Segmentation of textured images based on multiple Fractal feature combinations

Dimitrios Charalampidis[*], Takis Kasparis[*] and Jannick Rolland [†*]

* Department of Electrical and Computer Engineering
University of Central Florida
Orlando, FL 32816

[†] Center for Research and Education in Optics and Lasers
University of Central Florida
Orlando, FL 32816

ABSTRACT

This paper describes an approach to segmentation of textured grayscale images using a technique based on image filtering and the fractal dimension (FD). Twelve FD features are computed based on twelve filtered versions of the original image using directional Gabor filters. Features are computed in a window and mapped to the central pixel of this window. An iterative K-means-based algorithm which includes feature smoothing and takes into consideration the boundaries between textures is used to segment an image into a desired number of clusters. This approach is partially supervised since the number of clusters has to be predefined. The fractal features are compared to Gabor energy features and the iterative K-means algorithm is compared to the original K-means clustering approach. The performance of segmentation for noisy images is also studied.

Keywords: fractal dimension, Gabor filters, texture, segmentation, K-means

1. INTRODUCTION

Texture is a main characteristic of the surface of an object. In the case of an image, it defines the special relationship between the grayscale values of the pixels in a region of the image. Texture segmentation is an important task with many applications in pattern recognition and computer vision. It can be defined as identification of different regions with uniform textures, or as identification of the boundaries between them. For the purpose of segmentation, textures must be described by parameters, usually denoted as features. If the variation of the feature values that describe a uniform textural region is small, then the results obtained by the segmentation process are more accurate. Another desired feature property is insensitivity to different image transformations, such as changes in the intensity, zooming, scaling of the grayscale values, subjection to noise and rotation. Usually, a feature set needs to contain more than one feature to successfully characterize a textural region. Many features have been used for texture segmentation and classification. Some of them are Gabor energy features [1, 2, 3], Fourier transform energy[4], second order statistical features[5], wavelet features [6,7] and fractal dimension (FD) [8] which is also studied here. The last stage of segmentation involves a method of identification of the different texture regions, based on the corresponding features. This method can be clustering or classification of the feature sets that characterize different regions of the image. A very commonly used clustering algorithm is K-means.

2. BACKGROUND

2.1 Gabor filters and energy

Multi-channel Gabor decomposition is an approach to texture characterization and segmentation. The frequency spectrum of a signal, texture in this case, is decomposed into its spectral components using two dimensional Gabor filters

with specified bandwidths and center frequencies. The filters can have constant or octave bandwidth. Two-dimensional directional symmetric Gabor filters can be defined in the spatial and the frequency domain respectively as:

$$h(x, y)=g(x', y') [\cos(2\pi fo\, x') + j \sin(2\pi f o\, x')] \tag{1}$$

$$H(u, v)=4 \pi \sigma_x \sigma_y \exp \{ - [(u ' - fo)^2 / 2\sigma_u^2 + v'^2 / 2\sigma_v^2)]\} \tag{2}$$

where

$$g(x, y)=\exp\{ -[x^2/2 \sigma_x^2 + y^2/2 \sigma_y^2] \} \tag{3}$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos \Phi o & \sin \Phi o \\ - \sin \Phi o & \cos \Phi o \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4}$$

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} \cos \Phi o & \sin \Phi o \\ - \sin \Phi o & \cos \Phi o \end{pmatrix} \begin{pmatrix} v \\ u \end{pmatrix} \tag{5}$$

Here, g(x, y) is the Gaussian envelope, Φo denotes the orientation of the filter with respect to u axis, fo denotes the center frequency of the filter and $\sigma_u = 1/2\pi\sigma_x$, $\sigma_v = 1/2\pi\sigma_y$. It can be noticed that H(u, v) is also a Gaussian envelope centered at frequencies (fo, 0). The parameters σ_x, σ_y are the standard deviations of the Gaussian envelope in the spatial domain and in the direction of x' and y' respectively and they define the size of the envelope. The standard deviations of the Gaussian envelope in the frequency domain and in the direction of u' and v' respectively are σ_u, σ_v.

Real-valued Gabor filters, are a special case of the complex-valued Gabor filters and they can be defined in the spatial and the frequency domain respectively as:

$$h(x, y)=g(x', y') \cos(2\pi uo\, x') \tag{6}$$

$$H(u, v)=2\pi \sigma_x \sigma_y \left[\exp\{ -[(u'-uo)^2/2\sigma_u^2 + v'^2/2\sigma_v^2)] \} + \exp\{-[(u'+uo)^2/2\sigma_u^2+v'^2/2\sigma_v^2)] \}\right] \tag{7}$$

Filtering the original image using filter banks is the first step of feature extraction for texture characterization. The second step is to compute a measure in small overlapping windows of the filtered images. Texture energy is a measure that is widely used to characterize texture. It is the absolute deviation from the mean of the grayscale values in small windows of the image. The energy that corresponds to a window centered at x, y in the spatial domain is defined as:

$$E(x, y)=1/M^2 \sum_{(i, j) \in W_{xy}} |F(Z_f(I, j))| \tag{8}$$

where M x M is the size of the window W_{xy} centered at x, y and $Z_f(x, y)$ is the value of the filtered image with coordinates x, y. F(•) is a non-linear, sigmoidal function of the form

$$F(t) = \tanh(\alpha t) = [1-\exp(-2 \alpha t)] / [1+\exp(-2\alpha t)] \tag{9}$$

where α is a constant. The feature extraction scheme that involves multi-channel filtering and computation of the texture energy can be characterized as a blob detector.

2.2 Fractal Dimension

There are many definitions of the FD of an object, including box dimension, intersection dimension and Bouligand-Minkowski dimension. FD has been characterized as a measure of irregularity of an object. Any curve is an object with one topological dimension that occupies some part of a surface. 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 surface 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 "box" that the surface can occupy. Two methods that give good estimation of the FD are the differential box counting (DBC) and the variation methods.

2.2.1 Differential box counting

The concept of self-similarity can be used to estimate the FD. A bounded set A in the Euclidian n-space is self-similar if A is the union of N_r distinct (non-overlapping) copies of itself, scaled up or down by a factor of r. Using this definition FD can be defined by:

$$1 = N_r r^D \text{ or } FD = \log (N_r) / \log(1/r) \tag{10}$$

For this method the image of size M x M pixels is scaled down to s x s, where $1< s \leq M/2$, where s is an integer. Then, r = s/M. The image is considered as a 3D space, where two dimensions define the coordinates (x, y) of the pixels and the third (z) defines the grayscale values of the pixels. The (x, y) is partitioned into grids of size s x s. On each grid there is a column of boxes of size s x s x s. If the minimum and the maximum grayscale levels in the $(i, j)^{th}$ grid fall into the k^{th} and l^{th} box respectively, then the contribution of N_r in the $(i, j)^{th}$ grid is defined as $n_r(i, j) = l - k + 1$. For this method N_r is defined as the summation of the contributions from all grids that are located in a window of the image:

$$N_r = \sum_{i,j} n_r(i, j) \tag{11}$$

If N_r is computed for different values of r, then FD can be estimated as the slope of the line that best fits the points $(\log(1/r), \log N_r)$.

2.2.2 Variation method

Variation is a method that is used to compute the FD of an object. It has been shown[9] that the variation method gives accurate and robust estimation of the FD of a surface. An image Z(x, y) of size R x R can be considered as a surface of size R x R, where its value at position (x_o, y_o) is $Z(x_o, y_o)$. The variation method can be applied on an image so that its FD can be computed.

The definition of the FD for surfaces using the variation method, is that if a surface Z is a fractal, there exists at least one part of the interval [0, 1] where Z is nowhere or almost nowhere differentiable. If P(x, y, x', y') is the slope of the line passing through points (x, y, Z(x, y)) and (x', y', Z(x', y')), then this |P(x, y, x', y')| goes to infinity as the point (x', y') tends toward (x, y). FD is defined as the rate in which |P(x, y, x', y')| goes to infinity. The variation of Z can be defined as:

$$V_\varepsilon(x, y) = \max_{\text{dist}((x, y),(s, t)) \leq \varepsilon} Z(s, t) - \min_{\text{dist}((x, y),(s, t)) \leq \varepsilon} Z(s, t) \tag{12}$$

where $\text{dist}((x, y),(s, t))=\max(|x-s|, |y-t|)$ and $\varepsilon > 0$. The integral of $V_\varepsilon(x, y)$ tends to zero as ε tends to 0. The rate of growth of this integral is directly related to the FD of Z. The FD of the surface Z is given by:

$$FD_Z = \Delta_V(Z) = \lim_{\varepsilon \to 0}\left(3 - \frac{\log \int_0^1\int_0^1 V_\varepsilon(x, y)dxdy}{\log \varepsilon} \right) \quad (13)$$

$$= \lim_{\varepsilon \to 0}\left(\frac{\log \int_0^1\int_0^1 [\, V_\varepsilon(x, y)/\varepsilon^3 \,] dxdy}{\log (1/\varepsilon)} \right) \quad (14)$$

The slope of the log-log plot of the line that is defined by $\log \int_0^1\int_0^1 [\, V_\varepsilon(x, y)/\varepsilon^3 \,] dxdy$ and $\log(1/\varepsilon)$ gives the FD of the surface. The computation of the FD of a discretized surface involves substitution of the integrals with summations.

The FD of an image can be computed locally in all different regions of size R x R of the image, so that a FD space can be created. This FD space will be mapped one-to-one to the pixels of the image. The algorithm for computing the FD space of an image is implemented as follows: The difference $V_{\varepsilon/R}$ between the maximum and the minimum grayscale values is computed in a small window of size T x T, where T = 2ε+1. This window is centered at the pixel with coordinates (x, y). This computation is repeated for all pixels (x, y) of the image, for $\varepsilon = 1, 2, 3, \ldots, \varepsilon_{max}$. $V_{\varepsilon/R}(x, y)$ is the ε^{th} variation located at (x, y). If we define $E_{\varepsilon/R}$ as the average of $V_{\varepsilon/R}(x, y)$ over a window W of size R x R, then the FD located at the window W is the slope of the line that better fits the points $(\log(R/\varepsilon), \log\{(R/\varepsilon)^3 E_{\varepsilon/R}\})$ where $\varepsilon = 1, 2, 3, \ldots, \varepsilon_{max}$. The line that better fits these points can be found using the least mean square approach. This FD is mapped to the central pixel of the window W. The next step is to shift the window W and map its FD to the central pixel of the new window. The previous steps are repeated for all pixels of the image and the FD space is created.

2.3 Previous segmentation methods

Join and Farrokhnia[1] have proposed a method that uses a bank of Gabor filters. The filters are real-valued, even and symmetric, with octave bandwidth. The bank consists of twenty filters for four directions (0°, 45°, 90°, 135°) and five center frequencies √2x4, √2x8, √2x16, √2x32, √2x64 pixels x (image width)$^{-1}$. The standard variation was selected to be $\sigma \approx 0.5$ x (image width / uo) where uo is the center frequency of the Gabor filter. The minimum number of filters where outputs can be used to reconstruct the original image at least by a percentage R, are chosen out of this twenty. The feature set consists of the energy of the filtered versions of the image. The coordinates of the pixels are also used as features so that pixels that are close can be grouped together. A K-means-based method that estimates the number of different textures is used for the final segmentation.

Dum and Higgins[2] propose a method that uses complex symmetric filters. A filter quality measure is computed for the selection of optimal Gabor filters with respect to the different filter parameters. The energy measure that is computed to characterize the textures is defined as $O_h(I(x,y)) = |I(x,y)*h(x,y)|$. In their work it is assumed that the image contains two different textures and that the filter outputs can be modeled as Rician random variables. A threshold-decision rule is used to segment the two textures.

Teuner, Pichler and Hosticka[3] have proposed a method that uses complex and symmetric Gabor filters with octave bandwidth. Filter selection is based on a pyramid scheme that is used to identify characteristics of the textures for different resolutions. The features are the power of the filtered versions of the original image which can be defined as $O_h(x,y) = [I(x,y)*h_{real}(x,y)]^2 + [I(x,y)*h_{imaginary}(x,y)]^2$. The segmentation is done using a simple threshold function and a minimum distance method.

Chaudhur and Sarkor[8] propose texture segmentation using FD and the multi-fractal concept. Six FD features are computed based on the original image, the below and above average gray-level image, the horizontally and vertically smoothed image and the multifractal-dimension of order two. DBC is used for computation of the FD features. Feature

smoothing and unsupervised K-means like algorithm are used for the segmentation. Supervised techniques are also considered.

3. A COMBINED SEGMENTATION METHOD

The problem with FD was that as a single feature is not sufficient for texture analysis and characterization. The idea of using more than one FD features, has already been introduced [8, 10]. In this paper, a feature set consisting of the FD of twelve filtered versions of the original image is used. Directional, symmetric, real-valued, two-dimensional Gabor filters are used for the filtering. The idea here is to use the FD instead of the energy of the filtered images, because FD is insensitive to differences of the local intensity of the image and to local scaling of the grayscale levels. Ideally, FD is also insensitive to image zooming but in practice this is valid to a certain extent. The method we use for computing the FD is the variation method. The segmentation method that is described here consists of two steps: feature extraction and clustering of the feature vectors.

3.1 Feature extraction

The first step in feature extraction is filtering of the original image Z_o using Gabor filters with different center frequencies and orientations. The filter images are $Z(q, f, \sigma_x, \sigma_y)$, where q defines the orientation of the filter, f defines the center frequency and σ_x, σ_y are the standard deviations of the Gaussian envelope in the spatial domain. In our experiments we used $q = 0°, 45°, 90°, 135°$ and $f = 0, 0.05$ and 0.1 so that a set of twelve filters is used to filter the original image. The second step is to compute the FD space for all filtered versions of the image. The FD values that are computed over a window W centered at the pixel with coordinates (x, y) for all the twelve filtered versions of the image, are mapped to pixel with coordinates (x, y) of the original image. These twelve FD values comprise the feature set.

Four orientations are sufficient for the filters to detect the directional characteristics of the textures. The standard deviations are $\sigma_x = 6$ in the direction of the filter and $\sigma_y = 0.01$ in the direction which is perpendicular to the direction of the filter. These filters decompose the image into its frequency components only for the direction of the filters and at the same time do not affect the frequency characteristics of the other direction. The FD feature set that is computed for the filtered images gives better segmentation results if this type of filters is used. If σ_y is also large, each filtered image will not contain enough information for computation of the FD. The same reason leads to selection of lower center frequencies for the Gabor filters, since images contain most information at lower frequencies. Filters with higher center frequencies were also tried but the segmentation results did not change significantly.

For the computation of the FD the slope of the line that passes through two points $(\log(1/\varepsilon), \log\{(R/\varepsilon)^3 E_{\varepsilon/R}\})$, $\varepsilon = (1, 2)$ is considered. The reason for selecting only two points is that the main interest is not the exact value of the FD, but the robustness of the feature and its power to discriminate between different textures. Larger values of ε involve larger windows for computing the variations. The larger the window, the higher the probability of including regions of more than one texture. It is preferable to avoid this situation so that there is better estimation of the FD close to the boundaries. Also, larger windows deduce the insensitivity of FD to image intensity as it will be described later.

The FD is computed for windows of size R x R where R =16. Large window size gives more robust feature set, but at the same time it blurs the boundaries between textures. It was found experimentally that R = 16 includes enough information for the computation of FD and at the same time it does not create much blurring at the boundaries.

There are certain advantages of using FD over other features. One advantage is that FD is independent of the intensity of the image. This means that if a constant value is added to the grayscale values of the image, the FD remains the same. The proof is trivial. Each variation $V_{\varepsilon/R}$ around the pixel with coordinates (x, y), is equal to the difference between the maximum (max) and the minimum (min) grayscale values in a window of size T x T, where $T = 2\varepsilon + 1$. If *any* constant value C is added to all grayscale values in the region around (x, y) then the new maximum and minimum values are $\max_{(new)} = \max + C$, $\min_{(new)} = \min + C$. The new variation around (x, y) is $V_{\varepsilon/R}^{(new)} = \max_{(new)} - \min_{(new)} = (\max + C) - (\min + C) = \max - \min = V_{\varepsilon/R}$. Since all variations remain the same, the FD will be unchanged. Practically, C can vary slowly, since it can be assumed that it is approximately constant in a window of size T x T. Adding different constant values to different regions of the image will not affect the segmentation results. A small problem might occur around boundaries of regions where two different constant values have been added.

The FD is also insensitive to multiplicative noise. Multiplicative noise is defined as local compression or stretching of the grayscale levels, which can be considered as the result of multiplication by different factors of the grayscale levels in different regions of the image. The proof is that if the grayscale values in a region around pixel with coordinates (x, y), are multiplied by a constant factor C, then the new maximum and minimum values are $max_{(new)} = C\ max$, $min_{(new)} = C\ min$. If the variation around (x, y) before the multiplication was $V_{\varepsilon/R}$, the new variation is $V_{\varepsilon/R}^{(new)} = max_{(new)} - min_{(new)} = C\ max - C\ min = C\ [max - min] = C\ V_{\varepsilon/R}$. If the average variation around (x, y) was $E_{\varepsilon/R}$, then the new average variation will be $E_{\varepsilon/R}^{(new)} = C\ E_{\varepsilon/R}$. FD is equal to the slope of the line that better fits the points $(\log(R/\varepsilon), \log\{(R/\varepsilon)^3 C\ E_{\varepsilon/R}\})$ or $(\log(R/\varepsilon), \log\{(R/\varepsilon)^3\ E_{\varepsilon/R}\} + \log C)$. It can be noticed that if $\log(R/\varepsilon)$ defines the coordinate of the points on X - axis and $\log\{(R/\varepsilon)^3\ E_{\varepsilon/R}\} + \log C$ defines the coordinates of the points on the Y - axis, then the line that best fits these points has exactly the same slope as the line that best fits the points $(\log(R/\varepsilon), \log\{(R/\varepsilon)^3\ E_{\varepsilon/R}\})$, since it is the same line, but shifted with respect to Y - axis by a constant value equal to log C. In the case of two points, $\varepsilon = \varepsilon_1, \varepsilon_2$ where $\varepsilon_1 = 1$, $\varepsilon_2 = 2$, and the slope is equal to $\{[\log\{(R/\varepsilon_1)^3\ E_{\varepsilon 1/R}\} + \log C] - [\log\{(R/\varepsilon_2)^3 E_{\varepsilon 2/R}\} + \log C]\}/\{\log(R/\varepsilon_1) - \log(R/\varepsilon_2)\} = \{\log\{(R/\varepsilon_1)^3 E_{\varepsilon 1/R}\} - \log\{(R/\varepsilon_2)^3 E_{\varepsilon 2/R}\}\}/\{\log(R/\varepsilon_1) - \log(R/\varepsilon_2)\}$ which is equal to the slope of the line before multiplication by the factor C. So, the FD which is the slope of the line remains unchanged. Practically, C can vary slowly since it can be assumed that it is constant in a relatively small window. This result shows that if different regions of the image are multiplied by different constant factors, the segmentation results will not change.

Energy on the other hand, is not insensitive to intensity changes. Theoretically, high-pass Gabor filters could be used to remove the low-pass components, but practically, filters with low center frequencies are necessary for good segmentation results.

Energy is also sensitive to multiplicative noise. The output of a filter is the convolution $Z_f(x, y) = h(x, y) * Z_o(x, y)$ where Z_o is the original image and h is the two-dimensional impulse response of the Gabor filter. If the grayscale values of the original image are multiplied by a constant factor C, then the new output is $Z_C(x, y) = h(x, y) * C\ Z_o(x, y) = C\ Z(x, y)$. Obviously, the energy will be different in this case. If different regions of the same texture are multiplied by different factors, then they are not going to be identified as the same texture.

3.2 Clustering the feature vectors

After the computation of the twelve feature spaces, all twelve FD features that are mapped to the pixel with coordinates (x, y) comprise a feature vector. There is a total of S x S feature vectors of size twelve, where S is the width of the image.

A K-means-based algorithm is used to cluster the feature vectors. This algorithm considers the boundaries between textures so that finer segmentation at these regions is possible. It is very important for the features to be smoothened over a window, so that their robustness increases. This window becomes smaller as the window moves closer to the boundaries. The variable window increases the robustness at the inner texture regions and at the same time it decreases blurring at the boundaries. The algorithm works as follows:

Step 1. Smoothing of the features by averaging their values in a square window of size Ro x Ro and map the result to its central pixel, where Ro is the width of the initial smoothing window.

Step 2. Define the initial cluster centers and apply the K-means algorithm for an initial estimation of texture regions. The final cluster centers are kept unchanged for the remaining steps of the algorithm.

Step 3. A sliding median window is used to merge small regions to large regions. This median window clusters the feature vector that is mapped to central pixel of a window of size Ro/2 x Ro/2 to cluster j, if cluster j is the dominant cluster in this window. A cluster is dominant, if the number of feature vectors that are associated to this cluster is larger than the number of features that are associated to any other cluster.

Step 4. Clustering will remain unchanged for the feature vectors that correspond to all pixels that are far away from the boundaries for a distance D that is greater than half the width of the smoothing window. The reason is that the smoothing window with center that is far away from the boundaries for a distance D consists of only one texture. The

rest of the pixels that are closer to the boundaries are considered as an ambiguous cluster and they will be further examined.

Step 5. Apply the procedure to the ambiguous cluster for smoothing windows of size R_i x R_i and median windows of size $R_i/2$ x $R_i/2$ respectively, iteratively starting from step 3, where i = 1,2, ..., is the iteration index. R_i is smaller for larger values of i.

For the experiments three smoothing window widths are used, $R_o = 33$, $R_1 = 25$, $R_2 = 17$.

4. EXPERIMENTAL RESULTS

The segmentation method that is presented in this paper, is compared to the method that is proposed by Jain and Farrokhnia, where a filter bank of two dimensional, even, symmetric, directional and real Gabor filters with octave bandwidth, is used. Here, sixteen Gabor filters are used to filter the original image, for the four directions and for four center frequencies $\sqrt{2}$x4, $\sqrt{2}$x8, $\sqrt{2}$x16, $\sqrt{2}$x32 pixels x (image width)$^{-1}$. The energy of the filtered versions of the image is computed. Another segmentation method using non-symmetric Gabor filters and energy was also studied. In this case, sixteen Gabor filters are used to filter the original image, for the four directions, four center frequencies 0.1, 0.2, 0.3, 0.4, and for $\sigma_x = 6$ and $\sigma_y = 0.01$. The size of all images that are used for the experiments is 256 x 256.

Figure 1 illustrates an example where the properties presented in section 3 are verified. The image consists of four textures. Each of the four textures occupies four separate regions of the image. The segmentation algorithms using the FD vector and energy are applied for the original image and for the image that has been subjected to multiplicative noise. This noise has been applied as a multiplication of the grayscale values by a factor C, which is equal to C = 0.5 {[1+ (x-64)/64] + [1+(y-64)/64]}, where x, y, are the coordinates of the pixels. This factor compresses the grayscale values for regions closer to pixel with coordinates (64, 64) and stretches the grayscale values for regions far from this point. The segmentation results are similar for both the original and distorted images in the case that the FD vector is used. The percentages of correct clustering (PCC) are 92.57% and 93.00% respectively. Textures that are the same are clustered together even if they are subjected to multiplicative noise of different multiplication factor. The segmentation method that uses energy fails to cluster correctly the same textures of the distorted image.

The FD vector and the energy vector for symmetric Gabor filters are compared with respect to the segmentation results in cases that multiplicative noise is not present. The percentage of correct clustering (PCC) for FD vector is similar or a little greater than the PCC for energy in the case where the boundaries between different textures are smooth. The average PCC for all textured images that were tested was 95.35% for the case where FD vector is used and 93.36% in the case where energy is used. The PCC is better for the FD vector than for the energy in the case where the boundaries between textures have more irregular shapes.

In Figures 2 and 3, some more segmentation results are shown. In Figure 2 the segmentation result is better for the FD vector. The energy has been computed by applying the method that uses non-symmetric Gabor filters of constant bandwidth. Figure 3 shows that if the different texture regions have more irregular shape, FD works better than energy. The PCC for FD is 93.2%, and for energy is 88%. The energy for the segmentation results that are illustrated in Figure 3, has been computed by applying the method that uses symmetric Gabor filters of octave bandwidth.

The simple K-means algorithm is compared to iterative K-means algorithm. The iterative K-means algorithm uses window of variable size, as it was described before, for smoothing of the features. For the simple K-means two sizes for the smoothing window are chosen: small constant window of size 17 x 17 and large constant window of size 33 x 33. It can be noticed that the iterative K-means algorithm gives similar or better segmentation results than simple K-means algorithm. The iterative K-means gives much better segmentation results for more irregular boundary shapes. The simple K-means algorithm that also uses a large window for smoothing the features, works well if the boundaries between the textures are already smooth. If the boundaries are not smooth, the details at these regions are lost and the boundaries are smoothed out. The simple K-means algorithm that uses small smoothing window does not work well for the inner texture regions. The different feature extraction methods that use the FD vector, energy and filtering with symmetric Gabor filters, and energy and filtering with non-symmetric Gabor filters, are applied on different textured images, to compare the iterative and the original K-means algorithm with respect to the segmentation results. The average PCC is 95.78% for the iterative K-means algorithm, 93.92% for the original K-means algorithm and smoothing window of size 17 x 17, and 95.24% for original K-means algorithm and

smoothing window of size 33 x 33. Three examples are illustrated in Figures 4, 5 and 6. In Figure 4 it is shown that the details are lost at the boundaries for the case of the original K-means algorithm when a smoothing window of size 33 x 33 is applied. Also, in the case of the original K-means algorithm for a smoothing window of size 17 x 17 there is some extra misclassification at the upper left corner of the image. The PCC for the image in Figure 5, is 94.35% for the iterative K-means algorithm, 90.53% for the original K-means algorithm when a smoothing window of size 17 x 17 is applied, and 92.23% for the original K-means algorithm when a smoothing window of size 33 x 33 is applied. The PCC for the image in Figure 6 is 96.10% for the iterative K-means algorithm, 94.40% for the original K-means algorithm when a smoothing window of size 17 x 17 is applied, and 96.20% for the original K-means algorithm when a smoothing window of size 33 x 33 is applied.

Another case, where the image is subjected to uniform white noise is studied. Some results are shown in Figure 7. FD and energy were tested with respect to the segmentation results, for different values of the standard deviation of the noise. In this case, the segmentation results are similar for both features up to a standard deviation value close to 50.5. The results are a little better for the energy, for very large values of the standard deviation of the noise.

5. CONCLUSIONS

The performance of the FD features that have been computed over twelve filtered versions of the original image, is studied with respect to the segmentation results. It has been found that the FD vector is good for texture characterization since it is insensitive to the intensity of the image and to multiplicative noise. In this perspective, it is better than energy features. In any case, FD features give similar or better segmentation results than energy features. The advantage of FD features over energy features is greater for textures with more realistic boundary shapes. If the image to be segmented has been subjected to uniform white noise, FD features and energy give similar segmentation results if the standard deviation of the noise takes values below 50. For large standard deviation the segmentation results improve slightly for energy.

The performance of the clustering algorithm has been studied. The features need to be smoothened by averaging their values over a window, and by mapping the result to the central pixel. An iterative K-means clustering algorithm that uses smoothing window with variable size is compared to the original K-means algorithm. Before the original K-means algorithm is applied, the features are smoothened using a window of constant size. The iterative K-means algorithm gives similar segmentation results if the boundaries between textures are smooth, but it gives much better for more detailed boundaries.

6. REFERENCES

1. Anil K. Jain, and Farshid Farrokhinia, "Unsupervised texture segmentation using Gabor filters", *Pattern Recognition*, Vol. 24, No 12, pp. 1167-1185, 1991.
2. Dennis Dunn, and William E. Higgins, "Optimal Gabor filters for texture segmentation", *IEEE Transactions on image processing,* Vol. 4, No. 7, pp. 947-964, July 1995.
3. Andreas Teuner, Olaf Pichler and Bedrich J. Hosticka, "Unsupervised texture segmentation of images using tuned matched Gabor filters", *IEEE Transactions on image processing,* Vol. 4, No. 6, pp. 863-870, June 1995.
4. R. Bajscy, "Computer identification of visual surfaces", Computer graphics and image processing, Vol. 2, pp. 118-130, 1973.
5. P. C. Chen, and T. Pavlidis, "segmentation by texture using correlation", *IEEE Transactions on pattern analysis and Machine intelligence,* Vol. PAMI-5, pp. 64-69, Jan. 1983.
6. Michael Unser, "Texture classification and segmentation using wavelet frames", *IEEE Transactions on image processing,* Vol. 4, No. 11, pp. 1549-1560, November 1995.
7. Robin N. Strickland, "Wavelet transform methods for image detection and recovery", *IEEE Transactions on image processing*, Vol. 6, No. 5, pp. 724-734, May 1997.
8. B.B Chaundhuri, and Nirupam Sarkar, "Texture segmentation using fractal dimension", *IEEE Transactions on pattern analysis and machine intelligence,* Vol. 17, No. 1, pp. 72-77, January 1995.
9. B. Dubuc, C. Roques-Carmes, C. Tricot, and S. W. Zucker, "The variation method: a techique to estimate the fractal dimension of surfaces", *SPIE, Vol. 845, Visual Communications and image processing II*, pp. 241-248, 1987.
10. T. Kasparis, N.S. Tzannes, M. Bassiouni and Q. Chen, "Texture Description based on Fractal and Energy Features", *Computers and Electrical Engineering, Vol. 21, No. 1, pp 21-32, 1995.*

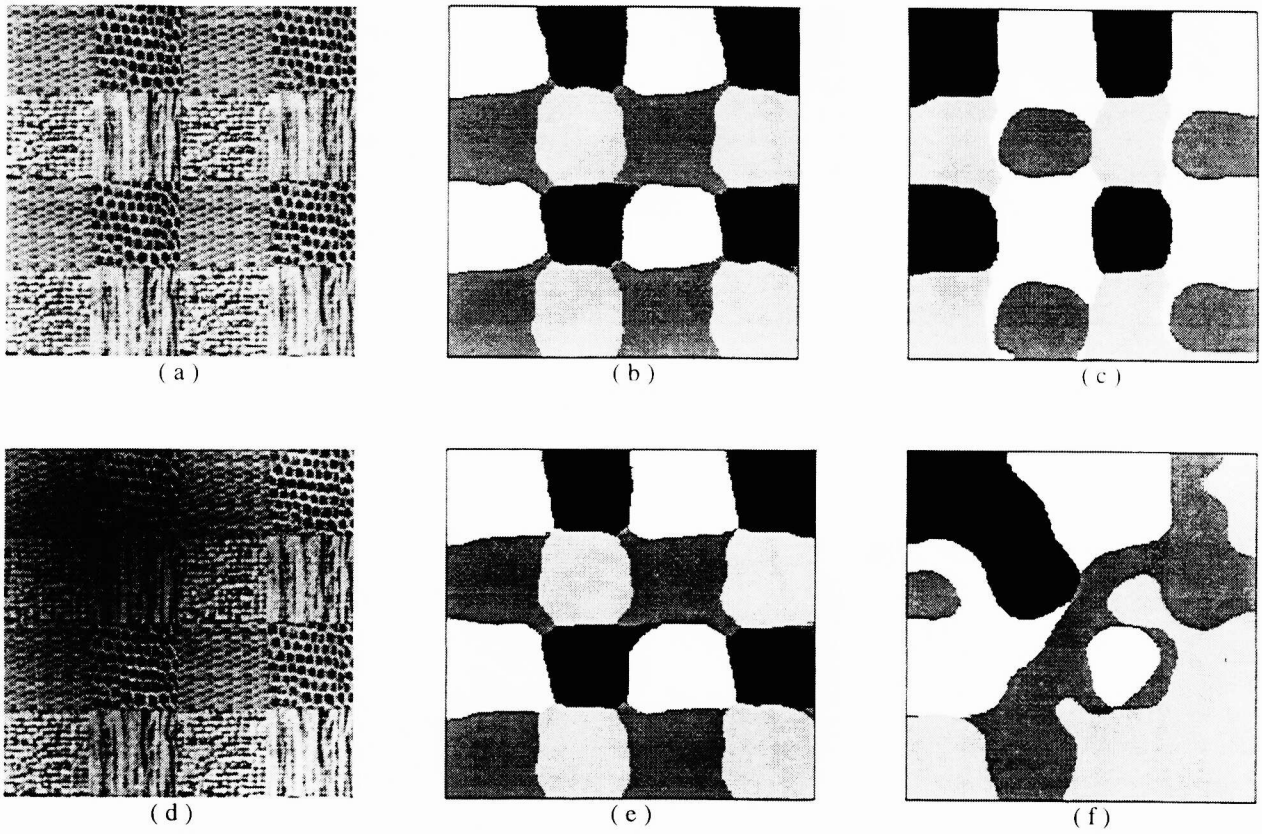


Figure 1: Comparison of FD and energy with respect to the segmentation of original image and of image subjected to multiplicative noise. (a) original image, (b) segmentation of the original image using FD, (c) segmentation of the original image using energy, (d) image subjected to multiplicative noise, (e) segmentation of the distorted image using FD, (f) segmentation of the distorted image using energy

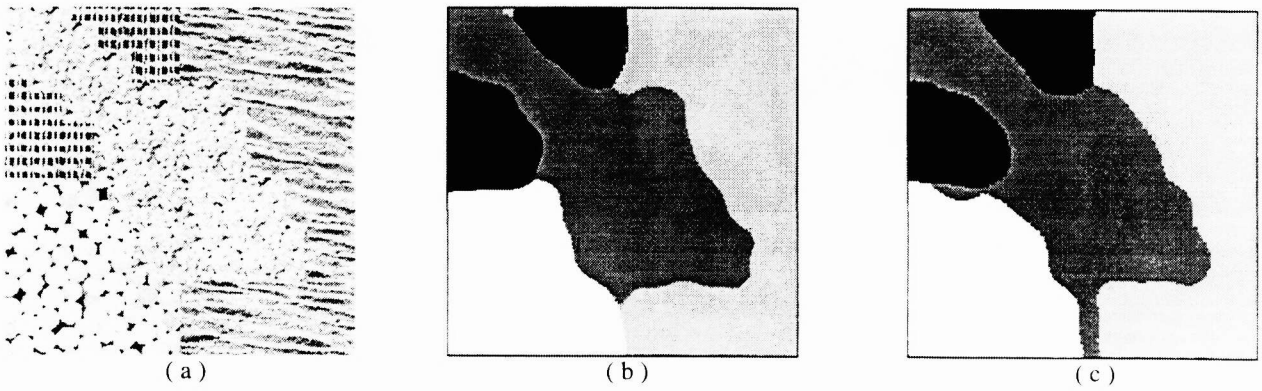
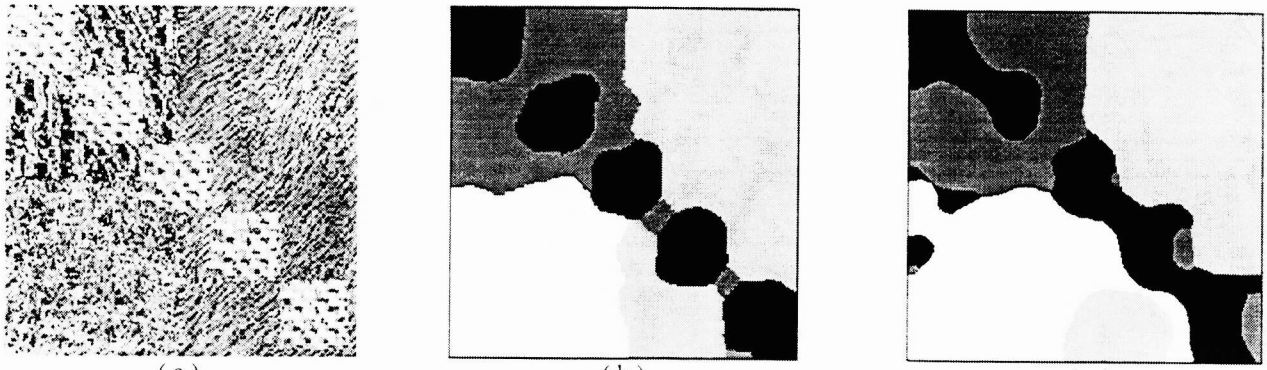
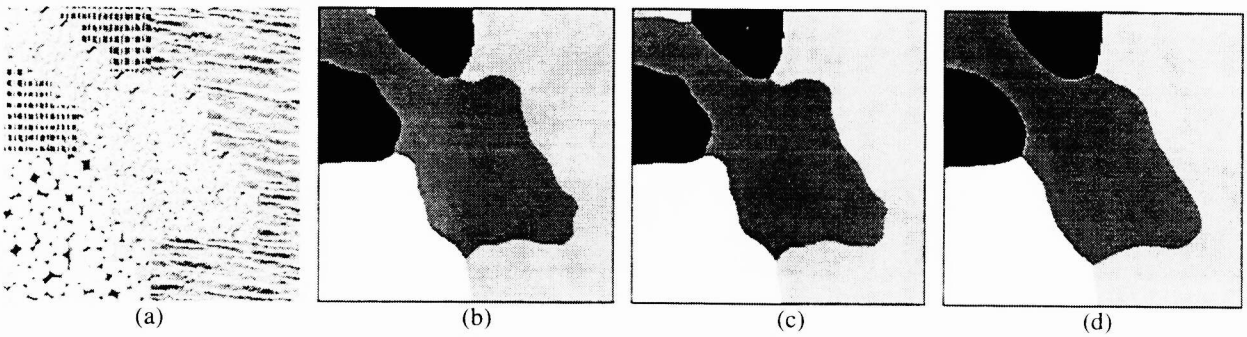


Figure 2: Comparison of FD and energy with respect to the segmentation of the original image (a) original image, (b) segmentation of the original image using FD, (c) segmentation of the original image using energy (for non-symmetric Gabor filters)



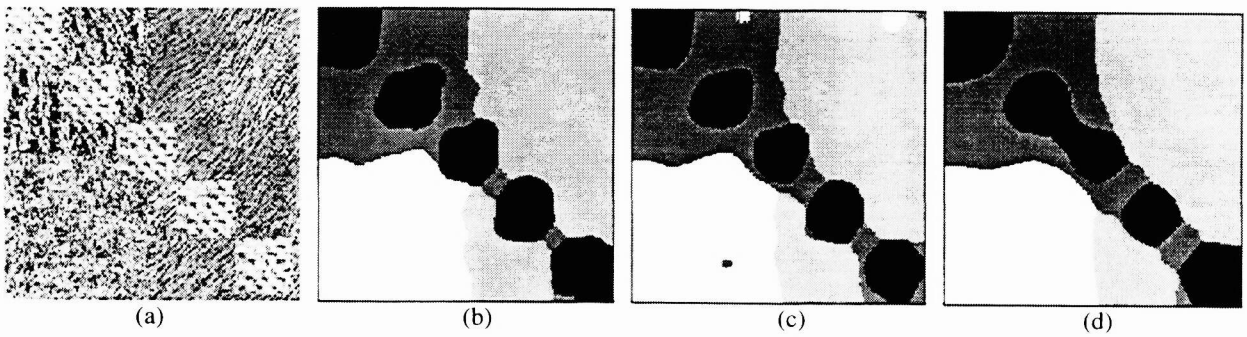
(a) (b) (c)

Figure 3: Comparison of FD and energy with respect to the segmentation of original image (a) original image, (b) segmentation of the original image using FD, (c) segmentation of the original image using energy (for symmetric Gabor filters)



(a) (b) (c) (d)

Figure 4: Comparison of the iterative and the original K-means algorithms. (a) original image, (b) segmented image using the iterative K-means algorithm, (c) segmented image using the original K-means algorithm and smoothing window of size 17 x 17, (d) segmented image using the original K-means algorithm and smoothing window of size 33 x 33. The FD feature vector has been used.



(a) (b) (c) (d)

Figure 5: Comparison of the iterative and the original K-means algorithms. (a) original image, (b) segmented image using the iterative K-means algorithm, (c) segmented image using the original K-means algorithm and smoothing window of size 17 x 17, (d) segmented image using the original K-means algorithm and smoothing window of size 33 x 33. The FD feature vector has been used.

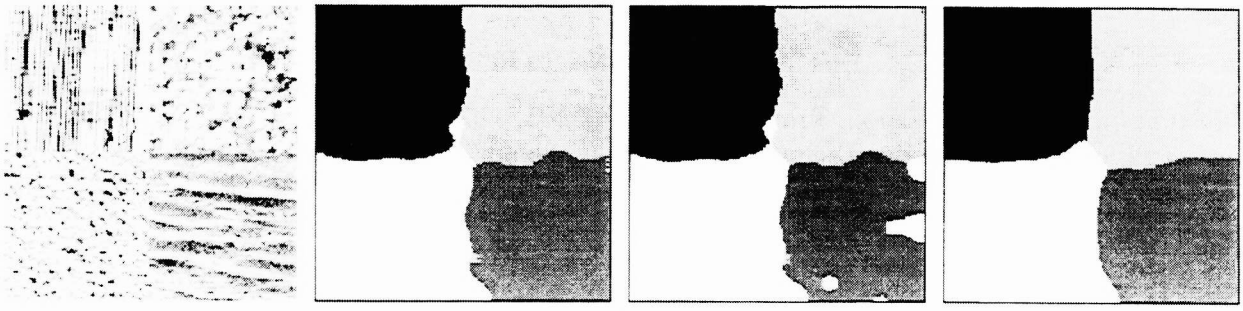
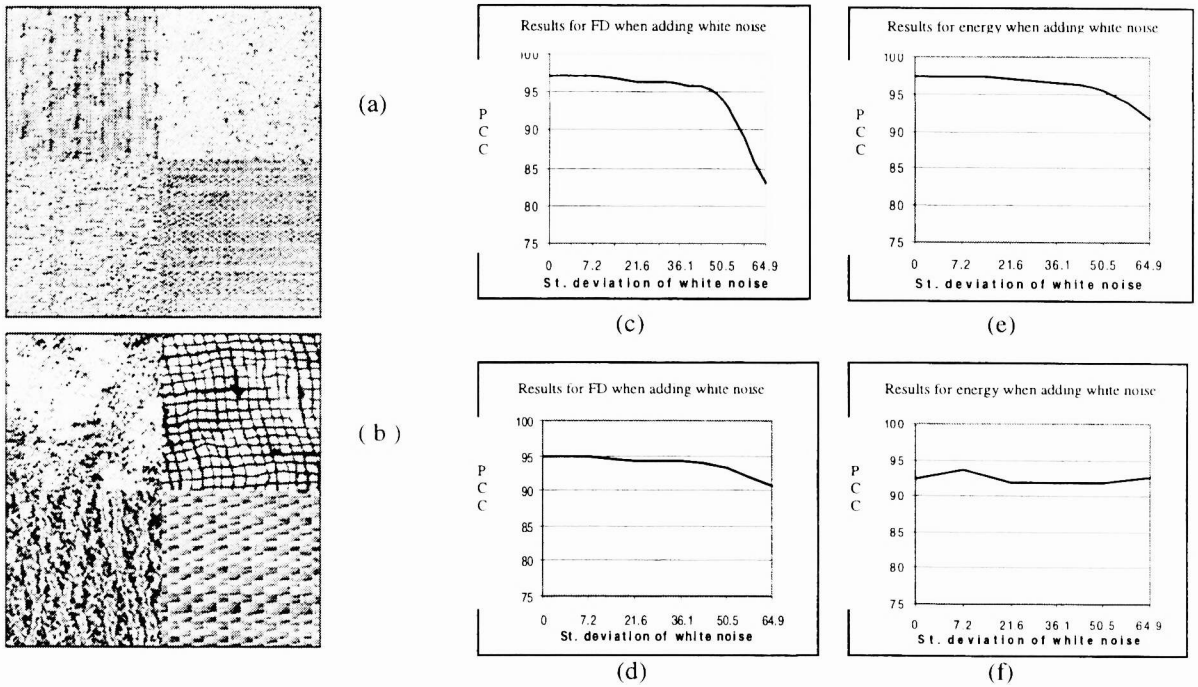


Figure 6: Comparison of the iterative and the original K-means algorithms. (a) original image, (b) segmented image using the iterative K-means algorithm, (c) segmented image using the original K-means algorithm and smoothing window of size 17 x 17, (d) segmented image using the original K-means algorithm and smoothing window of size 33 x 33. The FD feature vector has been used.



St. Deviation	RESULTS WHEN ADDING WHITE NOISE COMPARING FRACTALS AND ENERGY with respect to PCC			
	Image(a)		Image (b)	
	FRACTALS	ENERGY	FRACTALS	ENERGY
0	97.21	97.22	94.90	92.34
7.2	97.15	97.22	94.80	93.55
21.6	96.31	97.02	94.25	91.77
36.1	96.09	96.65	94.38	91.83
50.5	93.93	95.56	93.11	91.87
64.9	83.01	91.64	90.80	92.53

(g)

Figure 7: Comparison of FD and energy, when adding white noise to the image. (a), (b) original images, (c), (d) corresponding PCC with respect to the standard deviation of noise for FD, (e), (f) corresponding PCC with respect to the standard deviation of noise for energy, (g) table that contains the segmentation results.