Perspective Multiscale Detection of Vehicles for Real-Time Forward Collision Avoidance Systems

Juan Diego Ortega[1], Marcos Nieto[1], Andoni Cortes[1], and Julian Florez[2]

[1] Vicomtech-IK4, Paseo Mikeletegi 57, San Sebastian, Spain
{jdortega,mnieto,acortes}@vicomtech.org
[2] Tecnun, University of Navarra, Paseo Manuel Lardizabal 13, San Sebastian, Spain
florez@tecnun.es

Abstract. This paper presents a single camera vehicle detection technique for forward collision warning systems suitable to be integrated in embedded platforms. It combines the robustness of detectors based on classification methods with an innovative perspective multi-scale procedure to scan the images that dramatically reduces the computational cost associated with robust detectors. In our experiments we compare different implementation classifiers in search for a trade-off between the real-time constraint of embedded platforms and the high detection rates required by safety applications.

Keywords: Intelligent Transportation Systems, Object Detection, Machine Learning, Pattern Recognition.

1 Introduction

Problems concerning traffic mobility, safety, and energy consumption have become more serious during the past decades as the number of vehicles in the roads has increased. Particularly, a large emphasis has been given to develop Advanced Driver Assistance Systems (ADAS) incorporated to vehicles to try to help the driver reduce his workload while driving, preventing accidents and its associated societal and economical impact. Services like Lane Departure Warning, Blind Spot Detection and Forward Collision Warning had grown mature, just to mention a few, mainly due to the research in computer vision and the reduced costs of cameras and embedded processors.

Considering the challenges of extracting information about the environment of the vehicle with video processing techniques, vehicle detection is one of the more complex, mainly because of i) the varying appearance of objects of the same class, ii) the process of image acquisition (variable illumination, rotations, translations, scales, distortions, etc.), iii) the moving background induced by the motion of the vehicle, and iv) the real-time requirement of most applications.

Many approaches in the field of Forward Collision Warning Systems combine radar and vision (using supervised machine learning) technologies for detecting vehicles [3]. However, radar-based systems only work relatively well with metallic and reflectant objects and are very expensive. This makes vision-based a very

J. Blanc-Talon et al. (Eds.): ACIVS 2013, LNCS 8192, pp. 645–656, 2013.

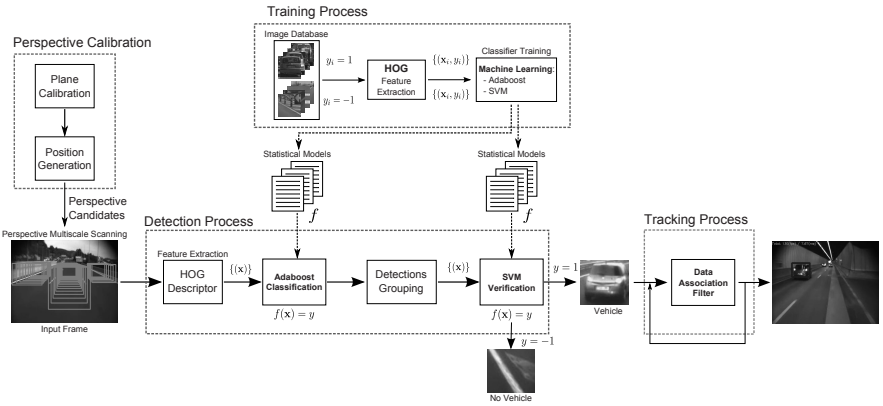


Fig. 1. Proposed approach of the system

interesting alternative due to its cost-effectiveness and ability to detect any object without relying on the object material's properties. Nevertheless, an existing drawback in vision-based approaches is the limited computational capabilities of embedded processors and the computationally demanding nature of vision systems.

The trend in vision-based systems applied to object detection is to use detection-by-classification approaches, combined with an exhaustive scanning of the whole image at different scales, looking for posible locations of the object. Such multiscale approaches are not practical for embedded platforms due to the elevated computational cost and the application real-time requirements. Hypothesis generation (HG) algorithms [2] are, therefore, normally used as a previous step to the hypothesis verification (HV) which usually involve using statistical classifiers. HG usually rely on simple techniques such as color segmentation, edge detection, symmetry and shadows, etc [12]. However, these are likely to be less robust than detection-by-classification and can throw a large number of undesirable false positives and false negatives, which within safety systems is not acceptable.

In this paper we propose to combine the use of detection-by-classification methods with calibration information of the scene to find a trade-off between the robustness of these detectors with the real-time requirements of vehicle detection in embedded platforms. We propose to exploit the known perspective of the scene, which can be computed with a calibration stage, to generate a perspective multi-scale scan grid of the image. Such perspective-based grid dramatically enhance the HG stage which is much more efficient focusing on positions in the image likely containing vehicles and decreasing the computational cost associated to traditional detection-by-classification.

The detection stage was carried out using a linear Adaboost classifier trained with Histograms of Oriented Gradient (HOG) [5] of the training images. It is common to combine HOG descriptors with linear Support Vector Machine (SVM) classifiers. Nonetheless, in this paper we show that it is possible to get similar results with Adaboost-HOG classifiers with lower classification time.

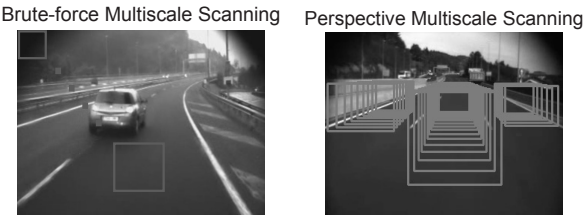


Fig. 2. Posible scanning aproaches: (a) Brute-force and (b) Perspective multi-scale (please note that we have used a very reduced amount of rectangles in this figure for a better understanding; the actual configuration of the grid is governed by two parameters defining the distances between 3D parallelepipeds)

An analysis of the training and testing errors of these statistical classifiers is shown in order to validate our results.

The detection of vehicles in our system consists in an initial phase of classification of perspective-related positions using an Adaboost classifier trained with HOG features. A posterior grouping of the detection is carried out to reduce the number of candidates which are finally verified using an SVM classifier, also trained with HOG features.

The proposed detection strategy in this paper can be easily used in combination with tracking strategies for applications like robust real-time forward collision avoidance systems. Our experiments hold our decisions and show that this is actually a good way to proceed, compared to other options or combination of methods. To understand the framework proposed, the entire system is depicted in figure 1. Besides, we show detailed performance numbers of our system in real conditions, using a hand labeled ground truth reference.

2 Perspective Analysis of the Scene

The calibration of the camera and the computation of its relative pose with respect to the ground plane offers valuable information for the detection of vehicles in images. In this work we propose to formalize the exploitation of the perspective of the scene by means of computing the projection matrix and defining a multi-scale detection approach according to it.

Figure 2 illustrates the proposed approach: (a) the simplest way to proceed once a detector has been trained is to run a multi-scale scanning of the image evaluating each image patch with the classifier in order to determine the presence of objects in the image, we have called this method *brute-force multiscale*; (b) when the projection matrix is known, we can determine a grid of positions in the 3D world in front of the camera where vehicles might appear, and project them into the image. The main difference between these alternatives is that the perspective analysis of the scene focuses significantly the effort of the classifier resulting in a much more efficient scan of the image.

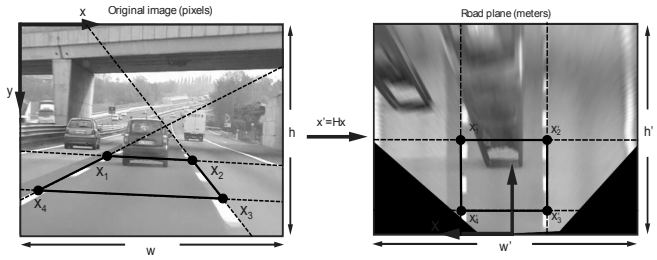


Fig. 3. Points correspondences between the image plane and road plane

2.1 Ground Plane Calibration

The calibration of the scene required to apply the proposed perspective multi-scale approach can be obtained in a two-steps process: (i) obtain the intrinsic parameters K of the camera using well known calibration methods [8]; (ii) determine the relative rotation R and translation $\mathbf{t}$ between the camera coordinate system, and a coordinate system on the ground plane (we assume the ground plane to be planar since it is a common practice by many authors [9]). For simplicity in terms of computational cost, the above extrinsic parameters are computed offline and keep constant.

The second step can be done in a variety of ways, although we propose to use a homography between the image and the ground plane. The world or road coordinate system can be selected such that the road plane is defined by $Z = 0$. In such situation, the projection of a point $\mathbf{X} = (X, Y, Z, 1)^\top$ into a image point $\mathbf{x}$ yields:

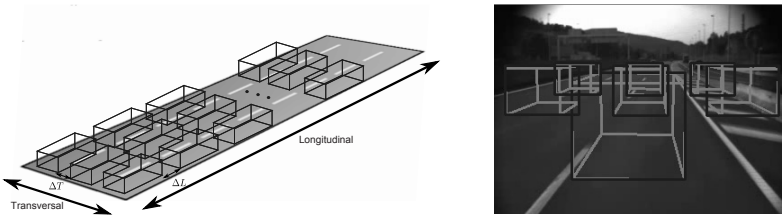
$$\mathbf{x} = K(R|\mathbf{t})\mathbf{X} = K(\,\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3\ \mathbf{t}\,)\left(\mathrm{X\ Y\ 0\ 1}\right)^\top \tag{1}$$

and therefore $\mathbf{x} = K(\,\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{t}\,)\left(\mathrm{X\ Y\ 1}\right)^\top$, which is a 3×3 homography between the image and world plane points: $H = K(\,\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{t}\,)$.

If we calibrate the homography matrix, we have $K^{-1}H = (\,\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\,)$.

This way, once we have computed and calibrated the homography we can extract the rotation and traslation from the columns of the resulting matrix. Note that since these are homogeneous matrices it is necessary to normalize the columns of the matrix in order to get the vectors: $\mathbf{r}_1 = \frac{\mathbf{p}_1}{\|\mathbf{p}_1\|}$, $\mathbf{r}_3 = \frac{\mathbf{p}_2}{\|\mathbf{p}_2\|}$ and $\mathbf{r}_2 = \mathbf{r}_1 \times \mathbf{r}_3$.

The homography H can be computed using a variety of methods, although the simplest one is to use the Direct Linear Transform (DLT [8]) which computes the homography from four point correspondences. Figure 3 illustrates the concept of point correspondences between the image plane, and the world plane (e.g. a road plane in this example).



(a) (b)

Fig. 4. (a) Grid of 3D vehicle positions: ΔT is the distance between positions in the transversal axis; and ΔL the distance in the longitudinal axis. These values can be defined to have overlaping parallelepipeds. (b) Parallepipeds projections in the image plane (light blue) and bounding boxes (dark blue). For seek of clarity only a few intances are drawn.

2.2 Vehicle Position Generation

Once knowing the camera calibration matrix K and the extrinsic parameters, the projection matrix can be constructed as $P = K[R|\mathbf{t}]$. It can be used to project 3D points into the image. Therefore, we can model a vehicle as a parallelepiped and create a grid of interest positions on the ground plane. The grid can be defined with two parameters, ΔL and ΔT, as the longitudinal and transversal distances between positions in the grid respectively in the directions of the plane. Figure 4 (a) depicts the composition of the grid.

The projection of a parallelepiped in an image is a convex polygon whose bounding box can be easily computed and used to determine the region of the image that will be evaluated with the classifier. Figure 4 (b) shows the set of projections of the grid into an image and the corresponding bounding boxes.

This parameterization makes the system very flexible and adaptable to the computational resources available. The higher values of ΔL and ΔT the lower processing cost of the algorithm, although at the cost of having more sparse detections.

3 Detection Stage

Supervised learning is commonly employed in detection-by-classification tasks in computer vision. The goal of supervised learning is to learn the function $y = f(\mathbf{x})$, where $\mathbf{x}$ is an unseen input feature vector and y is the output variable. In classification problems the output variable is the label to which the input feature vector belongs to (in our case, "vehicle" or "non-vehicle").

Discriminative learning methods have been used in the majority of works referred to object and vehicle detection [10]. Within this kind of methods, variations of SVM and Adaboost algorithms stand out in the literature[2]. The main idea underlying the training of classifiers is to find a model which could map the input feature vector to a set of output labels. The training stage involves the application of supervised training algorithms to a set of feature vectors

extracted from the image database. This database must have positive images (e.g. "vehicles") and negative images (e.g. "non-vehicles"). In this work we have used a public available database of rear-view vehicle images (GTI-UPM vehicle database [1]), which consists of 3425 positive and 3900 negative images. The result of the training is a statistical model used in the detection stage. Both Adaboost and SVM implementations of the training algorithms have been evaluated.

An important decision in pattern recognition and machine learning is the set of image features with which the images will be represented. This is, feature extraction process is carried out before classifier training. Different combinations of classfiers and feature decriptors have been reported in the literature such as Haar-like features and a cascade of boosted classifiers [13], Gabor Filters and SVM [11] or HOG and SVM [5]. In this paper we propose a combination of Adaboost and SVM classifiers trained with HOG descriptors due to its outstanding capabilities to visually describe objects.

3.1 HOG Feature Vector

The HOG method [5] consists on evaluating well-normalized local histograms of image gradient orientations in a dense grid. The basic idea is that local object appearance and shape can often be characterized by the distribution of local intensity gradients or edge directions, even without precise knowledge of the corresponding gradient or edge position. It captures edge or gradient structure that is very characteristic of local shape, upholding invariance to geometric and photometric transformations, except for object orientation. Therefore, it makes this method an interesting alternative to use in our task of vehicle detection.

In our system, four configurations of HOG decriptors were tested in combination with the learning algorithms. These configurations result in different feature decriptors which vary in length depending on the parameters chosen to compute the HOG. The proposed configurations are presented in table 1.

Table 1. HOG descriptor configurations used for training vehicle classifiers

	HOG-1	HOG-2	HOG-3	HOG-4
Image size (pixels)	64×64	64×64	32×32	32×32
Cell size (pixels)	8	8	4	4
Block size (pixels)	16	16	8	8
Num. of bins, β	9	18	9	18
Block stride (pixels)	8	8	4	4
Descriptor length	1764	3528	1764	3528

Note that the image size used for computing the descriptors restricts the minimum size of an object to be detectable. Therefore, this restriction could determine the HOG configuration to use in a practical problem.

3.2 Adaboost Classifiers

Binary classifiers are obtained using the Dicrete Adaboost algorithm described in [7]. Adaboost, which stands for *Adaptive Boosting* produces an additive classifier which is a linear combination of several weighted weak classifiers. We denote the set of N training points as $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$, where each $\mathbf{x}_i$ is an $n-dimensional$ input vector such that $\mathbf{x}_i \in \mathbb{R}^n, i = 1, \ldots, N$ and $y_i \in \{-1, +1\}$, indicates the class to which the point $\mathbf{x}_i$ belongs.

The goal of Adaboost learning algorithm is to estimate a function:

$$f : \mathbf{x}_i \in \mathbb{R}^n \mapsto y_i \in \{-1, +1\} \tag{2}$$

Moreover, it can be interpreted as a procedure for iteratively fitting an additive model using a set of basic functions or weak learners, h_t, repeatedly over a series of rounds $t = 1, \ldots, T$, [6]. The final model is a linear combination of the weak learners weighted by a set of values, α_t, $t = 1, \ldots, T$. The final strong model, thus, has the form:

$$f(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}), \tag{3}$$

Hence, the decision function is given by equation (4):

$$f(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right) \tag{4}$$

For our application, decision trees were selected as weak learners since they are the most popular weak classifiers used in boosting schemes. Additionally, their simplicity makes the training of Adaboost classifiers an easy and quick task.

3.3 SVM Classifiers

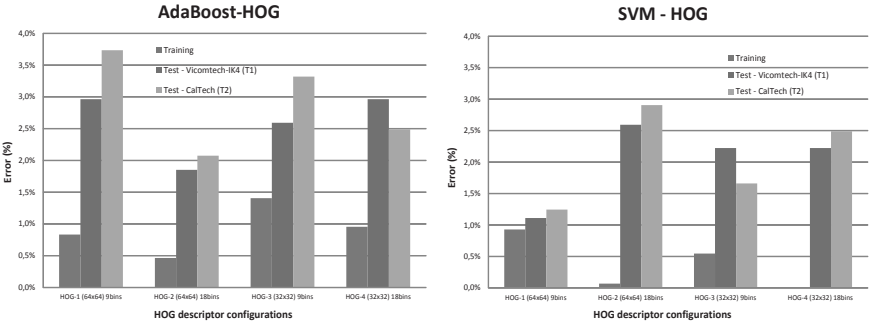
The SVM algorithm [4] finds the hyperplane defined in (5) which best separates the training examples by maximizing the distance between the closest elements of the two classes and the hyperplane. This distance is called margin.

$$(\mathbf{w} \cdot \mathbf{x}) + b = 0, \quad \mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R} \tag{5}$$

where $\cdot$ is the dot product and $\mathbf{w}$ is a normal vector to the hyperplane and b is the classification threshold of the model. Thus, the decision function yields:

$$f(\mathbf{x}) = sign\,(\mathbf{w} \cdot \mathbf{x} + b) \tag{6}$$

To simplify the training, linear SVM were used in conjuction with the four configurations of HOG decriptors of table 1. SVM parameters where chosen taking the recommendation of Dalal and Triggs [5]; where a soft linear SVM was used for training their classifiers. However, a trial and error study was done to



(a) (b)

Fig. 5. Training and testing errors: (a) Adaboost-HOG, (b) SVM-HOG classifiers

tune the parameters so that a better performance is obtained for our application. As stated in the decription of our system, SVM classifiers are used to verify the hypothesis generated by the Adaboost classifier. In this way, computation time will be reduced.

3.4 Classifiers Training and Testing

The vehicle detection strategy proposed in this paper establishes the use of Adaboost as a first stage and a SVM classification as a verification stage. Both classifiers were trained using HOG features. The GTI-UPM vehicle database was used for training, although we have tested our classifiers with two additional testing databases, denoted as "Vicomtech-IK4 (T1)" and "Caltech (T2)". The former contains 155 positive and 115 negative images, which were obtained from video sequences in roads in San Sebastian. The latter is a public available image database of vehicles [14] and consists of 126 positive and 115 negative high-resolution images.

Results of the training and testing errors for Adaboost and SVM are shown in figure 5. For our study, these errors were defined as the proportion of missclassified images over the total number of images in the database. To calculate the errors, the image sets undergo a classification process. The results of the classification are then compared with the labeled data, resulting in an error rate. The final error comprises the resultant misclassification rate. For training error the image database was the one used for training while for testing error the databases used where completely unkown by the classsifier.

The global training error gives some notions of how well the learning algorithm could separate the feature space used for training. Regarding the global testing error, the process is analogous but using a totally unseen set of images. In this matter, the global testing error comprises the generalization capacity of the statistical classifier. Notice that, in most of the cases, SVM classifiers achieved lower testing errors than Adaboost. However, SVM requiere much more time

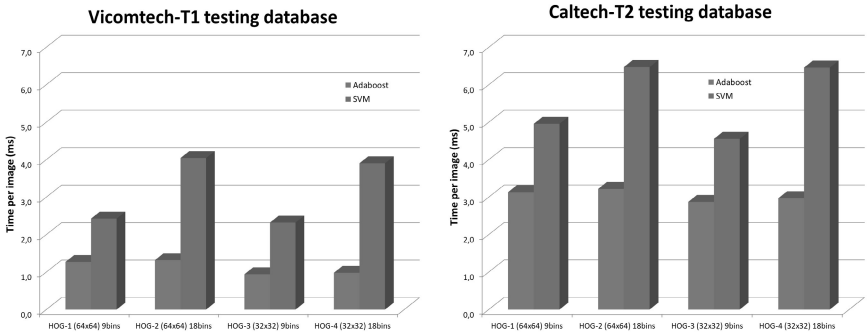


Fig. 6. Execution times between Adaboost and SVM classifiers for: (a) Vicomtech-T1 and (b) Caltech-T2 databases

for training the classifiers, namely, an average of 62% more time is needed for training an SVM. The difference in training time is likely to be due to the intrinsic properties of each of the learning algorithms used for learning. Adaboosts splits the data to generate the best decision trees while SVM solves a quadratic programming optimization problem [4].

The difference in behaviour was also evidenced in execution time. Time for processing each image in the database was also computed and compared. The improvement in time using Adaboost classification ranges between 30% to 70% when comparing the time for classifying an image with SVM (times per image are depicted in figure 6).

Thus, our detection strategy uses an Adaboost classifier to classify the candidate position retrieved by the perspective multiscale calibration. Then, a grouping process is carried out in order to reduce the number of hypothesis to be verified in the next stage. The latter will group in one candidate hypothesis all those bounding boxes which overlap each other. Finally, these candidates are verified using a SVM classifier.

4 System Test and Discussion

The proposed algorithm has been tested using image sequences of highways captured under different ilumination and weather conditions so that robustness could be verified. Additionally, a sample two minutes ground truth video containing several challenging situations (entering and exiting a tunnel, heterogeneous pavement color, casted shadows, overtaking maneouvres, etc) was annotated and has been used to extract values of true positives (TP) and false positives (FP) by applying a bounding box overlapping criterion.

For the sake of completeness we have applied a simple tracking algorithm which joins detections in time attending to their coherency in position and appearance,

and uses a Kalman Filter to predict the position of given tracks in the absence of detections (which tends to happen using detection-by-classification).

The objective of this section is to compare our system with the default brute-force multiscale detection strategy normally used with classifiers. This method consists on scanning the entire image at different positions and scales. In each position an SVM classifier is executed, which imply the extraction of the HOG features. Three parameters define the total number of evaluating windows in the brute-force multiscale method: the scale at which the searching window changes in size, s, the number of levels each window is resized, $nLevels$, and the stride at which the window moves across the image frame. For this comparison, we fixed the scale and window stride to 1.05 for the former and the cell size for the latter. In this way, only the $nLevels$ is involved.

On the other hand, our method is dependant of the number of longitudinal and transversal level used for creating the perspective grid. Different configurations of perspective multiscale and brute-force multiscale were tested for comparison and compared in terms of execution time and values of TP and FP.

The feature descriptor vector used for testing was the HOG-3. This is, the minimum window size is 32×32 pixels and hence, the minimimum vehicle size detectable. We have chosen this feature descriptor because our video size, 320×240 pixels, has vehicle instances which are very small. Depending on the frame resolution it is possible to use a different configuration of HOG descriptor, achieving similar results.

In table 2 are presented the results from applying **perspective multiscale** and **brute-force multiscale** to our ground truth video. Tests were done in an IntelÂő CoreâĎć i5-3330 CPU 3.00 GHz, 8.00 GB RAM.

Table 2. Results of Perspective Multiscale and Brute-force multiscale

Perspective Multiscale						Brute-force Multiscale				
# Hypotheses	ΔL	ΔT	T(ms)	TP	FP	# Hypotheses	$nLevels$	T(ms)	TP	FP
10	3000	1000	2.3	153	155	3869	1	11.2	95	263
18	3000	600	3.8	214	204	7613	2	11.4	114	492
20	1500	1000	4.5	236	189	11357	3	11.7	137	661
29	1000	1000	4.4	207	201	35746	10	30.6	264	844
34	3000	333	4.9	308	391	51156	15	32.7	332	878
37	1500	600	5.2	300	318	64287	20	33.9	320	1275
54	1000	600	6.3	279	329	74941	25	34.6	288	1513
70	1500	333	8.2	368	455	82787	30	36.8	283	1262
104	1000	333	10.4	352	512	87630	35	40.7	284	1238

As expected, the execution time of the brute-force multiscale increases considerably as the number of levels increase. The raise in the number of evaluation windows produce an increase in the value of TP; however more FP are also encountered. Moreover, tests had proven that brute-force multiscale could not process the video at real-time when the number of scales is greater than 10.

Fig. 7. Examples of detections in real videos

On the other hand, our method is able to process the image in much less processing time without reduction in detection performance. Brute-force multiscale generates more false positives as the number of levels increases. The advantange of using the knowledge of the perspective scene allows our classifiers to focus in regions where vehicles are more likely to appear. Then, no time is wasted scanning absurd hypothesis (very large and very small vehicles) or out the relevant regions of the image.

The use of Adaboost as a first classification stage provides the best performance. Adaboost was trained using decision threes as weak learners, which are very fast structures to be accesed. Hence, the classification time is very low comparing to SVM. The combination of Adaboost and SVM makes the algorithm robust and fast. SVM reduces the ammount of false positives that were generated by Adaboost classification without increasing significantly the computational cost of the algorithm.

Some results of the detection system are illustrated in figure 7. As can be observed, very different vehicles, in terms of size, colour or aspect-ratio are successfully detected, highly accurately delimiting their contour.

As a reference, we have shown the feasibility of our approach for embedded platforms implementing an instance of the algorithm in an ARM processor. The computing limitation of an ARM device are well-known, hence a fast and reliable system is required. Tests were carried out using the perspective multiscale in an ARM® dual-core Cortex™-A9 MPCore™ / 800 MHz, 512 MB DDR3 using 10 longitudinal positions and 3 transverse positions (i.e. 20 hypothesis positions). An approximate procesing time of 40ms (25fps) was obtained.

5 Conclusions

We have presented an efficient way of detecting vehicles in videos using the knowledge of the perspective of the scene in combination with machine learning classifiers. This approach has reduced processing times by detecting vehicles in two stages. First, a fast yet robust Adaboost-HOG classifier is applied on regions of the image defined by a perspective-based scan process, which allows to focus the attention in the regions where vehicle appearance is most likely neglecting uninteresting or absurd hypotheses. Then, an intermediate grouping phase is done to reduce the number of detections. Finally, an SVM classifier validates the hypotheses. The proposed detector can be combined with any kind of tracking system. In this work, a simple tracking system is proposed which will

be extended in future publications. Our perspective multiscale system has proved to achieve better execution times than using a regular brute-force multiscale, getting reductions in time of more than 50%. The results show that our approach is suitable to be integrated in embedded platforms achieved a trafe-off between robustness and accuracy, while keeping real-time operation.

Acknowledgements. This work has been partially supported by the program ETORGAI 2011-2013 of the Basque Government under project IEB11.

References

1. Arróspide, J.: GTI vehicle database (2012), http://www.gti.ssr.upm.es/data/
2. Arróspide, J.: Vision-based vehicle detection and tracking with a mobile camera using a statistical framework. Ph.D. thesis, Universidad Politécnica de Madrid (2012)
3. Bertozzi, M., Bombini, L., Cerri, P., Medici, P., Antonello, P.C., Miglietta, M.: Obstacle detection and classification fusing radar and vision. In: Intelligent Vehicles Symposium, pp. 608–613. IEEE (2008)
4. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. Computer Vision and Pattern Recognition, vol. 1, pp. 886–893. IEEE (2005)
6. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Vitányi, P.M.B. (ed.) EuroCOLT 1995. LNCS, vol. 904, pp. 23–37. Springer, Heidelberg (1995)
7. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: A statistical view of boosting. The Annals of Statistics 28(2), 237–407 (2000)
8. Hartley, R.I., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press (2004)
9. Kim, Z.: Robust lane detection and tracking in challenging scenarios. IEEE Transactions on Intelligent Transportation Systems 9(1), 16–26 (2008)
10. Papageorgiou, C., Poggio, T.: A trainable system for object detection. International Journal of Computer Vision 38(1), 15–33 (2000)
11. Sun, Z., Bebis, G., Miller, R.: Improving the performance of on-road vehicle detection by combining gabor and wavelet features. In: Proc. Int. Conf. on Intelligent Transportation Systems, pp. 130–135. IEEE (2002)
12. Sun, Z., Bebis, G., Miller, R.: On-road vehicle detection using optical sensors: A review. In: Proc. International Conference on Intelligent Transportation Systems, pp. 585–590. IEEE (2004)
13. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. Computer Society Conf. on Computer Vision and Pattern Recognition, vol. 1, pp. 511–518. IEEE (2001)
14. Weber, M.: Caltech rear view car image database (1999), http://www.vision.caltech.edu/html-files/archive.html