2004 IEEE Intelligent Vehicles Symposium
University of Parma
Parma, Italy • June 14-17, 2004

Multi-Resolution Vehicle Detection using Artificial Vision

Alberto Broggi, Pietro Cerri
Dipartimento di Ingegneria dell'Informazione
Università di Parma
Parma, I-43100, Italy
{broggi, cerri}@ce.unipr.it

Pier Claudio Antonello
Surround Sensing - Vehicle development
Centro Ricerche FIAT
Orbassano (TO), I-10043, Italy
pierclaudio.antonello@crf.it

Abstract

This paper describes a vehicle detection system using a single camera.

It is based on the search for areas with a high vertical symmetry in multi-resolution images; symmetry is computed using different sized boxes centered on all the columns of the interest areas.

All the columns with high symmetry are analyzed to get the width of detected objects. Horizontal edges are examined to find the base of the vehicle in the individuated area. The aim is to find horizontal lines located below an area with sufficient amount of edges. The algorithm deletes all the bounding boxes which are too large, too small, or too far from the camera in order to decrease the number of false positives.

All the results found in different interest areas are mixed together and the overlapping bounding boxes are localized and managed in order to delete false positives.

The algorithm analyzes images on a frame by frame basis, without any temporal correlation.

I. Introduction

Vehicle detection is a central problem in intelligent vehicles. A lot of research teams work on this problem; most of the solutions use a stereo system in order to reconstruct a 3D approximation of the framed scene [1] [2] [3] [4] [5] [6]. A 3D reconstruction is also possible with a monocular system; in fact the use of two subsequent frames can create a sort of stereo system, but this process needs a very precise pitch and yaw measurement [7]. All these methods provide a good precision in vehicle distance detection, incidentally the use of a single camera together with a precise calibration provides a satisfactory precision.

Some methods use models to identify a vehicle; many different models have been used, ranging from trivial models to complex ones, like deformable models that add details of the car approaching the camera [8], or 3D models that take into consideration vehicle misalignment with the camera. [9]. All these methods need models that match different vehicle types.

The authors gratefully acknowledge the grant provided by ATA as a support to this research.

The search for vehicle features provides a simplified way of localizing vehicles. For example symmetry is a characteristic that is common to most vehicles. Some groups have already used symmetry to localize vehicle [11] [12], they have tried out a lot of methods to find symmetry on images: using edges, pixel intensity, and other features. It has been chosen to use symmetry to search for vehicles, improving a previous version of an algorithm described in [10]. The limit of the old method is to be based on detection of a single vehicle, only in predefined area of the image in front of the camera. The goal is to enlarge the search area and to detect all vehicles present in the image. The algorithm is as simple as fast nonetheless it reaches good results.

In the next section the problem and the vehicle setup are explained, then in section III a description of the approach and the algorithm that has been developed can be found. In section IV the results are discussed and some final considerations exposed.

II. Problem and Setup

The goal is to detect all vehicles in the scene: large, small, far and close ones. Unfortunately a search for vehicles with variable size and distance can be very time consuming. Therefore a specific algorithm together with efficient optimization have been designed. Furthermore since the computation of vehicle distance is also required a precise camera calibration is mandatory. Indeed drifts from the static calibration are often experienced, due to vehicle movements; vehicle pitch adds noise to the distance measurement.

In the current setup the camera is mounted inside the cabin near the rear-view mirror; the camera aperture is 45 degree, the image resolution is 640 x 480 while the working part of the image is 640 x 300 pixels. The vehicle is shown in figure 1.

The test sequences were acquired on exurban roads and highways (fig. 2).

III. Algorithm description

The algorithm is divided into the following parts:

Fig. 1. The test vehicle.

(a) (b)

Fig. 2. Input images: (*a*) exurban road (*b*) highway.

A. creation of three areas of interest; on each area the following steps are performed:
   - vertical symmetry computation,
   - interesting columns identification,
   - bounding boxes generation and
   - position and size filtering.

B. Result mixing and real distance and size computation.

A. Areas of interest

Computing the symmetry on the whole image is very time consuming; moreover close vehicles can present too much details posing a significative problem in symmetry computation. Besides this, in order to speed up execution the algorithm considers three different interest areas, the first one for the detection of far vehicles (40-70 mt.), the second one for vehicles at medium distances (25-50 mt.) and the last one for close vehicles (10-30 mt.) (also see [13] for distance-based subsampling). The first area is a cropped version of the original image; the other areas are obtained subsampling the original image, or a part of it, at different steps in order to partially remove details, all areas are reduced to a fixed size (200 x 100 pixel). Figure 3 shows the three interest areas.

The following parts of the algorithm are applied to the three images.

B. Symmetry computation

Symmetry calculation is the main part of the algorithm, and the most time consuming as well. Only binarized edges are used in order to reduce execution time: grey levels symmetry is very time consuming and does not provide more information than edges symmetry. First of all the Sobel operator is used to find edges module and

Fig. 3. Interest areas for the detection of: far (left), medium distances (center), close (right) vehicles.

orientation, then three images are built, one with binarized Sobel modules, one with all the almost-vertical edges and the last with all the almost-horizontal edges. The idea is to find all the vertical and horizontal edges, even the weakest. Therefore a very low threshold is used on edges modules; vertical edges are labeled considering their orientation.

The symmetry is computed for every column of the three images, on different sized bounding boxes whose height matches the image height and with a variable width ranging from 1 to a predetermined maximum value. The computed value is saved in a 2D data structure (hereinafter referred to as an image) whose coordinates are determined as follows: the column is the same as the symmetry axis and the row depends on the considered bounding box width (fig. 4).

(a) (b)

Fig. 4. Symmetry calculation: (*a*) for a single column width and (*b*) for the largest width.

Symmetry is calculated as

$$simm = \frac{s^2}{n}$$

where s is the number of the symmetric points, and n is the number of all white pixels; this operation is used on the three images of the edges (binarized Sobel, vertical edges, horizontal edges); two vertical edges are considered symmetric if their orientation is opposite. Since this operation has to be executed for a large number of times (about 400000 times for every area) it must be as simple and quick as possible. A new image is created with a pixelwise *AND* of the horizontal and vertical edges symmetry images. This image is used to search for interesting columns.

C. Interesting columns

An interesting column is defined as having a high symmetry in: (*i*) the image that contains the result of Sobel binarization or in, (*ii*) the image that contains the AND between symmetry of horizontal and vertical edges. A columnwise histogram is then used to locate candidate columns. In correspondence to these columns the vertical edges symmetry is checked to obtain the expected vehicle width. More specifically if a high value of symmetry is present for small widths too, it means that the algorithm has detected a small object; in this case the column is discarded. Figure 5 shows an example: the leftmost peak is discarded because it presents a high symmetry value also for small widths, on the other hand the rightmost peak presents an appreciable symmetry value only for widths above a certain size.

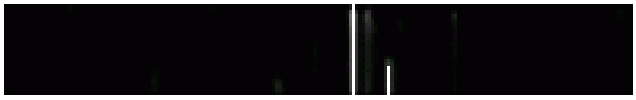


Fig. 5. Vertical edges symmetry: the horizontal axis represents the position of symmetry axis while the vertical axis represents the width of the symmetry box (small on the top and large on the bottom of the image).

The Sobel binarized symmetry of the discarded columns are further checked: if the histogram is greater than a very high threshold the column is considered valid, otherwise it is definitely discarded; this step is used to locate vehicles that do not present perfect vertical edges, for example vehicles misaligned with the camera (fig. 6).

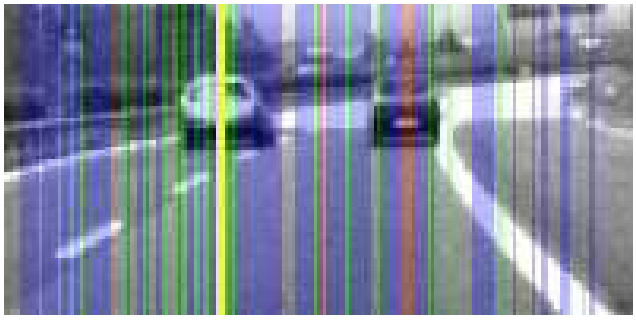


Fig. 6. Columns valuation: only red and yellow columns are processed.

D. Bounding Boxes generation

Up to now the algorithm gives information about the vehicle's center position only but since vehicle width is needed as well, a precise bounding box detection is mandatory. For each peak in the vertical edges symmetry image that survived the previous filterings the width of the box is given by the distance between the peak and the top of the symmetry image; the box is centered in the column (fig. 7). Otherwise -in case this is not possible- a columnwise histogram of the number of edges is considered to detect the box width (fig. 8).

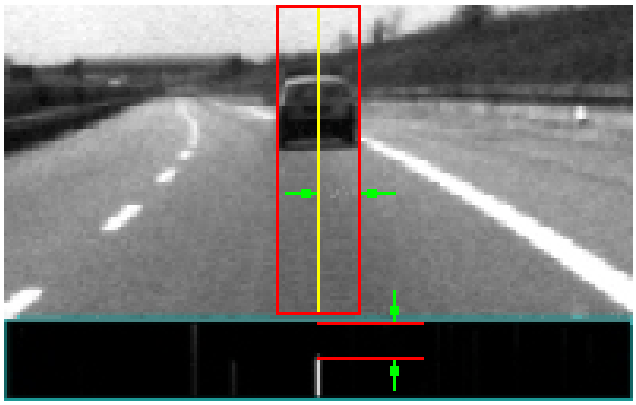


Fig. 7. Box width computed from peak height.

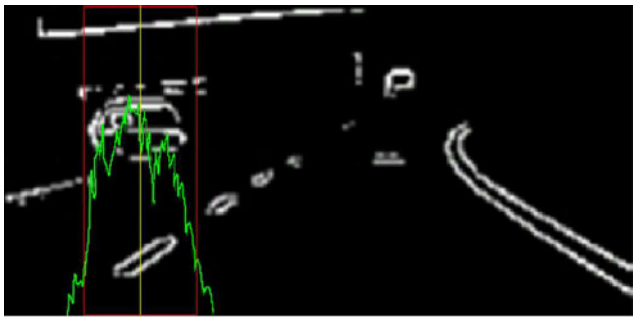


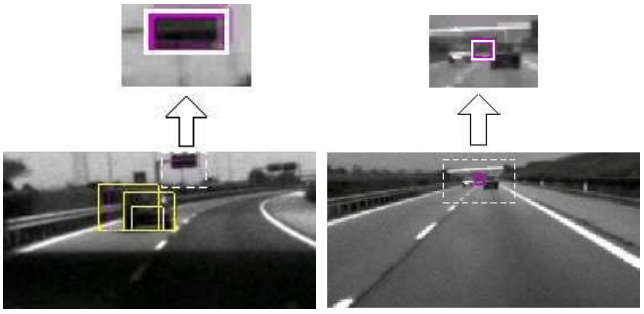
Fig. 8. Histogram of edges and relative box width.

The shadow under the car is searched for in order to find the box base. It is defined as a horizontal edge, but since other shadows, like bridges' ones, could be present on the road as well, and the algorithm looks for a high concentration of edges above the horizontal edge; if no base can be detected the column is discarded. The search for vehicle roof is not performed and a rectangle with aspect ratio equal to $\frac{4}{3}$ is displayed.

E. Filters

Unfortunately not all the detected boxes are correct: some false positives caused by road signs or other objects in the scene can be present as well. Two filters are used in order to discard some false positives; the former analyzes the box position and deletes the boxes too far from the camera taking into account perspective constraints, namely with the base too high in the image. The other one removes boxes too large or too small that probably do not identify a vehicle (fig. 9).

It is also possible that a car is detected in more than one column, so overlapping boxes may be present. Only one box per vehicle is expected, so a further step is required to merge similar boxes and eliminate redundant ones: the largest box is preferred. Furthermore when two rectangles with similar size have their base at the same height an average is computed, and this new box is considered.



(a) (b)

Fig. 9. Filters: (*a*) position and (*b*) size. Bottom row: original images, top row:enlargement of details showing the filtered boxes.

F. Results mixing

Three different lists of vehicles are produced, one for far vehicles, one for medium distance vehicles, and one for close vehicles. All the identified vehicles must be fused in a single list that contains all the vehicles in the image. The three lists are checked and the overlapping boxes removed: filters based on size (similar to the one described in III-E) are applied to bounding boxes coming from different lists.

Distance and size of vehicles are computed thanks to image calibration; all the bounding boxes with unreal size are finally eliminated.

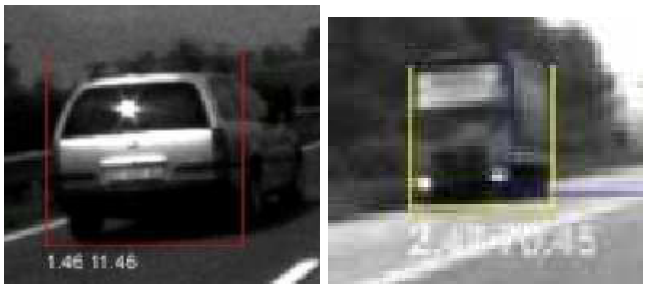
IV. Discussion of results and conclusions

The algorithm detects most of the vehicles in the provided test sequences, with a reduced number of false positives. Vehicles that proceed just in front of the camera are detected with a good precision (fig. 10.a), and also oblique ones can be identified (fig. 10.b). Trucks are detected as well since their square shape matches the algorithm assumptions (fig. 10.c). Also vehicles coming on the opposite lane can be detected (fig. 10.d).

Unfortunately sometimes the vehicle size can not be identified with a sufficiently high precision because of the simple histogram method. This basically happens with misaligned vehicles especially (fig. 11.c). Even if filters to remove false positives have been introduced, there are cases in which ghost vehicles are detected. Figure 11.a shows an example of false detection induced by the presence of vertical edges above a horizontal shadow line. Sometimes the base of the vehicle is detected with an error because of the presence of a black bumper (fig. 11.b).

Bridges shadows are very difficult to manage because they can generate a lot of false positive and can also interfere in base detection (fig. 11.d). Figure 11.d also shows a good performance of the algorithm in presence of a simple overlapping of two cars: both cars are detected.

The problem of misaligned vehicles (like overtaking vehicles or vehicles coming in the opposite direction) is the most critical aspect of this method. The algorithm does not always detect oblique vehicles, and when it does, vehicle width may not be precise. When vehicles rear vertical edges are clear on both sides the algorithm works fine and detects the vehicles properly (fig. 12.a), otherwise when other edges are present, vehicles width may be incorrect (fig. 12.b).



(a) (b)

Fig. 12. Misaligned vehicles: (*a*) correct detection, (*b*) width error.

The algorithm works efficiently on simple images. Problems arise when a large number of vehicles overlap or in urban scenes, where road infrastructures, road signs and shadows make the scene too complex.

Execution time is 120 ms with an Athlon XP 1.8 GHz; since 75% of the time is used for symmetry computation, the next research step will be the optimization of this part, moreover, a quantitative performance measurement is being carried out for comparison with other approaches.

References

[1] U. Franke, "Real-Time Stereo Vision for Urban Traffic Scene Understanding," in *Procs. IEEE Intelligent Vehicles Symposium 2000*, (Detroit, USA), pp. 273–278, Oct. 2000.

[2] H. Li, S. Bouaziz, and F. Devos, "An Embedded System for Autonomous Collision Avoidance and Line Tracking Using Artificial CMOS Retina Sensor," in *Procs. IEEE Intelligent Vehicles Symposium 2000*, (Detroit, USA), pp. 273–278, Oct. 2000.

[3] S. Heinrich, "Fast Obstacle Detection using Flow/Depth Constraint," in *Procs. IEEE Intelligent Vehicles Symposium 2002*, (Paris, France), June 2002.

[4] R. Labayarde, D. Aubert, and J. P. Tarel, "Real Time Obstacle Detection in Stereovision on non Flat Road Geometry through 'V-disparity' Representation," in *Procs. IEEE Intelligent Vehicles Symposium 2002*, (Paris, France), June 2002.

[5] C. Knöppel, A. Schanz, and B. Michaelis, "Robust Vehicle Detection at Large Distance Using Low Resolution Cameras," in *Procs. IEEE Intelligent Vehicles Symposium 2000*, (Detroit, USA), pp. 267–272, Oct. 2000.

[6] A. Bensrhair, M. Bertozzi, A. Broggi, A. Fascioli, S. Mousset, and G. Toulminet, "Stereo Vision-based Feature Extraction," in *Procs. IEEE Intelligent Vehicles Symposium 2002*, vol. 2, (Paris, France), pp. 465–470, June 2002.

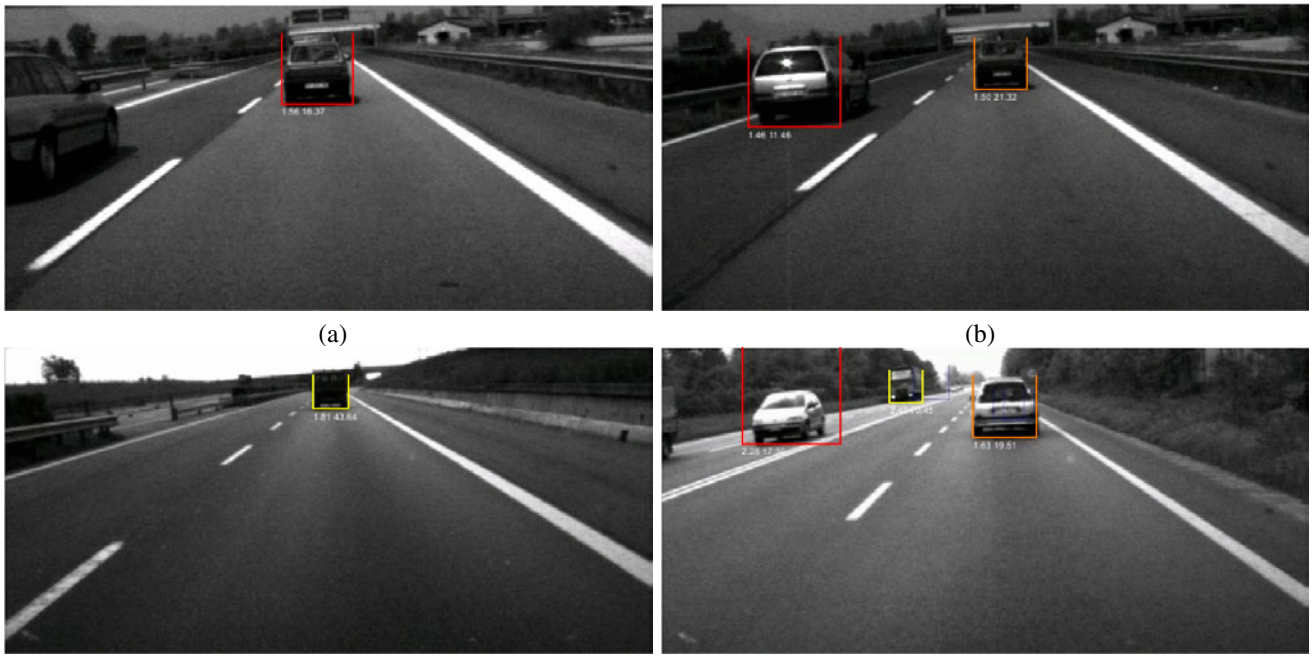
[7] Z. Hu and K. Uchimura, "Tracking Cycle: A New Concept for Simultaneously Tracking of Multiple Moving Objects in a Typical Traffic Scene," in *Procs. IEEE Intelligent Vehicles Symposium 2000*, (Detroit, USA), pp. 233–239, Oct. 2000.

[8] S. Denasi and G. Quaglia, "Obstacle Detection Using a Deformable Model of Vehicles," in *Procs. IEEE Intelligent Vehicles Symposium 2001*, (Tokyo, Japan), pp. i145–150, May 2001.

[9] K. Fleischer, H. H. Nagel, and T. M. Rath, "3D-Model-based-Vision for Innercity Driving Scenes," in *Procs. IEEE Intelligent Vehicles Symposium 2002*, (Paris, France), June 2002.

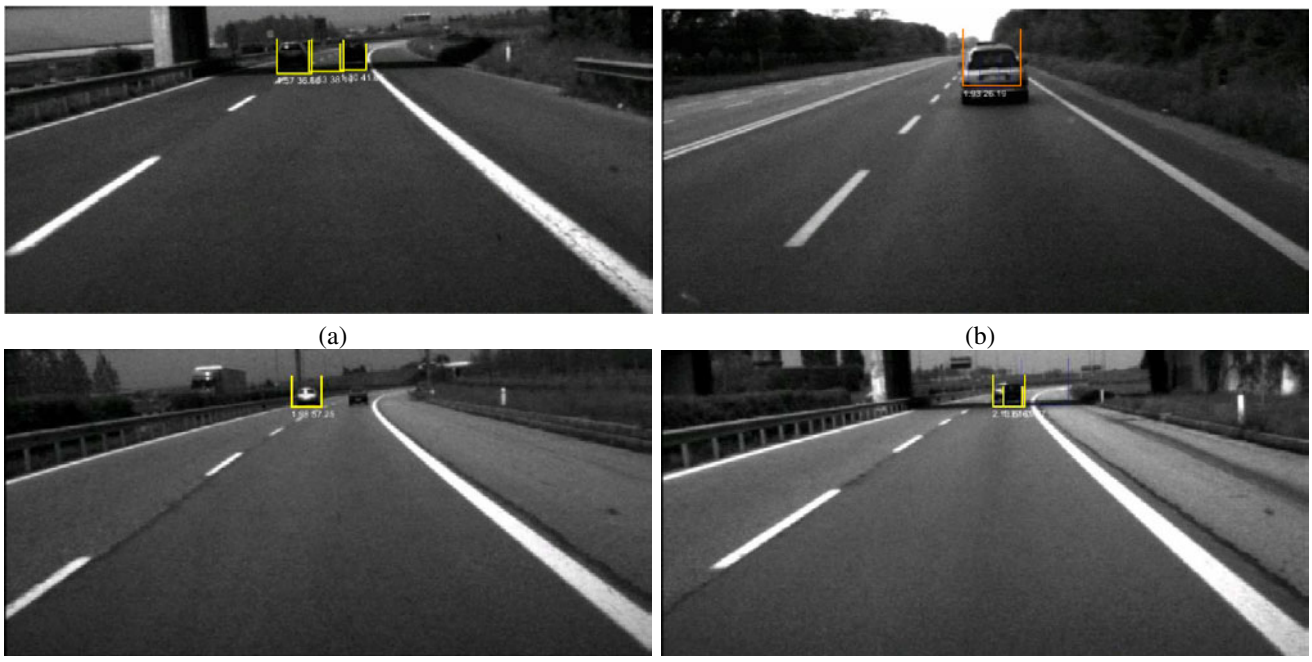
[10] M. Bertozzi, A. Broggi, A. Fascioli, and S. Nichele, "Stereo Vision-based Vehicle Detection," in *Procs. IEEE Intelligent Vehicles Symposium 2000*, (Detroit, USA), pp. 39–44, Oct. 2000.

[11] A. Kuehnle, "Symmetry-based vehicle location for AHS," in *Procs. SPIE - Transportation Sensors and Controls: Collision Avoidance, Traffic Management, and ITS*, vol. 2902, (Orlando, USA), pp. 19–27, Nov. 1998.



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

Fig. 10. Results: (*a*) a car driving in front of the camera (*b*) a misaligned car (*c*) a truck (*d*) vehicles on the opposite lane.



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

Fig. 11. Errors: (*a*) false positive (*b*) wrong base position (*c*) wrong size (*d*) another wrong base position.

[12] T. Zielke, M. Brauckmann, and W. von Seelen, "Intensity and Edge-based Symmetry Detection with an Application to Car-Following," *CVGIP: Image Understanding*, vol. 58, pp. 177–190, 1993.

[13] V. Graefe and W. Efenberger, "A Novel Approach for the Detection of Vehicles on Freeways by Real-time Vision," in *Procs. IEEE Intelligent Vehicles Symposium'96*, (Tokyo, Japan), pp. 363–368, Sept. 1996.