intensity image are equal. A useful technique for feature extraction is to extract or synthesize new features from the whole set of color and textural features. In this case, a weighted-sum approach is not sufficient due to the fact that textural features are differently influenced by color information. For this reason, this module is implemented using the fuzzy integral [13].

4. The TG and EPA Modules

4.1. TG (Texture Generation)

Texture Generation has been an active research area in computer graphics. Among many generation methods, as for example structural and reaction-diffusion-like ones, methods considering textures as samples from probabilistic distributions are of increasing interest. By determining the form of these distributions (i.e. the model), textures can be generated. The performance of the methods depends on the structure of the probabilistic density estimator being used. In this context, Markov Random Fields and autoregressive models have been successfully used to the generation of textures. For this reason, these methods are chosen to implement the TG module. Nevertheless, at the moment the authors are testing high-order statistical methods to improve the generation results. All mentioned methods are briefly described in the following subsections.

4.1.1. Markov Random Fields

The study of Markov random fields has had a long history, beginning with studies on ferromagnetism at the beginning of the last century. The model has been applied to the case of binary or Gaussian variables on a lattice. Extensions to the case of variables that have integer ranges, together with the use of estimation procedures, allow the application of the Markov random field to texture modeling. The following formulation has been taken from Cross and Jain [3].

Let $X(i,j)$ be the pixel value at a point (i,j) on a $N \times N$ lattice L. For simplicity let use $X(i), i = 1,2,\ldots,M$ where $M = N^2$.

Definition 1: The point j is said to be a neighbor of the point i if $p(X(i) \mid X(1), X(2), \ldots, X(i-1), X(i+1), \ldots, X(M))$ depends on $X(j)$. It should be pointed out that this definition does not imply that the neighbors of a point are necessarily close in terms of distance, although this is the usual case.

Definition 2: A *Markov random field* (MRF) is a joint probability density on the set of all possible X of the lattice L, subject to the following conditions:

1) Positivity: $p(X) > 0$ for all X.

2) *Markovianity:*
$p(X(i) \mid \text{all point in } L \text{ except } i) = p(X(i) \mid \text{neighbors of } i)$

3) *Homogeneity:* $p(X(i) \mid \text{neighbors of } i)$ depends only on the configuration of neighbors and is translation invariant.

In most cases, we are interested in the models where the point i is a neighbor of the point j if i is close to j. Nevertheless, the influence of the neighborhood depends on the probability function model that has been chosen.

Definition 3: The *order* of a Markov random field process ($O(N)$) on a lattice is the largest value of i such that $b(i,1)$ or $b(i,2)$ is nonzero ($b(i,1)$, $b(i,2)$ parameters of the Markov model).

Definition 4: A Markov random field is *isotropic at order i* if $b(i,1) = b(i,2)$. Otherwise, it is said to be *anisotropic at order i*.

Autobinomial-MRF Model

In this model, the probability of a point $X(i,j)$ having a gray level k is binomial, with parameter T determined by its neighbors $\theta(T)$:

$$\theta(T) = \frac{\exp(T)}{1+\exp(T)} \tag{1}$$

where a first-order model has the form:

$$T_1 = a + b(1,1)(t+t') + b(1,2)(u+u') \tag{2}$$

and a second-order model the form:

$$T_2 = T_1 + b(2,1)(v+v') + b(2,2)(z+z') \tag{3}$$

Additional high-order terms can be obtained by extending the orders in a similar way beyond those shown in figure 4.

Gaussian-MRF Model

In this case, pixel gray levels have a joint Gaussian distributions and correlation controlled by a number of parameters representing the statistical dependence of a pixel value on the pixel value in a symmetric neighborhood. The probability density function can be written as follows:

$$p(X(i) \mid X(i+r), r \in N) =$$

$$\frac{1}{\sqrt{2\pi v}} \cdot \exp\left(-\frac{1}{2v} \left[X(i) - \sum_{r \in N} \theta_r X(i+r) \right] \right) \quad (4)$$

		m		
	v	u	z	
l	t	X	t'	l'
	z'	u'	v'	
		m'		

Figure 4. Neighbors of the point X.

4.1.2. Autoregressive-Model

This is another statistical approach for texture modeling. Autoregressive-models can be either causal or non-causal [2]. Both of them consider that the gray level $X(i,j)$ at the pixel (i, j) is a linear combination of the gray levels at the neighborhood (N), and additive white noise which can have an arbitrary density of probability. Specific restrictions on the neighbor set N yield familiar representations in the image processing literature. For instance, the "causal" models are obtained when the gray level of the pixel (i, j) consider the terms "before" this pixel, i. e., the "earlier" values only. As an example, the work of Hering [7] considers the following dependence (see figure 5):

$$X(i,j) = a_{l_0} X(i-1, j-1) + a_0 X(i-1, j) + a_{r_0} X(i-1, j+1) + a_l X(i, j-1) + \sigma(i,j) \quad (5)$$

$(i-1, j-1)$	$(i-1, j)$	$(i-1, j+1)$
$(i, j-1)$	(i, j)	

Figure 5. Neighborhood considered in Hering's model [7].

$(i-K_l, j-L_l)$	$(i-K_l, j-1)$	$(i-K_l, j)$	$(i-K_l, j+1)$	$(i-K_l, j-L_r)$
$(i-2, j-L_l)$	$(i-2, j-1)$	$(i-2, j)$	$(i-2, j+1)$	$(i-2, j-L_r)$
$(i-1, j-L_l)$	$(i-1, j-1)$	$(i-1, j)$	$(i-1, j+1)$	$(i-1, j-L_r)$
$(i, j-L_l)$	$(i, j-1)$	(i, j)		

Figure 6. Arbitrary neighborhood for causal Autoregressive–Models.

This model can be extended by considering more pixels, like in the following formula:

$$g(i,j) = \sigma(i,j) + \sum_{k=0}^{N_k} \sum_{l=L_l}^{L_r} a(i-k, j-l) \cdot g(i-k, j-l) \quad (6)$$

where the neighborhood N is show in the figure 6.

4.1.3. Other Statistical Models

The models that had been shown has concentrated on exploitation of only second-order statistics of the data either explicitly by restricting attention to correlation properties of the random field, or implicitly by assuming that the random field is Gaussian. A consequence of this is that the impulse response of the underlying parametric model must possess certain symmetry (such a "symmetric non-causality"), in order to achieve parameter identifiability.

Recently has been interest in exploiting the high–order cumulative statistics of the random field, in addition to or in lieu of the usual second–order statistics. In [16] the authors develop a method that exploits both the second–order and the third–order correlations of the observed random field. The basic assumption is that the observed random field has been generated by driving 2D–ARMA (Autoregressive Moving Average) model by an i.i.d. non-gaussian sequence. They presented and analyzed a new family of inverse filter criteria for consistent estimation of the parameters of a 2D–ARMA model (with minimum phase or mixed phase, causal or non-causal) of a non-gaussian random field. The criteria are potentially useful for image representation and analysis, deconvolution, inverse as well as direct system modeling, image (texture) classification and hypothesis testing.

One major disadvantage of MRFs is that, in general, an explicit form of the joint probability of the random variables describing the model is not obtainable. However, a popular subclass of MRFs, called *Markov Mesh Models* (MMMs), allows the explicit description of the joint probability in terms of spatially local conditional probabilities. In [4] it is shown how *Partially-Ordered Markov Models* (POMMs) are a generalization of MMMs and it is demonstrated the versatility of POMMs to texture synthesis and pattern recognition in imaging.

Table 1. Models and their parameters.

Model	Number of Parameters	Name of Parameters
Autobinomial MRF	$2\,?O(N)$	$b(i,1)$ y $b(i,2)$
Gaussian MRF	$2?O(N),\quad O(N) \le 3$ $2+2?O(N),\ 4 \le O(N) \le 7$	θ_r
Autoregressive - Model	4	a_{l0}, a_0, a_{r0}, a_l
	5 – 18	$a(i-k, j-l)$

4.2. EPA (Evolutionary Parameter Adjustment)

The EPA (Evolutionary Parameter Adjustment) module allows the user to perform an interactive adjustment of the texture-generation model parameters by using (interactive) genetic algorithms. Interactive genetic algorithms can be defined as genetic algorithms that do not use a standard fitness function but a user's choice as fitness criterion.

In the here proposed system (see block diagram in figure 1) a set of generated textures are presented to the user at each iteration. The user selects a subset of textures that better achieve his requirements. This information is used to adjust the parameters of the texture-generation model. The system and the user iterate until the required texture is generated.

As usual, the parameters of the model are packed into a chromosome (array), whose size depend on the texture-generation model being used (see table 1). As an example, when the Gaussian model is being used, the chromosome looks like the one shown in figure 7.

θ_1	θ_2	θ_3	.....	θ_{10}

Figure 7. The chromosome structure corresponding to the Gaussian MRF model, with a neighborhood order of 4.

All real-valued parameters are encoded into binary values using interval numbering. From such bitstrings, 20 textures are synthesized and presented to the user. The user, if not satisfied by any of the presented textures, can select the five textures that come most close to his needs. This is the only interactive part in the evolutionary procedure, and replaces the usual fitness-proportionate selection. By selecting the texture images, the corresponding bitstrings are selected as well. From these five bitstrings, 20 new bitstrings are created by crossover and mutation operators. For crossover, two of the five bitstrings are randomly selected, and one-point crossover is performed. The new bitstrings are mutated as well, by flipping their bits with a low probability (<0.01). Then, the 20 new texture images are generated and presented to the user, who again could either stop or select the best five textures. The whole procedure is repeated as long as the user wishes.

For user convenience, a reject feature is also offered, which allows to drop the newly created bitstrings and to restart the operations for creating the next generation.

5. Preliminary Results

The texture images shown in figure 8 were generated using the Metropolis Algorithm [3], which corresponds to a very known implementation of autobinomial-MRF. As it can be seen from the images, the results are very good for some kind of textures like lines and clouds. Nevertheless, this algorithm is relative slow, compare with the others, because it need to perform a big number of iterations before reaching a stable texture image. On the other hand, the algorithm performance depends on the number of graylevels of the image and the calculations should be done with a variable internal precision of the numbers (256 to 1024 bits in floating point representation).

The texture images shown in figure 9 were generated using the Gaussian-MRF algorithm proposed by Chellappa [1], which has some advantages relatives to the Metropolis Algorithm. The calculation of the texture image is performed in only one iteration, the size of the image has only a small importance in the processing time, and the algorithm is independent of the number of graylevels of the image. Additionally, as it can be seen in figure 9, the generated images show a greater variability than the generated using the autobinomial model.

The images generated with the causal autoregressive model (see figure 10) have a similar characteristic: all of them are like lines, i.e. all images have a similar linear pattern. The values of the neighborhood parameters allow changing the orientation of the image and the medium gray level, but all the textures are rather similar.

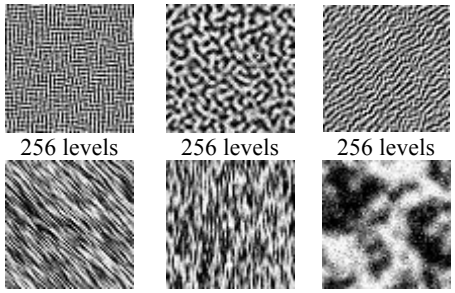

Figure 8. Textures generated using the Metropolis Algorithm. The size of the images is 64 x 64.

Figure 9. Textures generated using the Chellappa algorithm. The number of graylevels is 256 in all images.

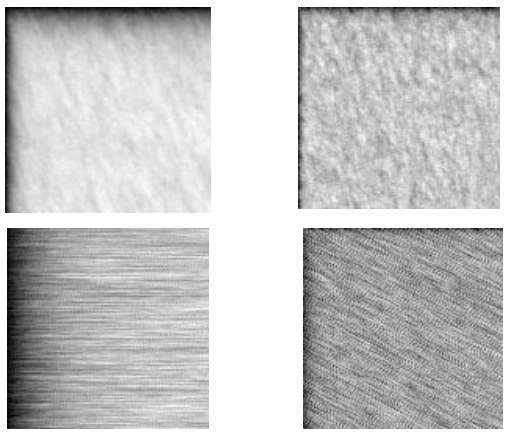

Figure 10. Texture Generated using the Autoregressive model. In each case the order of the neighborhood is 2.

Just to compare the obtained results, in figure 11 are shown some textures generated using a structural approach combined with a random placement of the basic patterns. The textures were generated using the TextureScape software.

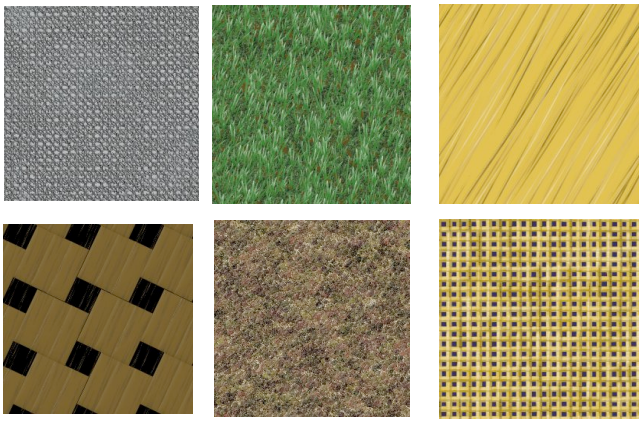

Figure 11. Textures generated using a structural approach.

It should be mentioned that the described texture generation methods greatly differ in their computational effort to generate a texture image, ranging from a couple of seconds to a few minutes, which has an influence on the comfortability of the whole framework.

6. Conclusions

Providing the facility of a user-friendly texture retrieval system comes out to be a very complex task. In this paper, a framework was presented, which is fully specified for solving all of the subtasks related to the retrieval of textures.

One of the most essential points is to relate literal texture descriptions to existing texture images. Two strategies are employed in the presented TEXRET system. The first strategy is to use a Neuro-Fuzzy network for training a mapping from literal texture descriptions to texture features. Then, the user query is translated into literal features, the corresponding textural features are computed and used for retrieving the texture images from a database. However, the texture database might be too small for finding a corresponding image. This is, when the second strategy comes into play. Missing textures are synthesized by an interactive genetic algorithm. When the user has interactively found his texture, the database can be expanded, and a new pair of training data is supplied as well. From this, even a small-at-the-beginning texture database can grow from its very beginning on.

Approaches for texture synthesis were presented as well. Regarding the fact that such approaches are more or less the "heart" of the generation part of the TEXRET system, the set of investigated approaches presented so far is not satisfying. Either the computational effort is high, or the variability among the generated textures is low. In future work, this set has to be expanded by non-Gaussian random field models, POMM models and structural approaches. In particular, for the last case a more straightforward inclusion of coloring is expected.

In this moment the authors are working in the final integration of all the modules and in the testing of the whole system.

Acknowledgements

This research was supported by FONDECYT (Chile) under Project Number 1990595 and by the join "Program of Scientific Cooperation" of CONICYT (Chile) and DFG (Germany).

References

[1] R. Chellappa, S. Chatterjee, and R. Bagdazian, "Texture synthesis and compression using Gaussian Markov random fields", *IEEE Trans. System, Man, Cibern.*, Vol. SMC - 15, no. 2, pp. 298 - 303, Mar /Apr. 1985.

[2] R. Chellappa, R. Kashyap, "Texture synthesis using 2D-Noncausal Autoregressive Models", *IEEE Trans. on Acoustics, Speech and Signal Processing*, Vol. ASSP - 33, No. 1, February 1985, pp. 194 - 203.

[3] G.R. Cross and A.K. Jain, "Markov random field texture models", *IEEE Trans. Pattern and Machine Intell.*, vol. PAMI- 5, pp. 25 – 39, January 1983.

[4] J.L. Davidson, N. Cressie, X. Hua, "Texture Synthesis and Pattern Recognition for Partially Ordered Markov Models", *Elsevier Pattern Recognition* 32, pp. 1475 – 1505, 1999.

[5] M. Hassner and J. Sklansky, "The use of Markov random field models for textures," *Comput. Graphics Image Processing*, vol. 12, pp. 357 – 370, Apr. 1981.

[6] M. Hassner and J. Sklansky,"Markov random fields models of digitized image texture," *Proc. Int. Joint Conf. Pattern Recognition*, Kyoto, Japan, Nov. 1978, pp. 357 – 370, 1980.

[7] F. Hering, "Modellbasierte Textureanalyse", Heidelberger Bildverarbeitungsforum: Textureanalyse, Germany, pp. 43 - 53, 1996.

[8] E. Khan and P. Venkatapuram, "Neufuz: Neural Network Based Fuzzy Logic Design Algorithms", *Proc. IEEE ICFS'93*, San Francisco, pp. 647-654, 1993.

[9] M. Köppen and J. Ruiz-del-Solar, "A fuzzy-based texture retrival system that considers psychological aspects", *Proc. of the Int. Conf. 5th Fuzzy Days, Dortmund, Germany, April 1997*, Springer, pp. 585-586, 1997.

[10] M.C. Maccarone, V. Di Gesù and M. Tripiciano, "An algorithm to compute medial axis of fuzzy images", *Proc. of the 9th Scandinavian Conf. on Image Analysis*, 525-532, Uppsala, Sweden, 1995.

[11] N.R. Pal, N.R and S. Mukhopadhyay, "A psychovisual fuzzy reasoning edge detector", *Proc. of the IIZUKA '96*, 201-204, Japan, 1996.

[12] J.Ruiz-del-Solar, "Neuro-fuzzy system for the administration of texture databases", *Proc. of the XIII Chilean Congress on Elect. Eng.*, 1999 (Spanish).

[13] M. Sugeno, *Theory of fuzzy integral and its applications*, Ph.D. Thesis, Tokyo Institute of Technology, 1974.

[14] M.J. Swain and D.H. Ballard, "Color Indexing", *Int. Journal of Computer Vision*, 7:1, 11-32, 1991.

[15] H. Tamura, S. Mori, and T. Yamawaki, "Textural features corresponding to visual perception", *IEEE Trans. on Sys., Man and Cyb.*, SMC – 8, no. 6, pp. 460 – 472, 1978.

[16] J. K. Tugnait, "Estimation of Linear Parametric Models of NonGaussian Discrete Random Fields with Application to Texture Synthesis", *IEEE Transaction on Image Processing*, Vol. 3 No. 2, pp. 109 – 127, March 1994.