Computer vision: the emerging cost-effective technology for vehicles

Marcos Nieto[1]*, Juan Diego Ortega[1], Oihana Otaegui[1], Andoni Cortés[1], Javier Barandiaran[1], and Luis Unzueta[1]
1. Vicomtech-IK4, Paseo Mikeletegi 57, San Sebastian, Spain
mnieto@vicomtech.org

Abstract: In this paper we analyze the recent exponential growth of applications based on video processing in the framework of Advanced Driver Assistance Systems (ADAS). Specifically, we focus on the cost-effective solutions provided by computer vision methods for services like lane departure warning systems, collision avoidance systems, etc. Along the paper our own contributions are described as examples of the state of the art from the perspective of real-time by design, searching a trade-off between the accuracy and reliability of the designed algorithms, and the restrictive computational, economical and design requisites of embedded platforms.
Keywords: ADAS, COMPUTER VISION, REAL-TIME, EMBEDDED SYSTEMS.

Introduction

Last decade has seen an exponential growth of the presence of computer vision systems in many sectors due to the explosion in consumer electronics, consolidation of the information society, the reduced costs of hardware production, and the significant contributions of scientific communities in the development of algorithms and methods.
Most noteworthy, in the field of Intelligent Transport Systems and intelligent vehicles, the technology has favoured the appearance of a number of systems to improve comfort, safety and efficiency of transport. Examples of these systems are GPS navigation systems, automatic cruise control, vehicle identification, communication and eCall, and more recently advanced driver assistance systems like lane departure warning, driver drowsiness detection, or semiautomated driving systems like platooning, parking assistance, etc.
The abovementioned systems cover a wide range of services for the driver that make intensive use of sensing systems, that allow obtaining data from the environment of the vehicle to understand the situations and take actions (e.g. send the information to the driver, actuate on the vehicle, launch communication channels, etc). The introduction of such systems to the

market is strongly conditioned to reaching cost-effective solutions, which is not an easy task since sensing systems typically are based on expensive technologies, or difficult to miniaturize.

Computer vision systems have broken through this market for several reasons: (i) richness of the information contained in images; (ii) advances in electronics and optics that allow reducing the size and cost of cameras; (iii) the growing computational capacity of embedded systems; and (iv) the outstanding contributions of scientific communities in image processing, projective geometry and machine learning (to mention a few). This last point is crucial because it opens the door to the definition of uncountable innovative methods and the creation of new services.

In this article we focus on the latter point, particularly describing our contributions in this field, in which we have collected an extensive expertise in both the design of algorithms, and their implementation in real-time platforms.

From research to market

During the last three decades there has been numerous research programs aiming to give steps towards the introduction of intelligent systems into vehicles, like the DARPA Grand and Urban Challenge, EUREKA Prometheus, or Google driverless car just to mention a few. These programs exemplify the effort of the scientific community, which has proposed since a vast number of algorithms, methods and technologies for intelligent vehicles. The target of these programs was to learn, test and evolve the technology in an exercise of exploring the possibilities that technology can offer to drivers in particular and the transport sector in general.

For the tests, intelligent prototype vehicles have been used, equipped with prototype sensors, large size radar, laser, GPS, antennas, multiple cameras, CPUs, batteries, etc. These cars demonstrate only the feasibility of the technical part of deploying intelligent vehicles. There is a very steep path from these vehicles to solutions for the market, considering also legal aspects, space, cost and consumption constraints of HW.

Historically, only the more basic ADAS (night vision, lane departure warning, safety distance warning) have reached the market, and only to high-end vehicles, as the result of working hard in the reduction of the complexity of algorithms, and their optimization into low capacity, low cost embedded platforms.

These days, ADAS systems in midrange vehicles are closer, thanks to the advances in more efficient computer vision methods, more easy-to-program and reduced cost embedded HW, and the growing interest of large manufacturers that compete for the market.

Among this wide range of aspects that make this technological success possible, in this paper we wanted to focus on the aspects of the design of computer vision methods, describing our own methods as example of how to find a trade-off between real-time and functionality.

Our own experience

The following subsections describe the vision-based work we have done in several of the research lines associated with specific ADAS. Details about the reduction of algorithm complexity are given, and details on the HW platform used are given in the next section.

Lane Departure Warning

The real-time detection of lanes in the road, joint with the determination of the position of the vehicle implies the possibility to warn the driver in case the vehicle is abandoning the lane involuntarily or to automatically correct the steering of the vehicle to keep it in the lane. Video processing systems allow detecting lane markings and determine whether the vehicle is abandoning the lane.

In our experience, it is critic to design an approach that avoids as many operations at pixel level as possible. In that sense we defined a method [1] that filters the image only once to detect the pixels in the image that belong to the lane markings for further fitting a lane model coherently with the perspective of the scene. We can see an example detection of such lane markings in Figure 1. The known perspective of the image allows us to create a perspective histogram which accumulates the detected lane markings pixels according to their relative orientation to the vanishing point which simplifies further computations. Such detections can be fed into a linear tracking stage like the Kalman filter, which adds the required time coherence to detections, smoothing the final result.

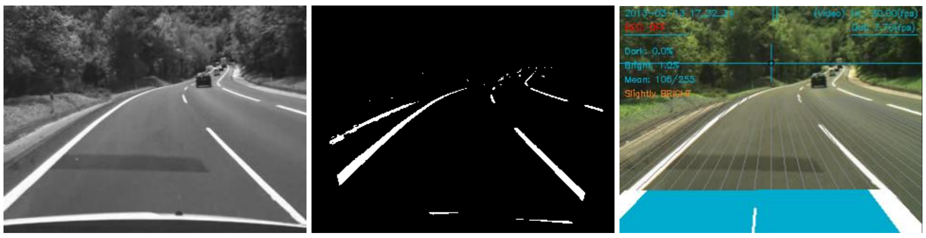

Figure 1: Detection of lane markings and the associated accumulated histogram.

The proposed method has been implemented in an ARM-based platform (see further sections for details about it) consuming about 10 ms filtering the image, and about 2 ms fitting the model.

For the evaluation of the system, we have analyzed a set of sequences with a total length of 129 minutes, in roads and highways, with varying illumination conditions (direct sun light, cloudy days, rain, night, etc). One way to objectively determine the correct functioning of the system is to check the correct detection of lane changes. The detection of such events shows that the system is well detecting the position of the lane markings. Lane changes are challenging situations that usually involve fast motion and partial absence of measurements (typically, only one lane marking is observed during such manoeuvre). Therefore, we hypothesize that the system is able to monitorize when the vehicle is getting out of its own lane if lane changes are detected correctly.

Table 1 shows the statistics of lane change detections. The recall and precision values are computed to show the performance of the proposed method. The recall, which is related to the number of correctly detected events, give values above 92 %, while the precision, related to the quality of the detections, is above 96 %.

	Duration	TP	FN	FP	R	P
Sunny1	25’35’’	14	1	0	0.933	1.000
Sunny2	24’39’’	21	1	0	0.954	1.000
Tunnels1	30’51’’	5	1	0	0.833	1.000
Tunnels2	22’51’’	23	2	1	0.920	0.958
Rainy1	16’00’’	13	2	1	0.867	0.928
Rainy2	9’27’’	13	0	1	1.000	0.928
TOTAL	129’23’’	89	7	3	0.927	0.967

Table 1: Detection of lane changes. TP: true positives, FP: false positives, FN: false negatives, R: recall = TP / (TP + FN), P: precision = TP / (TP + FP).

In our experiments we have observed that it is useful to increase the complexity of the road model in order to add the possibility to estimate as well the curvature of the road. Nowadays, vehicle detection systems, using time-of-flight sensors like radar are able to accurately determine the presence, distance and speed of objects in front of a vehicle. This feat allows services such as safety-distance warning, stop-and-go, etc. However these systems do not take into account the curvature of the road, therefore, many false alarms are generated when a vehicle is physically close, but in a different lane (see an example in Figure 2).

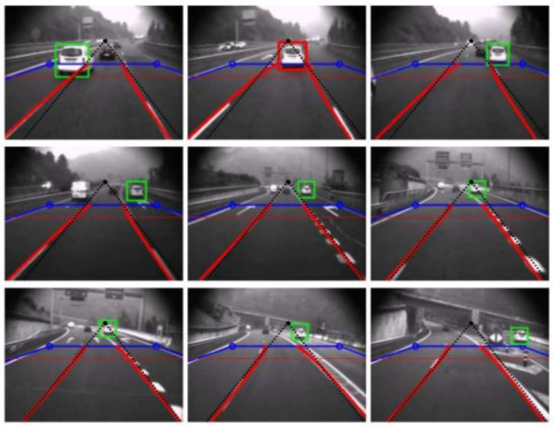

Figure 2: Computing the curvature of the lane allows determining whether a preceding vehicle is within the own lane (red), or not (green). This feature helps to reduce false alarms about safety distance.

In a previous work of the authors [2] we presented a road modelling approach that used a circumference model in the rectified view. It was solved using a Rao-Blackwellized particle filter which splits the lane model in two parts: (i) a linear part consisting in the position and size of the lane, which can be solve with the linear part of the filter; and (ii) a non-linear part that modelled the curvature of the road, and that is modelled as a set of weighted hypotheses. One of the major requirements of such systems is their operation in real-time. For that purpose the Rao-Blackwellized technique dramatically reduces the computational cost of such a curvature fitting module.
The accurate estimation of the curvature in this environment is pretty difficult, and heavily depends on the correct estimation of the lane position, and the visibility of the curvature ahead. Indeed, the curvature can only be computed when the lane exist, are visible at a far distance and the position of the lane has been correctly detected. Our method switches off the computation of the curvature in the situations in which the reliability of the lane markings detection does not exceed a defined threshold. In general, for the sequences we have analysed, the curvature is accurate in most situations in which there is at least one continuous lane markings, including different illumination conditions.

Vehicle detection and pedestrian detection

Detection and tracking of vehicles by means of video analysis can be accomplished in a two-stage fashion: hypothesis generation, and hypothesis verification [3]. The first usually implies a quick search, so that the image regions likely containing vehicles are broadly identified. Typical methods include edge analysis, color, and shadows.
In our work, we have used a Bayesian segmentation approach, which used a combination of shadows, intensity and gradient information [1], which is a fast hypotheses generation method.

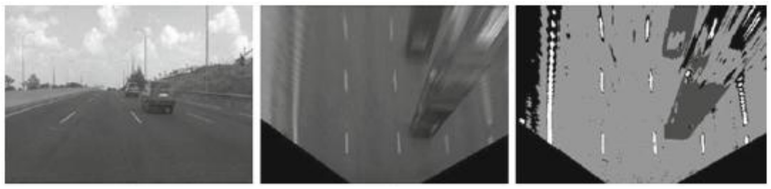

Figure 3: Segmentation process: (left) original image, (center) road-plane image, and (right) four-level segmentation

Model-based and appearance-based techniques can be then used to verify the existence of vehicles in the selected regions (hypotheses verification). We have used a supervised machine learning process, which involve a training stage in which visual features are extracted from a set of positive and negative samples to design a classifier. Neural networks and support vector machines (SVM) have been extensively used for classification [4] while strong efforts have been done to find a good feature extraction technique. In that sense, most widely accepted are histogram of oriented gradients (HOG), principal component analysis, Gabor filters, and Haar-like features.

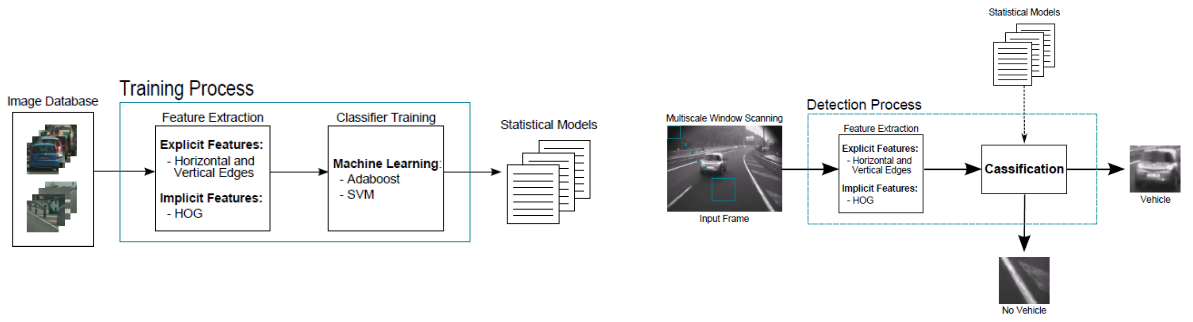

Figure 4: Block diagram of the proposed approach based on appearance features like HOG, edges, symmetry, and SVM classification.

In our work, we have adopted two main approaches, in one of them we exploit the power of SVM and HOG, and in the other we compute features like shadows, symmetry, edges, etc with an Adaboost classifier, which is a fast alternative to HOG-SVM (Figure 4 shows an illustrative block diagram of the approach).

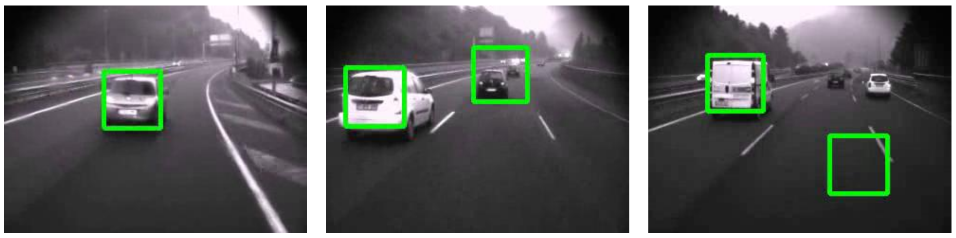

Figure 5: Example results of the detector using SVM and HOG.

SVM-HOG provides better detection results (see Figure 6), but the computation of HOG is much more computational, and it is hard to migrate to ARM architectures. We exploit the perspective of the scene to create the prior knowledge of the expected size of vehicles seen in the image at different distances. This prior information avoids the typical use of multiscale detection such that much less hypotheses need to be evaluated. Besides, we have tuned our algorithm so that it gives near 0% of false negatives. The cost is having some false positives (like in right-most image in Figure 5) that can be filtered afterwards with a tracking approach.
The following figure describes the training and testing error for different configurations of Adaboost and SVM classifiers using explicit and HOG features, respectively, on three different databases (GTI-UPM for training, and Vicomtech(T1) and Caltech (T2) for testing). Training times are also shown.

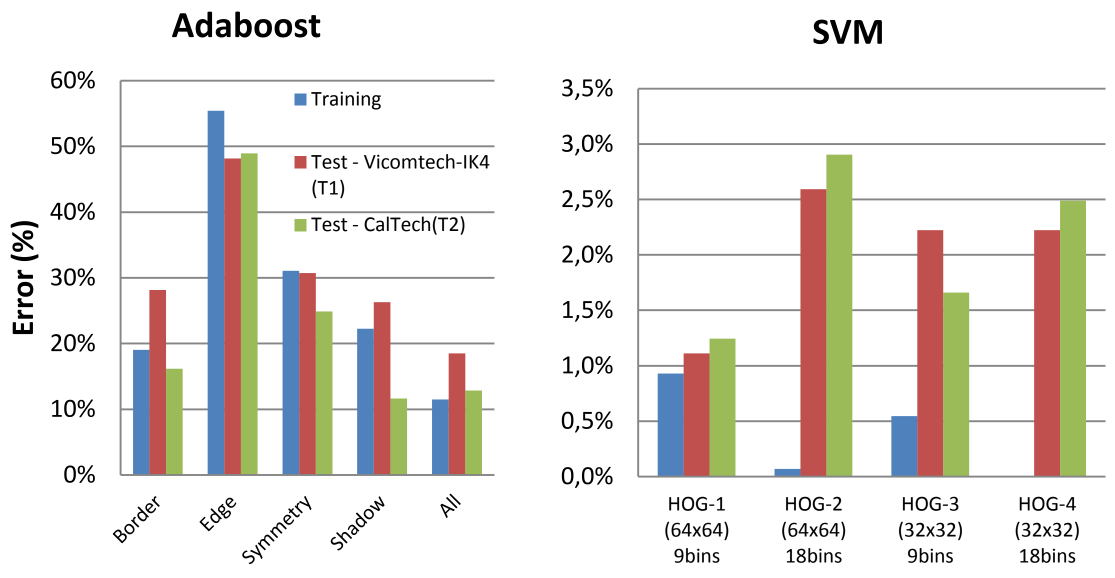

Figure 6: Example results of the detector using SVM and HOG.

The detection of pedestrians has been tackled in an analogous way, since we have used as well a supervised training methodology,using HOG descriptors and SVM

The detection of vehicles and pedestrians is sparse, i.e. there are a lot of missdetections. False negatives can be solved using tracking approaches, which are anyway needed to provide time coherence to detections, so that we can reconstruct the trajectory of objects.

For the tracking, we have implemented a Rao-Blackwellized Data Association Particle Filter (RBDAPF) [5]. This type of filter has been proven to provide good multiple object tracking results for even in the presence of "sparse" detections as the ones we have in these sequences, and can be tuned to handle occlusions. The Rao-Blackwellization can be understood as splitting the problem into linear/Gaussian and non-linear/non-Gaussian parts. The linear part can be solved with Kalman Filters, while the non-linear one must be solved with approximation methods like particle filters. In our case, the linear part is the position and size of a bounding box that models the persons. The non-linear part refers to the data association which is the process of generating a matrix that links detections (the HOG ones, for instance), with objects or clutter. The association process can be strongly non-linear so that sampling approaches can be used. In our case we have implemented ancestral sampling.

Preliminary results have shown that this approach is able to detect and track simultaneously up to 4 or 5 objects, which is a reasonable number for this type of scenarios.

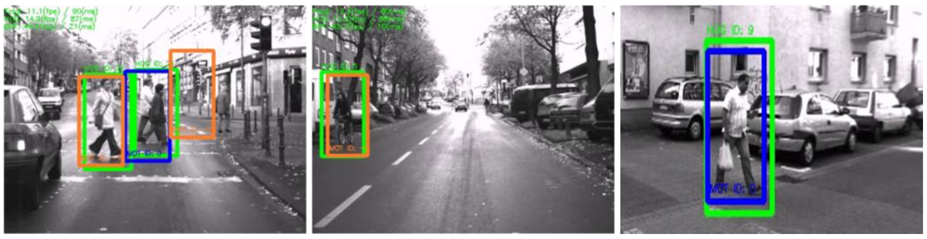

Figure 7: Pedestrian detection and tracking using HOG detection (green), and RBDAPF tracking (orange and blue meaning low and high confidence tracking values).

Besides, the control of input/output of new persons is handled thanks to the use of the data association filter, which classifies detections according to the existing objects, remove objects that have no detection for a too long period of time, and creates new objects when detections not associated to previous objects appear repeatedly.

Driver Drowsiness Detection

A driver may lose attention of the road for several reasons, some of them produced by voluntary actions (using the mobile phone, GPS, talking to passengers, etc), and involuntary actions like drowsiness or fatigue. Computer vision can be used to determine the fatigue of the driver by means of analysing biometrics, the more robust and easy to compute being the blinking speed.

Analogously to the other ADAS vision systems, this method is affected by the illumination conditions, and also by the great variability of the appearance of persons (including variable elements like glasses, skin color, facial hair, etc). For that reason we have devised a flexible solution to cope with all the variability of the scenario, focusing on a user-based detection and tracking of the face of the driver using a dynamic 3D model of the face. Eyes are then detected within the modelled face and eye blinking is measured and analyzed to generate alarms when the system determines the driver is getting drowsy.

The solution we have implemented is based on several algorithms. First, the face of the driver is detected in the image as an initialization stage. We have used the detector proposed in [6] using a cascade of weak classifiers (Adaboost) fed with generic features like LBP.

After this first detection stage, tracking and fitting is achieved by using algorithms based on online appearance models, proposed initially in [7] and further improved in [8]. This step fits a 3D model called Candide [9] to the face of the driver, from which the position of the eyes can be obtained quite accurately.

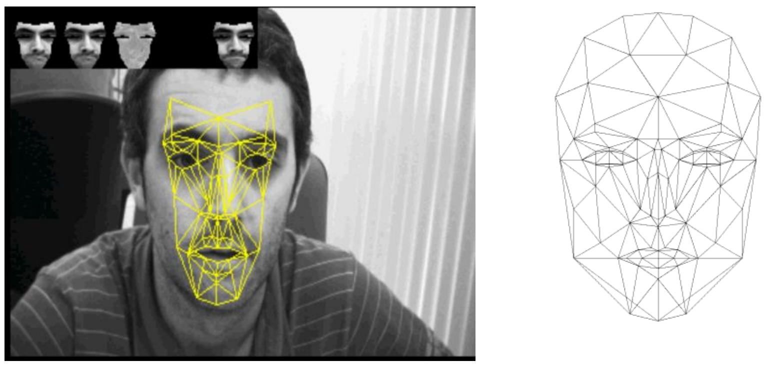

Figure 8: Fitting the Candide-3 model to the image of an user's face.

The third stage comprises the classification of the eye region into two classes: open and closed.

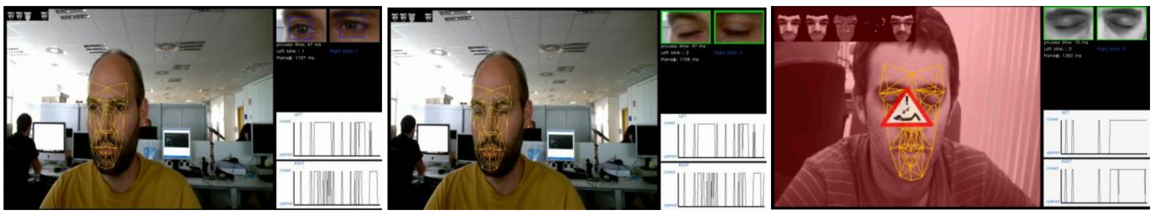

Figure 9: Detection of blinking in the proposed ADAS framework: (left) opened; (center) closed; (right) drowsiness detection using PERCLOS measurements.

This information is then interpreted in terms of the blinking frequency and also its duration. There exist several possible ways to normalize these measurements, being one of them the so-called PERCLOS measurement. This interpretation has been shown to be one of the most adequate and reliable to quantify the driver fatigue in vision-based systems. PERCLOS is defined as the percentage of time that the eyelids are 80-100% closed over a time window (e.g. over a 1-minute, 3-minutes or 20-minute time window). For example, if the eyelids are 80-100% closed for a total of 6 seconds within a one-minute time window, PERCLOS would be 6/60 or 10%.
In our experiments we have found that we can compute PERCLOS measurement in day-night scenarios by adding IR illumination to the cameras that capture the face of the driver. Additionally, this module has been tested independently in Apple platforms like iPad 2-3 and iPhone 4S-5 in order to check its behaviour in ARM-based processors. The following section depicts the details of its performance.

Implementation details

The methods described in this paper have been tested and deployed for real-time operation in an ADAS real framework. All the developments have been made as a multi-platform software, and built and tested in different computers to compare its performance in general purpose CPUs and embedded systems.

Type	Processor	RAM	GHz	OS	Language
PC	Intel Core 2 Q8300	4 GB	2.5	Windows 7 Ubuntu 12.04	C++
Industrial PC	Intel Atom N270	1 GB	2	Ubuntu 11.10	C++
Embedded HW 1	Zynq ARM Cortex-	512 MB	800 MHz	Specific Linux	C++, VHDL
Embedded HW 2	Apple A4 to A6X	512 MB – 1 GB	1 - 1.4	iOS	C++, NEON

Table 2: HW platforms used to test the described modules.

The PC has been used for design and debugging purposes, while the others have actually been used as final HW platforms. The industrial PC has been used for large vehicles, which had not hard space constraints and also could afford the price of a non-ARM processor. The embedded HW both use ARM architecture, although each one being very specific. The first one is part of the Xilinx's Zynq chip, which contain ARM – Cortex A9 and FPGA, running a personalized Linux. The other one is part of the Apple AX family of ARM for iPad, iPhone, using iOS. The following table summarizes the processing times associated with the different modules we have described in the paper running on the different mentioned platforms.

Method	PC	Industrial PC	Embedded HW 1	Embedded HW 2
Lane departure warning	0.5 ms	12 ms	18 ms	-
Vehicle detection	12 ms	70 ms	-	-
Pedestrian detection	30 ms	-	-	-
Driver drowsiness detection	2 ms	-	-	30 - 70 ms

Table 3: HW platforms used to test the described modules.

As shown, some methods have been tested already in ARM-based architectures, reaching real-time performances (below 40 ms for 25 fps input video) in most cases. The detection of pedestrian has not been yet reached that maturity level, but it is in our roadmap to reach the same level of optimization as with the other modules.

Conclusions and future work

The use of computer vision methods in vehicles is becoming a reality in the market of intelligent vehicles thanks to the optimization of algorithms, the increased power of low consumption embedded HW, and also the reduced costs of cameras and optics. Many different type of systems can be included in intelligent vehicles using computer vision, being the ADAS of particular relevance. The final goal is to make intelligent vehicles fully automatic and achieve accident free traffic scenarios. Since this is not yet feasible, the next steps in the field will likely be small steps towards automation, such as the introduction of semiatuomated systems (e.g. those that take control of the vehicles only in short periods of time), but also consolidate SW and HW advances, so that more complex computer vision solutions are available at lower costs. In that sense we are working on optimizing our algorithms, using parallelization of operations at pixel level by combining ARM architectures with FPGA and VHDL based implementation of algorithms. This will dramatically decrease the computational cost of algorithms and make them available for a wider range of cheap devices and thus to a wider market.

Acknowledgement(S)

The works described in this paper have been partially supported by the program ETORGAI 2011-2013 of the Basque Government under project IEB11. This work has been possible thanks to the cooperation with Datik – Irizar Group for their support in the installation, integration and testing stages of the project. Also, some of the works have been partially supported by the European Commission 7th Framework Program under project TRANSPORT & GALILEO: TAXISAT.

References

1. M. Nieto, J. Arrospide, and L. Salgado, "Road environment modeling using robust perspective analysis and recursive Bayesian segmentation," Machine Vision and Applications, vol. 22, issue 6, pp. 927-945, 2011
2. M. Nieto, A. Cortés, O. Otaegui, J. Arróspide, and L. Salgado, "Real-time lane tracking using Rao-Blackwellized particle filter," Journal of Real-Time Image Processing, (accepted), 2013.
3. J. Arrospide, L. Salgado: Region-dependent vehicle classification using PCA features. ICIP 2012: 453-456
4. Z. Sun, G. Bebis, and R. Miller, "On-Road Vehicle Detection Using Gabor Filters and Support Vector Machines," in Proc. IEEE International Conferenceon Digital SignalProcessing, 2002.

5. C. R. Del Blanco, F. Jaureguizar, and N. García (2010). Visual tracking of multiple interacting objects through Rao-Blackwellized Data Association Particle Filtering. In Proc. IEEE International Conference on Image Processing, pp. 821-825..

6. P. Viola and M. J. Jones, "Robust Real-Time Face Detection," Int. J. Comput. Vision, vol. 57, no. 2, pp. 137-154, 2004

7. A. D. Jepson, D. J. Fleety, and T. F. El-Maraghi, "Robust Online Appearance Models for Visual Tracking," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, vol. I, pp. 415-422, 2001.

8. F. Dornaika and F. Davoine, "On Appearance Based Face and Facial Action Tracking," IEEE Transactions on Circuits and Systems for Video Technology, vol. 16, no. 9, pp. 1107-1124, 2006.

9. J. Ahlberg, "CANDIDE-3 - an updated parameterized face," Report No. LiTH-ISY-R-2326, Dept. of Electrical Engineering, Linköping University, Sweden, 2001.