Image Classification by Texture Segmentation using GAF-SVM

Sergio Manuel Dorantes, Manuel Martín Ortiz, María J. Somodevilla, Jesús Lavalle Martínez, Ivo H. Pineda Torres

Facultad de Ciencias de la Computación, BUAP
sergiomanuel@hotmail.com, {mmartin, mariasg, jlavalle, ipineda}@cs.buap.mx

Abstract. Due to the amount of visual information that currently exists, there is a need to classify it properly. In this paper we present an alternative dual method for image categorization according to their texture content defined as GAF-SVM, this method is based in the use of Gabor Filters (GAF) and Support Vector Machine (SVM). To perform the image classification we rely on filtering techniques for feature extraction mixed with statistical learning techniques to perform the data separation. The experiments were carried out by taking a set of images containing coastal beach scenes and a set of images containing city scenes. A feature vector is obtained from applying a bank of Gabor Filters to the input images; the output feature space is then used as an input to the SVM Classifier. The Support Vector Machine is responsible for learning a model that is capable of separating the sets of input images. Experimental results demonstrate the effectiveness of the proposed dual method by getting the error classification rate to near 9%.

I. INTRODUCTION

The proposal for an alternative method of image classification requires analysis of the methods presented so far concerning the area. Extracting visual information from an image to obtain their most important features is essential for classification tasks; over the years various approaches have been presented regarding this field of study such as: color histograms, region-based classification and gray-level values of raw pixels; though one solution has been to incorporate the texture analysis as main feature descriptor. This is largely due to the fact that most surfaces on images contain some kind of texture. In the recent years, texture analysis has been used for object recognition, image interpretation, image segmentation and classification [1, 2, 6, 8, 9, 10].

In recent papers like [4, 6, 7, 10, 11, 12], texture is been understudy in an isolated manner to evaluate the performance of the proposed algorithms, in some cases it has been used artificial textures, which limits the application area of these methods. Textures are used by the human visual system to separate different objects within scenes as well as surface analysis [11]. Texture can be recognized as an irradiation patterns that are perceptually uniform. Textures can be explained as an efficient measure to estimate the structural differences of orientation, roughness, smoothness or regularity between different regions of an image [14].

But bring out a formal definition of what a texture really is, it became a subjective topic. As it was mention in [13] the definition of texture is dependent on the purpose for which it is being used and outlines some definitions:

1. The basic pattern and repetition frequency of a texture sample could be perceptually invisible, although quantitatively present. In the deterministic formulation texture is considered as a basic local pattern that is periodically repeated over some area.
2. An image texture may be defined as a local arrangement of image irradiances projected from a surface patch of perceptually homogeneous irradiances.
3. Texture is characterized not only by the grey value at a given pixel, but also by the grey value 'pattern' in a neighborhood surrounding the pixel.

Our proposal is based on the use of natural textures in real world images, for that reason the classification model must deal with more complex images in natural conditions.

The 2-D Gabor filters (2D-GF) have certain properties that make them suitable for textural identification in many ways: 2D-GF have tunable orientation and radial frequency bandwidths, tunable center frequencies, and optimally achieve joint resolution in space and spatial frequency. The demodulated Gabor channel envelopes generally contain only low spatial frequencies which are optimally localized in both domains [16].

Gabor filter based methods have been successfully applied for a variety of machine vision application, such as texture segmentation [10, 11, 12, 15, 16, 18], texture classification [9, 13, 19], iris recognition [21, 22, 23], on-road vehicle detection [17], fingerprint classification [20], and as mentioned in [15] edge detection, object detection, image representation, and recognition of handwritten numerals.

This paper is organized as follows: in section II it is mention the related work we based on to develop this article, in section III it is made a detail description of the proposed method, in section IV it is an explanation of the way the input data is processed as well as the Gabor filter's parameters selection, in section V the details of the SVM classifier parameters, and in section VI the experimental results.

II. RELATED WORK

The classification of images has been studied from various approaches, most of all through the mixing of methods, one for texture extraction and one for the classification process.

In [9] is emphasized the use of Gabor filters as a texture extraction method and classification is performed with maximum likelihood method for the classification of aerial and satellite digital images. In [3] is proposed a method of image classification using as an image representation their color histogram and as method of classification the Support Vector Machine. In [4], is not used an external feature extractor, instead the SVM classifier receives the grey level values of each pixel on the image, trying to prove that SVM can implement feature extraction methods within its architecture, this method is computationally expensive due to the number of regions that can define an image. Another approach is performed in [5] where a modification of the SVM is used for identification of regions among a group of images. In [6] the SVM is combined with the Discrete Wavelet Frame Transform for the classification of images of the Brodatz album. In [7] is mixed the use of wavelet transform as a feature extractor known as the pyramid-structured wavelet transform and SVM as the classification method. In [8] is proposed a method called Gaussian Mixture Model mixed with Independent Component Analysis (ICA) to perform the image classification, which is called ICA Mixture Model.

The first step to complete the proposed method is to extract the texture features with a bank of Gabor filters applied to each input image, and then take the filter's output to form a training dataset to feed the SVM classifier.

III. PROPOSED METHOD

In order to accomplish the image classification we rely on filter based techniques to perform texture feature extraction mixed with statistical learning theory techniques to achieve the image data separation. Gabor filters were selected to extract texture features from images due to their resemblance to the human visual system [13].

A. Gabor Filters

A number of authors have used a bank of filters to extract local images features [10, 11, 16, 19]. Different authors used different sets of Gabor Filters, from spatial domain to frequency domain.

A 2-D Gabor filter is a linear filter whose impulse response is defined by a harmonic function multiplied by a Gaussian function. In the spatial domain can be defined as follows:

$$\psi(x,y) = \frac{f^2}{\pi\gamma\eta} e^{-(\frac{f^2}{\gamma^2}x'^2 + \frac{f^2}{\eta^2}y'^2)} \cdot e^{j2\pi f x'} \quad (1)$$

$$x' = x\cos\theta + y\sin\theta$$

$$y' = -x\sin\theta + y\cos\theta$$

Where f is the central frequency of the filter, θ the rotation angle of the Gaussian major axis and the plane wave, γ the sharpness along the major axis, and η the sharpness along the minor axis (perpendicular to the wave). In the given form, the aspect ratio of the Gaussian is $\lambda=\eta/\gamma$. This four parameters (f,θ,γ,η) define the shape of the filter, and by changing them we can detect different textures.

The normalized 2-D Gabor filter function has an analytical form in the frequency domain.

$$\Psi(u,v) = e^{-\frac{\pi^2}{f^2}(\gamma^2 (u'-f)^2 + \eta^2 v'^2)} \tag{2}$$

$$u' = u\cos\theta + v\sin\theta$$

$$v' = -u\sin\theta + v\cos\theta$$

B. Filter Design vs. Filter Bank

There exist two aspects regarding the implementation of Gabor filters, on one hand the filter bank approach and in the other hand the filter design approach [25]. In the first one, a bank of filters is formed by grouping multiple filters tuned at different frequencies and different orientations. The decision of the parameters setting depends on the type of texture to be analyzed. The difficulty of using the filter bank approach relies on the fact that their parameters are established ad hoc and are not optimal for a specific processing task. One of the goals of this work consists on presenting results that would help to specify such parameters. Furthermore, if the bank handles many frequencies and orientations, resulting in a large bank with a lot of filters within, this translates in a large number of convolutions. The filter design approach focuses on designing one or a few filters for a particular application in an effort to reduce the difficulty provided by the filters bank and also to reduce the dimensionality of the output, as well as the processing cost. The disadvantage of this approach lies in the limitation of the tasks for which it was designed. When working with a single filter it is possible that some of the textures in the images are not identified or detected as the filter has a narrow range capacity to detect local texture features.

A filters bank allows the analysis of an image in a single pass way at several frequencies and in several orientations at once. According to the characteristics of our model, the use of a filters bank is the solution choice of deployment, although it could mean an increase, in the computational processing, this is not significant. The design of a Gabor filter bank consists, in general, in the selection, for each filter, of the proper values of the following parameters: *frequency, orientation,* γ and η, the last two parameters known as the smoothing parameters [26].

In this research it is defined a bank with up to 3 orientations and up to 2 frequencies, resulting in a bank with maximum 6 output filters, allowing us to accurately detect a texture among a large set of images. This decision was made based on the studies presented in [26], where is compared with various parameter selection approaches, and summarizes some parameter values adopted in literature.

Using many different orientations and scales (frequencies) ensures invariance; objects and some textures can be recognized al various different orientations, scales and translations [27].

C. Support Vector Machines

Support Vector Machines (SVM) were introduced by Vapnik as a powerful learning tool based on statistical learning theory, a Support Vector Machine is a binary classifier that makes its decision by constructing a linear decision boundary or hyperplane that optimally separates data points of the two classes in feature hyper space and also makes the margin maximized [20].

SVM starts from the goal of separating the data with a hyperplane, and extend this to non-linear decision boundaries using the kernel trick [29]. A hyperplane can be defined as:

$$w^T x + b = 0 \tag{3}$$

Where x represents a point (a vector), w represent the weight (also a vector). We want to choose w and b to maximize the margin, or distance between the parallel hyperplanes that are as far as possible while still separating the data. The hyperplane must separate data such as:

$$\begin{aligned} &w^T x_k + b > 0 \text{ for all } x_k \text{ of a class } y \\ &w^T x_j + b < 0 \text{ for all } x_j \text{ of another class} \end{aligned} \tag{4}$$

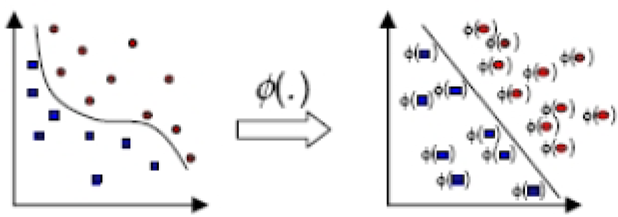
If data are separable in this way, there will probably be more than one way to do it. Among all the possible existing hyperplanes, SVM selects the one in which the distance between the hyperplane and the closest data is the widest possible [29].

When working with a dataset that is not linearly separable, it is necessary to turn to the use of a kernel function. The kernel function allows the SVM to form non-linear boundaries [29]. Data representation through kernel function offers an alternative solution to the nonlinearity problem, projecting the information to a higher dimension feature space [28]. This is accomplished by changing the representation of the function; this is similar to mapping the input space X to a new space H, called feature space, in the form:

$$\phi : X \subset \mathrm{R}^d \to H \tag{5}$$

Now instead of considering the input vectors $\{x_1, \ldots, x_n\}$ it is considered the transformed vectors $\{\varphi(x_1), \ldots, \varphi(x_n)\}$ as shown in figure 1. By doing this substitution, it is obtained a SVM raised in a new space (this is called the 'kernel trick'), it is important to mention that in practice the implementation of this nonlinear technique consumes the same amount of computational resources of its linear equivalent.

Fig. 1. Using the Kernel to transform (map) the input data space.



The general problem that SVM want to resolve is to search, for a given learning task, with a finite amount of data, an appropriate function that helps to carry out a good generalization, which results from a proper relationship between the accuracy achieved with a particular training set and the ability of the model [30].

The use of the ‘Radial Basis Function’ (RBF) kernel is based on the fact that this kernel is basically suited best to deal with data that have a class-conditional probability distribution function approaching the Gaussian distribution, like the texture present on the input images. It maps such data into a different space where the data becomes linearly separable. The kernel function is defined as follows:

$$k(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2} \right) \quad \textbf{(6)}$$

A disadvantage concerning this kernel is that is difficult to design, in the sense that it is difficult to obtain an optimum value for its parameter σ (sigma) and choose the corresponding C that works best for a given problem. The fact that certain combinations of σ and C make the SVM highly sensitive to training data also contributes to the error rate of the RBF-based SVM.

One of the advantages of the RBF kernel is that given the kernel, the weights, the number of support vector and the support vectors itself are automatically obtained as part of the training procedure, i.e. they don’t need to be specified by the training mechanism.

IV. SETTING THE EXPERIMENTS

As part of the experiments it was decided to work with two sets of images, one set consisting of coastal beach scenes, and the other set consisting of city scenes images. The processing of input images is done in order to reduce computational complexity.

The first set is conformed by 128 images of beach scenes content, the second set is conformed by 128 images of city scenes content, a total of 256 images.

The input images after processed end up being 8-bit per pixel grayscale images of dimension *128*x*128*, working with just one channel reduces de number of convolutions. The output of each filter is obtained by the convolution of the input image with a Gabor filter. The process is shown below:

$$G(x, y) = I(x, y) \otimes \psi(x, y) \quad \textbf{(7)}$$

where

$G(x, y)$ is the output of the filter

$I(x, y)$ is the original image

$\psi(x, y)$ is the Gabor filter

This computation can theoretically be done in the spatial domain however the Gabor filter is usually narrow. The filter is usually much larger in the frequency domain and thus less affected by aliasing effects due to sampling. It is thus more convenient to do all the computation process in the frequency domain. The convolution is then reduced to a simple and efficient point-wise multiplication of the Fourier transforms [11].

The family of Gabor filters selected to set up the filter bank for the experiments in the frequency domain are:

$$\Psi(u,v) = e^{-\frac{\pi^2}{f^2}(\gamma^2(u'-f)^2+\eta^2 v'^2)} \quad \textbf{(8)}$$
$$u' = u\cos\theta + v\sin\theta$$
$$v' = -u\sin\theta + v\cos\theta$$

A filters bank is constructed by changing values on the four parameters mentioned before, (***f***,***θ***, ***γ***, ***η***). For our experiments we only change of the parameters ***f*** and ***θ*** , which are the central frequency of the filter and the rotation angle of the filter, the other two parameters (***γ*** and ***η***) are set to be constant.

For the experiments it has been set up the bank with 2 frequencies and 3 orientations, to obtain a total of 6 filter outputs for each image on the respective set.

The selected parameters for the filter banks are: central frequency ***f = 0.1725***, which outputs the two frequencies 0.1725, 0.1220. Three orientations ***θ*** = 0°, 60° and 120°, sharpness along the mayor axis ***γ=0.5***, sharpness along the minor axis ***η=0.5***.

In order to apply the filters bank to the sets of images, each image needs to be transform into frequency domain via the Fourier transform. The 2-D Fourier transform used for images can be defined as:

$$I(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} i(x,y)e^{-i2\pi(ux+vy)}dxdy \quad \textbf{(9)}$$

where

$i(x,y)$ is the original input image

The convolution of the input image with the Gabor filter is performed. In domain frequency the convolution is represented in a point-to-point multiplication of the transformed image with the Gabor filter.

$$g(x,y) = i(x,y) \otimes \psi(x,y) \text{ - Spatial Domain} \quad \textbf{(10)}$$
$$G(x,y) = I(x,y)\cdot\Psi(x,y) \text{ - Frequency Domain}$$

Once the filter output is obtained, *G(x,y)* needs to be transformed back to its spatial representation using the Inverse Fourier Transform in 2-D.

$$i(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(u,v)e^{i2\pi(ux+vy)}dudv \quad \textbf{(11)}$$

where

$I(u,v)$ is the image in frequency domain

After the transformation, normalization is applied to the output image in order to avoid effects by illumination.

At the end of normalization, we have a certain number of square matrices per filtered image; each matrix dimension is *128*x*128*. The number of square matrices depends on the number of orientations and frequencies concerning the filters bank, in our experiments the filter bank consist of 2 frequencies and 3 orientations, so the number of output matrices is 6.

When convolution is performed some results are not useful especially if the image does not contain textures that respond meaningfully to the filter selected parameters. To reduce the problem all the outputs obtained by convolution are summed up to remove the results that are not relevant and to enhance those that helps to detect texture regions; this also helps to reduce dimensionality of the feature space by having one square matrix as an output, same size of the input image.

At this point we have one matrix per input image, reducing the dimensionality of input data. Each matrix is used to build up a *feature matrix*, which is going to serve as an input of the SVM classifier.

To complete the convolution of the input image with each one of the Gabor filters we take only the real part of the output filtered image. As mentioned in [31], by this way we can keep most the texture response information ignoring phase information.

$$\mathrm{Re}(G(x, y)) \tag{12}$$

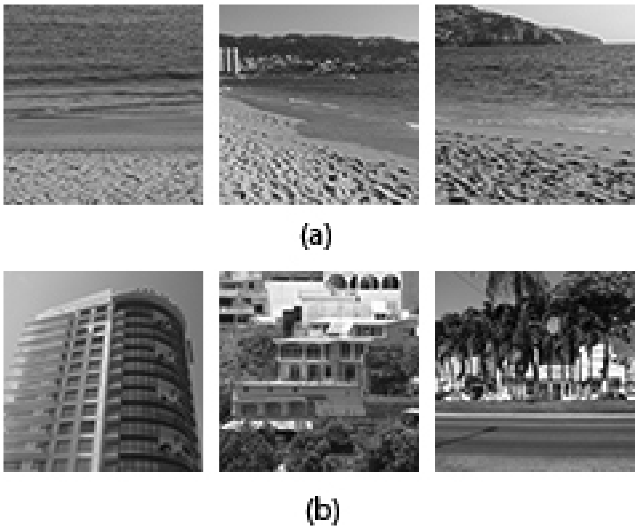
Then we modified each output matrix of dimension *128*x*128* to construct the *feature matrix*. We take each matrix and transform it in a *1*x*16384* vector, each vector is then piled up with the next transformed matrix to form the *feature matrix*.

Finally we have a *feature matrix* of dimension *256*x*16384* which serves as input to the classifier.

V. SVM CLASSIFIER

The goal of experimentation is to obtain a training model through SVM, which can be capable of separate a set of input images. Once we have the *feature matrix* with processed and filtered images we proceed with the SVM classification procedure. According to the nature of the classification process we need to define a training dataset, so the classifier could learn a model, and a test dataset, that let us test the learned model. The training dataset is conformed by 75% of the input dataset, and the test dataset by the remaining 25%. The selection criteria to build up the training dataset and the test dataset are done randomly. In fig. 2(a) is shown an example of coastal beach scene images, and in fig 2 (b) an example of city scenes images, which the classifier will try to separate.

Fig. 2. Example of images used in the experiments. (a) Beach scene images, (b) City images.



The experiments were performed using SPIDER [32], a MATLAB implementation of SVM, a complete object oriented environment for machine learning. Being SVM a binary classifier it is necessary to label the datasets for the classification experiments, the beach scenes images are labeled as 1, and the city scene images are labeled as -1.

In table I there is a list of kernels available in SPIDER, its formula and its parameters.

Table 1. List of Kernel opeators available in SPIDER

Kernel	Parameter	Formula
Linear		$k(x,y) = x \cdot y$
Poly	d, degree	$k(x,y) = (x \cdot y + 1)^d$
Radial Basis Function (RBF)	sigma	$k(x,y) = e^{\frac{-\|x-y\|^2}{2\cdot\sigma^2}}$
Gaussian	sigma	$k(x,y) = \frac{1}{2\pi^{N/2} \cdot \sqrt{\sigma}}$

VI. EXPERIMENTAL RESULTS

It is used the “RBF” kernel to execute the experiments with different sigma values. Another parameter used by SPIDER is the ‘soft margin parameter’, *C*, which penalizes the training errors. This value is set to 1000 in all the experiments.

Iteratively, the sigma values were changed until a significant error rate is obtained. The test results for the learned algorithm are presented in table II.

As it can be seen in table II the sigma value which represents the lower percentage error is $\sigma = 35$, with an error rate of 9.37%.

Table 2. Experimental results for different sigma values

RBF	error	RBF	error
σ=21	**0.1406**	σ=29	**0.1094**
σ=22	**0.1094**	σ=30	**0.1094**
σ=23	**0.1094**	σ=31	**0.1094**
σ=24	**0.1094**	σ=32	**0.1094**
σ=25	**0.1094**	σ=33	**0.1094**
σ=26	**0.1094**	σ=34	**0.1094**
σ=27	**0.1094**	σ=35	**0.09375**
σ=28	**0.1094**	σ=36	**0.09375**

VII. CONCLUSIONS

Extracting texture features by a Gabor filter bank and classify the filter outputs via the Support Vector Machines offers an excellent accuracy rate, 90.63% of the input images are correctly classified according to their class, belonging to beach scenes class or city scenes class.

The article proves the efficiency of using a dual model, first to extract de texture features and then classify them with SVM.

REFERENCES

[1] T. Randen and J.H. Husoy, "Filtering for texture classification: a comparative study", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol 21, Issue 4,, pp. 291 – 310, Apr 1999.

[2] F. Lumbreras Ruiz, "Segmentation, classification and modelization of textures by means of multiresolution decomposition techniques", Ph.D. dissertation, Dept. Informática and Computer Vision Center, Universitat Autònoma de Barcelona, Barcelona, España, 2001.

[3] O. Chapelle, P. Haffner, and V.N. Vapnik, "Support vector machines for histogram-based image classification", *IEEE Trans. On Neural Networks*, Vol. 10, Issue 5, pp. 1055 – 1064, Sep 1999.

[4] Kwang In Kim, Keechul Jung, Se Hyun Park, and Hang Joon Kim, "Support vector machines for texture classification", *IEEE Trans. On Pattern Analysis and Machine Intelligence*, Vol. 24, Issue 11, pp. 1542 – 1550, Nov 2002.

[5] I. Gondra and D.R. Heisterkamp, "Learning in region-based image retrieval with generalized support vector machines", *In Proc. of the Computer Vision and Pattern Recognition*, pp. 149 – 154, 2004.

[6] Shutao Li, J.T. Kwok, Hailong Zhu, and Yaonan Wang, "Texture classification using the support vector machines", *Pattern Recognition*, Vol. 36, No. 12, pp. 2883 – 2893, 2003.

[7] Bing-Yu Sun and De-Shuang Huang, "Texture classification based on support vector machine and wavelet transform", *In Proc. of the Fifth World Congress on Intelligent Control and Automation*, WCICA 2004. Vol. 2, pp. 1862 – 1864, June 15–19, 2004.

[8] V.P. Subramanyam Rallabandi and S.K. Sett, "Unsupervised texture classification and segmentation", *Proceedings Of World Academy of Science, Engineering and Technology*, Vol. 5, April 2005.
[9] J.A. Recio, L.A. Ruiz and A.Fernández-Sarriá, "Use of Gabor filters for texture classification of digital images", Física de la Tierra, Vol. 17, pp. 47 – 59, 2005.
[10] M.R. Turner, "Texture discrimination by Gabor functions", Biol. Cybern., Vol. 55, Num. 2–3, pp. 71 – 82, 1986.
[11] V. Levesque, "Texture segmentation using Gabor filters", Center for Intelligent Machines Journal, 2000
[12] P. Guha and R. Banerjee, "Segmentation and classification of multi-textured images", 2000, Available: http://www.cse.iitk.ac.in/~amit/courses/768/00/rajrup/, last visited: April 20, 2009.
[13] V.S. Vyas and P. Rege, "Automated texture analysis with Gabor filters", GVIP Journal, Vol. 6, Issue 1, pp. 35 – 41, July 2006.
[14] K.M. Rajpoot and N.M. Rajpoot, "Wavelets and Support Vector Machines for Texture Classification", *In proceedings of the 8th International Multitopic Conference*, INMIC 2004, 24-26 Dec., pp. 328 – 333, 2004.
[15] D.M. Tsai, "Optimal Gabor filter design for texture segmentation", Technical Report, Machine Vision Lab, Dept. of Ind. Eng. and Mgmt., Yuan-Ze University, Chung-Li, Taiwan, 2000.
[16] A.C. Bovik, M. Clark and W.S.Geisler, "Multichannel Texture Analysis Using Localized Spatial Filters", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, Num. 1, pp. 55 – 73, 1990.
[17] Zehang Sun, G. Bebis and R. Miller, " On-road vehicle detection using Gabor filters and support vector machines", *14th International Conference on Digital Signal Processing*, DSP 2002, Vol. 2, pp. 1019 – 1022, 2002.
[18] K. Hammouda and E. Jernigan, "Texture segmentation using Gabor filters", tech. rep., Biotechnology and health engineering centre, University of Waterloo, Dec. 2000.
[19] S.E. Grigorescu, N. Petkov and P. Kruizinga, "Comparison of texture features based on Gabor filters", *IEEE Trans. On Image Processing*, Vol. 11, Num. 10, pp. 1160 – 1167, 2002.
[20] D. Batra, G. Singhal and S. Chaudhury, "Gabor filter based fingerprint classification using support vector machines", *Proceedings of the IEEE First India Annual Conference, 2004, INDICON 2004*, pp. 256 – 261, 20-22 Dec. 2004.
[21] Q.A. Salih and V. Dhandapani, "IRIS Recognition based on multi-channel feature extraction using gabor filters", *Proceedings of the 2nd IASTED international conference on Advances in computer science and technology*, ACST'06, pp. 168 – 173, 2006.
[22] L. Ma, Y. Wang and T. Tan, "Iris recognition based on multichannel Gabor filtering", 5th Asian Conf. Computer Vision, Vol. 1, 2002.
[23] D. Carr, "Iris recognition: Gabor filtering", Connexions. Dec. 18, 2004, Available: http://cnx.org/content/m12493/1.4/, last visited April 20, 2009.
[24] K. Kämäräinen, "Feature extraction using Gabor filters", Ph. D. dissertation, Lappeenranta University of Technology, Finland, Nov. 2003.
[25] T.P. Weldon, W.E. Higgins and D.F. Dunn, "Gabor filter design for multiple texture segmentation", *Optical Engineering*, Vol. 35, pp. 2852 – 2863, 1996.
[26] F. Bianconi and A. Fernández, "Evaluation of the effects of Gabor filter parameters on texture classification", *Pattern Recognition*, Vol. 40, Num. 12, pp. 3325 – 3335, 2007.
[27] J. Ilonen, J.K. Kämäräinen and J.K. Kälviäinen, "Efficient computation of Gabor features", Research Report 100, Lappeenranta University of Technology, Dept. of Information Technology, 2005.
[28] J.A. Reséndiz, "Las máquinas de vectores de soporte para identificación en línea", Masters dissertation, Departamento de control automático, Centro de investigación y estudios avanzados, I.P.N., 2006.
[29] J.P. Lewis, "A short SVM (support vector machine) tutorial", CGIT Lab / IMSC, University Southern California, 2004.
[30] L. González, "Modelos de clasificación basados en máquinas de vectores de soporte", *Asoc. científica europea de econ. aplicada*. Anales de economía aplicada, 2003.
[31] D. Dunn, W.E. Higgins and J. Wakeley, "Texture segmentation using 2-D Gabor elementary functions", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, Num. 2, pp. 130 – 149, Feb 1994.
[32] SPIDER, A complete object oriented environment for machine learning in MATLAB. Available: http://www.kyb.mpg.de/bs/people/spider/, last visited May 15, 2009.