Traffic Signal Detector Report

Martí Cobos*, Jonatan Poveda*, *Student Member, IEEE,* Ferran Pérez*, *Member, GPI UPC,*
and Joan Francesc Serracant†,
*Universitat Politècnica de Catalunya, Barcelona, 08034, Spain
{marti.cobos, jonatan.poveda, ferran.perez}.@upc.edu
†Universitat Autònoma de Barcelona, Bellaterra, Barcelona, 08193, Spain
joanfrancesc.serracant@e-campus.uab.cat

Computer Vision Center

***Abstract*—This document summarizes the advances our team has achieved during the *Introduction to Human and Computer Vision* module inside the *Master in Computer Vision Barcelona* regarding the Traffic Sign Detection project. Included in this document you will find the algorithms developed, the evaluation compared with other teams, conclusions and further improvements. Each processing block builds on the previous ones and tries to compensate their weaknesses and improve the performance of the overall detector. We start by using a simple colour-based segmentation which is then improved by the addition of morphological filters, geometrical filtering, template matching and alternative segmentation methods. At the end, some experiments are shown, algorithms which were not fine-tuned enough to be integrated in the final processing chain.**

***Index Terms*—Computer vision, Feature extraction, Image classification, Image filtering, Image segmentation, Morphological operations**

I. Introduction

THIS document is a report on the project developed as the practical part of the *Introduction to Human and Computer Vision* module from *Master in Computer Vision Barcelona*. The goal of the project is to apply some of the classical methods and techniques introduced in the course in order to develop a traffic signal detection system.

The project was developed entirely in Matlab and a dataset was taken from the *KUL Belgium Traffic Signs Dataset*. The dataset contains pictures taken from cars while driving containing several types of signals (see Fig. 1). The dataset is split in two subsets: one for training which includes ground truth information, and the other one for testing. The first subset was split into two, training and validation splits, with a share of 70% and 30%. This split was not done completely random: the dataset was analysed and all the statistics extracted were kept as much similar as possible on each split. For instance, the same ratio of signal types appears into each split. Test dataset has been used week after week to compare results of all the participating teams in the master.

In the following sections, a list of methods and techniques used is introduced, together with some parameters found to fine tune them. Then, the results of our best solution are exposed and compared to the best results from other teams. Finally, a conclusions section will wrap-up our learnings during the project and provide some hints for possible improvements. All our code is published on (Github) under a MIT License.

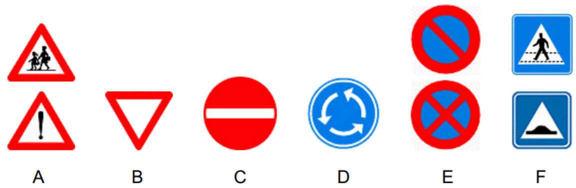


Fig. 1. Signal types in the dataset.

Also notice that other research has approached this problem before us [13] with a different dataset.

II. Methodology

A. Colour segmentation

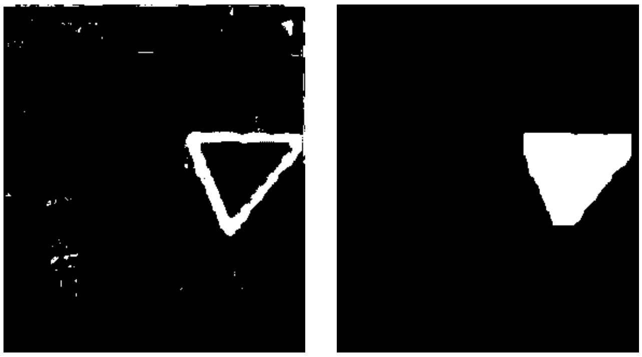
The first step of the traffic signal detector processing chain is the colour-based segmentation which converts an RGB image into a binary mask. For this segmentation, the image is converted to the HSV colour space [12] to separate the luminance information (S) from the colour information (H,V). With this transformation, the segmentation will detect signals inside building shadows (which cannot be detected with high accuracy by segmenting the image on the RGB colour space). It is also more robust to brightness changes because it is integrated in the channel V.

The main objective of this method is to detect red and blue section of the images which are the principal colours of the signals to be detected. White and black colours are also contained on traffic signals, however their detection is not implemented to avoid the false detection of other objects such as cars or the asphalt. Due to not detecting black and white colours, this segmentation can generate hollow signals that will be filled in later stages of the processing chain.

The H and S detection thresholds were hand-picked by inspecting the train image set. Three masks were generated for each image, one for red colour ($350 < H < 20$ and $S > 0.45$), one for a high saturated blue colour ($180 < H < 250$ and $S > 0.4$) and one for low saturated blue colour ($210 < H < 300$, $0.15 < S < 0.4$ and $V < 0.3$). Afterwards these three masks were combined using a binary *OR* operation to obtain the final mask. Fig. 2 shows ans example of the HSV based colour segmentation.

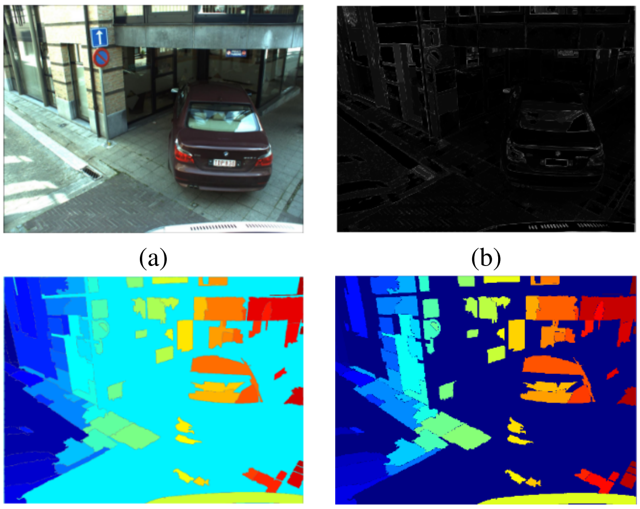
(a) (b)

Fig. 2. Example of usage of colour segmentation to generate a binary mask from a RGB image: (a) initial RGB image, (b) mask result of colour segmentation



(a) (b)

Fig. 4. Example of usage of morphological operations to denoise masks and reconstruct signals: (a) mask result of colour segmentation, (b) image denoised and signal reconstructed with morphological operations



(a) (b) (c) (d)

Fig. 3. Watershed example: (a) original image, (b) Watershed initial markers, (c) regions, (d) refined regions

B. *Alternative segmentation methods*

As an alternative to the former segmentation scheme, other more refined algorithms like *Mean Shift* and *UCM* were tested. Watershed algorithm[7] was finally chosen, as it was already implemented inside Matlab and it worked well enough. Also it allowed us to avoid a large increase in terms of processing time.

A set of initial markers were computed with a series of morphological filters using the gradient of channel V. Afterwards the Watershed regions were obtained, using the four-connectivity definition, and those areas that were larger than the maximum signal area we filtered out. It can be seen in Fig. 3 that the signal region is pretty well defined and if one finds the way to filter out non-signal regions, for instance using shape, colour, texture, or others, it seems that a really well defined regions could be retrieved.

The extracted regions were analysed and those with a percentage of pixels with the desired HSV values above a certain threshold, were considered as a potential signal. This technique yielded very good results when the colour information was reliable but, it performed poorly when this was not the case, like in the HSV segmentation. The initial markers could be improved, by defining them using colour or texture instead of relying solely on the local minima of the gradient.

C. *Morphological operators*

The colour segmentation proposed on Section II-A generated masks with noise and hollowed signals that had to be reconstructed for a correct segmentation, see Fig. 4 as an example. With this in mind, some morphological operations were used.

The hollow signals were reconstructed by applying an image hole filling. A hole is defined as an area of black (zero-valued) pixels surrounded by white (one-valued) pixels. Hole filling finds holes in the masks and changes the values of the pixels inside its contour from zero to one (Fig. 4). In order to reduce noise generated by the segmentation, a morphological opening was applied to the mask. This operation removes white objects smaller than the structuring element. The structuring element was determined by testing several shapes like circular, disk and square on the training dataset. In order to define its size, the granulometry over the whole training dataset was analysed. Finally, a 20 pixels square structuring element was selected for denoising the image with a morphological opening.

D. *Region-based detection*

From here onwards, the pixel-based model is replaced by a region-based model. Two steps were proposed to filter out regions which are not similar to traffic signs, taking advantage of our previous analysis of the training dataset.

1) Connected Component Labelling: with the goal of filtering out false positives, the connected components of the mask obtained after the morphological filtering were analyzed. To do so, Matlab built-in functions *bwconncomp* and *regionprops* were used to label the connected components in the mask and compute their area, bounding box, its centroid and list of pixel indices.

Afterwards, some constraints on the computed properties were defined by using the statistical analysis of the training set to filter out non-signals. In particular, the properties were modeled as gaussian distributions and thresholds to classify

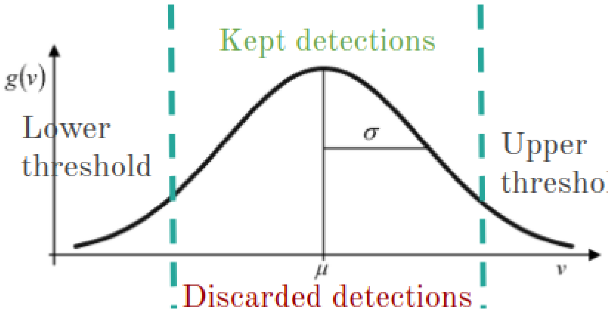


Fig. 5. Gaussian distribution that models the constrained geometrical properties

them were defined. However, the property *Area* was only limited based on the minimum and maximum signal area in train.
This choice came from two ideas. The first idea is that, as it is a binary classification an object can be positive or negative, for an small object, if this object is in fact a traffic signal, it means that the signal is really small or far away, so it is not as important to detect as if it was a big or close signal. The second idea is that, as we make the model of a signal smaller the probabilities for an specific area to look like the model gets higher. If we consider the size limit, it results that a one-pixel area would be classified as a positive. For this reason, it is better to consider a really small object as not a signal. After that, statistical parameters were fine-tuned based on thresholds to obtain the best results in the training set.

2) Sliding window: defines a window of a given size, moves it around the image and processes the pixels under it independently of the rest. In our case, a multi-scale solution based on the *gaussian pyramids* was implemented. The steps followed for this implementation were the following:

1. Generate some layers with *gaussian pyramids*.
2. Find the positive values ('1') of the mask.
3. Use naive *linear search* from the smallest to the largest layer with fixed size window.
4. To reduce the areas to look up, if a region with a traffic sign has been detected, this area is removed from all the layers.
5. Iterate until all layers have been processed.

In the step 4, it should be noticed that each time a positive is detected, the area that is removed from the larger layers is each time larger. Therefore the next iteration it is going to have less positions to analyse, lowering the processing time. In order to lower more the processing time one can consider to stop the iteration before analysing the largest layer, which is the original image. An example of three layers for a given mask in can be seen in Fig. 6.

E. Template matching

Once the window candidates were generated through the process defined in II-D the signal candidate detection had to be refined. Four models representing the traffic signal shape (circular, rectangular, upwards triangle and downwards triangle) were computed as a 40x40 grayscale pixel images. They are in fact a probabilistic model computed using the probability of a being a pixel from a traffic signal using the average of the signal masks in the training dataset. That allowed to fine-tune the model in later steps. A binarization of the final models is shown in figure 7. In order to determine whether a candidate contains a signal or not, two techniques were applied.

Fig. 6. Gaussian pyramid example with three scales

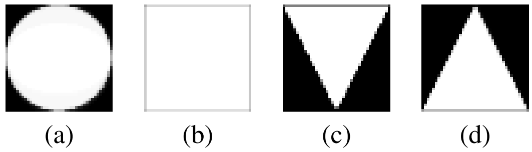


Fig. 7. Models used for template matching. (a) circular mode, (b) rectangular model, (c) downwards pointing triangular model, (d) upwards pointing triangular model

1) Based on correlation and subtraction: for each window candidate, the correlation was computed between the window mask and four models. The models were resized to the window candidate and then the correlation was computed. The decision if a window contains a signal was performed by comparing the correlation result with a certain threshold. If the result was higher than the threshold, the detection was considered positive. The classification thresholds were determined individually for each model using the training dataset. A boxplot of the correlation result were generated for each model (Fig. 8). The thresholds were chosen using the minimum value of the correlation, discarding the possible outliers beforehand.

2) Based on Distance Transform: this transform enables to compute an image with values that indicate the distance to the closest white *blob* (i.e.: object) given an input edge image. This edge image was computed using the *Canny* filter but, it also could be computed using any similar one like *Sobel* filter. Given the connected components computed in the

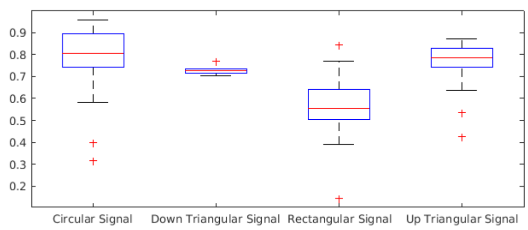


Fig. 8. Boxplot of correlation TM result between signals in train set and models. Left, circular model. Center left, downwards pointing triangular model. Center right, rectangular model. Right, upwards pointing triangular model

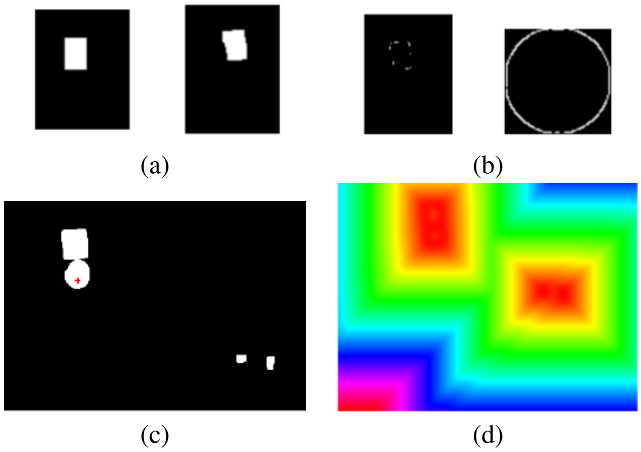


Fig. 9. Distance transform-based template matching correlation: (a) GT and detected CCs, (b) Canny of detected CC and circular template, (c) correlation maximum in original mask, (d) distance transform-based correlation output

Section II-D1, the algorithm implemented uses the following these steps:

1. Compute connected component edges with Canny filter.
2. Compute the Canny filter of the templates generated before (Fig. 7).
3. Compute the distance transform of the canny image from step 1.
4. Compute the normalized correlation between the image of step 3 (global distance transform of input image) and the Canny filtered of all templates (step 2).
5. Locate maximums (object locations) and accept it as signal based on a threshold on the correlation value.

Moreover, the steps before were repeated with several template scales and for each template type. The detection masks were combined with a logical OR of all of them.
Finally, an example of all the process with a circular template tested against an image with a circular and rectangular signal can be ssen in Fig. 9. Notice the correlation maximum location is marked with a red cross on (c).

F. Geometric heuristics

In order to improve the region-based detection (Section II-D), Hough transform [4] in its linear and circular variants were used. For each previously labelled connected component, a bounding box was computed and the representing area from the original image was cropped. Then the edge detection algorithm was applied. Afterwards, the Hough Transform was applied and a list of candidate lines or circles were obtained. Finally, such candidates could be heuristically filtered to met geometric constraints. For triangle signals, Prewitt edge detection [8] algorithm proved to be the best performer. After applying Linear Hough Transform to find candidate lines, vertical lines were discarded and we looked up for three lines to be linked forming a triangle. For square signals, Canny edge detection [3] was used instead. Candidate lines were chosen to be vertical or horizontal, two of each, and linked to form a square (Fig. 10). For circular signals, also Canny edge detection [3] was the chosen algorithm and the provided implementation of the circular Hough transform was used. Each match of geometric heuristics was replaced by its corresponding template mask resized to the size of the initial bounding box.

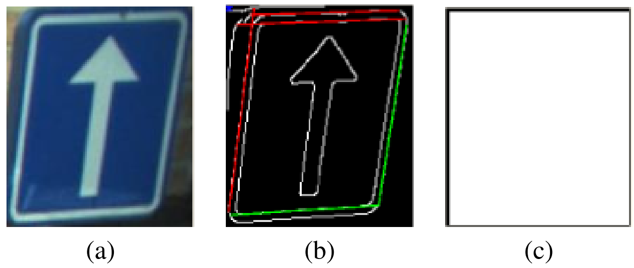


Fig. 10. Example of usage of linear Hough transform to detect square signals: (a) bounding box cropped from the original image, (b) green lines are those candidate from Hough, while red lines are those meeting geometric constraints, on the edge detection image, (c) resized mask of a rectangular signal template.

III. Results

During the project, several combination of the aforementioned methods have been tested. Metrics used to measure the quality of solutions have been:

$$\mathit{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\mathit{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\mathit{F1\ score} = \frac{2TP}{2TP + FP + FN} \tag{3}$$

While some methods, like colour segmentation or morphology, were measured at a pixel basis, other methods, such as connected components, were more naturally measured at a region basis (where ground truth bounding boxes of each traffic signal are compared to bounding boxes candidates).

The best performance has been obtained with the following sequence:

1) HSV colourspace colour segmentation
2) Morphological hole filling
3) Morphological opening filter with 20 pixels square structuring element
4) Connected components labelling with constraints on area, aspect ratio and filling ratio
5) Template matching using correlation

Final results of our best solution compared to other teams bests are shown in Table I and Table II.

IV. Conclusions and future work

This project detailed the development of a complete traffic sign detection system using classical image processing techniques. The results show that the system is very sensitive to the quality of HSV-based segmentation.

For instance, some signs were faded out and the output masks were poor. To overcome this, we used some alternative techniques, like *watershed*, and *mean shift*, but we found

TABLE I
PIXEL-BASED EVALUATION

Method	Precision	Recall	F1
Team 3	0.83	0.73	0.78
Our	**0.89**	**0.68**	**0.77**
Team 2	0.79	0.71	0.75
Team 6	0.71	0.80	0.75
Team 1	0.61	0.72	0.66
Team 5	0.52	0.81	0.63
Team 7	0.41	0.72	0.52

TABLE II
REGION-BASED EVALUATION

Method	Precision	Recall	F1
Team 3	0.76	0.79	0.77
Our	**0.83**	**0.70**	**0.76**
Team 5	0.87	0.65	0.74
Team 2	0.70	0.76	0.72
Team 6	0.63	0.71	0.66
Team 1	0.44	0.59	0.50
Team 7	0.25	0.80	0.38

their optimization requires several parameters to outperform the HSV model. Alternatively, a multiple descriptor could be combined, for example shape and colour descriptors, to generate a more robust mask.

Furthermore, an even better approach would be to compute several descriptors of the traffic signs in the training dataset using different features like shape, colour and texture, to feed a Support Vector Machine binary classifier [1, 2]. This could help to define a much better threshold to separate both classes. However this was out of scope from the very beginning of the project.

Alternatively, and following the state of the art in object detection, Convolutional Neural Network-like architecture (such as R-CNNs[5, 6, 11] or YOLO[9, 10]) could be employed to work directly with the original images if more data were available.

REFERENCES

[1] *The 2013 International Joint Conference on Neural Networks, IJCNN 2013, Dallas, TX, USA, August 4-9, 2013*. IEEE, 2013.

[2] S. Ardianto, C. J. Chen, and H. M. Hang. Real-time traffic sign recognition using color segmentation and svm. In *2017 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 1–5, May 2017.

[3] J Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, June 1986.

[4] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, January 1972.

[5] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[6] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[7] Fernand Meyer. Topographic distance and watershed lines. *Signal Process.*, 38(1):113–125, July 1994.

[8] J.M.S. Prewitt. Object enhancement and extraction. *Picture processing and Psychopictorics*.

[9] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[10] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[11] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[12] Shamik Sural, Gang Qian, and Sakti Pramanik. Segmentation and histogram generation using the hsv color space for image retrieval. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 2, pages II–II. IEEE, 2002.

[13] Gabriel Villalón-Sepúlveda, Miguel Torres-Torriti, and Marco Flores-Calero. Traffic sign detection system for locating road intersections and roundabouts: The chilean case. In *Sensors*, 2017.