Fast Traffic Sign Detection on greyscale images

Xavier Baró[+] and Jordi Vitrià[++]
Centre de Visió per Computador[+] & Dept. Informàtica UAB[++]
Edifici O - Campus UAB, 08193 Bellaterra, Barcelona, Catalonia, Spain
{xbaro,jordi}@cvc.uab.es

Abstract. This paper describes a traffic sign detection framework for greyscale images. The system is a heterogeneous cascade classifier formed by a rectangle features cascade followed by specific filters for each shape. We define two types of filters: The first one is based on local changes of the gradient direction and the second one is based on the idea of radial symmetry and gives us the centre of circular shapes. The filters use the position and the aspect ratio of the detected features to differentiate between real objects and false alarms of the rectangle cascade. At the end we use the PCA to eliminate the remaining false alarms.

Keywords: Mobile mapping, boosting, traffic sign detection, features extraction, radial symmetry

1. Introduction

The problem of traffic sign detection is studied for several purposes. Most of the work related with that problem is applied to autonomous driving under certain simplified conditions (e.g. [2], [3]), or in first instance to the assisted driving. Sign detection allows warning the driver for inappropriate actions and potentially dangerous situations. In this paper our work is focused on a different purpose: mobile mapping.

This paper explains the steps followed to detect traffic signs in images obtained by the ICC[1] Geomòbil. The images are in greyscale, therefore our methods do not use colour information. In section 2 of the paper we explain the concept of mobile mapping and the main features of the Geomòbil. In section 3 we explain the general structure of the detector and the different methods used. Section 4 explains the implementation of the test system and its results in terms of time and success. Finally, in section 5 there are the points we are trying to improve, conclusions extracted from our present work and the future work.

2. Mobile mapping and Geomòbil

Mobile mapping is the technique used to compile cartographic information from a mobile vehicle. The ICC is developing its own system, named Geomòbil, which incorporates inside a van all the sensors required for the capture of stereo-pairs of digital images and their subsequent georeferencing for the extraction of information. (Figure 1)

Geomòbil includes an image capture subsystem based on a pair of digital cameras of 1020 x 1024 pixels, a direct image orientation subsystem based on GPS/INS[2] and a synchronisation subsystem.

The cameras are calibrated to determine the GPS/INS orientation misalignment and to correct the errors due to the distortion of the optics of the digital cameras. This process is done by ICC's people [1] and is out of the scope of this paper.

---

[1] ICC (Institut Cartogràfic de Catalunya) www.icc.es
[2] Global Positioning System / Inertial Navigation Systems

Figure 1. Geomòbil

3. Detection

In order to detect traffic signs in Geomòbil images, the first approach was to adapt the object detection system proposed by Viola and Jones [4] and generalise it to detect traffic signs instead of faces. Viola and Jones describe an object detection system based on a classifier cascade. Using the integral image they obtain a real-time detection system. To train the cascade they use a powerful machine learning technique: AdaBoost. The problem of this method is that it needs to train many stages to have a reasonable false alarm ratio, but from a certain stage it begins to specialise on the training set, and then fail the test examples. The adopted solution has been to combine a cascade with fewer stages with other methods that eliminate the false alarms. The result is a heterogeneous classifier cascade (Figure 2). The input image goes directly to the rectangle features cascade, where each stage removes some negative examples, and positive examples pass all the stages. After the cascade, we use different kinds of features for each type of sign. Basically, we use the changes in gradient orientation for triangular signs and radial symmetry for circular signs.

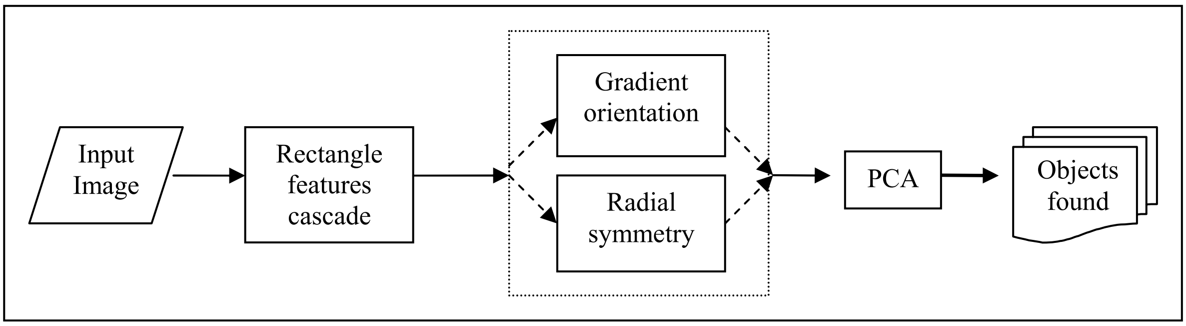

Figure 2. Detector structure

3.1. Rectangle features cascade

This cascade uses combinations of rectangle features to classify every possible subwindow in an image. Using that features instead of directly using the pixel values allows us to encode ad-hoc domain knowledge that is difficult to learn using finite amounts of training data. In addition, feature-based system operates much faster than pixel-based system.

Using the concept of integral image ($ii(x,y)=\sum I(x',y')$), we can calculate the rectangle features very fast, allowing real-time processing. The main idea is to eliminate as many windows as possible without losing the traffic signs.

To find a better representation of the objects in terms of rectangle features, we will use an extended set of features described in [5] (Figure 3). The rotated features also need to calculate the rotated integral image, but it seems that detecting triangular signs can be better. In order to use the representation in terms of rectangular features we have to learn the size and position of each feature and the weights assigned to them. This step is done by using an AdaBoost [6], which gives us a reduced set of features that can represent the objects. In our case, each stage of the cascade reduces the false alarm rate in a 50%, losing less than 0.05% of the positive examples.

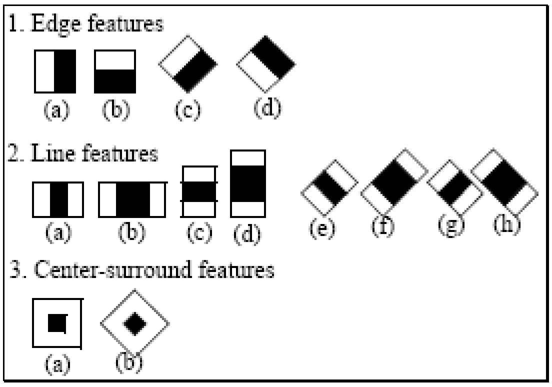

Figure 3. Set of Haar-like features. The black area has a negative weight and the white area a positive weight [5]

3.1.1. Boosting and Cascade of Classifiers

Boosting is a powerful learning concept that allows combining the performance of many simple classification functions to produce a strong classifier [6]. The learning algorithm of simple functions is called weak learner, because we do not even expect the best classification function to classify the training data well [10].

After each round of learning, the examples are re-weighted in order to emphasize those which were incorrectly classified by the previous weak classifier. The final strong classifier is a weighted combination of weak classifiers followed by a threshold [11].

Freund and Schapire proved that the training error of the strong classifier approaches zero exponentially in the number of rounds. They also demonstrate that boosting is especially effective at increasing the margins of the training, defining the margin of an example as the difference between the number of correct votes and the maximum number of votes received by any incorrect label [12].

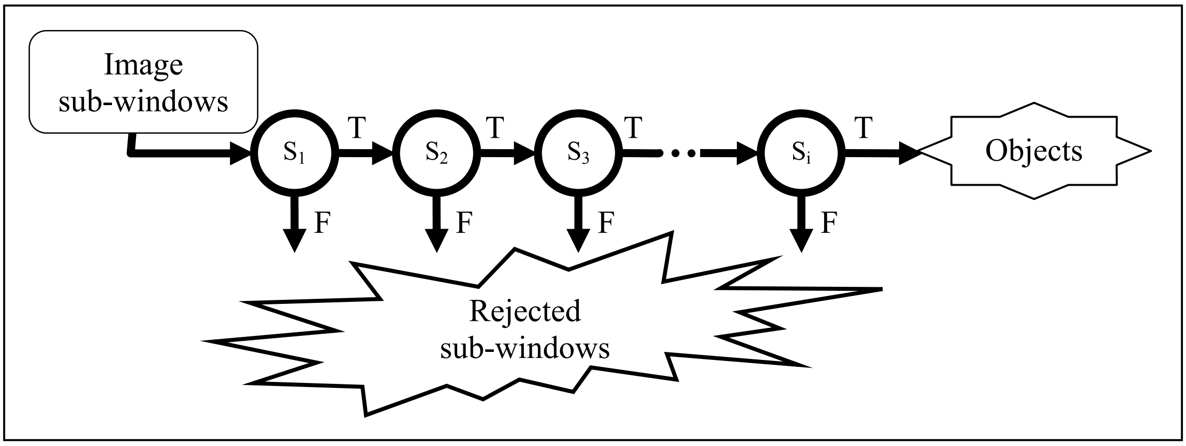

Figure 4. A cascade of classification is a degenerated decision tree. Any given sub-window will progress down through the cascade.

In the particular case of cascade classifiers (Figure 4), we do not have enough with a low error; we need a particular distribution of that error. We are looking for a classifier with a high detection rates and minimum false positive rates. That limitation makes necessary to use an external algorithm to decide the moment to finish the learning of the current stage (Figure 5).

Each stage of the cascade is trained by AdaBoost, increasing the number of weak classifiers until the target detection and false alarm rates are met for the stage. If the training set is easy to learn, just a few weak classifiers will be used, while if the training set is hard to learn or we are training an advanced level the number of weak classifiers will be considerable. In our case each weak classifier is a combination of a rectangle feature with a threshold and a polarity value.

- User selects values for f, the maximum acceptable false positive rate per layer and d, the minimum acceptable detection rate per layer.
- User selects target overall false positive rate, F_{target} .
- P = set of positive examples
- N = set of negative examples
- $F_0 = 1.0$; $D_0 = 1.0$
- $i = 0$
- while $F_i > F_{target}$
  - $i \leftarrow i + 1$
  - $n_i = 0$; $F_i = F_{i-1}$
  - while $F_i > f \times F_{i-1}$
    - $n_i \leftarrow n_i + 1$
    - Use P and N to train a classifier with n_i features using AdaBoost
    - Evaluate current cascaded classifier on validation set to determine F_i and D_i .
    - Decrease threshold for the ith classifier until the current cascaded classifier has a detection rate of at least $d \times D_{i-1}$ (this also affects F_i)
  - $N \leftarrow \varnothing$
  - If $F_i > F_{target}$ then evaluate the current cascaded detector on the set of non-sign images and put any false detections into the set N

Figure 5. Training algorithm for building a cascade detector. [10]

3.2. Radial symmetry

Circular signs are difficult to describe using rectangle features; therefore the cascade begins to lose positive examples at early stages. The number of false positive objects will be considerable if we want a high hit ratio at the end of the cascade. In that case, we will need a fast method to reduce the false alarm.

The fast radial symmetry method [7] uses local radial symmetry to extract points of interest within the scene. The problem we found by using that method was that it gives different points depending on the relation between the background and object intensity (Figure 6).

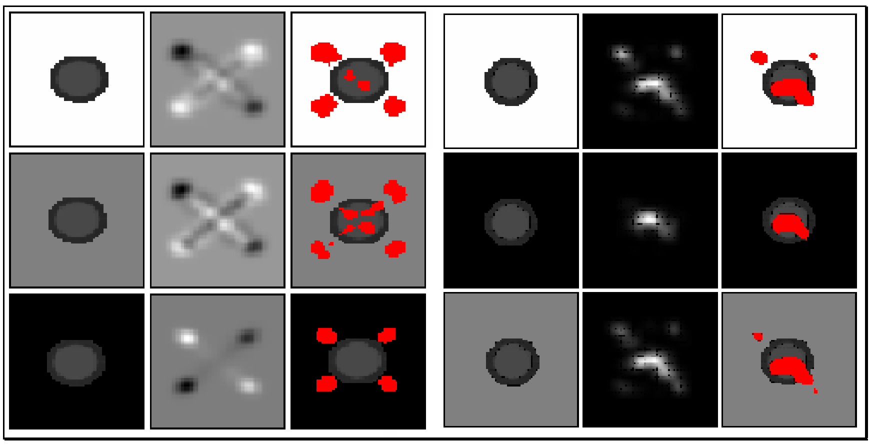

Figure 6. The same sign with different background colour gives different interesting points. Image shows the input image, the accumulate image and the output. *a)* Radial symmetry without convergence criteria. *b)* Using the convergence criteria we always find the centre.

Basically, when we have a prohibition sign with the darkest background the method works fine, because the points of the outer circle and of the inner circle accumulate on the same points, but in the reverse case, the inner points accumulate on the decreased points by the outer circle. We are looking for a method that always gives the same result for the same shape. Moreover, we always want to find the centre of the circumference, because it is the only point that we know where it must be and can be used to differentiate between correct objects and false alarms.

To obtain the highest result on the centre independently of the relation between object and background, we will use local information in a neighbourhood of each point. Using the gradient direction of the point and the direction of the consecutive points in the tangential direction, we will find the direction to which the gradients converge. Now no points are decreased, and all the points in the circle increase the centre (Figure 7).If the gradients do not converge we will ignore the point and will go to the next one. We will increase the point in a certain radius on the convergence direction.

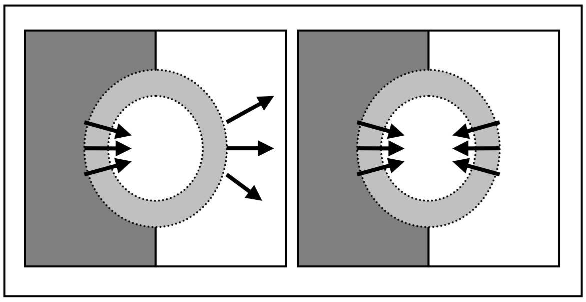

Figure 7. Draft illustrating the gradient direction differences without and with the use of convergence criteria.

3.3. Gradient orientation

Except the circular signs, the rest of the traffic signs are relatively long-sided shapes compared with the size of the edges (triangles, rectangles and squares). In all the sides the gradient direction is constant and on the edges it changes of well-known form. Using those changes we will find the edges of the sign.

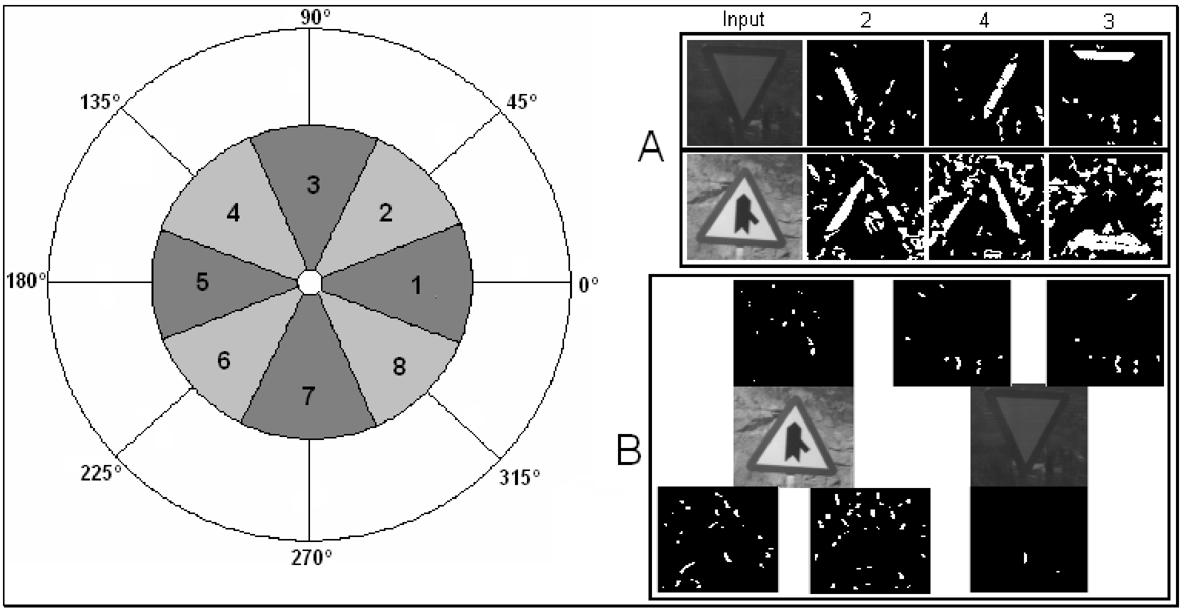

Figure 8. On the left side, the groups of gradient orientation. *A:* Points in the input image with orientation 2, 4 and 3 respectively. *B:* Candidate points to be each vertex of the triangle.

The first step is to calculate the orientation of the gradient in each point of the image. To avoid the changes produced by the noise, we will classify all points in 8 groups, as it can be seen on the Figure 8. Then we will look for changes on the value of the gradient orientation image.

This method uses the fact that the inner border of the sign has always the same gradient orientation because the red part always appears darker than the white part. The outer border will sometimes have the correct direction and sometimes the opposite one, but we can join the opposite groups in some cases if the shape we are looking for does not have both directions.

Once we have the edges image, we use geometrical relations between the different types of edges to find our shapes. In the case of triangular shapes, we will use a rigid template with a zone of uncertainty on the edges (Figure 9). Firstly, we look for the isolated edge and then increasing the value of *R,* we will look for the other two edges in a neighbourhood of radium *r*.

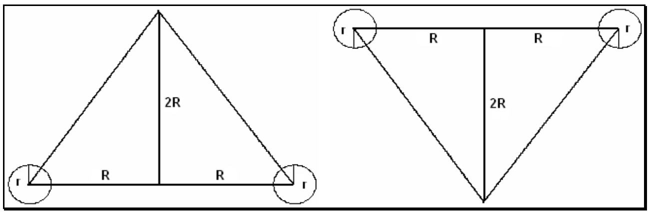

Figure 9. Template used to find the edges.

3.4. PCA

At the end on the cascade we will eliminate the remaining false alarms with the PCA[3]. That method uses global information instead of the local information used in the rest of the stages. The false alarms that arrive to the PCA are difficult to eliminate, because they normally have a shape similar to the sign. It usually happens when there are trees near the road, because they have a texture that can give response to radial symmetry and gradient orientation methods. There are a lot of objects around the road that can have similar shapes with the traffic signs (Figure 10).

A PCA with few components cannot reconstruct the texture; therefore the error between the original image and the reconstructed image is higher than the case where we have a sign. To avoid errors resizing the object window, we use a distance measure that depends on the scale.

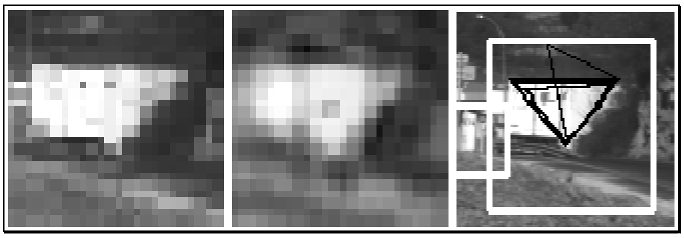

Figure 10. Example of a false alarm when a house near the road appears with a triangular shape. The left image is the input to the PCA. The central image is the retro projection using the PCA. On the right image there is the square zone given by rectangle features (white) and the triangles found using gradient orientation (black).

[3] Principal Components Analysis

4. Results

The test system is written in C++, using the OpenCV[4] library and an OpenMP[5] compatible compiler that allows us to take advantage of Multiprocessor computers and Pentium processors with Hyper-Threading technology. Due to the fact that we have different kinds of signs independent one of each other and that some steps are not interrelated, we can parallelize a lot of steps in the detection process. In a Pentium 4 3.2GHz HT with 512Mb RAM, the complete detection in a frame of 1020x1024 is carried out in less than a second. This time is directly related to the accuracy of the rectangle features cascade. If the cascade gives a lot of objects and we want a good hit ratio, we need to search a lot of features in the gradient orientation step. In the case of circular signs the time is shorter, but the detection speed is given by the slower case.

At this moment we are taking a subset of the whole set of possible signs. That subset is formed by three classes, where each class has a rectangle features cascade and a PCA trained specifically for the group. The rectangle features cascades are trained with 7000 positive examples and 12000 negative examples, and the number of steps is chosen in each case considering the evolution of hit ratio and false alarm. The PCA is trained using 5000 positive examples and the number of vectors is selected to have at least 98% of hit ratio. We need a false alarm rate of less than 0.00001 to obtain a reasonable performance.

Figure 11. Output of the system

5. Conclusions and future work

The main conclusion we extract from the results obtained from that first tests is that there not exist a perfect method to solve the problem. We are convinced that the way to solve that problem is find a good combination of simple methods instead of a try to develop a complex one.

Imminent future work on the detection process will be basically focused on making better the first part of the cascade to avoid losing signs, and to extend it to the rest of the signs. The main problem is to choose a compromised solution between the speed and the accuracy. If we make the test in all positions and all scales the speed decreases a lot, but if we scroll the window with larger steps, it looses some signs. To solve that problem we are going to consider several ways:

- Train the cascade with samples translated some pixels in each direction from the centre of the window. In theory if the cascade can learn that new set of samples, it will be less sensible to the scrolling window steps.
- Use improved versions of AdaBoost that give us better results.[9]

We also want to try other kinds of stages for the cascade. At this moment on this direction we are testing some processes to the input images to reduce the places where we

[4] Intel Open Source Computer Vision Library. http://www.intel.com/research/mrl/research/opencv/

[5] Specification for a set of compiler directives, library routines, and environment variables that can be used to specify shared memory parallelism in Fortran and C/C++ programs.

apply the cascade, or more sophisticated image comparison methods like the chamfer distance [8].

In our case we have additional information that we do not use at the moment, like the fact that we are working with stereo-pairs images, and it gives redundant information that we can use. For example, when we find an object on the left camera, we know the region where it must be on the right camera.

The next logical step on the process is the classification of the signs. At the end of the cascade we have a first classification of the signs, but it is necessary to identify the exact kind of sign.

Acknowledgements

This work has been partially supported by MCYT grant TIC2003-00654, Spain. This has been developed in a project in collaboration with the “Institut Cartogràfic de Catalunya” under the supervision of Maria Pla.

References

[1]. R. Alamús, A. Baron, E. Bosch, J. Casacuberta, J. Mirarda, M. Pla, S. Sànchez, A. Serra, J. Talaya. “On the Accuray and Performance of the Geomobil System”. *International Society for Photogrammetry and Remote Sensing (ISPRS ’04)*. Istambul July 2004.

[2]. U. Handman et al. “An image processing system for driver assistance”. *In Proc. of Intelligent Vehicles Conference*, pages 481-486, Stuttgart, Germany, 1998.

[3]. U. Franke, D. Gavrila, S.Görzig, F.Lindner, F. Pätzhold, and C. Wöhler. “Autonomous driving goes downtown”. *IEEE Intelligent Systems* vol.13, nr.6, pages 40-48, 1998.

[4]. P. Viola, J. Jones. “Robust Real-Time Face Detection”. *Technical Report 2001/01, Compaq CRL*, February 2001.

[5]. R. Lienhart and J. Maydt. “An extended set of Haar-like features for rapid object detection”. *In Proc. of the IEEE Conf. on Image Processing (ICIP '02)*, pages 155 - 162, USA, 2002.

[6]. Y. Freund, R. E. Schapire. “Experiments with a New Boosting Algorithm”. *In Machine Learning: Proceedings of the thirteenth International Conference*, 1996

[7]. G. Loy and A. Zelinsky. “Fast Radial Symmetry for Detecting Points of Interest”. *IEEE Transactions on pattern analysis and machine intelligence*. vol. 25, nr. 8, August 2003.

[8]. D.M. Gavrila, V.Philomin. “Real-Time object detection for “Smart” Vehicles”. *In Proc. of IEEE International Conference on computer vision*, pages 87-93, Kerkyra, 1999

[9]. Y. Freund. “An adaptive version of the boost by majority algorithm”. *Proceedings of the 12th Annual Conference on Computational Learning Theory*, 1999

[10]. P. Viola and M. J. Jones. “Robust Real-Time face detection”. *International Journal of Computer Vision* 57(2), pages 137–154, 2004

[11]. Y. Freund and R.E. Schapire. “A Short Introduction to Boosting”. *Journal of Japanese Society for Artificial Intelligence* 14(5), pages 771-780, September, 1999

[12]. R.E. Shapire, Y. Freund, P. Bartlett and Wee Sun Lee. “Boosting the margin: A new explanation for the effectiveness of voting methods.” In *Proceedings of the fourteenth International Conference on Machine Learning*, 1997.