Proceedings of the
2007 IEEE Intelligent Vehicles Symposium
Istanbul, Turkey, June 13-15, 2007



# Traffic sign shape classification based on Support Vector Machines and the FFT of the signature of blobs

P. Gil-Jiménez, H. Gómez-Moreno, P. Siegmann, S. Lafuente-Arroyo and S. Maldonado-Bascón
Dpto. Teoría de la Señal y Comunicaciones, Universidad de Alcalá
Alcalá de Henares, 28871 (Madrid), Spain
{pedro.gil,hilario.gomez,philip.siegmann,sergio.lafuente,saturnino.maldonado}@uah.es

***Abstract*— In many traffic sign recognition systems, one of the main tasks is the classification of the shape of the blob, which is intended to simplify the recognition process. In this paper, we have developed a new shape classification algorithm based on Support Vector Machines classifiers and the FFT of the signature of the blob. The FFT of the signature yields invariance to object scalings and rotations. Furthermore, the FFT is the vector input to the classifier. This classifier is trained to cope with projection deformations and occlusions.**

**The algorithm has been tested under adverse conditions, such as geometric distortions, i.e. scaling, rotations and projection deformations, and occlusions. The experimental results show good robustness when the system is working with real, outdoor road images.**

## I. INTRODUCTION

Automatic road sign detection and recognition have been studied in several works recently [1]–[4]. Recognition of traffic signs is an important issue for Driver Assistant Systems and unmanned vehicles. They can also be used as an inventory system in order to get a complete catalog of all the existing traffic signs in a particular road. Furthermore, these kind of systems can yield information about their state and condition.

### A. System overview

In figure 1 we can see the block diagram of a typical traffic sign recognition and tracking system, like the one we have developed. Although the system is mainly intended to work off-line, that is, a road sequence has been previously recorded and stored, and processed later, the system can be easily used as a real time system with the use of specialized hardware to cope with the real time requirements.

The system consists of four main blocks. The segmentation is the first step, which is designed to separate the objects of interest, in this case the possible traffic signs, from the background. Although many techniques have been proposed, most of them use color information to achieve the sign segmentation. For instance, for the Spanish traffic sign set, the most appropriate segmentations are based on red, blue, white and yellow colors. The output of this block is the list of blobs obtained from the computation of the connected components of each segmentation mask.

The shape classification block performs the identification of each blob according to its shape. As it will be seen later,

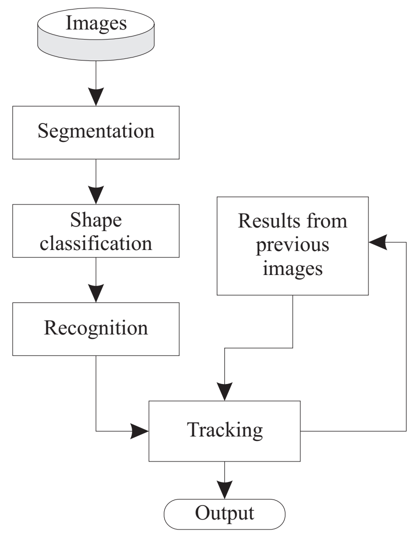



Fig. 1. Block diagram of a traffic sign recognition and tracking system.

in our case, these shapes are the triangle, rectangle, circle and semicircle.

The following step is the recognition of the meaning of the sign according to its shape, previously computed, and its content. The last step performs the tracking of the recognized signs using the information obtained from the current image, and information coming from the previous images of the sequence.

The algorithm described in this paper is related to the shape classification block in figure 1. The input to this block is a list of connected components computed from the different segmentation masks obtained on the segmentation block. The output will be the same list of blobs updated with the shape for each blob.

## II. SHAPE CLASSIFICATION

The goal of the shape classification step is the identification of the shape of all blobs obtained on the segmentation block. For the traffic sign recognition problem, in our case for the Spanish traffic sign set, these shapes are typically the equilateral triangle, the square and the circle. Besides, we have added the semicircle, since for some signs, specially those from the "end of prohibition" group, the segmentation step generally divides the whole circle into two similar semicircles.

In this work, the classification is achieved using a pattern recognition module based on Support Vector Machines (SVMs). The input will be the absolute value of the FFT of the signature of the blobs as defined and explained in section II-A.

### *A. Blob Signature*

The signature of the blob is an unidimensional representation of the contour of an object, and can be generated in several ways [5]. In this work, we will define the signature as the distance from the mass center of the object to the edge as a function of the angle. The mass center can be computed from the moments of the blob according to:

$$(x_c, y_c) = (m_{10}/m_{00}, m_{01}/m_{00}) \tag{1}$$

where $m_{00}$ is the area of the blob, and $m_{10}$ and $m_{01}$ are its first order moments. In figure 2 we can see the theoretical signatures for the four different shapes we have taken into account in the traffic sign recognition problem.

The signature allows the simplification of the bidimensional representation of the contour of the blob into a unidimensional one, which is easier to analyze using common signal processing techniques. The main limitation is that the signature is an accurate representation only for convex objects with a not so high eccentricity [6]. If this is not the case, the signature may not be a precise representation for such objects. Note however, that for the traffic sign recognition problem, the shapes defined are all convex and with a small eccentricity, and so, the signature can be an appropriate tool for the representation of the considered shapes.

The main advantage of the use of the signature for the classification of shapes is its invariance to object scaling and rotation with little modifications. Object scaling becomes a signature amplification, that is, if the object is imaged enlarged by some factor, the samples of the signature get multiplied by the same factor. A signal normalization can be carried out to get the scaling invariance property. In this work, the normalization is performed computing the total energy of the signal and dividing each sample by the square root of that energy, so that the total energy of the signal is equal to 1.

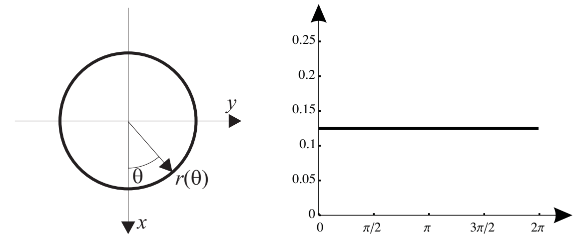
With another modification we can also make the algorithm invariant to object rotations. Rotation implies circular shifts on the signature of the object. Taking advantage of the invariance of the module of the DFT to shifts:

$$\begin{array}{ccc} y[n] & \Rightarrow & Y(\Omega) \\ y[n-n_0] & \Rightarrow & Y(\Omega)\, e^{j\Omega n_0} \end{array} \tag{2}$$

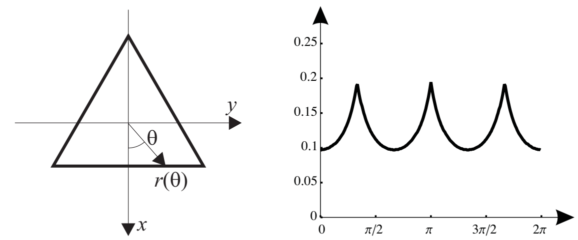
we can overcome the problem of object rotation by simply computing the absolute value of the DFT (AbDFT), using this vector as the input to the classifier. The computational complexity can be reduced computing the FFT of the signature instead of the DFT. This is always possible as long as the number of samples of the signature is a power of two. Therefore, the signature of the object is sampled at the following interval for the angle $\theta$:

$$\Delta\theta = \frac{2\pi}{2^n} \tag{3}$$

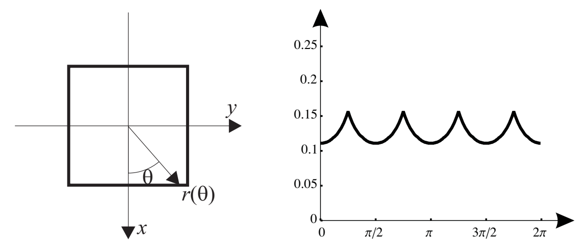
where $m = 2^n$ is the number of samples of the signature, which is a parameter that must be chosen. A small value for $m$ reduces the accuracy for the representation of the object through the signature. A large value for $m$ increases the accuracy, but also the computational complexity of the algorithm. For the current work, we chose $n = 6$, or equivalently, the number of samples of the computed signature $m = 64$. Since we are using the samples of the absolute value of the FFT (AbFFT) of the involved signatures, and the signatures are real signals, the AbFFT is symmetric, and so, only half of the samples is required.



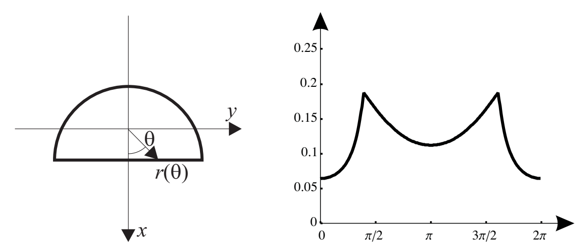

(a) Circle




(b) Triangle

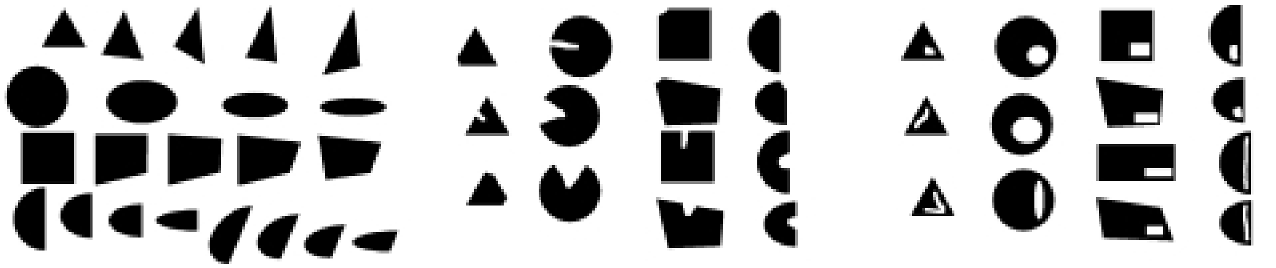



(c) Square




(d) Semicircle

Fig. 2. Reference shapes and their signatures.



(a) Different shapes and deformations (b) Opened segmentation errors and occlusions (c) Closed segmentation errors and occlusions

Fig. 3. Training set examples

### B. SVMs Classification

The next step is the classification of the AbFFT of the signature according to the classes we have defined: triangle, circle, square and semicircle. In previous works [7] we have performed the classification using a simple difference classifier. This classifier calculates the difference between the AbFFT of the signature to be classified with respect to the AbFFT of the signature of the four classes previously computed. The class assigned is the one with the smaller difference. This scheme needs some previous preprocess of the blobs in order to make it invariant to geometric deformations; the calculus of the second order moments and, according to this, an affine transformation to restore the object to its ideal form. Besides, due to physical occlusions and segmentation errors, the signature may be incomplete and some solution must be designed to overcome this lack of information. In [7] we proposed an interpolation algorithm to restore the lost information, based on the known samples of the signature.

In this work we explore the possibilities of the SVMs [8]–[10] in this task. Although the principles of SVMs are rigorously explained in [8] we extract here some concepts to best understand the rest of the paper. The goal of SVMs is to find a classification function based on some of the training vectors that are near the classification frontier, which are known as support vectors. This function is expressed as:

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i K\left(x_i, x\right) + b \qquad (4)$$

where $x$ is the vector to be classified, $x_i$ and $y_i$ are the support vectors and its associated labels, $l$ is the number of support vectors and the function $K\,()$ is known as the kernel, which allows us to obtain nonlinear decision functions. In the classification problem, the most general case is when not all vectors can be completely separated. In this case the SVMs training problem is formulated as follows:

$$\begin{array}{l} \text{minimize } \Phi\left(\omega, \xi\right) = \frac{1}{2}\left\langle \omega \cdot \omega \right\rangle + C\left(\sum_{i=1}^{l} \xi_i\right)^k \\ \text{subject to } y_i \left\langle \omega \cdot x_i \right\rangle + b \geq 1 - \xi_i \\ \text{with } \xi_i \geq 0 \text{ and } 0 < \alpha_i \leq C \end{array} \qquad (5)$$

where $\omega$ is the optimal separation hyperplane and, when using a kernel, is defined as follows:

$$\omega = \sum_{i=1}^{l} \alpha_i y_i K(x_i, \cdots) \qquad (6)$$

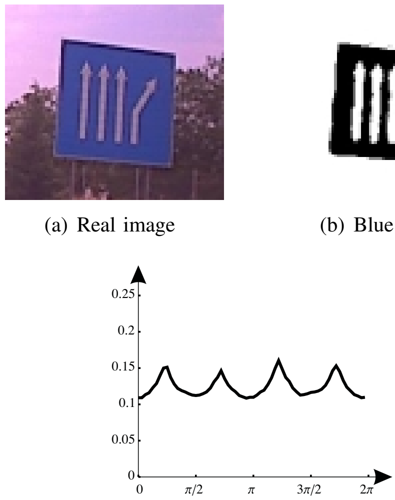
as appears in the decision function (4). The $\xi_i$ values in equation 5 represent the errors made with nonseparable vectors and $C$ is an “a priori” constant that gives more or less importance to errors in the minimization process. The $\alpha_i$ values are obtained during minimization and, in conjunction with the support vectors ($x_i$), give the optimal separation hyperplane.

The main idea behind the use of SVMs in this work is to avoid the preprocessing tasks before classification. This way, the blob could be classified directly using the AbFFT samples. The SVMs are known to have a good generalization ability [10] and thus, with a reduced set of training examples, a good general classifier can be designed. Therefore the election of the training set is a crucial point in order to obtain good classification results.

### C. Training set

This section describes how we have arranged the training examples in order to obtain the best generalization and classification. We have two options in order to build the training set. The first one is the use of real images extracted form sequences taken with a video camera. This option was abandoned since we do not have total control over the training process and also we would have to analyze a lot of images to obtain examples for all the deformations and occlusions cases we needed. The other option, which is the one we finally decided to use, is the manual generation of synthetic images, trying to simulate the same shapes, deformations and occlusions we can find in real images.

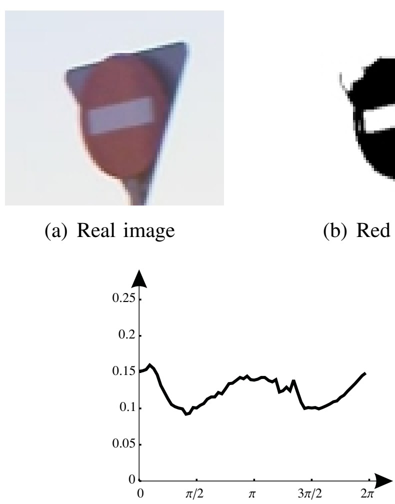
The first step is the manual generation of synthetic images with the reference shapes (triangle, square, circle and semicircle) which include some geometric deformations in order to train “generalized” shapes. Some examples of these training images are shown in figure 3(a) where triangles, circles, squares and semicircles appear directly and with some deformations. Ideally, only these images were needed to succeed in the classification of the shapes, but in real world the traffic signs can be occluded by obstacles and, sometimes, the segmentation step is not good enough to extract the whole shape, and hence we need more training images to include




(a) Real image (b) Blue mask

(c) Signature

Fig. 4. Example of rotation.




(a) Real image (b) Red mask

(c) Signature

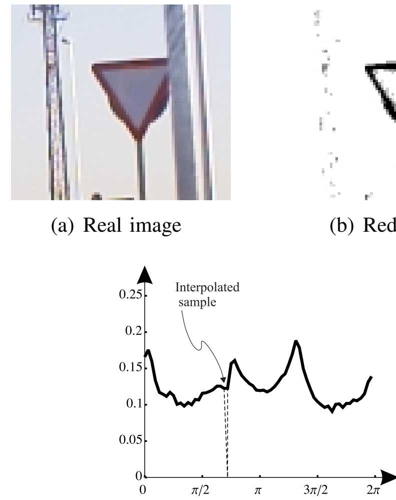
Fig. 5. Example of projection deformation.

these effects into the classifier. In figures 3(b) and 3(c) there are some examples of these effects, showing shapes with some parts erased, some of them opened (shape outside) and some of them closed (shape inside).

The images shown in figure 3 are only a part of the whole training set. We used a set composed of:

- 7 triangles, 4 circles, 20 squares and 13 semicircles with several deformations.
- 30 triangles, 24 circles, 144 squares and 39 semicircles with various opened occlusions and segmentation errors.
- 15 triangles, 12 circles, 48 squares and 26 semicircles with some closed occlusions and segmentation errors.

In the training process we used images like those in figure 3, except that in each image there are only one type of shape, e.g. we have images with only triangles, squares, circles or semicircles. These images are read, segmented and then the




(a) Real image (b) Red mask

(c) Signature

Fig. 6. Example of occlusion.

AbFFT of the signature of each blob is obtained and added to the training set. Afterwards, the training with these vectors is performed using the best parameters for the SVMs. The details about parameters selection and the implementation used are postponed until section III.

One advantage of using SVMs is that we can improve classification performance adding misclassified shapes to the training set. Thus, our training set may be increased if necessary. This option must be used with care, avoiding the addition of too noisy, or too specific examples. The retrained SVMs will always contain the added vectors as support vectors.

## III. EXPERIMENTAL RESULTS

The experimental results have been obtained using an implementation based on C language and the LIBSVM library [11]. The classifier we need must separate four classes, so then we need a multiclass SVMs. There are several ways to implement multi-classification with SVMs [12] but for simplicity we use the "one-against-all" scheme that is implemented in LIBSVM.

In order to evaluate the algorithm, the first step is the training of the SVMs, as we have described previously. We decided to use the gaussian kernel since it offers the best accuracy with the less number of support vectors. This kernel is defined as follows:

$$K(x, y) = e^{-\gamma\|x-y\|^2} \tag{7}$$

With SVMs based on gaussian kernels we have two parameters that must be set prior to the training process. The gaussian $\gamma$ must be set, taking into account that a high value makes the gaussian narrower, and then the SVMs tends to be specific for the training examples and, consequently, with poor generalization. A low value for $\gamma$ makes the gaussian wider, and the generalization is increased. The second parameter to be chosen is the constant $C$, that appears in the


(a) Correct classifications (b) Incorrect classifications

Fig. 7. Classification results in an artificial set of natural and deformed images

Fig. 8. Classification results examples in a real environment

minimization of the SVMs training process (equation 5). It is known that increasing this value the number of support vectors is sometimes reduced and, besides, the classification is improved since the nonseparable vectors are taken into account to find the decision function.

The faster way to find the optimal parameters for the SVMs is testing some of them and use those with the best results. In our case, after using a grid search over $C$ and $\gamma$ the optimal values were: $C = 100$ and $\gamma = 1$. With these parameters the training process gives 167 support vectors, while the number of training vectors used were 380.

The trained SVMs have been applied to different images in order to prove the performance of our algorithm. In figures 4, 5 and 6 we can see the computed signature for some traffic signs extracted from real images. In these images we can see the effect of geometric deformation and occlusions. We can also see the noisy nature of the signatures computed from real images. Nevertheless, we have to pay special attention to the occlusion problem. According to the definition of the signature as the distance from the mass center to the edge of the object, if an occlusion makes the edge disappear, as in figure 6, the occluded samples will not have a valid value. If, for simplicity, we set their value to zero, this will suppose a high frequency component, as can be seen in figure 6, (dotted line), that is, generally, undesirable. Although a complex interpolation algorithm can be designed to overcome this problem, the one proposed in this work is simply take the value of the previous sample as the value for the occluded sample. This method is faster than any other algorithm, while maintains a good performance, as was observed in our experiments.

In figure 7 we show the performance of the algorithm in a synthetic image that includes only traffic signs extracted from real images, where some of them have been projectively deformed in order to observe the robustness of the process. In this image the algorithm draws the detected shape over the image areas that must be classified. We can see that most of the forms has been correctly classified regardless of the size or the geometric deformation.

Once the correct performance were proved, the algorithm was tested in several sequences taken in outdoor environments. In figure 8 some examples of the results obtained are shown. In this figure we can see that, in real images, the segmentation can make appear some areas that are not actually traffic signs, and so their shapes are classified, as it would with real signs. This must not be considered as a failure of our algorithm, since we are only concerned in shape detection, and subsequent blocks of the system should discard these false alarms. Anyway, we can say that when the traffic sign is correctly segmented, the shape is generally correctly classified.

## IV. CONCLUSIONS AND FUTURE WORK

This paper describes a new algorithm for shape classification of traffic signs, based on Support Vector Machines, which uses as input the absolute value of the FFT of the signature of the blob. The shapes considered are the triangle, the square, the circle and the semicircle. The use of the signature of the blob has been proved to be highly invariant to object scaling and rotations. To deal with camera projection deformation and even sign occlusions, the Support Vector Machines are trained with a huge set of samples from each category, which includes different geometric deformations and occlusions. Furthermore, the samples were generated synthetically, in order to improve its performance and independence of real world images.

The algorithm has been tested with real outdoor noisy images which includes the different kind of geometric distortions considered, that is, scaling, rotations and projective deformation, and occlusions. These experiments have reported good results in the classification of the shapes of the traffic signs present in the images.

Our future work includes intensive testing of the algorithm with new sequences to see problems that may appear and then add new training images if necessary. New types of input vectors to the SVMs will be explored, for example, trying to use the signature directly instead of the FFT. Last, we want to test other kernels in order to reduce calculation complexity, trying to maintain the good results.

## V. ACKNOWLEDGMENTS

This work was supported by the project of the Ministerio de Educación y Ciencia de España with number TEC2004/03511/TCM.

## REFERENCES

[1] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, “A system for traffic detection, tracking, and recognition using color, shape, and motion information,” in *Proc. of IEEE Intelligent Vehicles Symposium*, Las Vegas, USA, June 2005, pp. 255–260.

[2] P. Paclick, J. Novovicova, and R. Duin, “Building road-sign classifiers using a trainable similarity measure,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 309–321, September 2006.

[3] A. de la Escalera, J. M. Armingol, J. M. Pastor, and F. J. Rodríguez, “Visual sign information extraction and identification by deformable models for intelligent vehicles,” *IEEE Trans. on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 57–68, June 2004.

[4] A. Soetedjo and K. Yamada, “Traffic sign classification using ring partitioned method,” *IEICE Trans. Fundamentals*, vol. E88-A, no. 9, pp. 2419–2426, September 2005.

[5] R. González and R. Woods, *Digital Image Proccesing*. Addison-Wesley, 1993.

[6] A. K. Jain, *Fundamentals of Digital Image Processing*. Prentice-Hall, 1989.

[7] P. Gil-Jiménez, S. Lafuente-Arroyo, H. Gomez-Moreno, F. López-Ferreras, and S. Maldonado-Bascón, “Traffic sign shape classification evaluation II: FFT applied to the signature of blobs,” in *Proc. IEEE Intelligent Vehicles Symposium*, Las Vegas, USA, June 2005, pp. 607–612.

[8] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 2000.

[9] N. Cristianini and J. Shame-Taylor, *Support vector machines and other kernel-based learning methods*. Cambridge, U.K.: Cambridge University Press, 2000.

[10] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[11] C. Chang and C. Lin, *LIBSVM: a library for support vector machines*, 2001, available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm.

[12] C. W. Hsu and C. J. Lin, “A comparison of methods for multi-class support vector machines,” *IEEE Transactions on Neural Networks*, vol. 13, pp. 415–425, 2002.