3D Vehicle Sensor based on Monocular Vision

Daniel Ponsa[†], Antonio López[†], Felipe Lumbreras[†], Joan Serrat[†] and Thorsten Graf[‡]

Abstract— **Determining the position of other vehicles on the road is a key information to help driver assistance systems to increase driver's safety. Accordingly, the work presented in this paper addresses the problem of detecting the vehicles in front of our own one and estimating their 3D position by using a single monochrome camera. Rather than using predefined high level image features as symmetry, shadow search, etc., our proposal for the vehicle detection is based on a learning process that determines, from a training set, which are the best features to distinguish vehicles from non–vehicles. To compute 3D information with a single camera a key point consists of knowing the position where the horizon projects onto the image. However, this position can change in every frame and is difficult to determine. In this paper we study the coupling between the perceived horizon and the actual width of vehicles in order to reduce the uncertainty in their estimated 3D position derived from an unknown horizon.**

I. Introduction

Determining the position of other vehicles on the road is an essential task to develop applications like adaptive cruise control and platooning, and it can be a key information to help driver assistance systems to increase driver's safety. The problem can be addressed by using active sensors as radar or lidar as many researchers have done. However, since vision is the most used human sense for driving and vision sensors (CCD/CMOS) are passive and cheaper, another possibility consists of applying computer vision techniques. Accordingly, the work presented in this paper addresses the problem of detecting the vehicles in front of our own one and estimating their 3D position relative to us by using a single monochrome camera placed in the rear–view mirror of our vehicle, now on called *host vehicle*.

There have been proposed different image features to identify vehicles as line structures and shadows [1] and symmetry [2]. These works share the viewpoint of deciding in advance which are the best image features for detecting vehicles and only vehicles. In this paper, however, we propose a different approach: given a training set of image regions showing vehicles and non–vehicles, we propose to apply a learning algorithm in order to select the collection of features that better distinguish between both region types. In this way, after an off–line learning phase we obtain a robust classifier that is used to detect the vehicles present in a single frame (image). Next, for each detected vehicle it is estimated its position with respect to the host vehicle. This last point is not possible using a single camera (no 3D information directly available) unless we put some constraint, that in our case consists of assuming a flat road. Under this assumption, however, the problem is not trivial since the horizon line perceived in the images is continuously changing (there is not a fixed homography from the road plane to the image plane). To minimize this problem we study the coupling between the actual width of vehicles and the possible horizon line perceived at the images. The aim is to consider only feasible pairs in the computation of the 3D information associated to each detected vehicle and, therefore, having more accurate 3D information than by just using a fixed average vehicle width and a fixed horizon line.

As the proposed system does not use temporal coherence, it performs as a sensor whose output is the 3D information of detected vehicles. This information can be used by either a posterior tracking procedure of the own computer vision system or can be fused with the information provided by other sensors (radar or lidar) to increase robustness.

The paper is organized as follows. In Sect. II we introduce our proposal to detect vehicles on images. In Sect. III we explain the algorithm used to derive the 3D information of the detected vehicles. Finally, Sect. IV presents current results and summarizes the main conclusions of the work.

II. Vehicle detection

From a general viewpoint vehicle detection is a problem of object detection, one of the always open issues in computer vision. Special difficulties that make vehicle detection a challenge for object detection methods are:

- *(D1)* since both camera and objects are in movement the perceived size and pose of the objects change, and events like occlusions can happen;
- *(D2)* the objects live in an environment that changes (*f.i.* its illumination and background);
- *(D3)* the actual aspect of vehicles is quite wide (*f.i.* color, size, presence/absence of different elements).

Among the different possibilities we rely on a technique whose main ingredients are *boosting* based learning and response to Haar filters as features to learn [3]. The main reasons for our choice are: 1) in this way we obtain a detection method based on vehicle appearance that can be applied independently to each single frame; 2) the difficulties *(D1)*, *(D2)* and *(D3)* can be addressed in this framework; 3) the detection can be performed in real–time; 4) each detection comes with a degree of confidence; 5) its usefulness has been proven in other difficult contexts as face detection (though probably with less variability than in our case).

This research has been partially funded by Spanish MEC project TRA2004-06702/AUT. We thank to Marc Seguer and Francisco Sánchez at SEAT for their collaboration in the capture of the video sequences.

([†]) Computer Vision Center and Dept. d'Informàtica, Universitat Autònoma de Barcelona. {daniel, antonio}@cvc.uab.es

([‡]) Volkswagen AG Group Research, Electronics.

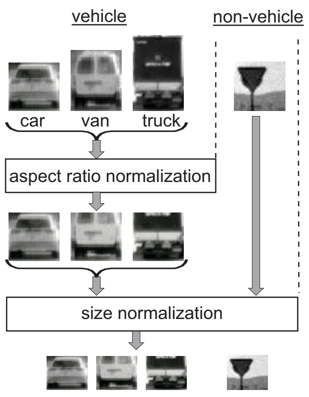

Fig. 1. Normalization procedure.

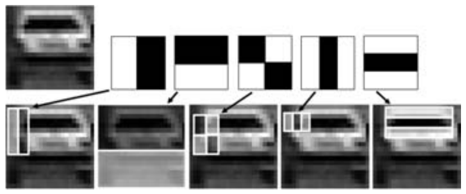

Fig. 2. Top: vehicle–like image region and basic forms of the Haar filters. Bottom: application of filters derived from the basic ones at particular pixels where the filters give a high response (in magnitude).

A. Learning of the vehicle detector

The detector we pursue is such that can classify any image region as vehicle or non–vehicle. Therefore, the learning method must characterize only one class of objects (the other is just the complement). Accordingly, the intuitive idea behind boosting algorithms is that, for the task of classifying, an appropriate combination of several simple classifiers can outperform the results of a unique complex classifier. This is the case of the named Real AdaBoost algorithm [4], the method we have chosen and whose application to our problem is summarized in this section.

We start with a *training set* of n_r image regions labeled as vehicles (examples) and non–vehicles (counter–examples). By *vehicles* we currently refer to cars, vans and trucks. Then, to cope with the variability expressed in *(D1)* and *(D2)* we introduce several steps. First, we only consider as examples the rear and front parts of the vehicles since they are well seen from the host vehicle in most of the situations. Second, we perform aspect ratio and size normalization to compensate the changes in the perceived size due to perspective. Since cars, vans and trucks have very different aspect ratios, the aspect ratio normalization implies that the upper–part of vans and trucks is cut–out while the width is preserved. Size normalization is done by reducing resolution until obtaining a region of 24×18 pixels, which is the dimension of the area that produces the projection of a vehicle at about 70 meters (for our acquisition system). Figure 1 illustrates these steps.

Once we have the normalized regions we must extract features to characterize them. At this moment we follow the approach used in [3] for face detection. Namely, image regions are described by their responses under a family of Haar–like filters which are sensitive to the presence of horizontal and vertical bars as well as symmetric structures. The basic filters are shown in Fig. 2. They are applied at each region pixel for different combinations of sizes and aspect ratios. It has been shown that computing a given feature requires a fixed low computational cost which is independent even of the filter size. It is also worth to mention that the filters are applied in a way equivalent to normalize in advance the image regions by their grey–level variance, which helps to gain invariance under illumination conditions *(D2)*.

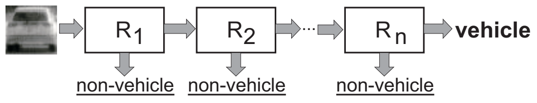

Fig. 3. Cascade of classification rules for vehicle detection.

Hence, each image region of the training set is first normalized and afterwards represented by its response $\mathbf{F} = \{f_1, \ldots, f_{n_f}\}$ under a fixed set of n_f Haar filters, achieving robustness to *(D1)* and *(D2)*. To gain robustness against the inherent variability of the vehicles *(D3)* we must ensure that we have selected a sufficiently representative training set. Now the input for the Real AdaBoost learning consists of the set $\mathscr{F} = \{(\mathbf{F}_1, l_1), \ldots, (\mathbf{F}_{n_r}, l_{n_r})\}$, where $\mathbf{F}_i$ contains the Haar features of the image region i and $l_i \in \{v, nv\}$ indicates if this region is a vehicle or not. AdaBoost output is a classification rule obtained as a linear combination of simpler classification rules. In particular, for each feature $f_j \in \mathbf{F}$ we have a simple rule r_j based on thresholding. The learning of $r_j(\mathbf{F})$ means determining what threshold t_j distinguishes better the vehicles from the non–vehicles in the training set according to f_j, if this threshold must be applied to f_j or to $\|f_j\|$ and if the vehicles are those above the threshold or those below. The learning of the rule is completed by determining the value it returns for each classification decision: for vehicles a positive real value whose magnitude can be understand as a degree of confidence on how well the rule identifies vehicles, and for non–vehicles a negative real value whose magnitude is the confidence of the rule for identifying non–vehicles. The final classification rule is:

$$R(\mathbf{F}) = \sum_{j=1}^{n_f} r_j(\mathbf{F}) \quad , \tag{1}$$

where the sign of R provides the classification decision (positive for vehicles, negative for non–vehicles) and its absolute value the confidence on that decision.

The effectiveness of the learned rule depends strongly on the training set. Therefore, since the class of counter–examples usually have such a large variability that it is very unlike to be well represented from a single training set, the classifier learned is prone to make false positive detections out of the training set. To minimize this problem, after identifying false positives using the learned rule, AdaBoost is run again with the same examples but using only these false positives as counter–examples. Thus, it is learned a new classification rule: now we have a rule R_1 that classifies

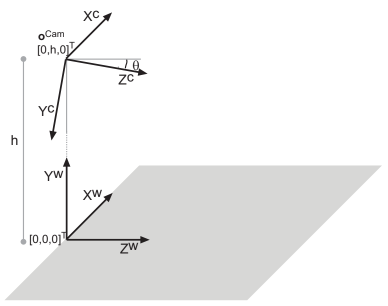

Fig. 4. Camera coordinate system (C) relative to the world coordinate system (W). The latter is supposed to be placed on the road that sustains the host vehicle. The plane $Y^W - Z^W$ is the same than the $Y^C - Z^C$.

the image regions as vehicles or not, and for those classified as vehicles we still have another rule R_2 that must confirm the classification. In this way, the rate of false positives is reduced. This process can be iterated up to construct a cascade of classification rules achieving desired classification ratios (Fig. 3).

B. *Vehicle detector*

The cascade of learned classification rules defines our vehicle detector. Therefore, given an image, first we determine which regions could contain a vehicle, next we compute their corresponding characteristic features and, finally, give them as input to the cascade. Most image regions are rapidly classified as non–vehicles by the first rules of the cascade and only those regions that actually contain vehicles arrive to the end of the cascade. Of course, the most a background region resembles a vehicle the most deep advances in the cascade.

It is interesting to point out that this process is very fast for a given image region thanks to the concept of *integral image* underlying the computation of the features at each region [3]: independently of the region size any feature can be computed by a few additions and subtractions, without performing explicit size normalization of the region. The integral image is computed very fast once for the whole input image.

With regards to real–time, a key point of this vehicle detection method consists of the number of image regions to test. Of course, a blind procedure could consists of considering all possible squared regions (different size and aspect ratio) centered at each image pixel. But this would turn in a very huge number of them. Fortunately, there is information we can use to reduce such number. To develop this point we need first to introduce the geometric model underlying image acquisition.

C. *Geometric model of the image acquisition*

Figure 4 presents the coordinate systems involved in the geometric model, where we remind that our camera is placed in the rear–view mirror of the host vehicle. The relationship between the two coordinate systems implies that the extrinsic parameters of the camera are fully determined by the camera origin $\mathbf{o}^{Cam} = (0,h,0)^T$ and the pitch angle θ describing its inclination with respect to the road surface. The projection of the 3D world onto 2D images (i.e. the camera intrinsic parameters) is modeled by a pin–hole camera of zero–skew [5], which requires the identification of the effective focal length on the image axes, namely (f^x, f^y), and the center of projection $(x^0,y^0)^T$. These values have been obtained by calibrating the camera using the software provided in [6].

With all these considerations, the image coordinates $(x,y)^T$ where a world point $(X,Y,Z)^T$ projects are defined by:

$$x = x^0 + \frac{f^x X}{Z\cos(\theta) + (-h+Y)\sin(\theta)} \ , \tag{2}$$

$$y = y^0 + \frac{f^y\left((h-Y)\cos(\theta) + Z\sin(\theta)\right)}{Z\cos(\theta) + (-h+Y)\sin(\theta)} \ . \tag{3}$$

D. *Feasible image regions to search vehicles*

Now, according to this geometric model and assuming a flat road we can establish some constraints regarding the feasible image regions where to look for vehicles.

First, we determine at which image coordinates can we place the bottom–left corner of an image region candidate to contain a vehicle. If such image region is a bounding–box for the rear/front part of a vehicle then such corner must touch the road. Since points on the road are of the form $(X,0,Z)^T$ and project below the perceived horizon line

$$y^h = y^0 + f^y \tan(\theta) \ , \tag{4}$$

we can safely discard those pixels above the image row y^h.

Second, we know the range of vehicle widths in the world (from cars to trucks) and we consider only a range of aspect ratios mainly defined to fit cars (the upper–part of vans and trucks is not used). These observations provide constraints in the 3D world that can be translated in 2D through (2)–(3). Then, at any pixel used as bottom–left corner for image regions of vehicle searching we have a range of sizes and aspect ratios that delimits the set of regions that can be the bounding–box of the rear/front part of a vehicle.

Finally, it is clear that if we search vehicles in image regions that have a lot of overlapping (*f.i* a region with the bottom–left corner at $(x,y)^T$ and another at $(x+1,y)^T$ of the same size) it is likely to detect the same vehicle in both regions. Therefore, we assume an uniform sampling in the 3D word that translates into a 2D sparse net of image pixels.

All together means that at each pixel of a sparse net below the horizon line we try several feasible candidate regions of different size and aspect ratio. Figure 5 illustrates the process. However, while the above observations allow to reduce the number of regions to explore, a remaining problem adds complexity to the system. We refer to the fact that the extrinsic camera parameters, h and θ, are not constant in practice. For instance they can change from frame to frame due to the vehicle suspension system (Fig. 6).

Evaluating how the projection of points on the road vary due to changes in these extrinsic parameters, it is found that the parameter with a more significant impact is θ. Ignoring variations on h provokes just an small error, assumable for our goals. For this reason, only the variation of θ is explicitly considered in this paper. In particular, during

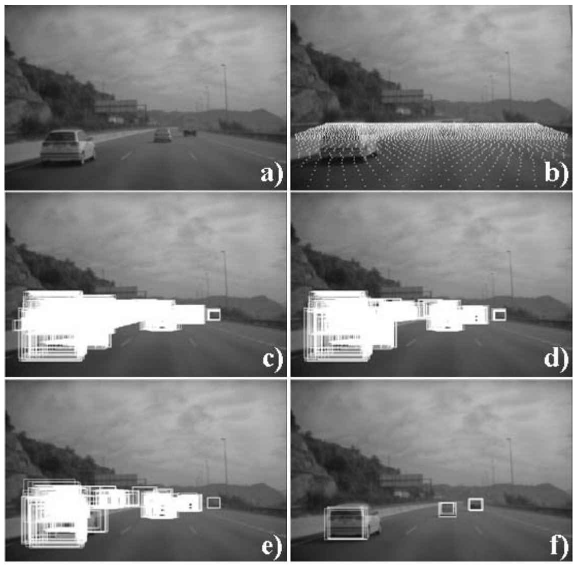

Fig. 5. a) Original image. b) Pixels used as bottom–left corner for candidate image regions where to look for vehicles. c)-f) Image regions that are like to be the bounding–box of the rear/front part of vehicles: c) shows the surviving regions after the first layer of the cascade (rule R_1) and f) shows the regions surviving after the last layer (rule R_n).

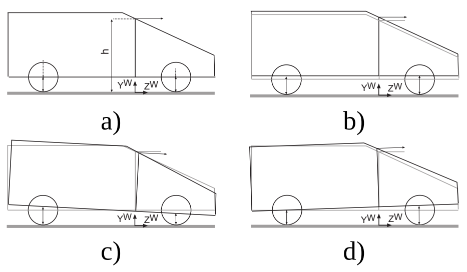

Fig. 6. Simplified vehicle suspension system. a) Neutral position. b) Camera height variation. c) & d) Camera position and pitch variation

vehicle detection we consider several θ values inside a feasible range. This implies that a sparse net of pixels like the one in Fig. 5b must be considered for each θ, thus, the proposed reduction becomes even more relevant.

E. Fusing redundant detections

Another point to consider is the presence of redundant detections. For instance, Fig. 5f shows that the same vehicle can give rise to more than one detection. As can be appreciated in Fig. 7 around the ideal detection (image region in which the vehicle perfectly fits) there are other possible detections with high confidence. The situation is more complex when we don't know the ideal detection (which is the actual situation) since then we must test more sizes, aspect ratios and θ values (having such confidence surfaces for each combination).

Unfortunately, the confidence surfaces are not monotonic, however, their high values or maxima are sufficiently close to the ideal as to allow an accurate detection. At the same time we appreciate that the classifier tolerates a wider region misalignment in the detection of vehicles near the host. Thus, it becomes reliable to have a sparser net of points when we get closer to the image bottom as the proposed sampling scheme induces (Fig. 5b).

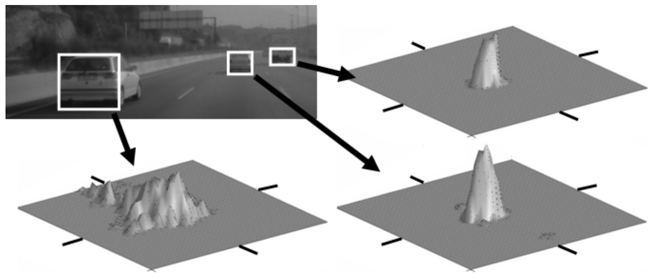

Fig. 7. For vehicles at near, middle and far distances we plot the positive values obtained using (1) (the negative values indicate non–vehicle), assuming a single horizon. At the image the white squares indicate the ideal detection (set manually). The surfaces have been generated by computing the positive values of Σ_R at the corresponding feasible image regions around this ideal: obtained with displacements from -20 to 20 pixels relative to the ideal and along both axes of the image. Since these are positive values of Σ_R they indicate the presence of vehicles, and the height of the response is the degree of confidence on this fact.

Since it is unavoidable to have several detections corresponding to the same vehicle, further processing is needed to fuse them. In the literature we can find different strategies to solve this problem. In [3], regions are partitioned into disjoint non–overlapping groups and each group gives a single detection located at the centroid of its associated regions. Two different algorithms are described in [7], which are based on the idea of non–maximum suppression. Provided that each detection has a confidence value, the strategy consists of identifying the one of highest confidence. This region is registered as output, and detections in a predefined neighborhood around it are deleted. Then this procedure is repeated for the remaining regions, until all detections are either output or suppressed.

In this paper we propose a mixture of both approaches. The mechanism of non–maximum is used, but with the purpose of clustering the non–maximum detections instead of delete them. Then a weighted average of the clustered regions is done, using the confidence value as weight. Our experiments indicate that in our particular problem this technique provides a better localization of the vehicles than just trusting the region of highest confidence.

The neighborhood criterion used to conform clusters is simple. Given a region of maximum confidence, another one is clustered to it if the intersection between both has an area greater than the 50% of the area of the bigger region.

III. Estimation of 3D information

The detection process provides a list of image regions supposed to be the bounding–box of the rear/front part of vehicles. For each region we know the $(x^r, y^r)^T$ coordinate of its bottom–left corner, which is supposed to be a point projected from the road surface, as well as its width w^r. In this section we address the problem of, given the (x^r, y^r, w^r) of a region, estimate the 3D position of the vehicle contained in it. In ideal circumstances we could follow two strategies:

- If the position of the camera with respect to the (flat) road is perfectly known we can determine the 3D

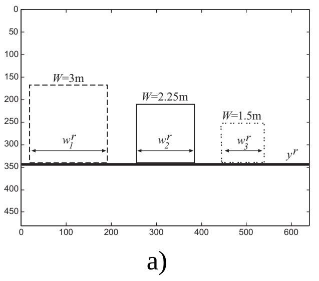
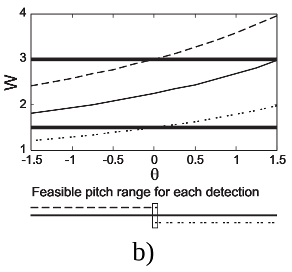
Fig. 8. a) Synthetic case where we suppose three vehicles of different size W detected at the same distance Z in a situation where $\theta = 0$. The rectangles correspond to their projection according to our acquisition model. b) For each case we take the corresponding w^r and y^r and run (9) for θ from -1.5° to 1.5°. Then what we have is the feasible values of W for each θ. Values outside the bold lines are discarded, because they correspond to unfeasible vehicle widths.

point on the road $(X,0,Z)^T$ corresponding to (x^r,y^r) by inverting (2) and (3):

$$X = \frac{f^y h(x^0 - x^r)}{f^x(y^0-y^r)\cos(\theta) + f^x f^y \sin(\theta)}, \quad (5)$$

$$Z = \frac{h((y^0-y^r)\sin(\theta) - f^y\cos(\theta))}{(y^0-y^r)\cos(\theta)+f^y\sin(\theta)} \ . \quad (6)$$

- If the actual width W of the vehicle is known, Z can be estimated taking into account that the effect of θ is meaningless when its value is close to zero 0:

$$Z = \frac{f^x W \sec(\theta)}{w^r} + h\tan(\theta) \sim \frac{f^x W}{w^r} \quad \text{if } \theta \to 0 \ . \quad (7)$$

Unfortunately, neither the exact position of the camera (basically θ) nor the actual width of vehicles are known in practice. To overcome these circumstances we could just fix θ and W to the average of their possible values accepting a fixed error. However, this error can be too large. For instance, if we consider the width of vehicles ranging from $1.5m$ to $3m$, fixing $2.25m$ as average width, then we are dealing with errors of the 33% in the worse case, which could be too much for some applications. For this reason, in this paper we propose a better use of θ and W. More specifically we propose to use the fact that once a detection is available, both parameters are coupled.

Given a vehicle whose rear/front part is parallel to the camera image plane, we consider its width at the road surface level. From (2), W projects into the image as:

$$w^r = \frac{f^x W}{Z\cos(\theta) - h\sin(\theta)} \ , \quad (8)$$

and using (6) to substitute Z and isolating W we obtain:

$$W = -\frac{f^y h w^r}{f^x(y^0-y^r)\cos(\theta) + f^x f^y \sin(\theta)} \ , \quad (9)$$

thus, given a detection with parameters (x^r,y^r,w^r), θ and W are related by this equation. Here we consider h as a known fixed value since as we have already mentioned its variation alone does not introduce a meaningful error. Therefore, given a range of θ values and (x^r,y^r,w^r), we can compute the range of coherent W values for that possible pitch angles and that detection. The claim is that many times we get a smaller region of uncertainty than just using default fixed ranges. For instance, in Fig. 8 we assume $[\theta_{min},\theta_{max}] = [-1.5^\circ, 1.5^\circ]$ and we asses the values of W that are consistent with three detections. In particular, looking at Fig. 8b we can say that if the detection of the bigger vehicle is correct then its actual size must be in the range $[W^r_{min}, W^r_{max}] = [2.5m, 3m]$ instead of assuming the default range $[W_{min}, W_{max}] = [1.5m, 3m]$. Besides, according to this detection the actual pitch must be in the range $[\theta^r_{min}, \theta^r_{min}] = [-1.5^\circ, 0^\circ]$. Therefore, we end with $[\theta^r_{min}, \theta^r_{max}] \times [W^r_{min}, W^r_{max}] \subseteq [\theta_{min}, \theta_{max}] \times [W_{min}, W_{max}]$, this is, with a smaller uncertainty region than the one we could have by default. In fact, if the three detections are correct we can infer that $\theta = 0$ (the actual value in the simulation) since it is the only coherent value for the three vehicles, therefore, we could go to (9) and deduce the actual width of all the vehicles. However, currently we are not fusing the information that can be inferred from the different detections, just to prevent the propagation of errors in case of a detection being a false positive (non–vehicle classified as vehicle).

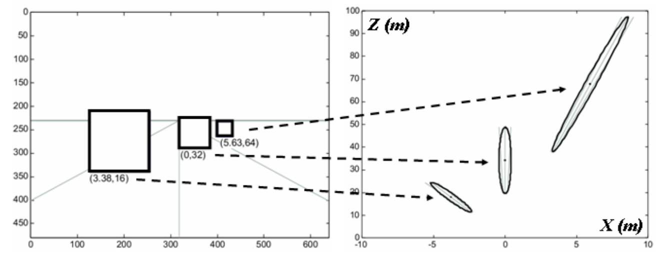
Fig. 9. Left: synthetic detections with actual (X,Z) values in meters. Right: 3D output of the system where, for each detection, a Gaussian is fitted to the region of feasible 3D estimations.

Once the coherent W and θ ranges are determined for a detection (x^r,y^r,w^r), the 3D road coordinates where the vehicle could be are computed solving (5) and (6) for each $\theta \in [\theta^r_{min}, \theta^r_{min}]$. These 3D coordinates lay on a straight line on the road plane. If we consider the fact that detection parameters can be altered by a given noise (so each parameter could have its correct value inside a given range), the 3D coordinates where the vehicle may lay conform a trapezoid on the road surface. This region could be the final output of the detection process. However, thinking on a practical use of this information (for example, in a Kalman filter), a Gaussian is fitted on this trapezoid and given as output. Figure 9 shows an example of the final output. It can be seen that the more distant the vehicles are, the more uncertain the estimation is. Also note that vehicles in front of the camera have their X coordinate more confidently estimated.

IV. Results and conclusions

Currently, we have based our training set on about 1200 vehicles (examples) and the learning iteration re–introducing counter–examples has given rise to a cascade of 6 classifiers. During the detection phase, for a pixel of the sampling net that correspond to a short distance we test about 10 regions and for pixels corresponding to large distance we test 1 region. If a region goes through the whole detection

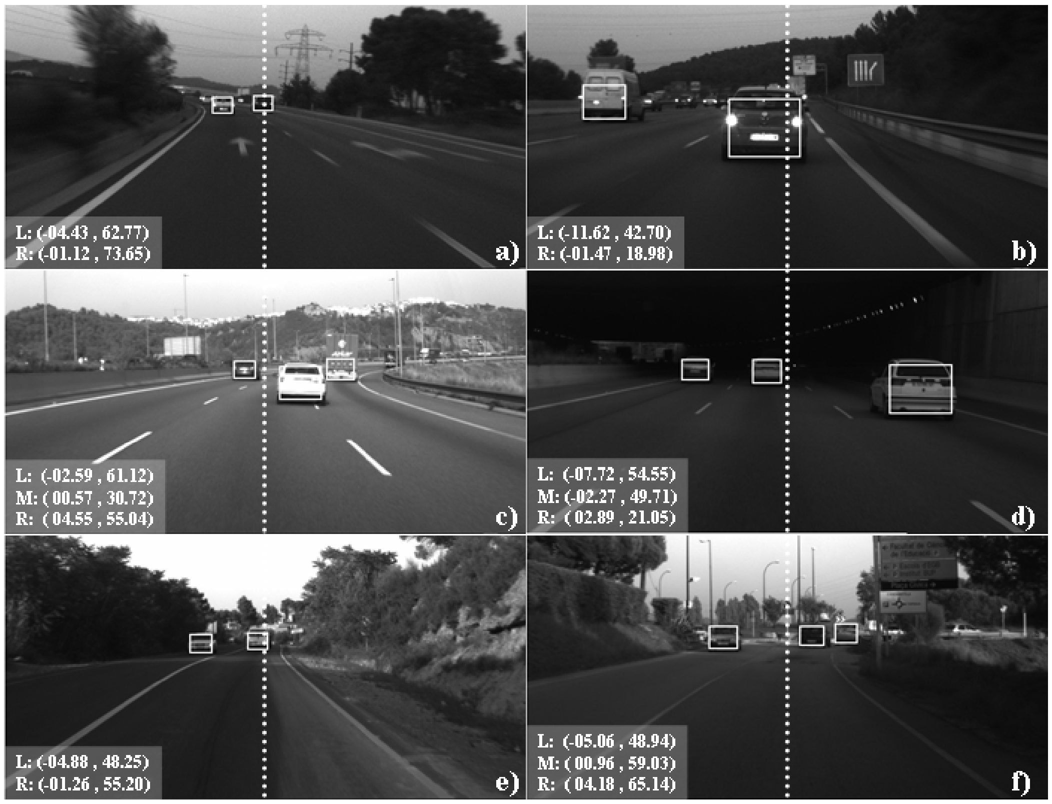

Fig. 10. 3D vehicle detection. The word line $X = 0$ is shown dotted, and we have annotated the (X,Z) position computed for each vehicle (L/M/R: left/middle/right). We see cars, vans (bL, eR) and a truck (cR), nightfall (a, b) and daytime, a tunnel (d), and front (eL, fL) and rear views of vehicles.

cascade then 777 features are computed from it (which basically only happens for vehicles). Besides, during the on–line detection we test 3 possible θ. Figure 10 shows detection examples. From them, we point out that vehicles at the nightfall are also detected despite the fact that the training set was based on daytime images with good lighting conditions. Besides, analogous results are obtained by changing the camera. Notice, that we can add to the detection itself a confidence (using the rules of the detection cascade). Of course, in some situations false positive appear but they tend to stay less than 5 frames. Therefore, we think we have a reliable detection approach that now must be further evolved in several points: 1) providing more examples (researchers training faces use about 10.000 examples and 30 cascades); 2) after this, determine a table with statistics of false positive an possible misdetections; 3) finally, asses if there are any better alternative to Haar filters.

In Fig. 10 we present 3D information too. Up to now, we have only performed an indirect validation of them (through the standard measures of line markings) and we can say that the number are realistic. However, it is still missing a more formal validation (comparing with radar/lidar data, etc.), which is our next future task regarding 3D information. Moreover, since in this process we have an estimation of the actual width of vehicles we want to test the possibility of using it for vehicle classification.

Regarding the applicability of the proposal we must work in improving real–time. *F.i* by using a first version pre–detector we have developed to discard clear homogeneous regions, we achieve processing ratios of 6 fps for images of 640×480 pixels in a Pentium IV 2GHz (without special optimizations of code). Without pre–detection it can take 1.5 seconds to process an image, mainly due to the necessity of testing several θ values to be robust to its variation.

In summary, despite the difficulties derived from using a single monochrome camera as sensor at a host vehicle in movement, we have presented a new framework for detecting vehicles and compute 3D information that we think will be useful for driver assistance.

REFERENCES

[1] M. Maurer, R. Behringer, S. Fürst, F. Thomanek, and E. D. Dickmanns, “A compact vision system for road vehicle guidance,” in *13th Int. Conference on Pattern Recognition*, Vienna, Austria, August 1996.

[2] A. Broggi, P. Cerri, and P. Antonello, “Multi-resolution vehicle detection using artificial vision,” in *IEEE Intelligent Vehicles Synposium*, June 14-17 2004, pp. 310–314.

[3] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 2001.

[4] R. E. Schapire and Y. Singer, “Improved boosting using confidence-rated predictions,” *Machine Learning*, vol. 37, no. 3, pp. 297–336, 1999.

[5] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[6] J. Bouguet. (2004, October) Camera calibration toolbox for matlab. MRL - Intel Corp. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[7] S. Agarwal, A. Awan, and D. Roth, “Learning to detect objects in images via a sparse, part-based representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1475–1490, 2004.