Classification of noisy patterns using ARTMAP-based neural networks

Dimitrios Charalampidis, Georgios Anagnostopoulos, Takis Kasparis, Michael Georgiopoulos
School of Electrical Engineering and Computer Science
University of Central Florida, Orlando, FL 32816

ABSTRACT

In this paper we present a modification of the test phase of ARTMAP-based neural networks that improves the classification performance of the networks when the patterns that are used for classification are extracted from noisy signals. The signals that are considered in this work are textured images, which are a case of 2D signals. Two neural networks from the ARTMAP family are examined, namely the Fuzzy ARTMAP (FAM) neural network and the Hypersphere ARTMAP (HAM) neural network. We compare the original FAM and HAM architectures with the modified ones, which we name FAM-m and HAM-m respectively. We also compare the classification performance of the modified networks, and of the original networks when they are trained with patterns extracted from noisy textures. Finally, we illustrate how combination of features can improve the classification performance for both the noiseless and noisy textures.

Keywords: Fuzzy ARTMAP, Hypersphere ARTMAP, texture, classification, neural networks

1. INTRODUCTION

During the past few years, neural networks (NN) have been extensively used for classification and pattern recognition tasks. Examples of such neural network models include Back Propagation NN[1], Radial Basis Function NN[2] and ART NN. The ART NN family includes Fuzzy ARTMAP (FAM)[3], Hypersphere ARTMAP (HAM)[4], ART-EMAP[5], Gaussian ARTMAP[6], etc. ARTMAP family members have advantages over many other NN models and are especially suited for classification problems. One advantage is that due to the small number of training epochs required by the networks to "learn" the input data, they are faster than other neural networks. Also, the classification results are easily interpretable. Furthermore, compared to the commonly used nearest neighbor techniques, the ARTMAP NN family requires less memory, since it uses a compressed representation of the data. Also for the same reason, members of the ARTMAP NN family require less processing time to complete classification tasks.

Applications of NN include classification of 1-D, 2-D or 3-D signals. In this work, we introduce a variation that is applied to the test phase of the ARTMAP family NN and exhibits superior generalization performance than the corresponding original ARTMAP NN, when the signals are corrupted by additive noise. The variation uses prior knowledge of the feature set's characteristics to adjust the regions of dominance for each class accordingly. We consider as an example the classification of textured images, which is a case of 2-D signals. However, our approach can be easily extended to 1-D or 3-D signals. Texture is a main characteristic of an object's surface. In the case of an image, it defines the spatial relationship between the grayscale pixel values in a region of the image. For the purpose of classification, textures must be described by parameters, usually referred to as features. The features that are selected must be sufficient to characterize the texture. Examples of features that have been used in the literature include Gabor energy[7,8,9], Fourier transform energy [10], second order statistical features[11], wavelet features[12,13] and fractal dimension (FD)[14,15,16].

The proposed modification is especially suited for applications that require the feature set capturing only the shape characteristics of the signal and excluding the actual amplitudes or average values. Examples of such applications include encephalographs and electrocardiographs used for medical diagnosis, classification and recognition of speech signals, classification and segmentation of textured images and satellite photos. In all these cases, it is important to achieve classification independent of illumination and signal amplitude. For instance, if the signal is transmitted through a communication channel, different attenuation may be introduced at different times. In the case of textures, the images may have been obtained using a camera in a poorly illuminated environment. This type of distortion is sometimes termed as multiplicative noise. Features sets that are fairly insensitive to linear transformations and multiplicative noise are desired.

In *Visual Information Processing IX*, Stephen K. Park, Zia-ur Rahman, Editors,
Proceedings of SPIE Vol. 4041 (2000) • 0277-786X/00/$15.00

Two such feature sets that we use in this work are based on fractal characteristics and normalized energy measures. The independence of fractal dimension (FD) to linear transformations of the signal and to multiplicative noise has already been illustrated in the literature[17,18,19]. Also, the energy features that we use maintain similar independence through proper normalization.

Even though the purpose of this paper is to illustrate the improved performance of our proposed, modified ARTMAP NN in the presence of noise, we also concentrate our interest in the specific application of texture classification, which will enable us to present more meaningful comparison results. We use rotation invariant features to be able to classify similar textures that are viewed from different angles. Also, the features are extracted using small image windows, which allows for more accurate identification of different texture regions of the same image. There is a vast literature on texture segmentation and classification. Among the various feature extraction techniques, Gabor filter banks and energy has been used[7,8,9]. These features are not rotational invariant, since they are extracted using directional filters. In Murino et al.[20], third order statistical features are used and it is demonstrated that they exhibit a significant tolerance to white noise. Chaunduri et al.[15] use FD-based features. Feature invariance to image transformations is a desirable property and it is often overlooked in the literature.

The modification that is proposed in this paper can be applied to a larger family of classification techniques, such as k-nearest neighbor techniques. For instance, it is easy to show that a FAM NN with number of nodes equal to the number of training patterns is a 1-nearest neighbor classifier in terms of the L_1 distance. It is a common technique to train a neural network with noisy patterns in order to improve its performance. We have examined the case, where the neural network is trained with patterns extracted from noisy textures. We will illustrate that our modification has certain advantages over this approach. Finally, we will show that even though combination of features can improve the classification of both the noiseless and noisy textures, our modification can still provide further improvement.

The paper is organized as follows. In Section 2, an overview of FAM and HAM architectures is presented. In Section 3, the proposed modification is discussed and Section 4 presents the feature sets that were used. Section 5 contains experimental performance results for texture classification that compare the original and modified neural networks. Also in the same section, we compare classification performance results of the original and the modified networks, when they are trained with patterns extracted from noisy textures. In addition, we illustrate how a combination of features can improve the classification performance for both the noiseless and noisy textures. Finally, in Section 6 we state some closing remarks.

2. ARTMAP NEURAL NETWORKS

In this section we only present a brief overview of FAM and HAM neural networks. The interested reader might find more details in the original papers[3,4].

2.1 Fuzzy ARTMAP Neural Network

For FAM's training phase, a list of *MP* training input/label pairs, such as $\{\mathbf{I}^1, \mathrm{O}^1\}$, $\{\mathbf{I}^2, \mathrm{O}^2\}$, . . . ,$\{\mathbf{I}^{MP}, \mathrm{O}^{MP}\}$, is presented repeatedly to the network until the desired mapping is established for all pairs. During the training phase a number N_a of nodes is created. For each node there is corresponding weight vector (template), which is designated by $\mathbf{w}_j^a = \{w_{j1}^a, \ldots, w_{j2M_a}^a\}$. Each template defines a M_a – dimensional hyper-box that includes within its boundaries all the training input patterns coded by the template. The first M_a elements of the template define a vector, which represents the lower endpoint of the hyperbox, and the last M_a elements define the complement $1\text{-}\mathbf{v}_j^a$ of a vector $\mathbf{v}_j^a$, which represents the upper endpoint. Consider the r-th input/label pair (i.e., $\{\mathbf{I}^r, \mathbf{O}^r\}$) from the training list. The bottom-up input of pattern $\mathbf{I}^r$ to node j is calculated according to:

$$T_j^a(\mathbf{I}^r) = \frac{|\mathbf{I}^r \wedge \mathbf{w}_j^a|}{\beta_a + |\mathbf{w}_j^a|} \tag{1}$$

where β_a is the ART_a choice parameter. From the set of nodes that satisfy the vigilance criterion[2], we choose the one that receives the maximum bottom-up input. If due to prior learning node j_{max} is mapped to a label different than $\mathbf{O}^r$ then the mapping is incorrect and the node is disqualified (reset). The input pair is presented repeatedly, until a suitable node is finally selected and the corresponding template is updated. Learning is considered complete, when no template changes have occurred after all patterns have been repeatedly presented. For the test phase, each input pattern $\mathbf{I}$ of the test set is assigned the label of the node that receives the maximum bottom-up input and satisfies the vigilance criterion. If no node satisfies the

vigilance criterion, the label of the input pattern is designated as "unknown". We must note that the fuzzy min operator ($\wedge$) of two vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ is a vector whose components are equal to the minimum of the corresponding components of $\mathbf{w}_1$ and $\mathbf{w}_2$, and the "size" ($|\cdot|$) of a vector $\mathbf{w}$ is defined to be equal to the sum of its components.

2.2 Hypersphere ARTMAP Neural Network

FAM uses hyperrectangles for category representation that allow for fast update rules and relatively compact coding (2 M_a parameters per category). On the other hand, HAM uses hyperspheres to geometrically describe categories, which results to slightly more complex update rules, but saves on memory usage per node (M_a+1 parameters per category). This is because each template j consists of a vector $\mathbf{m}_j$ and R_j representing the center and the radius of the hypersphere respectively. Due to the nature of the templates, complementary coding of the input patterns is unnecessary. The definitions of choice and match functions (bottom-up input in FAM) for a template j in HAM are based on the corresponding functions in FAM. Specifically, HAM's choice function is:

$$T_j^a(\mathbf{I}^r) = \frac{\overline{D} - R_j - \max\{R_j, \| \mathbf{I}^r - \mathbf{m}_j \|_2\}}{\beta_a + \overline{D} - 2R_j} \tag{2}$$

$\overline{D}$ is HAM's extend parameter and usually equals $(M_a)^{1/2}$, when the input feature space is $\{0,1\}^{Ma}$. HAM's vigilance and choice parameters have the same functionality and interpretation as the ones of FAM. Moreover, HAM's training and test phases follow the same steps as in FAM.

3. THE NEURAL NETWORK MODIFICATION

Consider the mapping between the image space $Z(x,y)$ and the feature space $\boldsymbol{F}^Z_{xy}$, where (x,y) are the coordinates of the image and $\boldsymbol{F}^Z_{xy}$ is the feature vector corresponding to these coordinates:

$$\boldsymbol{F}^Z_{xy} = \mathfrak{I}\{Z(x,y)\} \tag{3}$$

As it was mentioned earlier, in many applications it is required that the features capture only the shape characteristics and not the actual amplitude or average value of the signal. Equivalently, we can express this feature characteristic as insensitivity to linear transformations of the image:

$$\mathfrak{I}\{Z(x,y)\} = \mathfrak{I}\{a\, Z(x,y) + b\} \tag{4}$$

where a and b are constants. According to the previous notation, the feature vector $\boldsymbol{F}^N_{xy}$ extracted at location (x,y) of a realization of a pure noise surface $N(x,y)$ is defined as

$$\boldsymbol{F}^N_{xy} = \mathfrak{I}\{N(x,y)\} \tag{5}$$

In order to change the mean and variance of a random variable such as $N(x,y)$, we apply a linear transformation of the form $aN(x,y) + b$. According to (15), $\boldsymbol{F}^N_{xy}$ is independent of linear transformations of the noise surface, which means that $\boldsymbol{F}^N_{xy}$ is the same, independently of the mean and standard deviation of the noise $N(x,y)$. As a result, $\boldsymbol{F}^N_{xy}$ depends only on the location (x,y) on the noise surface, or equivalently on the specific realization of the noise. The mean $\boldsymbol{F}^N = E_{xy}\{\boldsymbol{F}^N_{xy}\}$ of the feature vector $\boldsymbol{F}^N_{xy}$ over all (x,y), is independent of the location (x,y). In order to be consistent with the usual terminology of FAMNN, we will refer to complementary encoded feature vectors as patterns, and to the complementary encoded mean feature vector $\boldsymbol{F}^N$ of the noise surface *as* the *noise pattern* $\mathbf{I}^N$. When we talk about *patterns extracted from a texture* we refer to the patterns which are the complementary encoded feature vectors extracted from a texture.

Assume that the size of the patterns after complementary encoding is $2M_a$. Then, we are dealing with an M_a-dimensional hyperspace. Each node created in the training phase can then be represented as an M_a-dimensional hyperbox. The training patterns that are associated with this node are included inside the boundaries of this hyperbox. When a pattern is tested, a hyperbox competes with other hyper-boxes to associate this pattern with the class that it represents. Basically, only the hyperboxes that are close to the pattern have some chance to be associated with it. Generally, each hyperbox is dominant in the region that it occupies and in some region around it. The latter, depends on the size of the hyperbox, on the parameter β_a and on its proximity to other hyper-boxes. It has been observed from experiments that patterns that are extracted from a noisy texture will move, at least in most of the cases, towards the direction of the pattern that is extracted from a pure noise

surface. Consequently, noisy patterns whose corresponding noiseless versions are closer to the noise pattern $\mathbf{I}^N$ have a better chance of being correctly classified than noisy patterns whose corresponding noiseless versions are further away from the noise pattern $\mathbf{I}^N$. Therefore, if in the test phase of the FAM we modify the region of influence of each hyperbox so that it gives more emphasis to coordinates that are further away from the noise pattern than the ones that are closer, we have a better chance of correctly classifying patterns extracted from noisy signals. Furthermore, the modification of the test phase of the FAM needs to be such that it allows the correct classification of noiseless textures as well. We have implemented a variation of the test phase of FAM (i.e., FAM-m) to achieve the aforementioned objectives. In the sequel, we present a simplified analysis that gives some intuition of how we have come up with this modification and what is its effect on the classification performance. First, we define the distance between a template $\mathbf{w}$ and a pattern $\mathbf{I}$ as:

$$\text{dist}(\mathbf{w},\mathbf{I}) = |\mathbf{w}| - |\mathbf{w} \wedge \mathbf{I}| \tag{6}$$

In the special case where the template $\mathbf{w}$ is a pattern $\mathbf{I}'$, the distance between $\mathbf{I}$ and $\mathbf{I}'$ as it is defined in (3) is equal to:

$$\text{dist}(\mathbf{I}',\mathbf{I}) = \|\mathbf{I}-\mathbf{I}'\|_1 = \sum_{i=1}^{M_a} |I_i - I_i'| \tag{7}$$

For a pattern $\mathbf{I}$ at the boundary between the regions of dominance of two nodes with templates $\mathbf{w}_1$ and $\mathbf{w}_2$, the corresponding bottom-up inputs $T_1(\mathbf{I})$ and $T_2(\mathbf{I})$ are equal:

$$T_1(\mathbf{I}) = T_2(\mathbf{I}) \Rightarrow \frac{|\mathbf{w}_1 \wedge \mathbf{I}|}{|\mathbf{w}_1|+\beta_a} = \frac{|\mathbf{w}_2 \wedge \mathbf{I}|}{|\mathbf{w}_2|+\beta_a} \Rightarrow \frac{|\mathbf{w}_1|-\text{dist}(\mathbf{w}_1,\mathbf{I})}{|\mathbf{w}_1|+\beta_a} = \frac{|\mathbf{w}_2|-\text{dist}(\mathbf{w}_2,\mathbf{I})}{|\mathbf{w}_2|+\beta_a} \tag{8}$$

In the case where the textures are affected by noise, the patterns extracted from the textures, move towards the noise pattern $\mathbf{I}^N$. Assume that the hyper-box defined by $\mathbf{w}_2$ is closer to the noise pattern than the one defined by $\mathbf{w}_1$, that is, $\text{dist}(\mathbf{w}_1,\mathbf{I}^N) > \text{dist}(\mathbf{w}_2, \mathbf{I}^N)$. Then, node 2 benefits from the movement of patterns due to noise over node 1 because it tends to "capture" patterns that belong to node 1 and move towards $\mathbf{I}^N$. In our approach we modify the boundary between the regions of dominance according to our previous discussion, by dividing the bottom-up input of a node with a term that depends on the corresponding template $\mathbf{w}$. We denote this term as $N(\mathbf{w})$. Assume that $\mathbf{I}'$ is a pattern on the new boundary that corresponds to the pattern $\mathbf{I}$ on the old boundary. We want $\mathbf{I}'$ to be further away from node 1 and closer to node 2, so that the boundary between the regions of dominance moves towards node 2. This boundary movement helps node 1 to "re-capture" the misclassified patterns. Then, for the pattern $\mathbf{I}'$ on the new boundaries:

$$\frac{T_1(\mathbf{I}')}{N(\mathbf{w}_1)} = \frac{T_2(\mathbf{I}')}{N(\mathbf{w}_2)} \Rightarrow \frac{|\mathbf{w}_1 \wedge \mathbf{I}'|}{N(\mathbf{w}_1)(|\mathbf{w}_1|+\beta_a)} = \frac{|\mathbf{w}_2 \wedge \mathbf{I}'|}{N(\mathbf{w}_2)(|\mathbf{w}_2|+\beta_a)} \Rightarrow \frac{|\mathbf{w}_1|-\text{dist}(\mathbf{w}_1,\mathbf{I}')}{N(\mathbf{w}_1)(|\mathbf{w}_1|+\beta_a)} = \frac{|\mathbf{w}_2|-\text{dist}(\mathbf{w}_2,\mathbf{I}')}{N(\mathbf{w}_2)(|\mathbf{w}_2|+\beta_a)} \tag{9}$$

Let us consider only the region between the closest boundaries of the two hyperboxes, to illustrate the effect of this modification. If a pattern $\mathbf{I}$ that exists in this region moves to pattern $\mathbf{I}'$ that exists in the same region then $\mathbf{I}'$ is further away from node 1 than $\mathbf{I}$ by a distance equal to $\text{dist}(\mathbf{I},\mathbf{I}')$ and closer to node 2 by the same distance. Equation (6) becomes:

$$\frac{|\mathbf{w}_1|-\text{dist}(\mathbf{w}_1,\mathbf{I})-\text{dist}(\mathbf{I},\mathbf{I}')}{N(\mathbf{w}_1)\,(|\mathbf{w}_1|+\beta_a)} = \frac{|\mathbf{w}_2|-\text{dist}(\mathbf{w}_2,\mathbf{I})+\text{dist}(\mathbf{I},\mathbf{I}')}{N(\mathbf{w}_2)\,(|\mathbf{w}_2|+\beta_a)} \tag{10}$$

Solving for $\text{dist}(\mathbf{I},\mathbf{I}')$, we find that it is equal to:

$$\text{dist}(\mathbf{I},\mathbf{I}') = \frac{[N(\mathbf{w}_2) - N(\mathbf{w}_1)][|\mathbf{w}_1| +\beta_a]\,[|\mathbf{w}_2| +\beta_a]}{N(\mathbf{w}_1)\,(|\mathbf{w}_1| +\beta_a) + N(\mathbf{w}_2)\,(|\mathbf{w}_2| +\beta_a)}\, T_1(\mathbf{I}) \tag{11}$$

We want to select the term $N(\mathbf{w})$ to satisfy the following property:

$\text{dist}(\mathbf{I},\mathbf{I}')$ should be proportional to $\text{dist}(\mathbf{w}_1, \mathbf{I}^N) - \text{dist}(\mathbf{w}_2, \mathbf{I}^N)$. *Motivation:* In the case where the textures are affected by noise, the patterns extracted from the textures, move towards the noise pattern $\mathbf{I}^N$. If $\text{dist}(\mathbf{w}_1,\mathbf{I}^N) > \text{dist}(\mathbf{w}_2, \mathbf{I}^N)$ then we want to move the boundary between the regions of dominance towards node 2 which is closer to the noise pattern $\mathbf{I}^N$, to increase the probability of recapturing the patterns that moved towards $\mathbf{I}^N$ and belong to node 1. It is reasonable then to move the boundary towards node 2 by an amount, which is proportional to the difference in distance of the two nodes from $\mathbf{I}^N$. For instance, if the distance of two nodes from $\mathbf{I}^N$ is the same: $\text{dist}(\mathbf{w}_1, \mathbf{I}^N) = \text{dist}(\mathbf{w}_2, \mathbf{I}^N)$ we do not want to favor one node versus the other.

We select $N(\mathbf{w}) = M_a - \gamma\, dist(\mathbf{w}, \mathbf{I}^N)$, where γ is a small constant associated with the amount of noise that has affected the textures. Next, we show that the desired property is satisfied. We should also emphasize the effect of the terms $N(\mathbf{w}_1)$ and $N(\mathbf{w}_2)$. For example, if $\gamma = 0.1$, $dist(\mathbf{w}_1, \mathbf{I}^N) = 0.5\, M_a$, and $dist(\mathbf{w}_2, \mathbf{I}^N) = 0.4\, M_a$. Then, $N(\mathbf{w}_1) = 0.95\, M_a$ and $N(\mathbf{w}_2) = 0.96\, M_a$ and $N(\mathbf{w}_2) - N(\mathbf{w}_1) = 0.01\, M_a$. On the other hand, if $dist(\mathbf{w}_2, \mathbf{I}^N) = 0.3\, M_a$ then $N(\mathbf{w}_2) = 0.97\, M_a$ and $N(\mathbf{w}_2) - N(\mathbf{w}_1) = 0.02\, M_a$. Then, what affects significantly the $dist(\mathbf{I},\mathbf{I}')$ is not the actual values of $N(\mathbf{w}_1)$ and $N(\mathbf{w}_2)$ which are very similar and close to M_a, but the difference of the two. We can rewrite (8) as:

$$dist(\mathbf{I},\mathbf{I}') \approx \frac{[N(\mathbf{w}_2) - N(\mathbf{w}_1)]\,[(|\mathbf{w}_1| + \beta_a)\,(|\mathbf{w}_2| + \beta_a)]}{M_a\,[|\mathbf{w}_1| + |\mathbf{w}_2| + 2\beta_a)]}\, T_1(\mathbf{I}) \tag{12}$$

The term $[N(\mathbf{w}_2) - N(\mathbf{w}_1)]$ should be proportional to $dist(\mathbf{w}_1, \mathbf{I}^N) - dist(\mathbf{w}_2, \mathbf{I}^N)$. If we select $N(\mathbf{w}) = M_a - \gamma\, dist(\mathbf{w}, \mathbf{I}^N)$, then $[N(\mathbf{w}_2) - N(\mathbf{w}_1)] = \gamma\,[dist(\mathbf{w}_1, \mathbf{I}^N) - dist(\mathbf{w}_2, \mathbf{I}^N)]$. Therefore, $dist(\mathbf{I},\mathbf{I}')$ is proportional to $[dist(\mathbf{w}_1, \mathbf{I}^N) - dist(\mathbf{w}_2, \mathbf{I}^N)]$ and the desired property is satisfied. The extra term

$$G = \frac{[(|\mathbf{w}_1| + \beta_a)\,(|\mathbf{w}_2| + \beta_a)]}{M_a\,[|\mathbf{w}_1| + |\mathbf{w}_2| + 2\beta_a)]}\, T_1(\mathbf{I}) = \frac{1}{M_a \left\{ \frac{1}{|\mathbf{w}_1| + \beta_a} + \frac{1}{|\mathbf{w}_2| + \beta_a} \right\}}\, T_1(\mathbf{I})$$

does not depend on the factors $N(\mathbf{w}_1)$ and $N(\mathbf{w}_2)$. Term G is not a direct result of our modification. It is a result of the characteristics of the bottom-up input as it is defined in the FAMNN architecture. It was mentioned earlier that the bottom-up input is not only a measure of proximity, but it also depends on the template sizes and on the β_a parameter. Term G reflects these bottom-up input characteristics to $dist(\mathbf{I},\mathbf{I}')$. The bottom-up input is constant inside the region defined by a template. The bottom-up input of a node decreases linearly with the distance of the pattern from it. The slope of the linear decrease is, for node 1 for example, equal to $1/(|\mathbf{w}_1| + \beta_a)$. Assume that the value of the bottom-up input on the boundary between nodes is given. Then, the larger the slopes $1/(|\mathbf{w}_1| + \beta_a)$ and $1/(|\mathbf{w}_2| + \beta_a)$, the closer the boxes defined by the templates are. The term G is inversely proportional to the sum of the two slopes, which is desired; the further away the nodes are, the larger the $dist(\mathbf{I},\mathbf{I}')$ is. According to our previous discussion, we define the bottom-up input for node j of FAMNN-m for the test phase equal to:

$$Tm_j^a(\mathbf{I}^r) = \frac{1}{M_a - \gamma\, dist(\mathbf{w}_j^a, \mathbf{I}^N)}\, \frac{|\mathbf{I}^r \wedge \mathbf{w}_j^a|}{\beta_a + |\mathbf{w}_j^a|} \tag{13}$$

where $\mathbf{I}^r$ is the r-th input pattern from the list of test patterns. The value of γ is selected as the largest possible so that the modification does not introduce more than a specified percentage of extra misclassification on the training set. We must note that γ depends on the specific feature set for a specific SNR.

The match function of the HAM neural network is modified in the test phase in a similar fashion:

$$Tm_j^a(\mathbf{I}^r) = \frac{1}{\overline{D} - \gamma\, \|\mathbf{w}_j^a - \mathbf{I}^N\|_2}\, \frac{\overline{D} - R_j - \max\{R_j, \|\mathbf{I}^r - \mathbf{m}_j\|_2\}}{\beta_a + \overline{D} - 2R_j} \tag{14}$$

It is important to mention that the noise vector (and consecutively the noise pattern) is extracted from a noise surface where the distribution of noise can be of any type, but it is uncorrelated.

4. FEATURE SETS

Although the focus of this paper is on the modification of the FAM and HAM, we also discuss briefly the textural energy and fractal dimension (FD) based feature sets that are used in the experiments.

4.1 Energy Feature Set

Multi-channel Gabor decomposition is an approach to texture analysis. Two-dimensional, non-directional, real and even Gabor filters can be defined in the spatial domain as:

$$h(x, y) = g(x, y)\cos(2\pi u_o \sqrt{x^2 + y^2}) \tag{15}$$

where

$$g(x, y) = exp\{ -(x^2 + y^2) /2\sigma^2] \} \quad (16)$$

Here, $g(x, y)$ is the Gaussian envelope, and u_o denotes the central frequency of the filter. The parameter σ is the standard deviation of the Gaussian envelope in the spatial domain and in all directions, and it defines the size of the envelope. The lack of directionality makes this type of filters relatively invariant to image rotation. We define $Z^{\sigma}_{u_o}(x,y)$ as the filtered version of the original image $Z(x,y)$ filtered by a Gabor filter with central frequency u_o and standard deviation σ. Then, the energy in a window W of size R x R that is centered at the pixel with coordinates (x,y) can be defined as:

$$E^{W}_{u_o\sigma}(x,y) = \sum_{x'=x-R/2}^{x+R/2} \sum_{y'=y-R/2}^{y+R/2} |Z^{\sigma}_{u_o}(x',y') - m^{W}_{u_o\sigma}| \quad (17)$$

where $m^{W}_{u_o\sigma}$ is the mean grayscale value in the window W of the filtered image. In order to make energy dependent only on the texture and not on the magnitude of the image, we normalize it by dividing with the energy in the corresponding window W of the original image which is defined as:

$$E^{W}(x,y) = \sum_{x'=x-R/2}^{x+R/2} \sum_{y'=y-R/2}^{y+R/2} |Z(x',y') - m^{W}| \quad (18)$$

Therefore, we define the normalized energy (NE) as:

$$En^{W}_{u_o\sigma}(x,y) = \frac{E^{W}_{u_o\sigma}(x,y)}{E^{W}(x,y)} \quad (19)$$

The first feature set consists of the NE of twelve filter versions of the original image as it is defined by (11), for twelve different central frequencies $u_o = 0.04k$, $k = 1, 2..12$, and constant standard deviation $\sigma = 23$.

4.2 Fractal Dimension Feature Set

A general definition is that FD is a measure of irregularity of an object. Any curve is an object that occupies some part of a surface and has topological dimension equal to 1. FD defines how much area of this surface is occupied by the curve. For instance, a highly irregular curve will have larger FD than a straight line. The FD of a curve can be between 1, which is equal to its topological dimension, and 2 which is equal to the topological dimension of the plane that it can occupy. The concept of FD can be extended to surfaces. The FD of a surface can be between 2, which is its topological dimension, and 3, which is the topological dimension of the 3-dimensional space that the surface can occupy.

One method that gives a good estimation of the FD of surfaces is the variation method (Dubuc et al., 1987). An image Z of size R x R can be considered as a discrete surface, where its value at position (x, y) is $Z(x, y)$. The variation of Z at the location (x,y) can be defined as:

$$V_{\delta R}(x, y) = \max_{dist((x,y),(s,t))\le\delta} Z(s,t) - \min_{dist((x,y),(s,t))\le\delta} Z(s,t) \quad (20)$$

where dist((x, y),(s, t))=max(|x-s|, |y-t|) and $\delta > 0$. The algorithm for computing the FD in an image window W of size R x R is implemented as follows: The δ^{th} variation $V_{\delta/R}(x,y)$ at the location of the image with coordinates (x,y) is the difference between maximum and minimum grayscale values in a small window $W_{\delta}^{(x,y)}$, of size (2δ+1) x (2δ+1) centered at (x,y). The $V_{\delta/R}(x,y)$ is computed for all pixels (x,y) of the window W, for $\delta = 1, 2, \ldots, \delta_{max}$. If we define $E_{\delta/R}$ as the average of $V_{\delta/R}(x, y)$ over the window W of size R x R for a specified δ, then the FD located at the window W is the slope of the line that best fits the points $(\log(R/\delta), \log\{(R/\delta)^3 E_{\delta/R}\})$ where $\delta = 1, 2, 3, \ldots, \delta_{max}$. The best fit can be found by linear regression. This FD is mapped to the central pixel of W. The same steps applied to window W are repeated for windows of equal size, centered to each one of the pixels of the image.

The second feature set consists of six FD-based features. We use the FD, the FD of the higher grayscale values and the FD of the lower grayscale values. For each one of the three cases we compute the FD as the slope of the line that passes through two points $(\log(1/\delta), \log\{(R/\delta)^3 E_{\delta/R}\})$ where $\delta = (1, 2)$ and we also compute the FD as the slope of the line that

passes through two points $(\log(1/\delta), \log\{(R/\delta)^3 E_{\delta/R}\})$ where $\delta = (2, 3)$. Therefore we have a total of six FD-based features. The FD of the higher and lower grayscale values is actually equal to the FD computed on the images defined as

$$Z^h(x,y) = \begin{cases} Z(x,y), \text{ if } Z(x,y) > m^W \\ FLAG, \text{ if } Z(x,y) < m^W \end{cases} \qquad Z^l(x,y) = \begin{cases} Z(x,y), \text{ if } Z(x,y) < m^W \\ FLAG, \text{ if } Z(x,y) > m^W \end{cases} \tag{21}$$

respectively, where m^W is the average grayscale value in a window W of size R x R around (x,y). A *FLAG* means that the corresponding pixel value is not included in the calculation of FD.

5. Experimental Results

In this section we present experimental results with respect to four key issues. First, we illustrate that our modification is effective independently of the ARTMAP architecture type. Second, we illustrate that the modification is effective independently of the type of noise, as long as it is uncorrelated. Third, we show that our modification has certain advantages over the approach where the network is trained with patterns extracted from noisy textures. Finally, we present a case where combination of features improves the classification performance of the networks, while the modification remains effective. In our previous work we have shown examples for different network sizes and other texture sets, where the modification applied to FAM's test phase showed its potential[21,22].

For our experiments we have used two different texture sets and the two feature sets that are discussed in section 4. The first texture set consists of 20 aerial photos that are shown in Figure 1. The second set consists of 16 textured images, which show different rocks, grass, asphalt, and concrete that are shown if Figure 2. The training was performed using patterns extracted from noiseless textures, while testing was performed on noiseless and noisy textures that represent a different realization of the corresponding textures in the training set. The number of training patterns extracted from the aerial photos was 1280 and 1024 patterns were extracted from the second texture set. The number of test patterns extracted from the aerial photos was 1280 for each case where noise of a specific standard deviation affected the textures. Similarly, 1024 patterns were extracted from the second texture set when it was affected by noise of specific standard deviation.

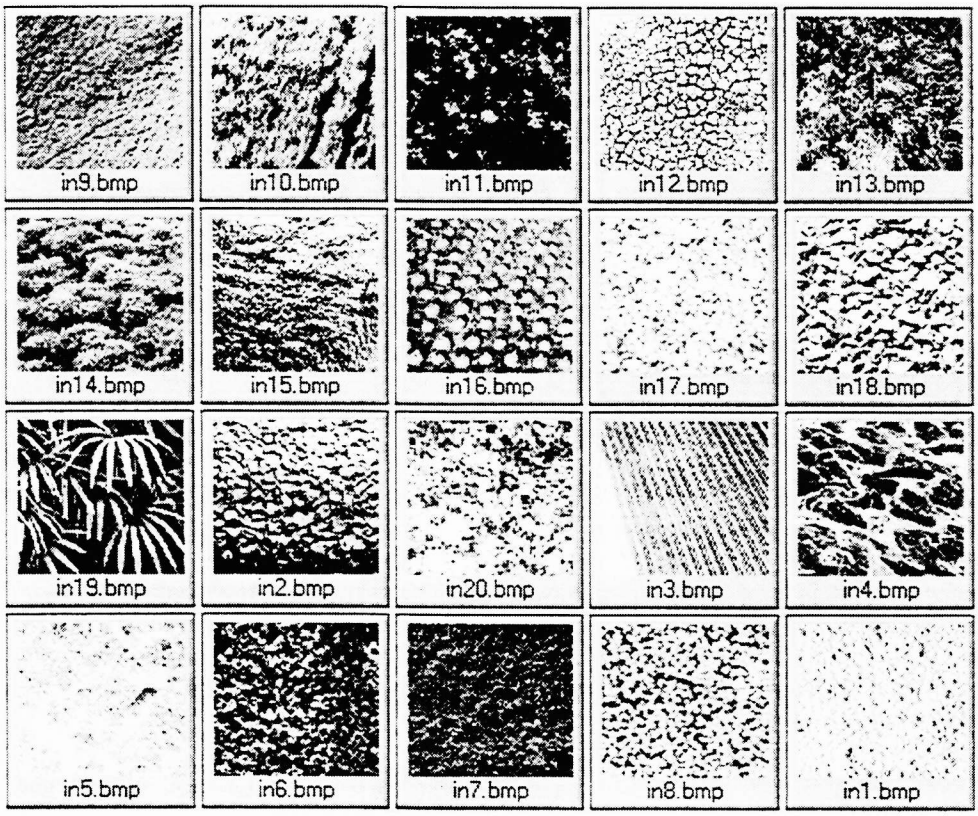


Figure 1: Samples from the aerial textures.

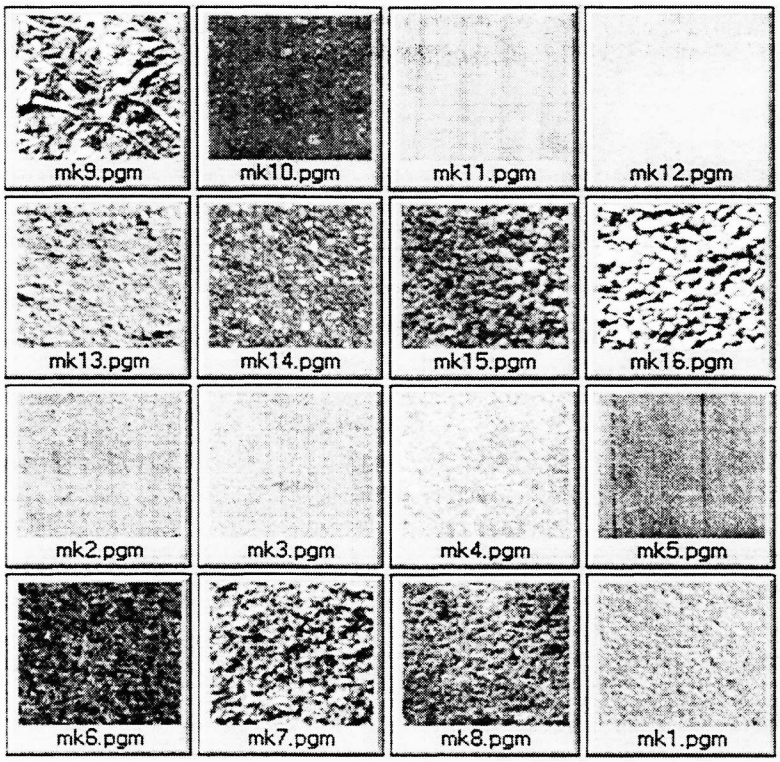
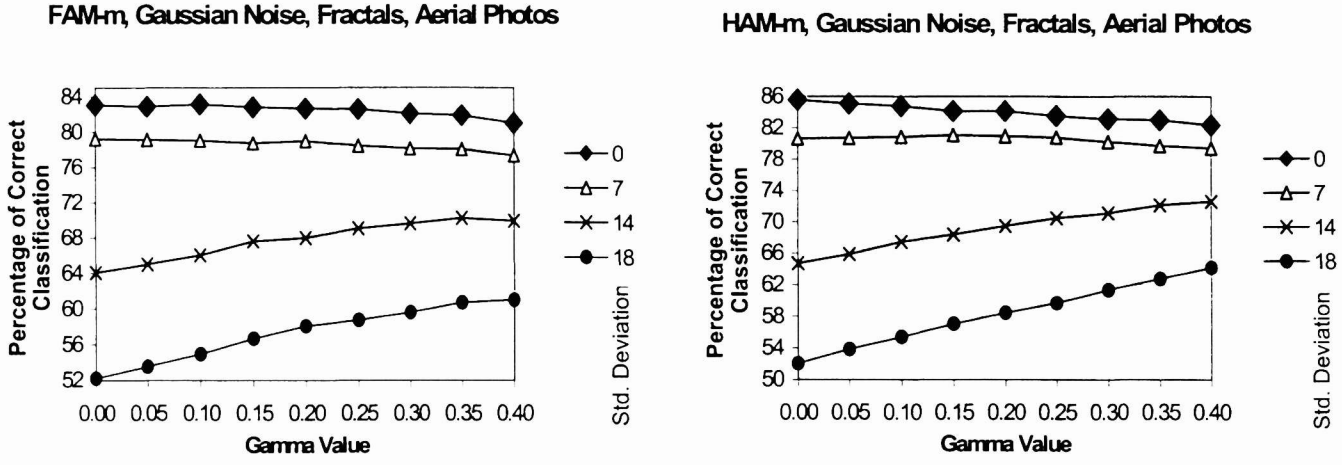


Figure 2: Samples from the second texture set.

5.1 FAM and HAM and the γ parameter

The effect of the parameter γ is shown in Figures 3(a) and 3(b) for the FAM and the HAM respectively. We see that as the γ value increases, the percentage of correct classification (PCC) is slowly decreasing if the textures are noiseless or if the standard deviation of noise is small. On the other hand, the PCC is increasing significantly with γ. Our modification takes advantage of this fact. If we select an appropriate value of γ, the PCC for the noiseless textures will not be significantly smaller than the PCC of the original neural network, while the PCC for the noisy textures will be significantly improved.



(a) (b)

Figure 3: Percentage of correct classification (PCC) for FAM-m and HAM-m for different standard deviations of noise with respect to γ.

If we compare Figures 3(a) and 3(b) we notice that the way that the modification affects PCC is similar for both FAM and HAM. Therefore, the modification is effective independently of the shape of the templates used to encode the input patterns.

The value of γ is selected as follows:

- Vary the value of γ and test the training set.
- If the PCC of the modified network drops bellow a certain value with respect to the PCC of the original network, then stop, and keep this value of γ.
- The estimated value of γ shows how much additional classification error we allow to be introduced. This value is used in the test phase.

In the example of Figure 3, γ was estimated to be equal to 0.3, when the desired drop of PCC was 0.75%. Table 1 shows the PCC for the FAM and HAM, for aerial textures and for the fractals feature set. The network sizes were: 727 nodes for FAM and FAM-m, and 814 nodes for HAM and HAM-m. We should emphasize that each node in FAM is represented by a template of size $2M_a$ where M_a is the dimensionality of the feature vector, while that each node in HAM is represented by a template of size M_a +1. The PCC for the noiseless textures was 83.1% for FAM and 82.1% for FAM-m. The PCC for the noiseless textures was 85.1% for HAM and 84.1% for HAM-m.

FAM	**Standard Deviation of noise:**	**7**	**14**	**18**
Gaussian Noise	FAM	79.2	64.1	52.2
	FAM-m	**78.1**	**69.6**	**59.6**
Uniform Noise	FAM	80.0	62.0	52.6
	FAM-m	**78.4**	**65.9**	**59.4**
Exponential Noise	FAM	79.6	69.7	60.6
	FAM-m	**78.1**	**70.7**	**64.7**

HAM	**Standard Deviation of noise:**	**7**	**14**	**18**
Gaussian Noise	HAM	80.6	64.7	52.0
	HAM-m	**80.9**	**69.5**	**58.4**
Uniform Noise	HAM	80.9	63.4	52.7
	HAM-m	**81.1**	**68.0**	**56.8**
Exponential Noise	HAM	82.7	71.6	62.3
	HAM-m	**82.3**	**72.6**	**65.0**

Table 1: Percentage of correct classification for FAM and FAM-m and for HAM and HAM-m. The fractal feature vector was used.

In the examples of Table 1, we have sacrificed 1% of the PCC for the case where the textures that are tested are noiseless, in order to increase the PCC by 5-6% when noise of standard deviation 14 or above is present.

5.2 Different Noise Distributions

We present some more results to illustrate that the modification is effective when the images are corrupted by different types of uncorrelated noise. Some results were already presented in Table 1. More results for aerial textures where the energy feature vector is used, are shown in Table 2 for FAM and FAM-m. In this example the value of γ was estimated to be equal to 0.05. The number of nodes for the FAM and FAM-m networks was 200.

FAM	**Standard Deviation of noise:**	**14**	**21**	**28**
Gaussian Noise	FAM	74.0	65.0	58.4
	FAM-m	**75.6**	**68.4**	**61.9**
Uniform Noise	FAM	74.5	64.9	58.4
	FAM-m	**76.2**	**67.2**	**62.7**
Exponential Noise	FAM	76.6	68.6	63.4
	FAM-m	**77.3**	**71.7**	**67.0**

Table 2: Percentage of correct classification for FAM and FAM-m. The energy feature vector was used.

The PCC for the noiseless textures was 78.6% for FAM and 77.8% for FAM-m. Again, we see that we have sacrificed 0.8% of the PCC for the case where the textures that are tested are noiseless, in order to increase the PCC by 3-4% when noise when noise of standard deviation 18 or above is present.

5.3 Training with noise

Generally, it is a common procedure to train the neural network with noisy data in order to increase the PCC. We have examined the case of training the neural network with patterns extracted from noiseless textures together with patterns extracted from textures affected by noise with standard deviation 7. The energy feature vector was used. The resulting training set consisted of 2560 patterns.

We have compared the PCC obtained from this approach with the PCC obtained from our modification. In order to make the comparison as fair as possible, we obtained in both cases the "optimum" network size, in the sense that the PCC for the noiseless test set was the highest one. Furthermore, we have selected a value of γ, so that the PCC obtain from both approaches is the same for the noiseless test set. The comparison results are presented in Table 3. We notice that the PCC for both cases is relatively close, which means that training with patterns extracted from noisy textures improves the PCC, but generally, the PCC for our modification is larger. Furthermore, the number of nodes created after training with both noiseless and noisy data is larger (368 nodes) compared to the number of nodes when the network is trained with only noiseless data (200 nodes). This result is expected, since training with noisy data increases the number of training examples and also increases the conflicts between nodes when it comes to the point where a pattern needs to be associated with a specific node. A direct result of creating more nodes, is that the training phase is more time consuming. Another advantage of our method is flexibility, since only one parameter, γ, has to be estimated after one list presentation of the training set to the already trained network, in order to trade between higher PCC for the noiseless textures and higher PCC for the noisy textures. If we train the network with patterns extracted from noisy textures, we require a new training process each time we need to trade between higher PCC for the noiseless textures and higher PCC for the noisy textures.

FAM-Energy Feature Vector

Standard Deviation of noise:		**nonoise**	**14**	**21**	**24**
Gaussian Noise	FAM (trained with noisy data)	78.1 For all	74.5	66.7	60.2
	FAM-m		**74.7**	**67.3**	**60.2**
Uniform Noise	FAM (trained with noisy data)		75.2	65.5	59.6
	FAM-m		**76.1**	**66.7**	**61.0**
Exponential Noise	FAM (trained with noisy data)		75.5	68.9	64.1
	FAM-m		**77.2**	**70.6**	**65.6**

Table 3: Percentage of correct classification for FAM-m, and FAM when it is trained with patterns extracted from noiseless and noisy textures.

5.4 Feature Combination

In Table 4 we present the PCC for FAM and FAM-m and when the second texture set was used. In these examples the test set is corrupted by Gaussian noise. In Table 4(a) we present the PCC when the energy feature vector was used. We notice that the PCC is relatively small. The number of nodes was 332. In Table 4(b) we present the PCC when the fractals feature vector was used. The PCC is larger than for the energy feature vector for both FAM and FAM-m. The number of nodes was 736. In both Tables 4(a) and 4(b) we notice the PCC improvement of FAM-m over FAM. In Table 4(c) we present the PCC result when a combined vector of size 18 (6 fractal and 12 energy features) is used. We notice the improvement in PCC of both FAM and FAM-m over the corresponding PCC in Tables 4(a) and 4(b). The number of nodes was 552.

From this example we can conclude that increasing the size of the feature vector can increase significantly the PCC, for both the case where noiseless and noisy textures are tested. Nevertheless, our modification is still effective.

FAM – Second Texture Set – Gaussian Noise

Std. Dev. of noise:	nonoise	7	14
FAM	63.0	59.4	40.6
FAM-m	**62.5**	**61.3**	**49.5**

(a)

Std. Dev. of noise:	nonoise	7	14
FAM	82.1	65.6	46.0
FAM-m	**82.0**	**68.8**	**50.5**

(b)

Std. Dev. of noise:	nonoise	7	14
FAM	84.9	74.6	55.1
FAM-m	**84.4**	**76.7**	**63.8**

(c)

Table 4: Percentage of correct classification for FAM and FAM-m. (a) Energy feature set, (b) Fractals feature set, (c) Combined energy and fractals feature vector.

6. SUMMARY AND CONCLUSIONS

In our previous work we have introduced a modification of FAM, namely FAM-m, that exhibits superior generalization performance than the standard FAM in the case where textures are affected by uncorrelated noise, without significantly worsening the classification performance of the networks when noise is not present. This modification works when the features extracted from textures possess the insensitivity to linear transformation property, which is a necessary property in many applications. In this paper we extended our previous work by examining the effectiveness of our modification on another member of the ARTMAP family, the HAM neural network. We name the modified HAM as HAM-m. Experiment show that HAM-m that exhibits superior generalization performance than HAM in the case where textures are affected by uncorrelated noise, in a similar way that the classification performance of FAM-m is related to the classification performance of FAM. This result makes our modification more general, since it illustrates that the modification can be extended to other classification techniques including k-nearest neighbor techniques. Furthermore we have shown that the modification improves the classification performance in the case that different types of noise have affected the textures. Finally, we have illustrated with an example that it is possible for the percentage of correct classification to be improved by combining different types of features, and as a result, increasing the size of the feature vectors. Even in the latter case, the modification can offer significant improvement in the classification performance of the neural networks when the textures under consideration are affected by noise.

7. REFERENCES

[1] Rumelhart, D.E., Hinton, G.E., and Williams, R.J., "Learning internal representation by error propagation", *Parallel Distribution Processing: Explorations in the Microstucture of Cognition*, Vol. 1, Chapter 8, Cambridge, MA:MIT Press, 1986.

[2] Moody, J.E., and Parker, C.J., "Fast learning in networks of locally tuned processing units", *Neural Computation*, Vol. 1, pp. 281-294, 1989.

[3] Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., and Rosen, D.B. , "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps", *IEEE Transactions on Neural Networks,* No. 3, pp. 698-713, 1992.

[4] Anagnostopoulos, G. and Georgiopoulos, M., "Hypersphere ART and ARTMAP for Unsupervised and Supervised, Incremental Learning", to appear in *Proceedings of the International Joint Conference on Neural Networks*, Como, Italy, 2000.

[5] Carpenter, G.A., Ross, W.D. (1995). "ART-EMAP: A neural network architecture for object recognition by evidence accumulation", *IEEE Transactions on Neural Networks*, Vol. 6, pp. 805-818, 1995.

[6] Williamson, J.R., "Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps", *Neural Networks*, Vol. 9, No. 5, pp. 881-897, 1996.

[7] Jain, A.K., and Farrokhinia, F., "Unsupervised texture segmentation using Gabor filters", *Pattern Recognition*, Vol. 24, No 12, pp. 1167-1185, 1991.

[8] Dunn, D., and Higgins, W.E., "Optimal Gabor filters for texture segmentation", *IEEE Transactions on Image Processing,* Vol. 4, No. 7, pp. 947-964, July 1995.

[9] Teuner, A., Pichler, O., and Hosticka, B.J., "Unsupervised texture segmentation of images using tuned matched Gabor filters", *IEEE Transactions on Image Processing,* Vol. 4, No. 6, pp. 863-870, June 1995.

[10] Bajscy, R., "Computer identification of visual surfaces", *Computer Graphics and Image Processing*, Vol. 2, pp. 118-130, 1973. Brodatz P., *Textures: A Photographic Album for Artists and Designers*, New York: Dover, 1966.

[11] Chen, P.C., and Pavlidis, T., "Segmentation by texture using correlation", *IEEE Transactions on pattern analysis and Machine intelligence,* Vol. PAMI-5, pp. 64-69, Jan. 1983.

[12] Unser, M., "Texture classification and segmentation using wavelet frames", *IEEE Transactions on Image Processing,* Vol. 4, No. 11, pp. 1549-1560, November 1995.

[13] Strickland, R.N., "Wavelet transform methods for image detection and recovery", *IEEE Transactions on Image Processing*, Vol. 6, No. 5, pp. 724-734, May 1997.

[14] Dubuc, B., Roques-Carmes, C., Tricot, C., and Zucker, S.W., "The variation method: a techique to estimate the fractal dimension of surfaces", *SPIE, Visual Communications and Image Processing II*, Vol. 845, pp. 241-248, 1987.

[15] Chaundhuri, B.B., and Sarkar, N., "Texture segmentation using fractal dimension", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 17, No. 1, pp. 72-77, January 1995.

[16] Kasparis, T., Tzannes, N.S., Bassiouni, M., and Chen, Q., "Texture description based on fractal and energy features", *Computers and Electrical Engineering,* Vol. 21, No. 1, pp. 21-32, 1995.

[17] Charalampidis, D., Kasparis, T., Rolland, J., "Segmentation of textured images based on multiple fractal feature combinations", *Proceedings of the SPIE, Visual Information Processing VII*, pp. 25-37, 1998.

[18] Garding, J., "Properties of fractal intensity surfaces", *Pattern Recognition Letters*, pp. 319-324, 1988. Georgiopoulos, M., Fernlund, H., Bebis, G., and Heileman, G.L., "Order of search in Fuzzy ART and Fuzzy ARTMAP: Effect of the choice parameter", *Neural Networks*, Vol. 9, No. 9, pp. 1541-1559, 1996.

[19] Pentland, A.P., "Fractal based description of natural scenes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, No. 6, pp. 661-674, November 1984.

[20] Murino, V., Ottonello, C., and Pagnan, S., "Noisy texture classification: A higher – order statistics approach", *Pattern Recognition*, Vol. 31, pp. 383-393, 1998.