Latin Character Recognition using Neural Networks

Jamal S. Majeed **Aseel W. Ali** **Amar S. Majeed**

jamal-alneamy@uomosul.edu.iq *aseelwaleed@uomosul.edu.iq*

College of Computer sciences and Mathematics
University of Mosul

Received on: 05/06/2007 **Accepted on: 15/08/2007**

ABSTRACT

The aim of this work is to recognize the printed Latin's characters. In this work two methods for constructing the feature space are used. These methods are Variance and Fractal dimension methods, as a result they have real values for every character in the Latin's language, and from these values they constructed the feature space extractions for every character in the Latin's language. After that, these features are given to the Back Propagation network for recognizing the characters.

The result is a highest recognition for the characters is obtained, it is about 82.75% characters while the unrecognized characters are 17.25.

Keywords: Pattern recognitions, Bag propagation, Feature extraction, Fractal dimension.

تمييز الحرف اللاتيني باستخدام الشبكات العصبية

جمال صلاح الدين مجيد **اسيل وليدعلي** **عمار صلاح الدين مجيد**

كلية علوم الحاسوب والرياضيات، جامعة الموصل

تاريخ الاستلام: 2007/06/05 **تاريخ القبول: 2007/08/15**

الملخص

إن الهدف من البحث هو تمييز الحروف اللاتينية المطبوعة في الحاسبة.

تم استخلاص الخواص للحروف اللاتينية بالاعتماد على مقياسين للتدرجات الرمادية وهما مقياس التباين، ومقياس البعد ألكسري للحرف، وفي النتيجة تم الحصول على قيم حقيقية لكل حرف في اللغة ومن هذه القيم تم بناء الخواص المستخلصة لكل حرف لاتيني، وأخيرا أَُعطي هذا الملف إلى شبكة الانتشار العكسي التي تعتبر من الشبكات التي تحتاج إلى معلم في طور التدريب ليتسنى لها تمييز الحرف المجهول في طور التمييز. تم تدريب الشبكة على 29 حرفا لاتينيا وتم الحصول على نسبة تمييز عالية تقريباً 82.75% قياساً بعدد الأحرف التي لم يتم تمييزها ونسبتها 17.25 %.

الكلمات المفتاحية: تمييز الانماط، التغذية العكسية، الصفات المستخلصة، البعد الكسوري.

1. Introduction

Pattern Recognition could be considered as one of the most important and widest branches in the field of Digital Image Handling, which was paid a lot of attention by several scientists and researchers, and many methods and techniques had been suggested in this field [4].

Optical character recognition, usually abbreviated to OCR, is a type of computer software designed to translate images of handwritten or typewritten text (usually captured by a scanner) into machine-editable text, or to translate pictures of characters into a standard encoding scheme representing them. OCR began as a field of research in pattern recognition, artificial intelligence and machine vision. Though academic research in the field continues, the focus on OCR has shifted to implementation of proven techniques [2].

Optical character recognition (using optical techniques such as mirrors and lenses) and digital character recognition (using scanners and computer algorithms) were originally considered separate fields. Because very few applications survive that use true optical techniques, the optical character recognition term has now been broadened to cover digital character recognition as well. Early systems required training (the provision of known samples of each character) to read a specific font. "Intelligent" systems with a high degree of recognition accuracy for most fonts are now common. Some systems are even capable of reproducing formatted output that closely approximates the original scanned page including images, columns and other non-textual components [2].

1.1 Overivew on OCR

In 1929, Gustav Tauschek obtained a patent on OCR in Germany, followed by Handel who obtained a US patent on OCR in USA in 1933. In 1935 Tauschek was also granted a US patent on his method (U.S. Patent 2,026,329). Tauschek's machine was a mechanical device that used templates. A photodetector was placed so that when the template and the character to be recognised were lined up for an exact match and a light was directed towards them, no light would reach the photodetector.

The United States Postal Service has been using OCR machines to sort mail since 1965 based on technology devised primarily by the prolific inventor Jacob Rabinow. The first use of OCR in Europe was by the British General Post Office or GPO. In 1965 it began planning an entire banking system, the National Giro, using OCR technology, a process that revolutionized bill payment systems in the UK. Canada Post has been using OCR systems since 1971. OCR systems read the name and address of the addressee at the first mechanized sorting center, and print a routing bar code on the envelope based on the postal code. After that the letters need only be sorted at later centers by less expensive sorters which need only read the bar code. To avoid interference with the human-readable address field which can be located anywhere on the letter, special ink is used that is clearly visible under ultraviolet light. This ink looks orange in normal lighting conditions. Envelopes marked with the machine readable bar code may then be processed.

2. Work description of the recognition system

The recognition system used in this work consists of several steps [Fig.1]:

2.1 Scanning.
2.2 Segmentation.
2.3 Normalization.
2.4 Feature extraction of character.
2.5 Neural Network.
2.6 Training (Learning) Phase.
2.7 Recognition Phase.

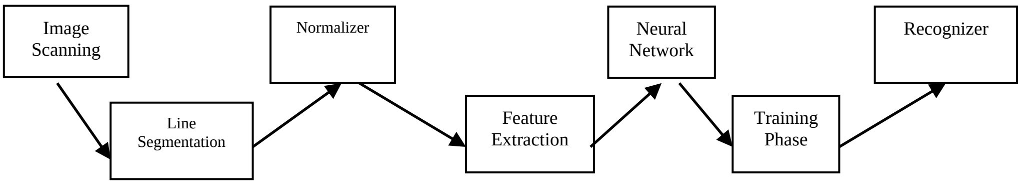

Figure 1: Work description of the recognition system

2.1 Scanning

The first step of this work is to scan the image that contains the characters, the scanned image is either used to build feature space or for the purpose of segmentation. The following steps are done in the scanning:

1- The image of the text is scanned using flatbed scanner which has the resolution 600 dpi.
2- The image characteristic is bitten to white (background) and black (foreground).
3- Then the image is saved as .bmp format.

2.2 Segmentation

Image segmentation is counted as one of the most important steps that used in image analysis for the purpose of shape recognition, and this is done by recognition of symmetric area in the image according to some properties of symmetry. Image segmentation is counted as the critical step in almost all vision problems, because there exists many factors that affect the quality of the segmentation, that is selecting the proper algorithms for feature extraction in addition to classification algorithms [6].

The segmentation is the process of separating the characters in the scanned image to be passed to be a recognition process. This process is done in two steps:

2.2.1 Line segmentation.
2.2.2 Cluster (Character) segmentation.

2.2.1 Line segmentation

In this process, the boundaries of each line of the text in the image is isolated. Line boundaries can be found by checking for the horizontal gaps in the image [1]. The horizontal gaps are found by a full row of pixels with zero value. The identified rows are then checked top down to determine the top and bottom of each text line. This process should consider the dots above and below the characters. Otherwise it gives wrong results (fig.2). In the coding this is done by making two loops, the boundaries of the loop are the dimensions of the image (width & length). The pixel line numbers that contain black (zero value) pixels are saved in matrix to be used in the second step (the horizontal segmentation).

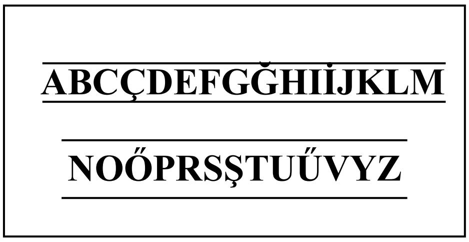

Figure 2: Line Segmentation (the text contains 2 lines)

2.2.2 Cluster (Character) segmentation

In this step the boundaries of each letter is identified in a separated line [1]. Word boundaries are found by looking for vertical gap in the segmented line and checking them to identify the beginning of words, but unfortunately this process may also isolate portions of word [fig 3].

In this portion two loops are used, the counters reach the length and the width of the line segmented in the vertical segmentation step. The segmented line is scanned vertically and the number of the columns that contain black pixel (zero values) is recorded in a variable, this variable is then used to segment each cluster.

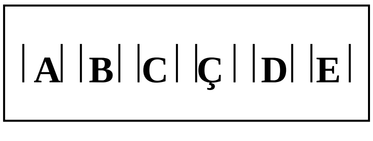

Figure 3: Cluster Segmentation

The Algorithm of the segmentation is working as follows:

A- Line segmentation:

```
1- Restore the counter I=0
2- Do the following until I=linenumber
     begin
        increment the counter I=I=1
        restore the counters b1=0 , linehight=0
3-      compute the beginning of each line
                j==0
                Do while (j<endline)&&(b1==0)
                If page(I, j)<>0 then
                        b1=I
                endwhile
4-      compute the ending of the line
                n=0
                Do while (n<linenumber) && (linehight==0)
                        b1=b1+1
                        if page(I, n) ==0 then
                        I=I+n
                endwhile
        endwile
```

B- Cluster segmentation

```
                1-   restore the counter r=0
                2- for m= b1 to linehight do
`                       r=r +1
                        for j=0 to endline do
                         copying the line into an array named XX
                           XX(r, j)= page(m j)
                3- for k=0 to r do
                        restore the counters b2=0, linewidth=0
                4- finding the beginning of the coulmn
                        for m=0 to endline do
                          if XX(m, k)<> 0 then
                           save the beginning of the column to a variable
                            b2=m
                           if (XX(m, k)==0) && (b2<>0) then
                           save the ending of the character to a variable
                             end2=m
```

```
5-  Saving the pixels of each character
      r2=0
         for m=b1 to linehight do
          r2=r2+1
          for mm=b2 to linewidth do
            X(r2, mm)= page(m, mm)
```

Note that the array X now has all the data needed to pass to the feature extraction.

2.3 Normalization

After the segment process the bitmap array into a plurality of separate bitmap arrays, these separate bitmap arrays are fed to the normalizer to scale_down the segments obtained to pre-determined sizes [1].

2.4 Feature Extraction

Another important step before recognition is the feature space construction which will be used as a standard in the recognition phase [9]. This will contain all the letters that are presented in the language, the feature space is constructed after the page which is containing the character(s) is scanned and converted to a black and white image (Note that the image that is used in this work is in .bmp format because it hasn't any compress and no lossy).

There are several methods to generate the feature space for the character; two methods of them are used in this work that is [fig.4]:

2.4.1 Variance
2.4.2 Fractals

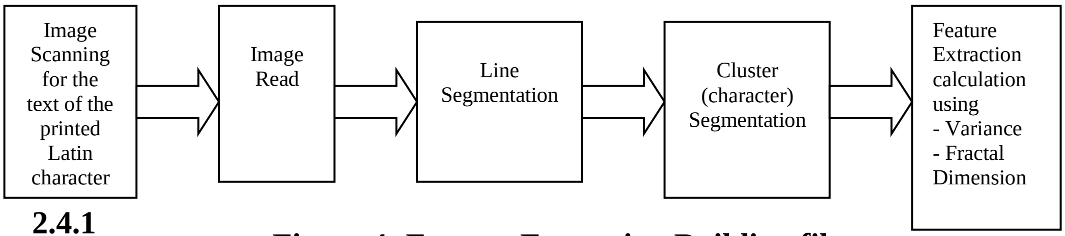
Image Scanning for the text of the printed Latin character
Image Read
Line Segmentation
Cluster (character) Segmentation
Feature Extraction calculation using
- Variance
- Fractal Dimension

Figure 4: Feature Extraction Building file

2.4.1 Measures of Variance

The Variance is a measure of how spread out a distribution is. In other words, it is a measure of variability.

The Variance is computed as the average squared deviation of each number from its mean.

The formula (in summation notation) for the Variance in a population is:

$$\sigma^2 = \frac{\sum_{i=1}^{n} (X_i - \mu)^2}{N}$$

Where μ is the mean and N is the number of scores.
When the Variance is computed in sample, the statistic :

$$S^2 = \frac{\sum_{i=1}^{n} (X_i - M)^2}{N}$$

where M is the mean of the sample can be used. S^2 is a biased estimate of σ^2, however, by far the most common formula for computing Variance in a sample is:

$$S^2 = \frac{\sum_{i=1}^{n} (X_i - M)^2}{N-1}$$

which gives an unbiased estimate of σ2. Since samples are usually used to estimate parameters, S^2 is the most commonly used measure of Variance .

Calculating the Variance is an important part of many statistical applications and analyses.[3]

- Experimental Result:

The Variance of the letter (Ç) (after input the array X which has each pixel in the character) is 0.748.

2.4.2 Fractals

The fractal Dimension depends on the property of self similarity, that means, if a shape is divided into a number of parts, each part is similar to base shape, a predetermined scale is used to segment the shape into parts depending on the scale and the number of parts which has the calculation of fractal dimension for the shape [10].

There are many methods for calculating the fractal dimension [7]:

1. Box counting method (used for binary images).
2. Two dimension variation method (used for gray images).

2.4.2.1 Box Counting Procedure

A comfortable estimator for the fractal dimension of a binary image is the box dimension [5]. The image can be covered with a grid of square cells with cell size r. For binary images the cell size is expressed as numbers of pixels. Fig. 5 shows the Sierpinsky gasket stored as a **688*612** matrix overlaid with a grid of squares. The number of grids containing a part of the structure is **N=51**.

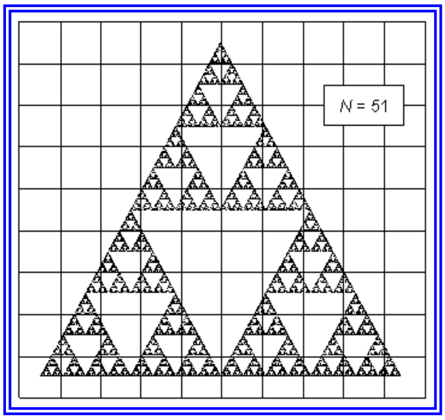

Figure 5: Sierpinsky gasket overlaid with a grid of squares

The number **N(r)** of squares needed to cover the structure is given by a power law:

$$\mathbf{N(r) = Const\,.\,r^{-D_B}} \qquad \text{..... (1)}$$

where $\mathbf{D_B}$ is the box dimension.

The image processor uses a modified algorithm which determines the area of cells containing parts of the structure [fig. 6]. Using equ. (1) the total area **A** covered by the squares of size **r** is:

$$\mathbf{A(r) = N\,.\,r^2 = Const\,.\,r^{-DB}\,.\,r^2 = Const\,.\,r^{2\,-DB}} \qquad \text{.....(2)}$$

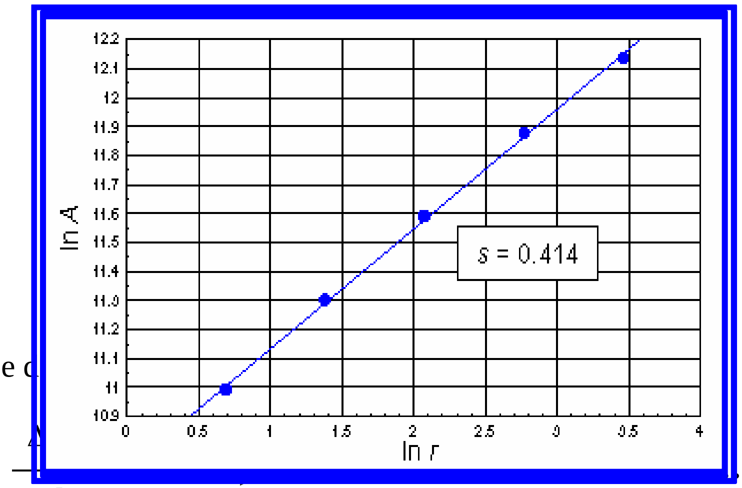

Figure 6: Richardson-Mandelbrot plot for the area of a binary structure at different box sizes (in pixels)

$\mathbf{D_B}$ can be c

$$\mathbf{2 - D_B = \frac{\Delta}{\Delta\,ln\,r}} \qquad \text{..... (3)}$$

This matches nicely with the Hausdorff-Besicovitch and self-similarity dimension, respectively, which is $\mathbf{D_H = D_S = 1.585}$ [5].

- Experimental Result:

The fractal dimension of the letter (Ç) (after input the array X which has each pixel in the character) is calculated using box counting method in normal size 14, Times New Roman, the result is as follows:

Character shape	Character size	Fractal Dimension
Ç	14	1.2306

2.5 Neural Network

One of the original aims of the artificial neural network was to understand and shape the fractional characteristics and computational properties of the brain when it performs cognitive processes such as sensorial perception, concept categorization, concept association and learning. However, today a great deal of effort focused on the development of the neural networks for the applications such as pattern recognition and classification, data compression and optimization [8].

2.6 Training (Learning) Phase

The Back Propagation network is a learning network because it is a guaranty to converge and it has a stability, also it is multilayer, used a sigmoid activated function, and the initial weight for this network is random. The Bp network is sensitive for the error because it is a gradient rule and the rate of converge is a linear [9].

The BP consists of input, hidden, output layers [fig.7]. The input layers depend on the problem itself, in this work, they have two nodes due to the two values of the characters feature (the Fractal and the Variance) that entered to the network.

The hidden layer is also two nodes because it depends on the input layer, it should be less than or equal to the input nodes.

The output layer consists of 29 nodes depending on the number of Latin's characters to be recognized.

The network is training on 29 characters each of them has two values which are entered to the network simultaneously, the weight is computed many times until it is stabilized for all the characters.

At that time, the testing phase is beginning where the two values of feature extractions are entered for each lateen's character, then the BP network try to recognize this character automatically.

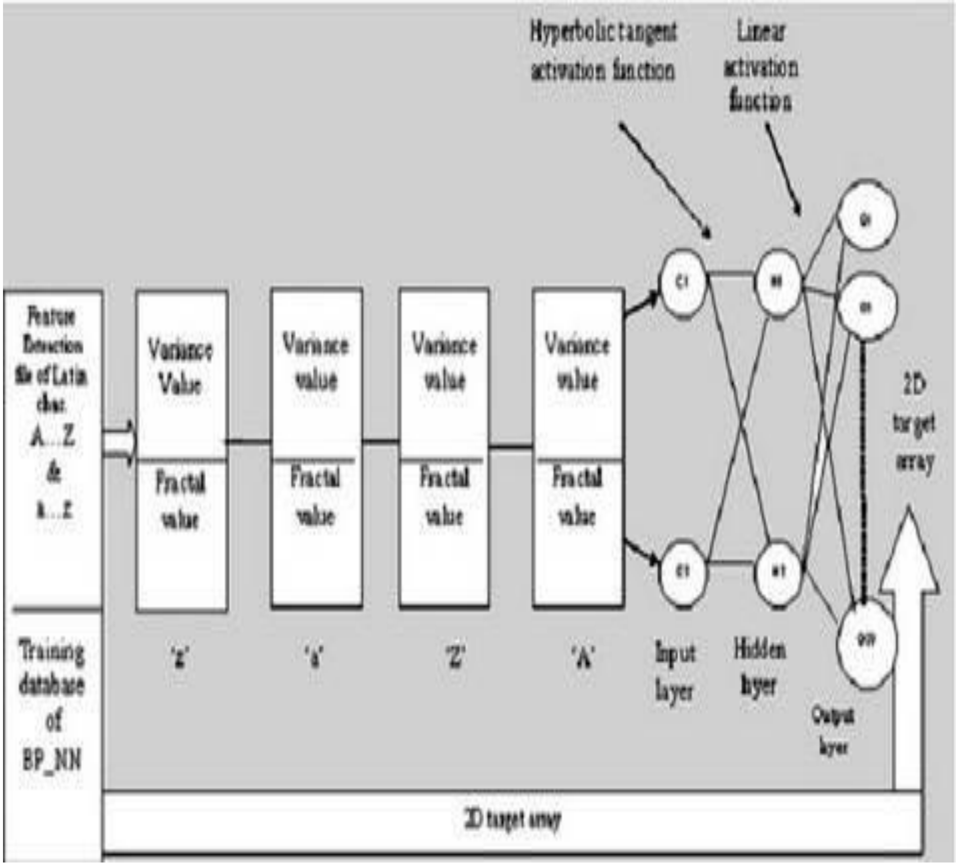

Figure 7: Back Propagation Neural Network of 2--29 (training phase)

2.7 Recognition Phase

The recognition phase is based on calculating two measures which are: fractal dimension, and Variance for each single character, then using these values to recognize the characters by comparing their values with those which are presented in the feature space, also the checking process is done using these three measures [Fig.8].

The three measurements are calculated for each character and checked with the presented feature space for the character using minimum distance algorithm which its equation is as follows:

$$Z_i = \sum_{i=1}^{n} \sqrt{X_i^2 - Y_i^2}$$

All the calculated values of minimum distance (z) are saved in a matrix, this matrix is checked for the minimum value, after that the index of the minimum value was taken and matched with the file that contained the index of each letter.

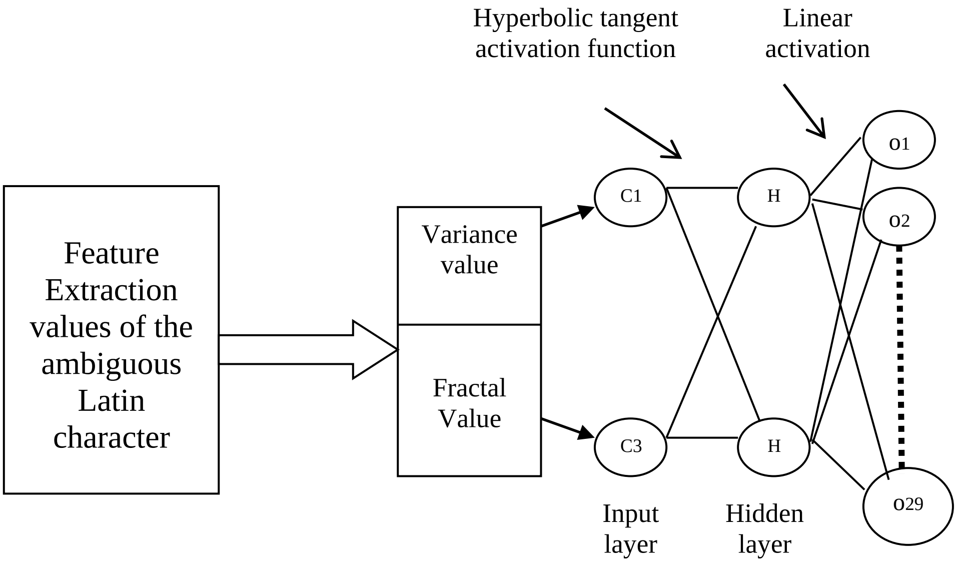
Hyperbolic tangent activation function
Linear activation
Feature Extraction values of the ambiguous Latin character
Variance value
Fractal Value
C1
C3
H
H
o1
o2
o29
Input layer
Hidden layer

Figure 8: Back Propagation Neural Network of 2-2-29 (testing phase)

3. Results

The Latin alphabet is composed of 29 letters, it has all the letters in the English alphabet, except “Q”,”W”, and “X”. in addition, it has the characters “Ç”, “Ğ,’ “İ”, ”Ő”, “Ş”, and “Ű”.

A text of Latin characters is an input to the recognition system as an image using the scanner, this text was segmented separately, then the feature space was constructed for each character using three measurements, after that the three matrices will be entered to the BP network to be recognized.

The results of this work is as follows:

1- Number of recognized characters = 24.

2- Number of unrecognized char. = 5.

Input text:	ABCÇDEFGĞHIİJKLMNOŐPRSŞTUŰVYZ
Trained Text :	ABCÇDEFGĞHIİJKLMNOŐPRSŞTUŰVYZ
Output (Recognized)Text:	**Λ**BCÇD**F**FGĞHI**I**JKLMNOŐP**P**SŞTUŰV**V**Z

4. Conclusion

1- Highly Accurate recognizes even the most complex of document.

2- Easily guides you through the OCR process. A useful tool for the infrequent OCR user.

3- Feature extractor is that for machine printed generate fractal dimension and Variance.

4- The highest percentage of correctly classified characters was 82.75%

5- We have used Neural Network (BP) for recognizing different Latin's character, the experimental results as shown above, show that the high performance is graceful and predictable. We can conclude that our Neural Network approach to recognizing Latin's characters is proved as a viable concept.

REFERENCES

[1] Al Ne'amy, J.S. (1996) "Pattern Recognition approach for Arabic letter processing", M.Sc. Thesis, **Department of Software Engineering**, University of Mosul..

[2] Bazzi, I. and Schwatz, R. (1999) "an Omnifont Open-Vocabulary OCRsystem for English and Arabic". **IEEE Transactions on pattern analysis and machine intelligence**. Vol. 21, No.6, PP. 495-504.

[3] Giudici, Paolo (2003) **Applied data Mining**, Faculty of Economics, University of Pavia, Italy.

[4] Ian, Turton (1997) "Application of Pattern Recognition to Concept Discovery in Geography", **Pattern Recognition**, Vol.25, PP. 210-273.

[5] Kraft, R. (1995) "Fractal and Dimensions", **HTTP-Protocol at www. weihenstephan.de.**

[6] Laws, K. I. (1980) "Texture Image Segmentation", Ph.D. Thesis, **University of Southern California**.

[7] Neary, D. (2002) "Fractal Method in Image Analysis and Coding", M.Sc. **for Dublin city University**, Dublin, (Ireland).

[8] Rumelhart, D.; Hintor, G. and Williams, R. (1986) "Learning Representations by Back_propagating Errors", **Nature, 323**, PP.533-536.

[9] Sharma, M. (2000) "Texture Analysis", Ms.c. of Philosophy in Computer science, **Univ. of Exeter**, New York.

[10] Voss, R.F. (1998) "Fractal in Nature: from characterization to simulation", H.O. Peitgen and D. Saupe, Editors, The Science of Fractal Images, Springer Verlag, New York.