Combining Snakes and Neural Networks for Off-Line Signature Verification

José F. Vélez, Ángel Sánchez, Ana B. Moreno, José L. Esteban

Combining Snakes and Neural Networks for Off-Line Signature Verification

José F. Vélez Ángel Sánchez Ana B. Moreno José L. Esteban
Grupo de Algorítmica para la Visión Artificial y la Biometría (GAVAB),
Escuela Superior de Ciencias Experimentales y Tecnología, Universidad Rey Juan Carlos, 28933 Madrid.
jose.velez@urjc.es, angel.sanchez@urjc.es, belen.moreno @urjc.es, joseluis.esteban@urjc.es

Abstract

This paper introduces an improved snake algorithm based on the work by Kass et al. Our approach is applied to the off-line signature verification problem where signatures are scanned and then converted into binary images. This way no dynamic information of the signers is available. We also have considered some real conditions for the verification problem when applied to bank check. For example our system uses only one training signature per subject. Involved system parameters are tuned to solve the task in an effective and efficient manner. A two-layer perceptron is build for signature classification and it uses only two signature features (distance and matching factor) provided by the adjusted snake. Finally, a study of the system for a signature database is provided.

Keywords: off-line signatures, verification, snakes, neural networks.

1. Introduction

Signatures are a special case of handwriting subject to intra-personal variations and inter-personal differences. As stated by Justino et al [10]: "signature verification problem aims to maximize the interpersonal differences and to minimize the intrapersonal differences". This variability makes necessary to analyse signatures as complete images and not as collection of letters and words. Human signatures provide secure means for authentication and authorisation in legal and banking documents. Therefore, the need of research in efficient automatic solutions for the involved signature recognition and verification problems has recently increased. In the signature recognition (or identification) problem, a given signature is searched in the database to establish the signer's identity. Signature verification problem is concerned to determine if a particular signature is genuine or it is a forgery. Techniques for solving both the recognition and verification problems can be classified as on-line and off-line. In the first ones, data are adquired using an electronic tablet and other devices, and in the second ones, the images of signatures written on a paper are scanned and then dynamic information is not available.

Automatic signature verification is an active research area since 1975 [12]. There are many related works concerned with the location of a signature in a noisy environment [18]. Others consider the problem of having only one signature per writer during the system training stage [7], the scalability problem [4], or those problems produced by the skilled forgeries [6], among others.

Generally, the stages in a signature verification system are: a signature pre-processing stage, a segmentation task, a feature extraction stage, and finally a classification algorithm based on the extracted features [12]. In general, the proposed techniques use either a type of features (global [20], local, statistical, geometric [21], etc) or a combination of different types of features, extracted from the signature images. Some of the most used signature features are: centres of gravity, baselines, upper and lower signature limits, number of holes, signature skeleton, bounding box, signature contour or perimeter, major and minor signature axis, area-to-perimeter ratio, density of points the different signature regions, slant angle, number of signature strokes, crossing points, and so on [19].

Among the many referenced off-line signature verification techniques, HMM-based approaches [10][4], fuzzy logic [9], NNs and SVMs [2][15], genetic algorithms [14], elastic graph matching techniques [6] or optimal displacements functions [7] can be mentioned.

In short, the off-line signature verification problem has been considered by many authors under controlled conditions with promising results. However, we are far away from an automatic verification system which performs this task under practical conditions with the same effectiveness of a human minimally trained for solving the considered problem.

The paper is organised as follows. Section 2 illustrates the practical off-line signature verification problems. Section 3 describes the signature database used in the experiments. In section 4, we give an overview of the snakes applied to signature images. Section 5 describes the proposed *snake*-based algorithm for automatic off-line signature verification. Section 6 describes the experimental results produced by a neural classifier which uses the features extracted by the *snake's* method. Finally the last Section outlines the conclusions and describes the future work.

2. Signature verification problems

Real problems involved in off-line signature verification can be classified in two main categories: (a) those related to the extraction of the signature from the document and (b) those derived from the verification task itself. The first group includes problems originated

by the need of segmenting the signature from the image document. In general, there is a lack of knowledge about the exact position of the signature in the document. Other related problems are the presence of white noise (caused by document scanning) or the existence of structured noise (caused by textures and logotypes in the document background, or by stamp superposition or typed text mixed with the signature among others). Figure 1 shows some examples of situations where the signature segmentation becomes a difficult task.

(a) (b) (c)

Figure 1. (a) and (b) noised and textured background, (c) typed text and stamp superposition.

Among the problems related to the own verification process, some authors have remarked the lack of sufficient signature samples from each writer for training up the system. This difficulty is a very realistic problem due to the private nature of signatures. It would also be desirable to collect signatures from a writer in different periods of time to capture the intrapersonal signature variations. Moreover, with an insufficient number of training samples, the characterization of interpersonal signature differences among writers could result unreliable. Other related signature verification problems are the scalability of the system when adding new writers, the acceptable response time of the system when automatically processing a large amount of documents, and how to consider FAR (False Acceptance Rate) and FRR (False Rejection Rate) values depending on the particular application for the system.

It is also very important how to work with the presence of forgeries. These can be roughly classified in two groups: skilled and simple forgeries [12]. The first group corresponds to the falsifiers who know the subject's signature and can reproduce this signature with a high quality. The second group corresponds to those falsifiers who have never seen the subject's signature and therefore they can only reproduce with a very low degree of fidelity with respect to the original signature. It is interesting to observe that about 95% of related bank fraud corresponds to simple forgeries [8].

We have analysed these real practical problems when designing and implementing our off-line signature verification system.

2.1. Practical requirements of an off-line signature verification system

A first functional requirement is that each system client can not be required for signing many times. Of course, a reasonable number of signatures per writer would be very desirable for the system training stage. To have a more realistic scenario, we have considered that only one original signature for each system user will be available. However, it is reasonable to consider that this signature is captured under controlled conditions of position and reolution.

Another important requirement is the system scalability with respect to new clients whose signature needs to be verified. Additionally, the system needs to be independent with respect to spatial and radiometric signature resolution conditions.

Finally, high accuracy and fast response time requirements are also demanded by a practical verification system. Also, the approach needs to be adapted to the particular conditions of the application area. For example, in bank environments is preferable that the system accepts a false signature (FAR error) to the rejection of a true signature (FRR errors), whenever the economic cost of the error is low.

3. Signature database

Due to the private nature of a signature, there is a lack of standard databases referenced in the literature. For this reason, we have created our own signature database that can be downloaded from the following URL: http://gavab.escet.urjc.es.

This database has actually the signatures of 56 individuals with 6 samples per writer. These patterns were acquired in different periods of time and also using different types of pens. Signatures were scanned as binary images with a spatial resolution of 300 dpi and stored as BMP files. Fig. 2 shows some samples of our signature database.

Figure 2. Four signature samples from two different subjects.

4. S*nake* overview

A *snake* [11][5] is a type of *Active Contour Model* [7], based on the analysis of the movement of a closed or open contour over an image to which it tries to adjust. An associated energy function, with both internal and external constraint components, is associated to the snake. The internal energy is due to some restrictions of elasticity and flexibility imposed to the snake shape. The external energy component is caused by the influence of the image which guides the snake movements. The aim is to minimize the snake energy which is attracted by the image features such as time or edges.

A parametric representation is generally used to describe the snake position on a two-dimensional image. In this way, this position can be represented as an ordered list of points: $v(s)=(x(s),y(s))$, and the snake energy as:

$$E_{snake} = \int_0^1 E(v(s))ds =$$

$$\int_0^1 \left[E_{\text{int}}(v(s)) + E_{imagen}(v(s)) + E_{cons}(v(s))\right]ds \qquad (1)$$

where s represents a positional parameter in the snake, E_{int} represents the energy due to the relationships among

the snake points, E_{image} represents an energy related to the positions of the *snake* points on the image, and finally E_{cons} considers the energy associated to other external conditions.

4.1. Internal energy

For many authors [11], the term E_{cons} is not considered in eq. (1), and for E_{int} the following equation is proposed:

$$E_{\text{int}} = \frac{1}{2}\left(\alpha(s)\left[x_s^2(s) + y_s^2(s)\right] + \beta(s)\left[x_{ss}^2(s) + y_{ss}^2(s)\right]\right) \qquad (2)$$

In (2), x_s and y_s represent the respective first derivatives with respect to the positional parameter s. Similarly, x_{ss} and y_{ss} are the respective second derivatives. Coefficients $\alpha(s)$ and $\beta(s)$ respectively correspond to the influence of the *snake* elasticity and flexibility. High values of $\alpha(s)$ produce the *snake* shape shrinks. High values of $\beta(s)$ produce that the *snake* shape tends to be smoother.

Different *snake* formulations have been proposed to solve related snake problems under particular conditions [5]. In general, all the models are based on spring-like equations.

4.2. Image energy

An energy function is defined for the image where the *snake* is placed and it moves. This function assigns a value to each image pixel in such a way that the lowest energy values are assigned to the desired final position of the *snake*. The gradient image is obtained to give the lowest energy values to those pixels with highest gradient in order to promote the *snake* displacements in a gradient descent direction.

The original *snake* formulation by Kass et al [11] considers different terms that respectively attract the active contour model to the, previously segmented, image lines, edges and termination pixels. Thus, the image energy E_{image} was formulated as:

$$E_{image} = E_{lines} + E_{edges} + E_{term}$$

The first term of the previous equation is proportional to the own image function $I(x,y)$:

$$E_{lines} = w\, I(x,y) \qquad (3)$$

To attract the snake towards image edges, the following gradient function is proposed:

$$E_{edges} = -\left|\nabla I(x,y)\right|^2$$

Finally, to attract the *snake* towards the ending image points, a curvature function is defined on the smoothed image C that is obtained using a Gaussian filter:

$$E_{term} = \frac{\partial \theta}{\partial n_{\perp}}$$

where: $\theta = \text{arctg}(Cy/Cx)$ and $n_{\perp} = (-\text{sen}(\theta), \cos(\theta))$

4.3. Minimizing the *snake* energy

When searching a minimum of the snake energy, several assumptions are established to simplify the problem. First, the contour is discretized and it is represented by a sequence of n control points $\{v_i\}_{i=1}^{n}$ that define a *spline* or even a simple polygon. Second, the coefficients of the internal energies, $\alpha(s)$ and $\beta(s)$, are converted into constants. Third, the equation (1) is transformed into an iterative formulation. This allows the computation of the successive snake positions in time control points $\{v_i(t)\}$, thus obtaining a *variational problem*.

Amini et al. [1] have proposed a *dynamic programming* algorithm to avoid the expensive evaluation of the multiple positions associated to each of the *snake* movements. We have used a modification of this approach in our implementation. According to this formulation, the energy defined by the equation (1) can be decomposed into successive stages according to:

$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$

The computation of each energy term can also be decomposed, using the dynamic programming scope, into a sequence of functions of a variable s_i where the set $\{s_i\}_{i=1}^{n-1}$ is obtained by the following system of recurrence equations:

$$s_1(v_2) = \min_{v_1} E_1(v_1, v_2)$$

$$s_2(v_3) = \min_{v_2}\left\{s_1(v_2) + E_2(v_2, v_3)\right\}$$

$$s_3(v_4) = \min_{v_3} s\left\{_2(v_3) + E_3(v_3, v_4)\right\}$$

$$\ldots$$

$$\min_{v_1,\ldots,v_n} E(v_1,\ldots,v_n) = \min_{v_{n-1}}\left\{s_{n-2}(v_{n-1}) + E_{n-1}(v_{n-2}, v_{n-1})\right\}$$

To evaluate the complete equation (2), we have to consider that computing the second derivative involves a third point. The energy term can be rewritten as follows:

$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2, v_3) + E_2(v_2, v_3, v_4) + \ldots + E_{n-1}(v_{n-2}, v_{n-1}, v_n)$$

where: $E_{i-1}(v_{i-1}, v_i, v_{i+1}) = E_{img}(v_i) + E_{int}(v_{i-1}, v_i, v_{i+1})$ (4)

In consequence, the set $\{s_i\}_{i=1}^{n-1}$ will be expressed as:

$$s_i(v_{i+1}, v_i) = \min\left\{s_{i-1}(v_i, v_{i-1}) + \alpha\left|v_i - v_{i-1}\right|^2 + \beta\left|v_{i+1} - 2v_i + v_{i-1}\right|^2 + E_{ext}(v_i)\right\}$$

5. Proposed signature verification method using *snakes*

The signature verification problem consists in minimizing the adjustment energy applied to the *snake*, obtained from a model signature, and the image of the test signature.

The proposed verification algorithm can be resumed in the following steps:

1.- For each known model signature, create a polygonal line (*snake*) P, composed by equal-spaced control points, using a model signature image.

2.- Place approximately P over a signature to verify.

3.- Use the considered *snake* algorithm to adjust P, as best as possible, to the new image.

4.- Compare a measure of the energy necessary to deform the *snake* after the adjustment process. This value is compared with a threshold and is used to decide whether or not the test signature is genuine or is a forgery.

5.1. Initial adjustment method

As a first approximation, the original *snake* definition by Kass et al [11] was used. As the signature images are binary, finding the edges and ending image points makes no sense to adjust the *snake*. Only the energy associated to each image pixel, represented by eq. (3), was considered to attract the *snake* towards the signature image. For simple images, the results were promising since the snake adjusted accurately in a few number of iterations to the signature image. However, this model has two main problems: the influence of the position where the *snake* was initially placed and the significant lost of the original shape of the *snake*.

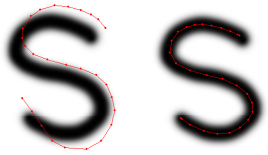

Figure 3. Adjustment of a snake with parameter values (α,β,ω) = (0.1,1.0,-10.0), where α,β and ω appear in eqs. (2) and (3).

If the *snake* was not very close to the test signature, the algorithm usually failed. Besides, if the signature image was not found after a small number of iterations, the original *snake* lost its original shape. To solve these problems, a Gaussian smoothing was performed over the signature image to increment the region of influence of the signature edges. Fig. 3 shows the results of this initial approach over a 'S' image.

This method is now detailed. The polygonal line (*snake*) P is manually obtained from the strokes of the model image signature. Besides, the centres of gravity C and C', corresponding respectively to the black pixels of the training and test images, are computed. Next, P is placed on the signature image to verify and both centres of gravity are made coincident. Finally, P is proportionally rescaled to make coincident both widths corresponding to the *snake* and to the signature-to-adjust.

After different experiments, we noticed that this first method became not practical for real signatures since it was only useful in the signature regions affected by the Gaussian filtering.

5.2. Improved adjustment method

As a second approximation, a new *snake* energy definition E'_{snake} was adapted with two considered aspects. First, to solve the snake locality problem, a *potential map* for the energy term E'_{image} associated to the test image was defined. Second, to avoid a significant lost of shape of the *snake*, the term E'_{shape} was introduced to maintain the *snake* shape. Now, E'_{snake} can be represented as:

$$E'_{snake} = E'_{image} + E'_{shape}$$

Potential maps

Cohen and Cohen [16] have proposed the use of an external force, based on the idea of *potential map*, defined as the Euclidean distance in pixels $m_{image}(x,y)$ from each image point to the nearest signature point. In this way, the algorithm first computes the distance values beginning from each of the adjacent points to the signature strokes and then moving away. Fig. 4 shows the potential map corresponding to an example signature. The intensity level associated to each image point reflects the distance to the signature pixels. Using the potential map, the energy term E'_{image} attached to each snake control point v_i is defined as:

$$E'_{image}(v_i) = m_{image}(v_{i-1}) + m_{image}(v_i) + m_{image}(v_{i+1})$$

where i is defined in the range 1..n.

Figure 4. Example of a potential map for an example signature where the distances values have been discretized to 16.

Shape conservation

To avoid an excessive *snake* deformation during the iterations, the internal energy of the snake is characterized as:

$$E'_{shape} = E'_{angle} + E'_{prop}$$

The term E'_{angle} serves to maintain the angle between each pair of adjacent *snake* segments into a controlled interval. This can be expressed as:

$$E'_{angle}(v_{i-1}, v_i, v_{i+1}, t) = \begin{cases} \infty & \text{if } \delta > U_a \\ k_a \cdot \delta & \text{if } \delta \le U_a \end{cases}$$

where:

$$\delta = \left| ang(\overrightarrow{v_{i-1}(0)v_i(0)}, \overrightarrow{v_i(0)v_{i+1}(0)}) - ang(\overrightarrow{v_{i-1}(t)v_i(t)}, \overrightarrow{v_i(t)v_{i+1}(t)}) \right|$$

and *ang* represents the angle defined by the three vertices: v_{i-1}, v_i and v_{i+1}, and k_a is a constant.

The term E'_{prop} is introduced to preserve the proportions between adjacent segments in P. The ratio ϕ is defined by the quotient between a pair of adjacent segments in the *snake* after t iterations and the same pair of the adjacents segments before the iterations. The function defined by E'_{prop} can be expressed as:

$$E'_{prop}(v_{i-1}, v_i, v_{i+1}, t) = \begin{cases} \infty & \text{if } 1-U_p \text{ or if } \phi > 1+U_p \\ k_p \cdot \phi & \text{if } 1-U_p \le \phi \le 1+U_p \end{cases}$$

where:

$$\phi = \frac{\left\| \overrightarrow{v_i(t)v_{i+1}(t)} \right\| / \left\| \overrightarrow{v_{i-1}(t)v_i(t)} \right\|}{\left\| \overrightarrow{v_i(0)v_{i+1}(0)} \right\| / \left\| \overrightarrow{v_{i-1}(0)v_i(0)} \right\|}$$

and k_p is a constant.

Equations corresponding to E'_{angle} and E'_{prop} are compatible with the energy minimization formulation given by equation (4).

5.3. Similarity measure

After a variable number of iterations, the *snake* will converge to a solution. To study the similarity between the *snake* and the signature to which it is adjusted, other elastic matching algorithms were tested [17][3], achieving poorer results.

We now introduce two discriminant features which are used for signature classification: the coincidence measure and the distance measure.

Coincidence measure

This factor f_c uses a potential map $m_{image}(x,y)$ to assign a value to each snake point (not only the control points). In order to obtain a normalized value between 0 and 1, the measure f_c is computed as follows:

$$fc = 1 - \frac{1}{N}\sum_{i=0}^{N} c(p(i))$$

where:
$$c(p(i)) = \frac{m_{\text{imagen}}(p(i))}{k_{fc}/g}$$

and N represents the number of *snake* points, $p(i)$ are all the points belonging to the *snake* segments, g is the average thickness in pixels of the signature strokes and k_{fc} is a scaling factor.

Distance measure

This factor f_d also uses a potential map $m_{snake}(x,y)$, which permits to compute the distance of the signature pixels with respect to the *snake* position. The distance measure is computed as:

$$fd = \sum_{i=0}^{M} d(r(i))$$

where:

$$d(p(i)) = \begin{cases} 1 & \text{if } m_{snake}(r(i)) < g \\ 0 & \text{if } m_{snake}(r(i)) \geq g \end{cases}$$

and $r(i)$ are the signature pixels, M are the number of these pixels and g is the average thickness in pixels of the signature strokes. This second measure can be influenced by the structural noise.

6. Experimental results

A total of 28 from 56 individuals of the database were used for training a NN clasifier. The other 28 individuals were used for test purposes. For each person *i,* we have six genuine signatures: one is used to build the snake and the other five to train the NN. For each person all the signatures of the 27 remaining individuals are used to define the set of patterns to reject by the *snake* (class) *i*. By applying this method for the 28 train individuals, we have a set of 28×5=140 train signatures as patterns to accept, and a set of 28×27×6=4536 test signatures as patterns to reject. All these patterns are presented in Fig. 5 in a 2D feature space.

We used a 2-layer perceptron as signature classifier. This *neural network* (NN) has two input neurons corresponding to the values of the computed similarity measures. The number of neurons in the hidden layer is experimentally set to 10 units and the output layer has only one neuron (see Fig. 6). For training this neural network, the free software JNNS was used [http://www-ra.informatik.uni-tuebingen.de].

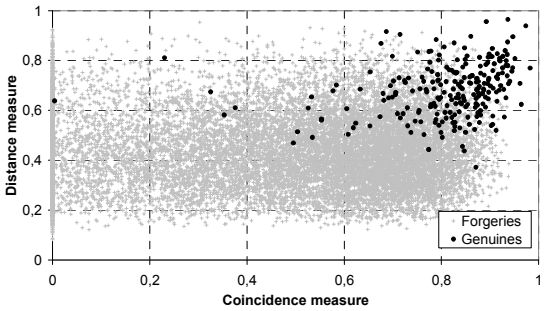

Figure 5. Training patterns in a 2D feature space.

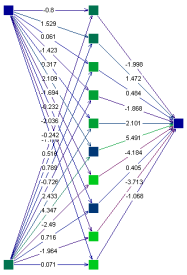

Figure 6. Image of the trained neural network with the updated values of weights.

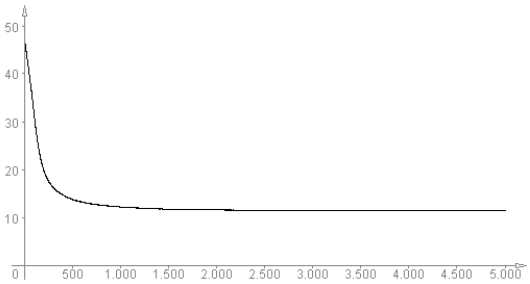

Figure 7. MSE curve obtained during the NN training stage.

The NN was trained during 5000 iterations using a learning rate equal to 0.01. Fig. 7 shows the MSE evolution during the training stage of the neural network using an standard backpropagation algorithm. The error converges in about 1000 iterations.

Each test signature is compared with the corresponding *snake*, using the verification method proposed in section 5. After this process, two selected features f_c and f_d are computed, and used to train the 2-layer perceptron. The obtained result (a real number between 0 and 1) defines the similarity degree with the corresponding model signature. Finally, a threshold value must be chosen in order to decide whether to accept or to reject a test signature. Fig. 8 presents the obtained False Acceptance Rates (FAR) and the False Rejection Rates (FRR) values for different thresholds.

From Fig. 8, we observe that both FAR and FRR curves produce an Equal Error Rate (EER) value (crossing point) of 7.5%. Finally, Table 1 shows the verification results of a threshold value of 0.82.

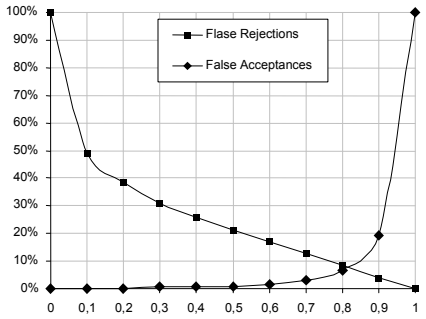

Figure 8. FAR and FRR curves for different threshold values.

Table 1. Results for a threshold equal to 0.82 where FAR equals to FFR.

Signatures	Genuine	Forgery
Total	140	4536
Error	10	340
Correct	130	4196

Percentage	Genuine	Forgery
Errors	7.14%	7.50%
Correct	92.86%	92.50%

Preliminary tests show that the convergence process of the snake is slieghtly affected by structured noise. However the convergence is strongly affected by white noise. Fortunately, white noise is easily removed by a previous simple morphological filtering.

7. Conclusion and future work

A new snake-based off-line signature verification method is proposed. This algorithm works in practical conditions requiring only one training signature for each user and taking about 5 seconds for each test on a 3 Ghzs P-IV computer.

Currently, we are working in developing two components that are needed to have a complete automatic off-line signature verification system:

(a) A method to automatically define the snake using a model signature image.

(b) A morphological algorithm for the location and the segmentation of the signature in a generic document.

We also have to improve the comparison method between the iterated snake and the signature to be verified. Then, a more exhaustive test in presence of structured noise will be done.

8. References

[1] A. Amini et al, "Using Dynamic Programing for Solving Variational Problems in Vision", *IEEE Trans. on Pat. Analysys and Machine Intel.*, V. 12, no. 9, 1990.

[2] R. Bajaj and S. Chaudhury, "Signature Verification using Multiple Neural Classifiers", *Pat. Recog*, Vol. 30, no. 1, pp. 1-7, January 1997.

[3] A. Blake and M. Isard, "Active Contour"*s*, Springer 2000.

[4] J. L. Camino et al, "Signature Classification by HMM", *IEEE 33th International Carnahan Conference on Security Technoly*, 1999.

[5] N.E. Davison, "Snakes simplified", *Pattern Recognition*, Vol. 33, pp. 1651-1664, 2000.

[6] B. Fang et al, "Off-line signature verification with generated training samples", *IEE Proc.-Vis. Image and Signal Processing*, Vol. 149, no. 2, pp. 85-90, 2002.

[7] Z. Hou and C. Han, "Force field analysis snake: an imporved parametric active contour model", *Pattern Recog. Letters,* Vol. 26, pp. 513-526, 2005.

[8] J. L. Esteban, "Técnicas de Reconocimiento de Información y de Gestión en Medios de Pago", Proyecto presentado a las ayudas del Programa Nacional de la Sociedad de la Información en PROFIT, 1-36, 2000.

[9] M. A. Ismail and Samia Gad, "Off-line arabic signature recognition and verification", *Pattern Recognition*, Vol. 33, pp. 1727-1740, 2000.

[10] E.J. Justino et al, "The Interpersonal and Intrapersonal Variability Influences of Off-Line Sign. Verification using HMM", *XV Brazilian Symp. on Comp. Grap.s and Image Proces.* (SIBGRAPI 2002), 2002.

[11] M. Kass et al, "Snakes: Active Contour Models", *Inter. Journal of Computer Vision*, pp. 321-331, 1988.

[12] F. Leclerc and R. Plamondon, "Automatic Signature Verification: The State of the Art", *Intl. J. Pat. Recog and Arti. Intelligence*, Vol. 8, no. 3, pp. 643-660, 1994.

[13] Y. Muzukami et al, "An off-line sign. verification system using an extracted displacement function", *Pat. Recog. Letters*, Vol. 23, pp. 1569-1577, 2002.

[14] V.E. Ramesh and M. Narasimha, "Off-line sign. verification using genetically optimized weighted features", *Pat. Recog.*, Vol. 32, pp. 217-233, 1999.

[15] J. Vélez, A. Sánchez and A.B. Moreno, "Robust off-line sign. verification using compression networks and positional cuttings", *Proc. of the IEEE Conference on NN for Signal Procesing* (NNSP'03), 2003.

[16] L. D. Cohen and I. Cohen, "Finite element method for active contour models and ballons for 2-D and 3-D images", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 11,1131-1147, 1993.

[17] K. Yoshida and H. Sakoe, "On-line handwritten character recog. for a personal computer system", *IEEE Trans. Consum. Electro.*, Vol. 28, No.3, 202-099, 1982.

[18] M. Ammar et al, "Off-line preprocessing and verification of signatures", *Int. Journal of Pattern Recognition and Artificial Intelligence*, 1987.

[19] R. Plamondon, "Progress In Automatic Signature Verfication", *Series in Machine Perception a Artificial Inteligence,* Vol. 13, 1994.

[20] Y. QI, B. Hunt , "Signature verification using global and grid features", *Patt. Recog.*, Vol. 12, 1994.

[21] K. Huang, H. Yan, "Off-line sign. Verif. based on geometric feature extraction and NN classification", *Patt. Recog.*, Vol. 30, Elsevier Science Ltd., 1997.