Signal Processing ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Evaluation of shape classification techniques based on the signature of the blob

Pedro Gil-Jiménez *, Hilario Gómez-Moreno, Javier Acevedo-Rodríguez, Roberto J. López-Sastre, Saturnino Maldonado-Bascón

Dpto. de Teoría de la señal y Comunicaciones. Universidad de Alcalá, 28805 Alcalá de Henares (Madrid), Spain

ARTICLE INFO

Article history:
Received 29 December 2010
Received in revised form
27 April 2011
Accepted 7 June 2011

Keywords:
Imagen processing
Pattern recognition
Blob signature
Shape classification

ABSTRACT

In this paper, we report a study of different preprocessing and classification techniques that can be applied to shape classification using the signature of the blob, or its FFT, as the main feature. Eight well-known classification methods were tested and compared.

The results obtained show that, for shapes with a small to medium amount of distortion, all the methods obtained an almost 100% success probability. However, as distortion increased, those not based on the FFT performed better than the other algorithms, at the expense of a small increase in computational time.

The samples employed for training and testing purposes were not hand-selected, but were generated by an application developed as part of this study. This application simulates the main distortions that can be produced by a real camera, including shifts, scalings, rotations, affine transformations and noise. We demonstrate that the use of these synthetic images for the training process, instead of manually selected ones, had proven to perform well with real images.

A study of the false positive problem is also included, showing that, with the use of SVMs and careful selection of the training set, a large number of false positives can be discarded in the detection step.

1. Introduction

Shape classification can be seen as the task of identifying the shape of an object according to a set of previously defined reference shapes, including, if necessary, false positives as another class. Many object recognition problems are more easily solved in terms of shape recognition than with other features, such as color or texture. In recent years, the problem has been given considerable attention, with potential applications for image processing problems such as handwriting recognition, industrial inspection or computer aided diagnosis.

The problems that must be solved are, firstly, the choice of a shape descriptor that encompasses all the relevant information concerning the shape as accurately as possible, and secondly, the design of a classifier that, using the previously defined descriptor, compares and classifies each shape into one of the predefined reference shapes. Much effort has been expended on the first step, and many approaches have been described in the literature. Nevertheless, it is worth noting that the choice of the descriptors is driven by the kind of shapes to classify.

Some of the most popular descriptors reported in the literature are the well-known *Fourier Descriptors* (FD) [1]. The FD are used, for instance, in [2] for the identification of American Sign Language hand shapes, or more recently, in [3,4], for the recognition of paper and metal defects, or human silhouettes. In [5], FD were extended to become the *Generic Fourier Descriptors* (GFD), for the retrieval and identification of pictographs.

Another well-known shape descriptor is the *Contour Curvature* [6], where a one-dimensional curve can be

* Corresponding author.
E-mail address: pedro.gil@uah.es (P. Gil-Jiménez).

0165-1684/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2011.06.007

obtained from the curvature of the object contour. Some examples can be found in [7], for instance, for the classification of general objects, or in [8], for the recognition of *diatoms*, a kind of unicellular algae. In [9], the curvature of the contour was further tested with occluded objects. An extension of the curvature of the contour is the *Area Matrix*, which is more affine-invariant than the curvature itself, and was tested, for example, in [10] for the recognition of sign language hand shapes.

Another way to describe a shape is through the use of Zernike moments. Zernike moments can be found in a number of papers, such as [11–14]. In these studies, the Zernike moments were used to classify different kinds of shapes, such as traffic signs, complex pictographs and handwritten texts. [15] reports a comparative study of Zernike moments and Fourier Descriptors.

Other shape descriptors in the literature include the *Level Sets*. In [16], the shape descriptor of known objects, such as handwritten text, was used for the segmentation of low quality images. In [17,18], the Level Set Function was used for the identification of different shapes in medical images, such as Magnetic Resonance or Computed Tomography scans.

Although all the descriptors mentioned so far are capable of describing complex shapes, their main disadvantage is reduced performance in the presence of high geometric distortions, especially segmentation noise. In this paper, we focus on the use of the *signature* as the main descriptor of the shape of the object [1]. Shape signature has not been widely reported in the literature, most probably because it cannot describe complex shapes. However, the signature has proven to be a very robust and efficient descriptor of simple shapes in the presence of noise and geometric distortions, as demonstrated here.

The research reported here represents the continuation of previous research described in [19,20]. Although these studies were exclusively oriented to traffic sign systems, part of our efforts has been devoted to the design of a more generic algorithm, that is, to be able to process and recognize a wider range of predefined shapes. Thus, we conducted a study comparing the performance of several classifiers which has enabled us to improve the performance of the algorithm in terms of classification success. All the results were obtained with the use and processing of synthetic images generated automatically with an application which was also developed as part of this research, although the algorithm was also tested with real images. Lastly, we investigated the false positive detection problem, which enabled the algorithm to eliminate those objects that did not correspond to any of the predefined shapes, thus reducing the computational complexity of the entire system.

2. Signature extraction and preprocessing

Fig. 1 shows a block diagram of the implemented system. In this system, a segmentation block was designed to extract the objects of interest from the background: in the example given here, a red traffic sign. The block also computes the connected components, so that the output provides a list of blobs that must be processed by the next block. It is not the aim of this paper to describe the different algorithms that can be designed at this stage. A wide-ranging study of segmentation techniques can be found, for instance, in [21]. Rather, we worked with segmented images, i.e. we designed an application that generates binary images, so that segmentation becomes unnecessary. In this way, we were able to conduct the study described in this paper independently of the segmentation process, and could concentrate our efforts on the shape classification process itself.

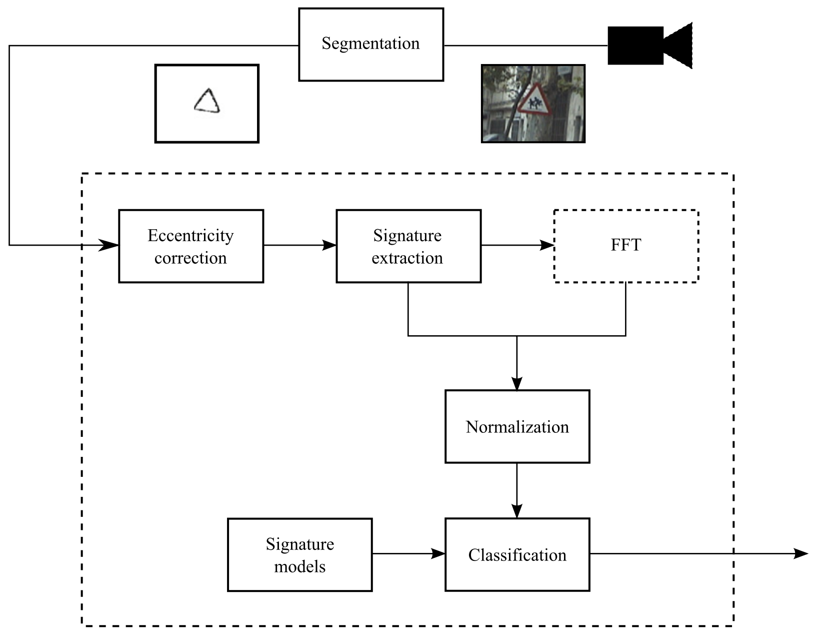

Fig. 1. Detection block diagram.

More information about the application designed to generate the binary images can be found in Section 8.

Focusing on the shape classification block, our system used the signature of the blob to perform the classification. The signature is a unidimensional representation of the object contour, and in this study it was obtained as the distance from the mass center of the blob to the edge as a function of the angle. In Fig. 2 we give two examples of a signature, one for a square and the other for a triangle.

Although the signature is a powerful tool in shape classification problems, geometric distortions such as shifts, scalings, rotations and affine distortions can worsen the performance if they are not taken into account. In this study, we addressed each problem as follows:

Shifts were automatically handled by the signature extraction process itself, since in this case the signature is defined from the mass center of the blob. Scalings must be addressed with some kind of normalization. From our previous research, we found that the best strategy was either energy normalization or amplitude normalization, depending on the classification process employed. Specifically, amplitude normalization is used when working with *Earth Mover's Distance*, whereas energy normalization is used for the rest of the classifiers.

Rotations are reflected as circular shifts in the signature. This problem can be tackled using either of the following strategies. Since the classification can be seen as a comparison between the current signature and the previously computed *models* placed in a reference position, one can simply perform this comparison with shifted versions of the models, in this case, as many of them as the number of signature samples. The other option takes advantage of the invariance property of the absolute value of Discrete Fourier Transform (DFT) in the presence of shifts. Thus, by computing the DFT of the signature and comparing only its absolute value, the rotation problem is overcome. Both strategies are tested and discussed in Section 9.

Affine camera distortions are reflected in the image in a much more complicated way. Nevertheless, the eccentricity of the blob represents a parameter that enables the amount of affine distortion to be estimated. In binary mask processing, the eccentricity can be computed from the second order moments m_{20} and m_{02} when the blob is reoriented to the axis of less moment of inertia [22]. With this information, the blob can be scaled anisotropically so that the new blob has an eccentricity equal to one. Although this process does not undo the affine transformation itself, it did improve the overall performance of all the classifiers tested in this paper.

The solution implemented will not be described in full here, since a complete description can be found in [19]. Nevertheless, Fig. 3 illustrates the basis of the algorithm.

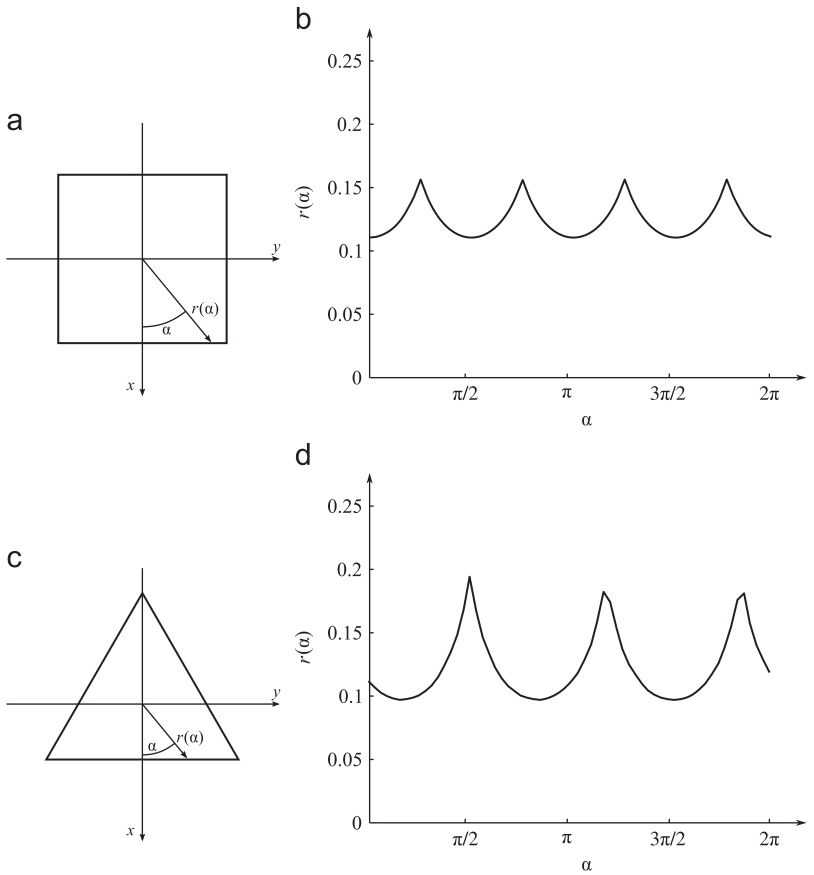

Fig. 2. Two examples of a shape and its corresponding signature. (a) Square; (b) square signature; (c) triangle; and (d) triangle signature.

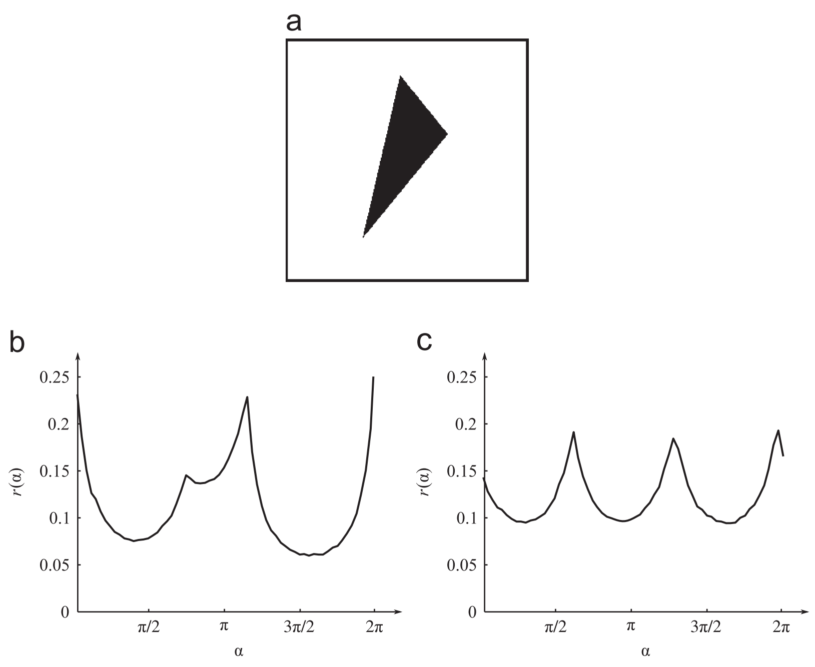

Fig. 3. An example of the performance of the eccentricity correction: (a) Original object with a clear affine distortion. (b) Signature extracted from (a) without any correction. (c) Signature extracted using the eccentricity correction process.

Fig. 3(a) shows the object that will be used in this explanation. If we extract the signature of the object as it is defined for instance in [1], that is, simply the distance from the mass center as a function of the angle, we obtain the signal shown in Fig. 3(b). However, if the process is modified with the eccentricity correction method proposed, the new signature is the one shown in Fig. 3(c). If we compare both signatures with the original one for a triangle, shown in Fig. 2(d), it is clear that 3(c) resembles the original signature more closely than 3(b), thus making the probability of success higher.

The algorithms designed to deal with all these problems were implemented, and fully described, in [19]. There, we found that eccentricity correction and energy normalization should be included independently of the classification method employed. However, in this study, we went one step further, and tested the signature extraction process with different classifiers, as described in the following sections. One of the purposes of this paper is to report the behavior of each classifier under different conditions, and extract some conclusions about their performance.

3. Classification using the *Mean Squared Error* (MSE)

In this section, we describe the use of the *Mean Squared Error* to compare the signature of the object being analyzed with those of the reference shapes.

3.1. MSE classification through the signature

As discussed above, in order to overcome rotation problems, the MSE must be computed for every possible position of the current signature with respect to the reference one, according to:

$$\mathrm{MSE}_{\mathbf{xy}}(j) = \frac{1}{N}\sum_{i=0}^{N-1}(x_{i+j}-y_i)^2, \tag{1}$$

where x_i is the ith sample of the current signature $\mathbf{x}$ to classify, y_i is the ith sample of a reference signature $\mathbf{y}$, and N is the number of samples extracted from the signature. Furthermore, the expression x_{i+j} represents a circular shift, that is, the samples that disappear from one side are moved to the other side of the signature, and j ranges from 0 to $N-1$. Through this process, N measures of the error are obtained, one for each possible shift of the current signature. Since the MSE is a measure of difference, the sample with the minimum value must be chosen:

$$M_u = \min\,(\mathrm{MSE}_{\mathbf{xy}}(j)), \tag{2}$$

where M_u indicates how similar the current signature is to the uth reference shape. This process is performed for all the reference shapes considered, and for classification purposes, the minimum of all the shapes will indicate the shape S which the current object most closely resembles:

$$S = \arg\min_u(M_u). \tag{3}$$

3.2. MSE classification through the FFT of the signature

As can be deduced from the previous discussion, the MSE must be computed N times for each shape, and this is intended to overcome rotation distortions. Computational load can be reduced by taking advantage of the shift invariance of the absolute value of the DFT (AbDFT). Thus, it is only necessary to compute the AbDFT of the current signature, and compute the MSE according to:

$$\mathrm{MSE}_{\mathbf{xy}} = \frac{1}{N/2}\sum_{i=0}^{N/2-1}(|X|_i - |Y|_i)^2, \tag{4}$$

where X and Y are the corresponding Fourier Transforms of the current ($\mathbf{x}$) and reference ($\mathbf{y}$) signatures, respectively. Note that since the signatures are signals of real samples, the AbDFT is symmetric, and so only half of the samples are used in the comparison, thus reducing the computational load. Furthermore, if the number of samples is fixed to a power of two, the FFT can be implemented, instead of the DFT, reducing computational complexity still further. Similar to the previous schema, (4) must be computed for all the reference shapes considered, and the minimum will indicate the shape which is most similar to the current object.

In short, the use of the absolute value of the FFT (AbFFT) reduces N times the computational load, since the MSE is evaluated only once, instead of N when not using it. Of course, some time must be spent in the computation of the AbFFT, but even so, the overall computational complexity is smaller, as will be seen later.

4. Classification using the *Cross-Correlation* (CC)

As with the MSE, the *Cross-Correlation* can be seen as another tool for comparing and classifying signals. In this case, the CC is defined as

$$\mathrm{CC}_{\mathbf{xy}}(j) = \sum_{i=0}^{N-1} x_{i+j}\cdot y_i, \tag{5}$$

where again x_{i+j} implies the circular shift of signal $\mathbf{x}$. As can be seen, the definition of the CC itself implies the shift of the current signature for every possible position, in contrast to the MSE, where the shift must be explicitly included. Nevertheless, calculating the CC implies the computation of the right hand side of the equation N times, one for each possible shift, and therefore the computational complexity will be similar to that of the MSE.

Unlike the MSE, the CC is a measure of similarity, and so, from the N samples obtained for one reference shape, the one with the highest value will indicate the similarity of the current signature to the uth reference one.

$$C_u = \max(\mathrm{CC}_{\mathbf{xy}}(j)), \tag{6}$$

where j ranges from 0 to $N-1$. This process must be performed for all the reference shapes, and for all of them, the one with the highest value will be the shape to which the current object is most similar:

$$S = \arg\max_u(C_u). \tag{7}$$

5. Classification using the *Earth Mover's Distance* (EMD)

The *Earth Mover's Distance* is defined as the measure of the distance between probability distributions [23]. Although this metric was originally oriented to compare probability distributions, its application can be extended to the comparison of two signals with no extra work. Intuitively, given two distributions, the EMD can be understood as the minimal amount of work needed to transform one distribution into another. The EMD can be computed according to

$$\mathrm{EMD}_{\mathbf{xy}} = \frac{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1} f_{ij}d_{ij}}{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1} f_{ij}}, \tag{8}$$

where f_{ij} is the flow between x_i and y_j that minimizes the overall cost of transforming distribution $\mathbf{x}$ into distribution $\mathbf{y}$, and d_{ij} is the *ground distance* between samples i and j. In our case, we chose the Euclidean distance as the ground distance. It can be seen that if both distributions are similar, then the flow between different samples tend to be null, and so, the EMD will be zero, and as the distributions become different, its value increases.

An important fact is that only when the total weights of the two distributions are equal, the EMD is a true metric. For that reason, only amplitude normalization is acceptable, as opposite to the other methods investigated in this paper, where the tests conducted showed that energy normalization is preferred than amplitude normalization.

In this section, we propose the use of the EMD to measure the similarity between signatures. As it happened with the MSE classification, the comparison can be done either over the signature itself, or using the FFT of the signature. While both strategies yielded acceptable results, only the FFT option has a computational load comparable to the rest of the classifiers. Thus, only the FFT version was included in the results section.

6. Classification using *Support Vector Machines* (SVMs)

Another way to perform classification between signals is through the use of Pattern Recognition techniques. In this section, we analyze the performance of the *Support Vector Machines*. Although a complete description can be found in [24], we will outline here the main points of this kind of classifiers.

In this case, the classification task consists in finding a decision frontier which separates the data into the considered classes. The simplest decision problem comprises a number of vectors divided into two classes, and the optimal decision will be the one that maximizes the distance from the frontier to the data. In the simplest case, the decision function has the following form:

$$f(\mathbf{x}) = \sum_{i=1}^{L}\alpha_i p_i\langle \mathbf{y}_i\cdot\mathbf{x}\rangle + b, \tag{9}$$

where $\mathbf{x}$ is the input vector, in our case, the current signature to classify, and p_i takes $+1$ when vector i belongs to one class, and -1 to the other. In addition, the inner product is performed between each training input $\mathbf{y}_i$, and the input vector $\mathbf{x}$. Thus, a set of training data $(\mathbf{y},p)$ is needed in order to build the classification function. Furthermore, α_i are the Lagrange multipliers, obtained through a minimization process, and L is the number of vectors $\mathbf{y}$ that, during the training process, contribute to form the decision frontier. These vectors are those with α not equal to zero, and are known as *Support Vectors*.

In order to improve the performance when the vectors are not linearly separable, the SVMs map the input data into a higher dimensional feature space, using the well-known *Kernel Trick*. In this case, the non-linear expression for the classification function becomes:

$$f(\mathbf{x}) = \sum_{i=1}^{L} \alpha_i p_i K(\mathbf{y}_i, \mathbf{x}) + b \tag{10}$$

where K is the kernel that performs the non-linear mapping. Several kernels have been defined in the literature, being the more important ones the Radial Basis Function (RBF), the Polinomial and the Sigmoidal kernels. Nevertheless, for many practical applications, the RBF kernel has shown to be a good option [25–28]. In our work, we have implemented and tested the performance of SVMs with all the three kernels, but only RBF was included in the result section, due to its better performance.

With SVMs, or any other classifier of this kind, a problem may arise in the sense that it becomes necessary to design a method to collect the samples that will be used in the training process. This set should include sufficient samples from each class to allow the classifier to *learn* the features of each one. The collection of samples can be carried out manually, but this method entails a heavy workload for the designer of the system. In our case, however, the application designed to create the image set for testing the performance of the previous algorithms can also be used to generate the samples for the training process, although some care must be taken when choosing the most appropriate ones. In Section 9, the process for creating the training set is fully described.

In brief, the process followed started with the generation of the training set using the application described in Section 8. SVMs with RBF kernel were trained with all these images, or more specifically, with their signatures. Next, we generated a new set of images and classified them, as we have done with the other two algorithms. Similarly, the system was capable of working either with the original signatures, or with the AbFFT to improve the performance against geometric rotations. Note however that in the first case we did not need to take the problems related to rotations into consideration as we did with MSE, CC or EMD. In this case, the training set was composed of a large number of images, and thus included sufficient versions of each shape at different shifts. This did not imply a reduction in the computational complexity with respect to the MSE or CC implementation, since the time employed by the SVMs on classification is related to the number of Support Vectors chosen. For the AbFFT version of the algorithm, the previous point does not apply, and the same advantages described for the MSE solution also applied here.

7. Classification using *k-Nearest Neighbors* (*k*-NN)

The *k*-Nearest Neighbors algorithm is amongst the simplest of all Pattern Recognition algorithms. Basically, given a set of labeled vectors, the distance from every vector to the test one is computed, and the class assigned is the one most common among the k nearest vectors, being k a positive integer, typically small [29]. It is worth noting that, if $k=1$, and the Euclidean distance is utilized to compute distances between vectors, then, the k-NN algorithm becomes the MSE classifier discussed before, except that in this work MSE uses only one reference sample per shape, whereas k-NN uses a larger number of samples, which includes noise and geometrical distortions.

Although the training processes for k-NN and SVMs are different, with regard to the generation of the training set, the process is similar to the one used for SVMs, and will not be discussed here. Again, the classification can be performed both with or without computing the FFT of the signature of the blobs.

8. Generation of the image set

Although the entire system, including the algorithms described in this paper, can work with real images, preparing a database with a number of samples large enough to obtain statistically valid results would be unfeasible in practice. Consequently, we developed an application that generates images from a template model simulating all the possible distortions for an image of the object. The block diagram of this application is shown in Fig. 4. It basically consists of a database containing images of the template models and a block that computes a projection of the template from a homography plus a contour noise addition.

The database of templates must include all the shapes that will be considered in the classification process. In our case, it consisted of the shapes shown in Fig. 5, although different ones can be added with little additional effort.

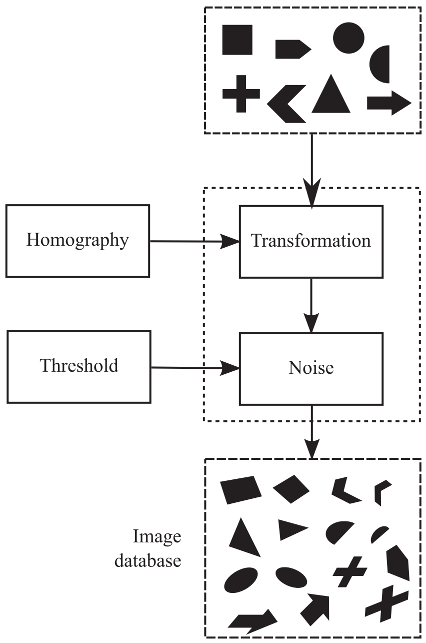

Fig. 4. Block diagram of the Image Database Generator.

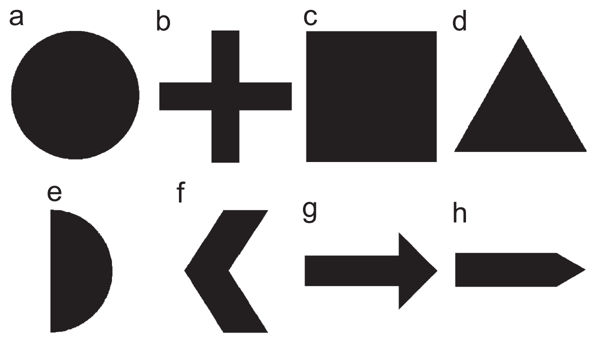

Fig. 5. Shape set considered in the classification. (a) Circle; (b) cross; (c) square; (d) triangle; (e) semicircle; (f) arrow1; (g) arrow2; and (h) indicator.

Computation of the homography is the key step of the application. It is designed to generate random homographies, but all of them must be constrained to a maximum geometrical distortion. To accomplish this, the homography can be decomposed into several simpler ones, that is, a scaling, a rotation, an affine transformation and a projective transformation. Shifts are not considered, since signature extraction itself is shift invariant. Thus, the complete homography is decomposed according to

$$H = H_S\, H_R\, H_A\, H_P, \tag{11}$$

where H_S is the scaling matrix, H_R is the rotation matrix, H_A is the affine matrix, and H_P is the projective matrix. Since H_A can be decomposed into

$$H_A = H_{R_1}\, H_C\, H_{R_1^{-1}}\, H_{R_2}, \tag{12}$$

where H_{R_1} is a rotation matrix, $H_{R_1^{-1}}$ its inverse, H_C is an anisotropic scaling matrix and H_{R_2} a second rotation matrix, the whole transformation can be written as:

$$H = H_S\,(H_R H_{R_1})\, H_C\, (H_{R_1^{-1}} H_{R_2})\, H_P = H_S\, H_{R_V}\, H_C\, H_{R_U}\, H_P, \tag{13}$$

$$\begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} = \begin{pmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & b & 1 \end{pmatrix}. \tag{14}$$

For the scaling homography H_S, one random value is generated for k. To prevent large or small samples, this value must be limited between $1/t < k < t$, where $t > 1$ is a threshold that controls how large or small the samples can be. For the rotation matrices H_{R_V} and H_{R_U}, random values are generated for α and β, in both cases ranging from $0 < \alpha, \beta < 2\pi$. For the anisotropic scaling H_C, another value is obtained for k_y with the same restriction as the one for k above. Finally, the matrix for the projective transformation H_P needs to be explained in more detail.

To prevent the shape from undergoing excessive deformation, it is only necessary to ensure that the line at infinity does not lie close to the object, as shown in Fig. 6. As the line at infinity approaches the object, the shape becomes increasingly deformed. It can be demonstrated that for the matrix corresponding to the

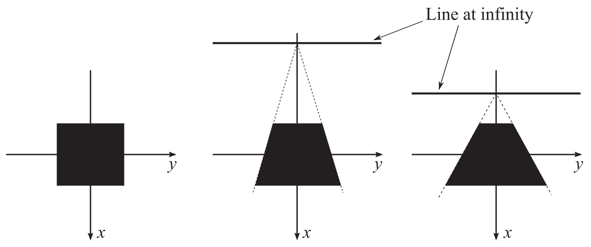

Fig. 6. Example of a projective transformation.

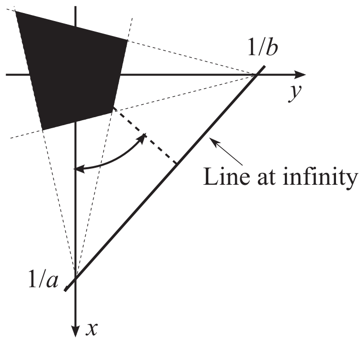

Fig. 7. Computation of the projective homography.

projectivity in (14), the point $1/a$ corresponds to the cross point between the line at infinity and the x axis, as shown in Fig. 7, and the same is true for the other point. Thus, it is only necessary to obtain two random values, a and b, to generate the projective homography. To limit the amount of deformation, these values need to be limited. As the value of a approaches 0, the cross point between the line at infinity and the x axis approaches infinity. Conversely, as the value of a rises, the deformation of the shape increases. Thus, it is only necessary to set a limit for these values, such that $0 < a, b < t_p$, where t_p is the threshold for the projective transformation.

The last step is the addition of noise to simulate the distortions undergone by the contour of the object in real images. To this end, the application adds circular patches at random positions near the contour of the object. The algorithm works as follows: once the projected image has been generated, the contour of the object is extracted, and an array of coordinates is created from all these points. Afterwards, one point P is chosen at random, as well as an angle γ and a displacement Δ, which define the center of a circumference of radius $R > \Delta$, also chosen at random. Fig. 8 shows an example of all these variables. If the center of the circle lies outside the object, a circle with the same color as the background is drawn. On the other hand, when the point lies inside the object, the circle has the same color as the object. This process is repeated a sufficient number of times, to allow the application to draw these circular patches along the whole contour.

Finally, to be able to simulate and evaluate different scenarios, it is necessary to control the amount of noise added σ. This can be achieved by limiting the maximum value of the radius R, and consequently Δ. For instance, if the amount of noise σ is set to 5 pixels, all the patches

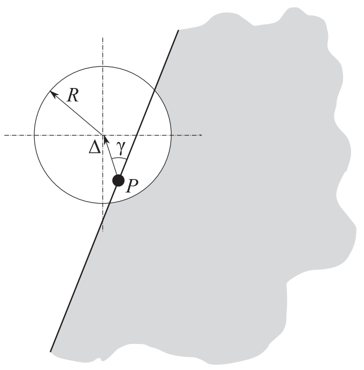

Fig. 8. Noise addition to the transformed shape.

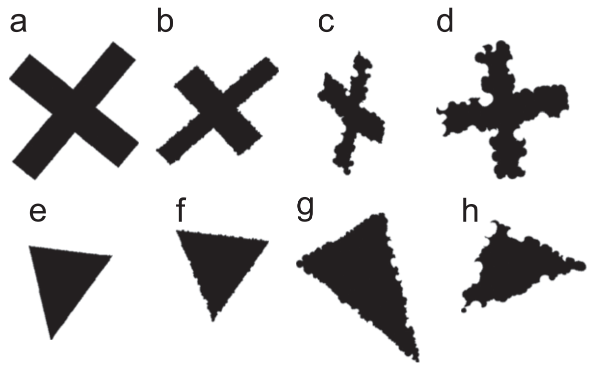

Fig. 9. Images generated for different levels of noise, in pixels. (a) Noise=0; (b) Noise=4; (c) Noise=8; (d) Noise=12; (e) Noise=0; (f) Noise=4; (g) Noise=8; and (h) Noise=12.

drawn over the figure will have radii ranging from 0 to 5 pixels. Fig. 9 shows some images obtained with the application, in this case for triangles and crosses, for different levels of noise.[1]

9. Results and discussion

In order to test the proposed algorithms, and compare their effectiveness, the application described in Section 8 was used to generate distorted replicas of the reference shapes shown in Fig. 5. The goal of this section is twofold. On the one hand, we present a comparison of the performance of all the algorithms proposed in this paper. On the other hand, we report testing of the algorithms in the presence of different levels of noise. All the algorithms were implemented in C++ using OpenCV 2.1. For the EMD, RBF-SVMs and k-NN classification, we employed the implementations in OpenCV 2.1 library [30]. For RBF-SVM, we fixed $C=1000$ and $\gamma=8$ after an exhaustive search. Likewise, $k=5$ for k-NN.

To test the performance of the algorithms, 200 images of each shape were created, that is, 200 triangles, 200 squares, and so on, so that a total of $200\times 8=1600$ images were created for each level of noise. The level of noise σ ranged from 0 to 20 pixels in steps of 2 pixels (see Section 8 for a description of the meaning of the level of noise σ). Since the performance of the system against projective distortions was not analyzed in this study, the images in this set only included scalings, rotations and affine distortions.

9.1. Signature and FFT length setting

Two parameters regarding the length of the signature needed to be defined prior to any other experiment. Firstly, it was necessary to choose the length of the signature itself in terms of accuracy and execution time. Since for some classifiers, a power of two for this length is needed, the performance of the MSE, MSE FFT and CC classifiers was only evaluated for 32, 64, 128 and 256 samples of the signature. The results show that, from lengths equal to 64 onwards, the probability of success was approximately the same, and execution time rose as the length of the signature increased. For this reason, a length of 64 samples was chosen for the signature, and all the remaining evaluations were made with this length.

Secondly, it was necessary to select the number of samples to use for those algorithms using the AbFFT. Since the actual length of the signature had been fixed at 64 samples, the number of useful AbFFT samples was half of that length, that is, 32. Experiments have shown that using the sample in 0, that is, the mean value, does not contribute at all to the classification process, thus it is possible to remove this sample without any detriment to the process.

Furthermore, we also tested the performance of the MSE FFT algorithm as a function of the number of samples used. To this end, we analyzed the success probability of the algorithm when the classification vector was composed only of the sample in 1, the samples in 1 and 2, and so on, until the vector was composed of the samples in 1–32. Fig. 10 gives the results for the different values, showing that the best performance occurs at 5 samples, and also that from this length onwards, as the number of samples increases, the performance decreases. This is due to the fact that these samples belong to the medium–high frequencies, and for a sign such as the signature defined in

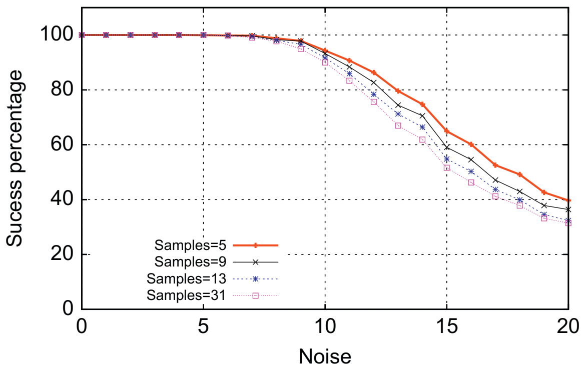

Fig. 10. Performance of the MSE FFT classifier as a function of the noise of the test images, for a different number of AbFFT samples.

[1] All the images generated for the evaluations described in this paper can be found at http://agamenon.tsc.uah.es/Investigacion/gram/papers/Elsevier10.

this study, medium–high frequencies are strongly affected by noise, and thus, do not contribute at all to the classification process.

In short, for all the experiments described below, the length of the signature was 64 samples, and for the algorithms using the AbFFT of the signature, the length of the classification vector was 5 samples, specifically, samples 1–5.

9.2. SVMs and k-NN training set design

For the SVMs and k-NN algorithms, one additional set of images for training was required. In this case, we did not need a complete range for the level of noise, but rather a set of figures that accurately characterized each shape considering the different geometric distortions. Consequently, it was necessary to place careful limits on the maximum amount of noise, called the *Training Noise*, σ_T, added to the figures. High levels of training noise will result in shapes which are quite different to the original one, and this would lead to a decrease in the performance of the classifiers, rather than in an improvement. To this end, we tested the performance of the SVMs considering different levels of training noise.[2]

This process is illustrated in Fig. 11. The horizontal axis represents the value of the training noise σ_T, that is, for $\sigma_T = 0$, the training set was composed only of shapes without noise. For $\sigma_T = 2$ the training set was composed of shapes with noise σ ranging from 0 to 2 pixels, where σ is a random variable. For $\sigma_T = 4$, shapes with noise ranging from 0–4 pixels were included, and so on. All the training sets included the same amount of figures, in this case, 50 images per shape; or consequently, $50 \times 8 = 400$ figures all the shapes together. Furthermore, each curve in Fig. 11 corresponds to the success probability for a particular test set of a fixed level of noise. We can see that, for figures with a small amount of noise, the results are almost independent of the training noise. On the other hand, for figures with a high amount of noise, the success probability increased as the training noise increased. For figures with a medium amount of noise, the top were obtained at about 10 pixels of noise. Since we were interested in figures with a small to medium amount of noise, because, as we can deduce by simple inspection of the test set, figures with noise levels above 15 pixels do not resemble the original shape, we focused only on curves for $\sigma = 0$–10. For these sets, the optimal value of the training noise was near 10 pixels.

The other question to answer regards how many samples of each class are necessary in the training process. In normal systems, with a limited number of manually annotated samples, all the available samples are used to train the SVMs. In our study, we had an unlimited number, since they were generated automatically. However in this case, we had to determine how many of them were needed. With a small number of samples, it is unlikely that the SVMs will capture all the characteristics of each class.

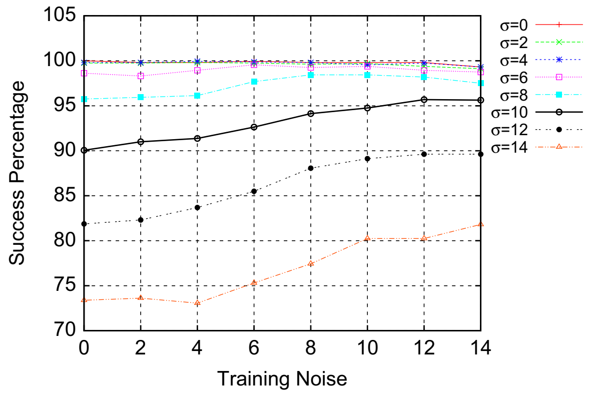

Fig. 11. Performance of the SVM classifier as a function of the noise of the training images, for different levels of image test set noise.

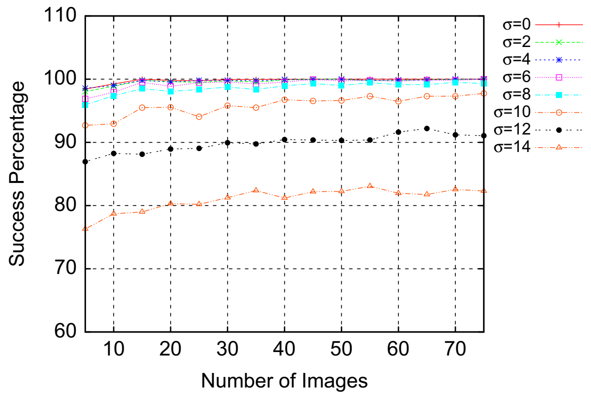

Fig. 12. Performance of the SVM classifier as a function of the number of images used for the training process, for different levels of image test set noise.

A large number, on the other hand, will lead to a large number of support vectors [31], increasing the computation time in the test process. Fig. 12 shows the success probability as a function of the number of images per class employed in the training process, and it can be seen that as the number of images increased, the overall success probability also increased. Nevertheless, the figure also shows that the use of about 50 images per class was more than enough for an acceptable performance.

Therefore, for the rest of the experiments performed in this study, the SVMs, and also the k-NN algorithm, were trained with $50 \times 8 = 400$ images, each of them with a level of noise that varied randomly between $0 < \sigma < 10$ pixels.

9.3. Comparison of methods

In this section, we compare all the described classification methods and extract some conclusions about their performance. Fig. 13 shows this comparison.[3] As can be

[2] A similar experiment was conducted with the k-NN algorithm, but it will not be discussed here as the differences between both algorithms were not relevant.

[3] The complete set of results, and all the images employed in the experiments can be found at http://agamenon.tsc.uah.es/Investigacion/gram/papers/Elsevier10.

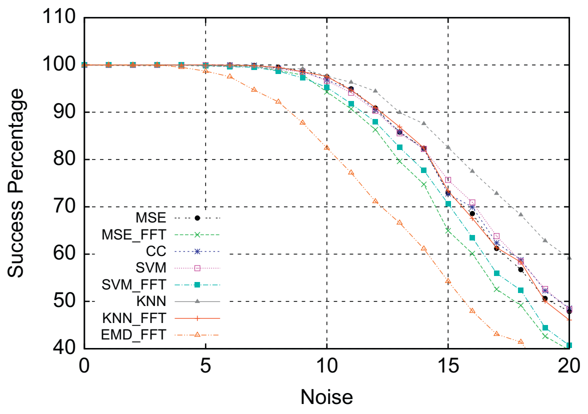

Fig. 13. Performance comparison of the eight classifiers described in this paper.

Table 1
Execution time.

Algorithm	Time (s)
MSE	5.05
MSE FFT	3.9
CC	4.91
EMD FFT	4.06
SVMs	4.34
SVMs FFT	3.82
k-NN	5.36
k-NN FFT	4.06

seen, for the classification of figures with a level of noise below 10 pixels, the performance of all the methods was almost identical, and close to 100%. However, as the noise increased, the performance of those classifiers not using the FFT presented better performance than those which did, and, among all the algorithms, k-NN yielded the best results for figures with a level of noise above 15 pixels.

The other parameter evaluated was the computational time employed in processing all the images.[4] Table 1 summarizes the time employed by each algorithm. As expected, the algorithms using the FFT took less computational time at the expense of a reduction in classification accuracy. Nevertheless, the reduction in time with respect to the non-FFT version was not entirely clear, showing that the use of the FFT in the classification process did not yield any clear advantage. Besides, the worst algorithms in terms of computational time are the k-NN and the MSE ones.

9.4. False positive detection

So far, several algorithms that allow a system to classify a shape into one of a finite number of classes have been described. However, we have not yet taken into account the detection of false positives. In this section, we describe two different implementations of this part of the system.

[4] Time elapsed in the computation of 1600 images of size 500×500 pixels over a Pentium 4 2.4 GHz on a Linux kernel 2.6.32-25.

The first method consists in setting a threshold μ on the measure evaluated for the classification purpose. For instance, for classifications based on the Mean Squared Error, it is necessary to identify not only the reference shape that yields the minimum error given in (3), but also to verify whether this value is greater than a given threshold or not. This threshold can also be set for the MSE FFT, the CC or the EMD FFT classification systems proposed here, or any other potential classification strategy.

The second approach, which only can be applied to SVMs and k-NN classifiers, consists in the addition of one extra class, labeled *noise*, and the classifier in this case must classify every shape into one of the predefined classes, including this new one. All those objects classified as *noise* will be treated as false positives and, most probably, discarded for further steps.

For this second approach, one question arises immediately, which is how to generate noise samples. In Section 8, we propose an application that can generate replicas of a reference shape with several geometric distortions, including variable contour distortions. For the latter, as the distortion of the contour increases, the shape becomes progressively more deformed, up to a point where the object generated no longer resembles

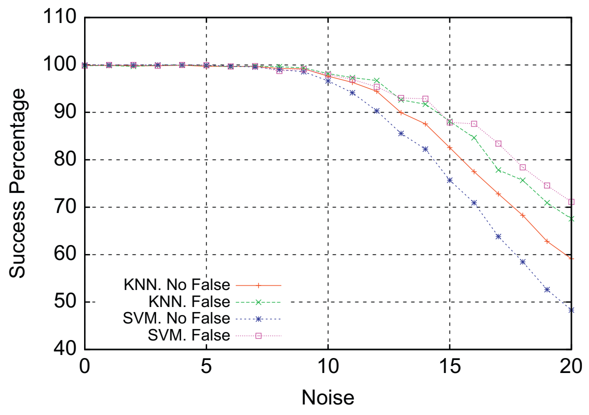

Fig. 14. Comparison of the success probability of the SVMs and k-NN methods with and without *False Positive* detection.

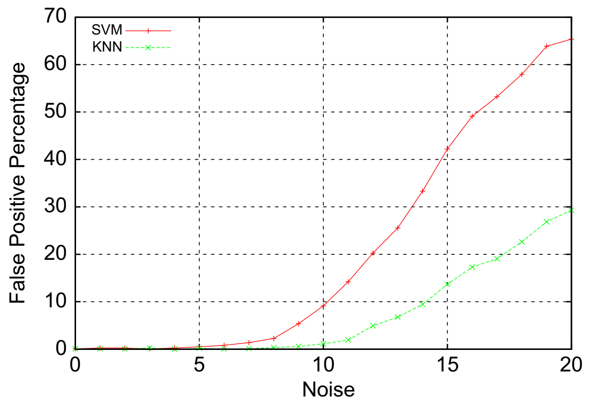

Fig. 15. Percentage of figures discarded as *False Positive* as a function of noise.

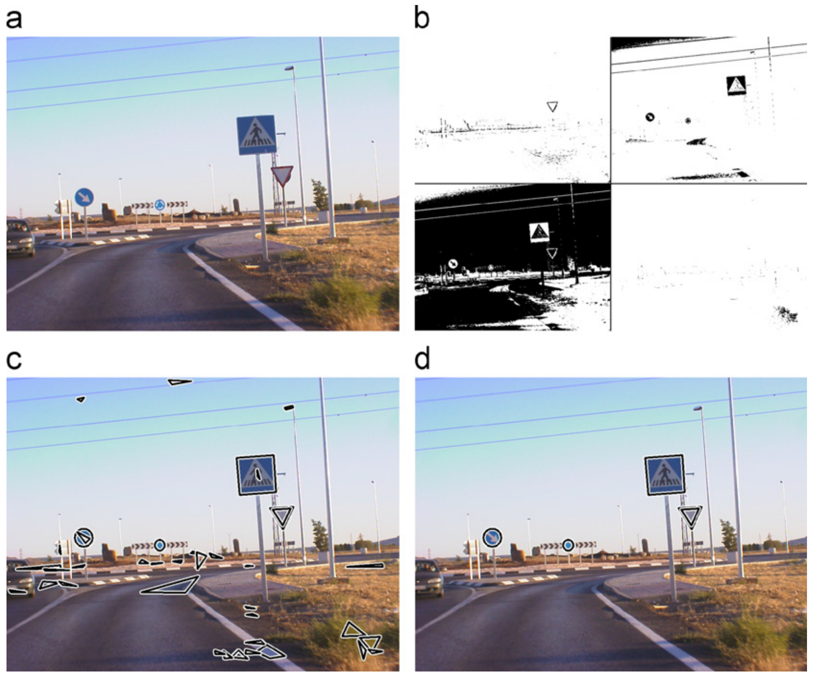

Fig. 16. Experimental results with real images. (a) Original image. (b) Segmentation masks. (c–d) Detection and localization without false positive removal (c) and with false positive removal (d). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

the original shape. Accordingly, it is possible to use the same application to additionally generate as many samples of noise as desired in an automatic fashion. To set the value of σ used to generate the noise samples, the results shown in Fig. 13 can be used. Here, it can be seen that, for figures with $\sigma = 20$, success probability reached the minimum value, showing that these figures do not preserve the original shape and thus, can be used as noise samples. Accordingly, the training set was composed of 50 samples of each shape, with σ ranging from 0 to 10 pixels, and 400 samples of noise, generated by setting $\sigma = 20$ pixels.

To show the improvement achieved, we performed the following experiment. Having trained the SVMs, or the k-NN, with the new training set described above, we compared the success probability of this new schema with the one obtained previously, as shown in Fig. 14. However, in this case, we computed the success probability as the number of correctly classified shapes with respect to all the shapes evaluated, except those classified as false positives. Fig. 15 shows the number of figures discarded as *False positives*. As expected, the algorithm discarded few or no samples for figures with a small amount of noise, whilst for figures with a noise level of approximately 15 pixels or higher, the SVMs discarded more than half, while the k-NN about one third. This indicates that the algorithms were able to detect the amount of noise in the figure, and eliminate all those that no longer resembled the original shape. Looking again at Fig. 14, we can see how the success probability increased, especially for figures with a level of noise ranging from 10 to 15 pixels. For the FFT version of the algorithm, the same experiment was carried out. However, it showed a worse performance with respect to the non-FFT version, and, as in the previous comparison, only a small reduction in the computational time.

For the first proposal, that is, thresholding the measure evaluated, we performed a similar experiment, but the results were not as good as expected, showing that this strategy is not suitable for this task. To summarize, from the results obtained, we conclude that, when false positive detection is needed, SVMs become the best option in terms of accuracy and execution time.

Lastly, we tested the performance of the algorithm, including false positive detection, in real scenarios. To this end, the system implemented in [19] incorporated a segmentation module in addition to the shape classification module described here, together with a localization module for rectangles, triangles and circles. For this experiment, only SVMs, which was proven to be the best classifier when false positive detection is needed, was used. Furthermore, the system was able to superimpose the detected shape over the original image to test whether the shape has been correctly classified and if so, determine the accuracy of the localization of its position. An example is given in Fig. 16.[5] In Fig. 16(a), the original image is shown whilst Fig. 16(b) shows the segmentation mask for the colors red, blue, white and yellow. In Fig. 16(c) all the shapes detected are drawn over the original image. Since the localization module is able to localize triangles, rectangles and circles, only these shapes were considered for the shape classification process. This implies that training for new SVMs must only include

[5] More images, and their corresponding results can be found at http://agamenon.tsc.uah.es/Investigacion/gram/papers/Elsevier10.

the considered shapes. Fig. 16(d) illustrates the effectiveness of the False Positive detection. This image was obtained by including the False Positive set of images in the training set, along with the triangles, rectangles and circles. As we can see, a significant number of noise objects were removed, and those remaining comprised the false positives that cannot be removed since they are reasonably similar to the considered shapes.

10. Conclusions

In this paper, we have presented a study of different classification techniques for shape classification based on the signature of the blob. Although the signature is not an appropriate tool for working with complex shapes, when working with simple ones it has proven to be very robust against typical geometric distortions, such as shifts, rotations, scalings, affine transformations and segmentation noise. The algorithm was tested with randomly generated images, using an application specifically designed for this purpose. The parameters evaluated were the shape classification success probability as a function of the segmentation noise and the computational time. The algorithm was also tested with real images, showing an overall good performance, thus demonstrating its suitability for tasks such as traffic sign recognition.

Furthermore, several kinds of classifiers were tested, from the simpler ones, such as the *Mean Squared Error* to the more complex ones, in this case the well-known *Support Vector Machines* and the *k-Nearest Neighbor*, with and without the use of the FFT to overcome rotation problems. One of the novelties of this study was the fact that the samples were not manually selected, but generated automatically. Thus, a complete study of the composition of the training process was also possible, and conclusions regarding the number and kinds of the samples used are given. The advantage underlying this process is that the workload of manually selecting the training samples is not needed.

The results obtained led to the following conclusions: firstly, when the quality of the images and the segmentation process is adequate, any one of the proposed methods yields a success probability close to 100%, but as deformation of the segmented object increases, *k*-NN performs better than the rest of the approaches, although with a higher computational time than the rest of the algorithms. Furthermore, the use of the FFT to overcome rotation problems is not the best option, as it yields a loss in accuracy and little reduction in the computational time.

The study was extended to examine the false positive problem, and results show that, with careful design of the process for generating the false positive samples for the training set, the use of SVMs can greatly improve the performance of the system. For this case, *k*-NN does not yield as good results as the SVMs do.

For future research, a study of the effects of projective distortions on the object is required. In this case, SVMs appear to be the main candidate for solving the problem, but a more exhaustive design of the generation of the training set, which should include projective distortions, is necessary. Furthermore, other potential classifiers, such as Neural Networks or Decision Trees should be compared to obtain conclusions about their relative performance.

Acknowledgments

This work was supported by the Spanish Ministry for Science and Innovation project number TIN2010-20845-C03-03 and the Comunidad de Madrid-Universidad de Alcala project number CCG10-UAH/TIC-5965.

References

[1] R. González, R. Woods, Digital Image Processing, Addison-Wesley, 1993.

[2] T. McElroy, E. Wilson, G. Anspach, Fourier descriptors and neural networks far shape classification, Proceedings of the International Acoustics, Speech, and Signal Processing ICASSP-95. Conference, vol. 5, 1995, pp. 3435–3438.

[3] I. Kunttu, L. Lepisto, J. Rauhamaa, A. Visa, Multiscale fourier descriptor for shape classification, in: Proceedings of the 12th International Image Analysis and Processing Conference, 2003, pp. 536–541.

[4] Z. Ling, C. Zhao, Q. Pan, Y. Wang, Y. Cheng, Analyzing human movements from silhouettes via fourier descriptor, in: Proceedings of the IEEE International Automation and Logistics Conference, 2007, pp. 231–236.

[5] D. Zhang, G. Lu, Enhanced generic fourier descriptors for object-based image retrieval, Proceedings of the IEEE International Conference Acoustics, Speech, and Signal Processing (ICASSP '02), vol. 4, 2002.

[6] H. Kauppinen, T. Seppanen, M. Pietikainen, An experimental comparison of autoregressive and fourier-based descriptors in 2d shape classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 201–207.

[7] D. Shen, W. him Wong, H.S. Horace, Affine-invariant image retrieval by correspondence matching of shapes, Image and Vision Computing 17 (7) (1999) 489–499.

[8] R.E. Loke, M.M. Bayer, D.G. Mann, J.M.H. du Buf, Diatom recognition by convex and concave contour curvature, Proceedings of the OCEANS '02 MTS/IEEE, vol. 4, 2002, pp. 2457–2465.

[9] K.-H. Guo, J.-Y. Yang, An improved curvature coding for planar shape representation, in: Proceedings of the Chinese Conference on Pattern Recognition CCPR '08, 2008, pp. 1–5.

[10] C.R.P. Dionisio, H.Y. Kim, New features for affine-invariant shape classification, Proceedings of the International Conference Image Processing ICIP '04, vol. 4, 2004, pp. 2135–2138.

[11] W.-Y. Kim, Y.-S. Kim, A region-based shape descriptor using Zernike moments, Elsevier Signal Processing 16 (1) (2000) 95–102.

[12] Y. Mei, D. Androutsos, Robust affine invariant region-based shape descriptors: the ICA Zernike moment shape descriptor and the whitening Zernike moment shape descriptor, IEEE Signal Processing Letters 16 (10) (2009) 877–880.

[13] V.N. More, P.P. Rege, Devanagari handwritten numeral identification based on Zernike moments, in: Proceedings of the TENCON 2008—2008 IEEE Region 10 Conference, 2008, pp. 1–6.

[14] A. Goyal, E. Walia, H.S. Saini, Improved accuracy in shape based image retrieval with complex Zernike moments using wavelets, in: Proceedings of the 2nd International Congress on Image and Signal Processing CISP '09, 2009, pp. 1–5.

[15] N. Ezer, E. Anarim, B. Sankur, A comparative study of moment invariants and fourier descriptors in planar shape recognition, in: Proceedings of the 7th Mediterranean Electrotechnical Conference, 1994, pp. 242–245.

[16] J. Kim, M. Çetin, A.S. Willsky, Nonparametric shape priors for active contour-based image segmentation, Elsevier Signal Processing 87 (12) (2007) 3021–3044.

[17] A. Tsai, W.M. Wells, S.K. Warfield, A.S. Wilsky, An EM algorithm for shape classification based on level sets, Medical Image Analysis 9 (5) (2005) 491–502.

[18] M.E. Leventon, W.E.L. Grimson, O. Faugeras, Statistical shape influence in geodesic active contours, in: Proceedings of the 5th IEEE EMBS International Biomedical Imaging Summer School, 2002.

[19] P. Gil-Jiménez, S. Maldonado-Bascón, H. Gómez-Moreno, S. Lafuente-Arroyo, F. López-Ferreras, Traffic sign shape

classification and localization based on the normalized FFT of the signature of blobs and 2D homographies, Elsevier Signal Processing 88 (12) (2008) 2943–2955.
[20] P. Gil-Jiménez, S. Lafuente-Arroyo, S. Maldonado-Bascón, H. Gómez-Moreno, Shape classification algorithm using support vector machines for traffic sign recognition, in: Computational Intelligence and Bioinspired Systems, Lecture Notes in Computer Science, vol. 3512, 2005, pp. 873–880.
[21] H. Gómez-Moreno, S. Maldonado-Bascón, P. Gil-Jiménez, S. Lafuente-Arroyo, Goal evaluation of segmentation algorithms for traffic sign recognition, IEEE Transactions on Intelligent Transportation Systems (ISSN: 1524-9050) PP (99) (2010) 1–14.
[22] A.K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, 1989.
[23] Y. Rubner, C. Tomasi, L.J. Guibas, A metric for distributions with applications to image databases, in: Proceedings of the Sixth International Conference on Computer Vision, 1998, pp. 59–66.
[24] V. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, New York, 2000.
[25] S. Maldonado-Bascón, J. Acevedo-Rodríguez, S. Lafuente-Arroyo, A. Caballero-Fernádez, F. López-Ferreras, An optimization on pictogram identification for the road-sign recognition task using SVMs, Computer Vision and Image Understanding 114 (3) (2010) 373–383.
[26] J. Acevedo-Rodríguez, E. Domínguez, S. Maldonado-Bascón, A. Narváez, F. López-Ferreras, Probabilistic support vector machines for multi-class alcohol identification, Sensors and Actuators B 122 (2007) 227–235.
[27] D. Meyer, F. Leisch, K. Hornik, The support vector machine under test, Neurocomputing 55 (1-2) (2003) 169–186 Support Vector Machines.
[28] J.H. Min, Y.-C. Lee, Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters, Expert Systems with Applications 28 (4) (2005) 603–614.
[29] C.M. Bishop, Pattern Recognition and Machines Learning, Springer, 2006.
[30] G. Bradski, A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, first ed., O'Reilly Media, 2008.
[31] I. Steinwart, Sparseness of support vector machines—some asymptotically sharp bounds, in: Proceedings of the 16th NIPS Conference, 2004, pp. 169–184.