Lane-Change Decision Aid System Based on Motion-Driven Vehicle Tracking

Javier Díaz Alonso, Eduardo Ros Vidal, Alexander Rotter, and Martin Mühlenberg

***Abstract*—Overtaking and lane changing are very dangerous driving maneuvers due to possible driver distraction and blind spots. We propose an aid system based on image processing to help the driver in these situations. The main purpose of an overtaking monitoring system is to segment the rear view and track the overtaking vehicle. We address this task with an optic-flow-driven scheme, focusing on the visual field in the side mirror by placing a camera on top of it. When driving a car, the ego-motion optic-flow pattern is very regular, i.e., all the static objects (such as trees, buildings on the roadside, or landmarks) move backwards. An overtaking vehicle, on the other hand, generates an optic-flow pattern in the opposite direction, i.e., moving forward toward the vehicle. This well-structured motion scenario facilitates the segmentation of regular motion patterns that correspond to the overtaking vehicle. Our approach is based on two main processing stages: First, the computation of optical flow in real time uses a customized digital signal processor (DSP) particularly designed for this task and, second, the tracking stage itself, based on motion pattern analysis, which we address using a standard processor. We present a validation benchmark scheme to evaluate the viability and robustness of the system using a set of overtaking vehicle sequences to determine a reliable vehicle-detection distance.**

***Index Terms*—Collision-avoidance systems, lane-change decision aid systems, machine vision, safety.**

I. Introduction

THIS PAPER was carried out within the framework of the ECOVISION Project (http://www.pspc.dibe.unige.it/~ecovision/). One of the objectives of the ECOVISION consortium is to develop precognitive visual models for use in real-world environments. Within this context, we describe here a side-view-mirror, blind-spot monitor, and driver-distraction alert system in which motion processing can provide useful cues for the motion pattern segmentation of an overtaking vehicle.

Although the use of an image processing system in a vehicle is not straightforward, as it requires complex algorithms and high computational power, the use of video cameras with computer vision techniques offers an attractive alternative with multiple applications in traffic scenarios [1]–[4]. Most of the current techniques concerning vehicle tracking usually focus on road-traffic monitoring for incident detection using static cameras [1], [5]. Onboard cameras and efficient processing devices open the door to advanced vision systems for driving assistance [2]. We focus on overtaking scenarios, which are an important cause of accidents. In fact, the European Commission is studying specific actions to eliminate the blind spot on motor vehicles [6]. A lane-change assistant would recognize vehicles in the blind spot and warn the driver if he/she starts changing lanes. A standardization committee has been formed to study the subject of a Lane-Change Decision-Aid System (*LCDAS*). To evaluate this system, we have applied the norms set out in a preliminary draft of the International Organization for Standardization (ISO) (ISO/TC204/WG14/N40.27).

Over the last few years, driver-assistance systems have become a priority with car manufacturers. Nowadays, onboard cameras and image processing platforms are in high demand to help in lane keeping and to detect and avoid collisions from fast-approaching or lane-changing vehicles [7], [8], even to the extent of including stereo cameras [9], [10]. Although radars constitute a very valid alternative, such as in [11], other authors propose sensor fusion schemes, such as radar and stereovision fusion [12] or laser scanner and stereovision [13], as effective methods for avoiding collisions. Some companies, such as Mobileye N.V. [8], Volvo [14], and Fico S.A. [15], have developed aids to lane-change decision making and claim that their products work properly, but no reports on their technical details or the performance of these approaches have been published yet. Their initiatives only cover the application itself but with no benchmark information to validate their systems, making it impossible to compare the different approaches. There is also a product based on radar sensors [16] to solve the same problem, but again, it lacks any validation information.

In our approach, we use a monocular camera within the car, which allows us to detect the overtaking vehicle by using an optical-flow algorithm. This system can be used to generate alarm signals to the driver. The use of monocular cameras as well as motion information has been highlighted elsewhere by other authors as a useful strategy to solve dangerous driving situations [2], [17]. The optical-flow-driven scheme has several properties that can be very useful for car segmentation. Basically, by focusing on the optical-flow field, we should find static objects and landmarks moving backwards (due to our ego-motion) and the overtaking vehicles moving forward toward our vehicle. Nevertheless, there are several artifacts, such as

Manuscript received September 21, 2006; revised August 5, 2007, October 18, 2007, and November 15, 2007. This work was supported in part by the VI EU research framework through the European Projects ECOVISION under Grant IST-2001-32114 and DRIVSCO under Grant IST-016276-2, by the National Spanish DEPROVI Grant DPI2004-07032, and by Junta de Andalucía Project under Grant P06-TIC-02007. The review of this paper was coordinated by Dr. S. Anwar.

J. D. Alonso and E. R. Vidal are with the Computer Architecture and Technology Department, ETSI Informatica, University of Granada, 18071 Granada, Spain (e-mail: jdiaz@atc.ugr.es; eros@atc.ugr.es).

A. Rotter and M. Mühlenberg are with Advanced Development Systems and Products, Department GE-ADS, Hella KGaA Hueck and Company, 59552 Lippstadt, Germany (e-mail: Alexander.Rotter@hella.com; Martin.Muehlenberg@hella.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2008.917220

perspective deformation and camera vibration, that can affect the performance of the system. The proposed scheme needs to address these kinds of artifacts.

The application involves significant challenges. Most of the contributions developed for traffic analysis work with static cameras [1], [5], [18]. Onboard cameras considerably increase the complexity of the system, partly because the algorithm needs to deal with nonstatic scenarios (which means complex algorithms to analyze the scene) and partly because the processing frame rate becomes a critical factor for such an analysis. Onboard cameras have been used for lane tracking [19], [20] as well as in front/rear vision for obstacle avoidance [12], [17], but the application we present here focuses on a different field of view, i.e., the side-view mirror. It is important to emphasize that we have to deal with such important factors as perspective deformation [21], and to satisfactorily perform, the proposed system needs to overcome this problem.

One important implementation issue concerns the codesign strategy, i.e., deciding the software/hardware code partitioning, which will have an important impact on the final flexibility of the system and its cost. The working scheme that we adopted is composed of two very different stages. In the first step, we customized an FPGA device (to be used with embedded systems) for real-time motion processing [22]. The chosen optical-flow scheme uses a gradient model based on the classical approach of Lucas and Kanade [23], [24]. As mentioned in [22], this model achieves satisfactory optical-flow accuracy using affordable hardware resources. We also used a high-level hardware description language (Handel-C, cf. [25]), which allows us to describe hardware using high-level (C-like) algorithmic structures. This makes it easy to decide the critical code to be implemented on a customized *digital signal processor* (DSP). In the second step, based on the previous motion-salience map, we combined Kalman filtering techniques with appropriate filtering operations to compensate the effects of perspective deformation to arrive at a reliable estimation of a vehicle's position in the scenario. Furthermore, this stage, which is now run on a standard processor, has been carefully designed to allow efficient implementation on an embedded processor in the same FPGA chip to get a standalone embedded system (system-on-a-chip).

II. System Description

What is the aim of the system? The system should warn the driver of impending critical situations during a lane change. Critical situations occur in different possible scenarios.

1) Another vehicle is beside the lane-changing vehicle in the so-called blind spot, and the driver does not realize that his/her lane change would cause a critical situation.
2) A vehicle is coming up from behind at relatively high speed, which would also result in a dangerous situation if the driver were to change lanes.
3) The absent-minded driver begins to change lanes without noticing that an overtaking vehicle is approaching.

Within this context, we are only interested in the approaching vehicle closest to us, and therefore, the local search is performed, starting within the right-hand area of the image (for the sake of clarity, we only consider right-hand driving with the steering wheel on the left and left-hand overtaking). We are interested in detecting the vehicle as soon as possible and not losing track of it, particularly when the vehicle is close to us. Furthermore, the proposed application needs to direct alarm signals to the driver to prevent an accident. Therefore, we estimated the vehicle's position and the confidence level. This facilitates the generation of the alarm signal.

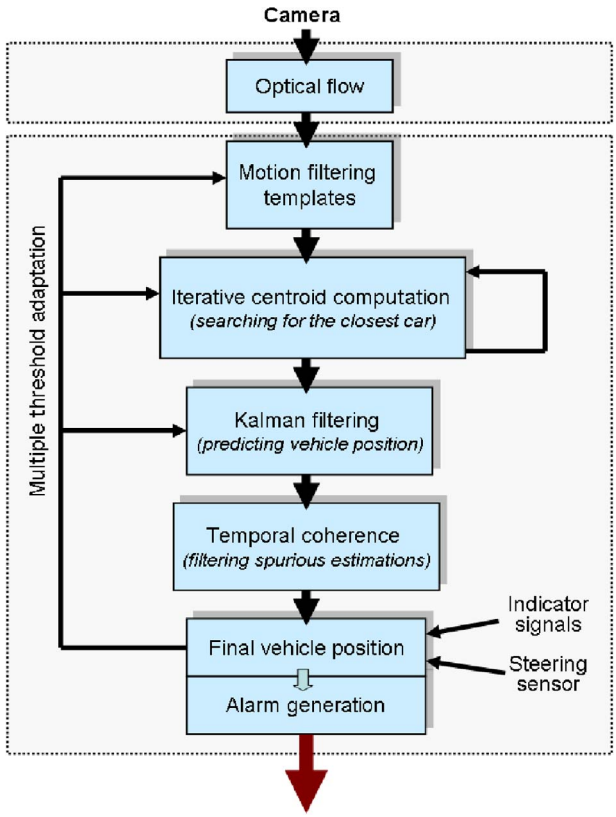


Fig. 1. System functional blocks. Note that the different thresholds dynamically adapt according to the evolution of the recent scenario. Final alarm decision uses overtaking-car position, car-steering sensors, and blinkers.

We are currently working on the hardware implementation of the whole system, and as a result, the algorithms we use must take into account the target hardware. For this purpose, we have taken into account several parameters such as the type of arithmetic operations, bit width, memory requirements, and so on. For extra details about the hardware system architecture, a preliminary system is described in [26].

The next sections describe the processing stages of the system. They are schematized in Fig. 1.

A. Pattern Selection and Optical-Flow Filtering Templates

Optical flow is a well-known method used for motion-based segmentation, cf. [27], and according to our previous results [28], we have validated this approach for the onboard segmentation of overtaking vehicles. In our system, some simplifications can be made because of the structure of the problem addressed. We only consider rightward movements. During overtaking maneuvers, the approaching vehicle is moving to the right-hand side of the side-view image, so we do not need to consider leftward velocities (Fig. 2). Wrong velocity estimations of the optical flow are frequent, so we need to clean up these erroneous

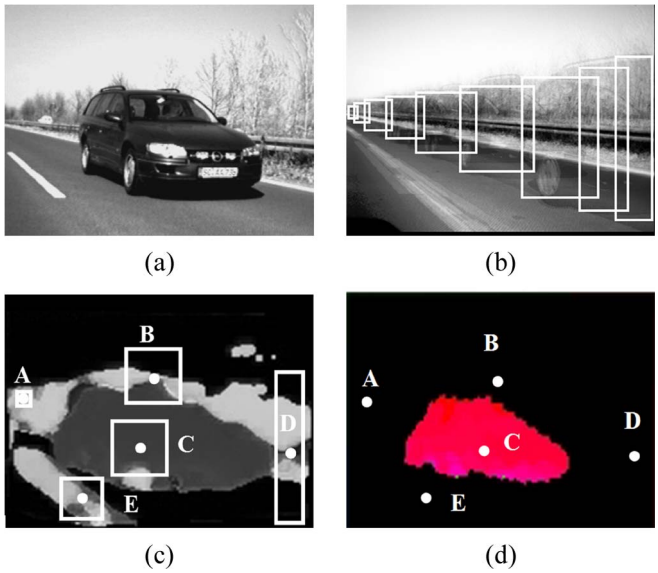


Fig. 2. (a) Overtaking car sequence. (b) Templates for filtering the optical flow based on rectangles to estimate the car's position. (c) Original optical-flow saliency map (grey levels indicate velocity orientations: Grey encodes rightward motion, and light pixels encode leftward motion). Several template examples are also shown. [Note that the template surrounding pixel D is not square because the rest of the template lies outside the image and can, therefore, be neglected. This is indicated in (1) and (2).] (d) Final result (segmented car) after applying the filters. This figure only shows the active pixels that are used for centroid computation. In pixels A and D, velocity values indicate leftward motion, and therefore, these pixels are rejected for vehicle-position estimation because they do not fulfil the velocity direction rule. In pixels B and E, although the estimated velocity is in the right direction, there are not enough active pixels inside the filtering templates. Finally, pixel C is used for centroid computation because there is a percentage above α of pixels with rightward motion inside this template. Note that different sizes and shapes of template are used to correct the deformation in the side-mirror perspective.

patterns in the next few steps. We define an image coordinate system placed at the bottom left-hand corner of the image, with positive values growing from left to right and from bottom to top. The coordinate x stands for the pixel column and y for the pixel row. If V_x is the x component of the velocity of one pixel, V_y is the y component, and $V_{\min}$ is a threshold that represents the minimum reliable velocity component module, the pixels of the scene are taken into consideration only if their velocity verifies $V_x > V_{\min}$ and $|V_y| < V_x$ (with a typical $V_{\min}$ value $= 0.25$ pixels/frame).

This allows us to only consider rightward motion and takes into account the focus of expansion in the side-view mirror, which apparently produces both vertical and horizontal patterns for the moving objects.

The proposed system uses templates that filter the motion-saliency map. We use them to clean up the optical flow of the previous stage and maintain only the more reliable data to compute the position of the overtaking vehicle. The key idea is to convolve the image with uniform rectangular kernels of different sizes. They collect information regarding homogeneous areas with well-structured motion, which has been filtered according to the condition of the previous paragraph. The template forms are rectangles that grow along the x-axis toward the right-hand side of the image, where the vehicle is expected to be larger (cf. Fig. 2). Each spatial position has an associated template that establishes the minimum number of points and the neighborhood area to carry out the search. If we call H_r and V_r the horizontal and vertical image resolution (width and height of the images) and N the number of regions into which we divide the image (typically $N = 20$ in our experiments), the size and shape of the template change every H_r/N columns. For each image pixel, their corresponding horizontal and vertical template sizes T_x and T_y are computed according to (1), shown at the bottom of the page, where we have defined the following:

- x_n and y_n: the pixel column and row image position, where we use values normalized to one (dividing by H_r and V_r sizes of the image);
- $Round\ ()$: rounding to the nearest integer operation;
- Δ: defined as $1/N$;
- T_{x0} and T_{y0}: initial horizontal and vertical template sizes (corresponding to the first image columns on the left and typically of two pixels).

Equations (1) and (2), shown at the bottom of the page, define the template sizes in units of pixels. They have been experimentally determined to roughly compensate for any deformation in the side-view mirror perspective. For an analytical study of this perspective and the way to determine perspective parameters, see [21]. A study concerning perspective deformation and different techniques available to reduce its effect when tracking vehicles can be found in [29] and [30].

For each pixel, their value, after convolving with the template, will be nonnull only if the number of pixels with rightward movement exceeds a predefined threshold that we have called template threshold (T_T) and is computed according to

$$T_T = \alpha \cdot T_x \cdot T_y \tag{3}$$

where α is a constant factor that represents the percentage of nonnull pixels in the template neighborhood (with a typical value of 0.5). Pixels that pass to the next processing stage are called active pixels (only these pixels will not have null values). An illustrative example of how the templates function is shown in Fig. 2. This scheme fits quite well into specific

$$T_x = \begin{cases} T_{x0} + round(x_n/\Delta), & \text{if } x_n < 0.25 \\ T_{x0} + round(0.25/\Delta) + 2 \cdot round\left((x_n - 0.25)/\Delta\right), & \text{if } x_n \geq 0.25 \text{ and } x_n < 0.75 \\ T_{x0} + 3 \cdot round(0.25/\Delta) - 2 \cdot round\left((x_n - 0.75)/\Delta\right), & \text{if } x_n \leq 0.75 \end{cases} \tag{1}$$

$$T_y = \begin{cases} T_{y0} + round(x_n/\Delta), & \text{if } x_n < 0.25 \\ T_{y0} + round(0.25/\Delta) + 2 \cdot round\left((x_n - 0.25)/\Delta\right), & \text{if } x_n \geq 0.25 \text{ and } x_n < 0.75 \\ T_{y0} + 3 \cdot round(0.25/\Delta), & \text{if } x_n \leq 0.75 \end{cases} \tag{2}$$

hardware because the required operations are pixelwise and can be implemented as convolvers and comparators, producing binary outputs.

After template filtering, we count the remaining number of active pixels (NAPs). A vehicle (an overtaking truck for instance) will pass to the next stage if it has enough of these active pixels. A simple centroid of active pixels can be computed to estimate the vehicle's position. Although this centroid computation of the saliency map gives us an estimation of the vehicle's position, this is correct only for continuous overtaking. Some more complex and realistic situations still need to be solved.

1) Static overtaking: An overtaking vehicle seems to stop (and its optical flow vanishes) because it maintains the same velocity as the vehicle being overtaken. In this situation, we need to maintain the vehicle's estimated position for a certain time.
2) Multiple vehicle overtaking: This is a very common situation on highways that needs to be solved.

Another subject to address is the minimum number of overall valid pixels that gives us reliable vehicle-position estimations. After template filtering, a small number of pixels remain active (nonnull values) and represent our saliency map. They correspond to homogeneous areas with basically rightward motion. When there are no vehicles in the sequence, or they maintain the same speed as our own vehicle, no valid data should appear in the saliency map. We use a second confidence threshold called *active pixel threshold* (T_{AP}) after template filtering, which is experimentally determined using the overtaking vehicle sequence database provided by Hella KGaA Hueck and Company [31] (see the Appendix for typical values). This threshold represents the minimum NAPs in the saliency map to keep only reliable features active (i.e., the minimum NAPs to reject spurious artifacts from the camera or overall motion). The NAPs represent the number of confidence points used to estimate the vehicle's position. The threshold T_{AP} is dynamically adapted according to the system's recent record using a threshold function that linearly decreases with time in the absence of inputs (enhancing the system sensibility) and increases when a high number of inputs are presented (improving the system's reliability). This threshold also varies according to the vehicle's estimated position using our *a priori* knowledge about the mirror perspective deformation. Higher thresholds are used in the right-hand area of the image, where vehicles are expected to be larger. Furthermore, the motion extracted in this area is noisier because speeds are higher. The method used to compute this threshold is

$$T_{\mathrm{AP}}(k) = T_{AP0} + x'_{\mathrm{est}} \cdot T_{AP0}^2 + (C(k-1)-1) \cdot T_{\mathrm{AP}}(k-1)/4 \quad (4)$$

where k is the frame number, x'_{est} is the normalized horizontal vehicle-position estimation at this frame k (computed as a centroid of the active pixels), and C is a binary variable that stands for the confidence value computed from

$$C(k) = \text{NAPs} > T_{\mathrm{AP}}(k). \quad (5)$$

Note that this threshold is computed for the case of only one overtaking vehicle presented in the scene. For a multiple overtaking scenario, this threshold value is iteratively refined as described in Section II-C.

B. Solution for Static Overtaking: Kalman Filtering

We need to use a memory system to retain the vehicle position when it remains stationary relative to our vehicle. Traditionally, Kalman filtering has proved to be satisfactory in resolving many problems involved in predicting the position of moving targets [32], [33] and is even useful for object motion prediction and segmentation [34]. It is also advisable because of the inherent latency of the system's processing. For an introduction to Kalman filtering theory, see [35].

Although the proposed platform can compute 25 frames/s, the optical-flow processing unit has a latency of three frames. This means that the estimated position of the vehicle undergoes a short delay with respect to its real position. This is not a problem for low relative velocities, but when the velocity is high, it might result in the system underestimating the vehicle's position. The capability of Kalman filtering to predict position allows us to overcome the artifact produced by this inherent processing latency, thus increasing the system's reliable detection distance.

As far as hardware feasibility is concerned, we have used linear Kalman filter equations that basically act as a short-term memory system with prediction capability.

After optical-flow postprocessing, we can estimate from the saliency map the vehicle position that we note at frame k as $x'_{\mathrm{est}}(k)$, $y'_{\mathrm{est}}(k)$ by means of centroid computation. We can also compute the average velocity from the optical-flow values noted henceforth as $v^x_{\mathrm{est}}(k)$, $v^y_{\mathrm{est}}(k)$. Kalman filtering can be used to refine the vehicle position, leading to a more accurate estimation, which is written as $x_{\mathrm{est}}(k)$, $y_{\mathrm{est}}(k)$. This can be done in the following way. Using the standard Kalman filter nomenclature, the process and measurement model we use can be described by

$$\begin{aligned} s_k &= As_{k-1} + Bu_k + \xi_{k-1} \\ s_k &= \begin{pmatrix} x_{\mathrm{est}}(k) \\ y_{\mathrm{est}}(k) \\ v^x_{\mathrm{est}}(k) \\ v^y_{\mathrm{est}}(k) \end{pmatrix} \\ A &= \begin{pmatrix} g_m & 0 & g_v & 0 \\ 0 & g_m & 0 & g_v \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \xi_{k-1} &= \begin{pmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \\ \xi_4 \end{pmatrix} \\ u_k &= \begin{pmatrix} x'_{\mathrm{est}}(k) \\ y'_{\mathrm{est}}(k) \\ 0 \\ 0 \end{pmatrix} \\ B &= \begin{pmatrix} (1-g_m) & 0 & 0 & 0 \\ 0 & (1-g_m) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{aligned} \quad (6)$$

$$
\begin{aligned}
Q_k &= \sigma_Q^2 \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
z_k &= H s_k + \mu_k \\
H &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
\mu_k &= \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{pmatrix} \\
R_k &= \sigma_R^2 \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \qquad (7)
\end{aligned}
$$

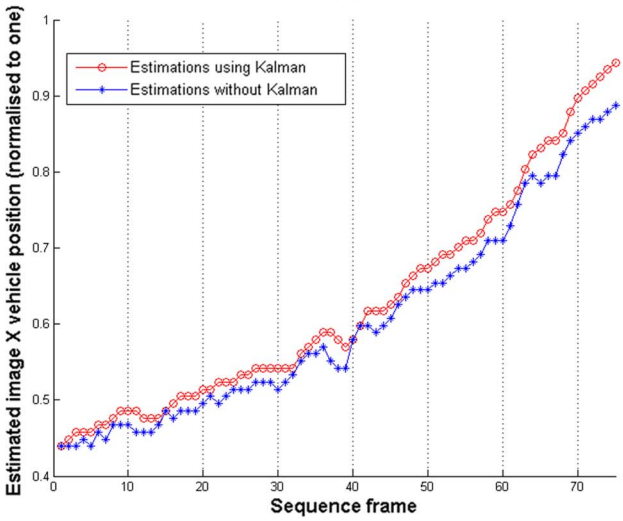
The system state s_k is described by matrices A and B in (6). Matrix B constrains our parameter prediction according to the current position measurement, i.e., the state control input u_k. The system state also depends on the position and velocity estimations at the previous instants, which are represented in matrix A. Dependencies are modeled using the position memory gain parameter g_m and the velocity gain parameter g_v. These parameters are constraints that imply smooth velocities and vehicle position. For the measurement model, which is described by z_k, we use (7). Vectors ξ and μ represent the process and noise measurement, respectively. They are supposed to be independent of each other and modeled via a random Gaussian white-noise vector of zero means. They have diagonal covariance matrices Q_k and R_k, which are also defined in (6) and (7) with σ_Q^2 and σ_R^2 as model parameters.

This model makes the assumption that the velocity is constant between two frames and that any acceleration of the object will be seen as noise. The range and typical values of the parameters of the Kalman filters are set out in the Appendix. An illustrative example of the properties of the Kalman filter is shown in Fig. 3.

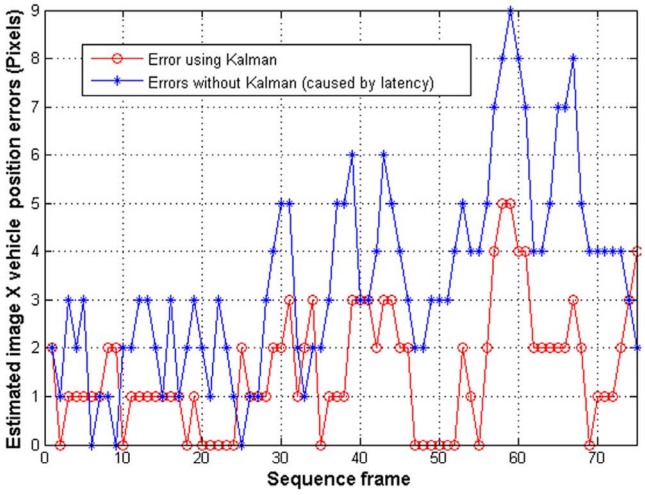
Fig. 3 shows the two main advantages of the Kalman filter in tracking vehicle position. First, the position estimation is stabilized, avoiding spurious errors. This is graphically shown by the smoother curve for the Kalman-based tracking [Fig. 3(b)]. We can quantitatively evaluate this effect just by computing the standard deviation of the position estimation derivative. We find a derivative standard deviation of 1.06 pixels/frame for non-Kalman tracking and a value of 0.78 for the Kalman-based one. We conclude from this that the Kalman filter helps to stabilize the data and reduce spurious measurement variations.

The second advantage of this technique is its prediction capability. Since the system provides the vehicle position with a latency of three frames, the Kalman filter is used to predict the position three frames ahead, i.e., to estimate the current vehicle position. To analyze the Kalman filter prediction capability, we computed offline the vehicle position without any latency with and without Kalman filtering. We delayed these estimations by three frames to take the latency of the system into account and compared the results with the estimations with neither latency nor Kalman filtering. These values have been taken as reference positions. We estimated the error between the delayed

(a)



(b)



(c)

Fig. 3. Kalman filter refining the estimation of the vehicle position. (a) Image taken from an overtaking sequence of 75 frames of a car driving in a straight line with a closing speed of 5–10 m/s (the checkered square indicates the estimated position of the vehicle). (b) Estimated vehicle position (x-axis) using (circles) Kalman filtering and (stairs) without Kalman filtering. Note that Kalman-computed estimations predict a forward position due to their prediction capability. (c) Error estimation. We estimate the vehicle position offline and compare our estimation with Kalman and without Kalman filtering, both with a delay of three frames (simulating latency effect). This shows that the Kalman technique effectively helps to reduce latency errors.

estimation and the original estimation with and without Kalman filtering [Fig. 3(c)]. This graph clearly shows that latency error is reduced thanks to the use of Kalman filtering. Quantitatively, we obtained an RMS error of 3.97 pixels and a mean error of 3.4 for the system without a Kalman filter. With this filter, the RMS error was reduced to 1.98 and the mean error to 1.53 pixels. Similar results were achieved for the rest of the overtaking vehicle sequences. This confirms that the Kalman filtering technique effectively helps to reduce the latency problem. This translates into early detection of the overtaking vehicle, thus improving system performance.

C. Solution for Multiple Vehicle Overtaking: Iterative Process

For our application, a multitarget tracking system is unnecessary. We only need to know whether there is at least one vehicle in a potentially dangerous situation. We use an iterative computation with several steps to compute the vehicle's position. In the first step, we use all the saliency map points of the whole image to give the estimated position of the vehicle, which will be the correct position if there is only one present. When there are several targets in the scene, however, the main goal is to detect the position of the vehicle closest to us. Therefore, we focus on a restricted image region, i.e., the right-hand area of the image, using the computed centroid position as the left-hand image boundary. We try to calculate a centroid of this restricted area in the image if we have significant features (i.e., if the number of elements of the corresponding saliency map is above T_{AP}); otherwise, we take the value calculated in the larger image region. We can repeat this computation several times until the estimation converges, or we can use a limited number of iterations. For our system, we have used only three iterations to get adequate results.

The main operations can be described as follows, where N_i stands for the number of iterations, the function $Estimated\_vehicle\_position()$ computes the centroid of the saliency map pixels, and the function $Compute\_T_{\mathrm{AP}}(x)$ evaluates this model threshold at the centroid position. Note that, as shown in the pseudocode, the value of T_{AP} is refined in each iteration because the searching area is modified at each stage. This is consistent with the scenario geometry described in Section II-A and is necessary to take into account the mirror perspective deformation.

Iterative tracking pseudocode

$search\_area = 1$: H_r;
For $Number\_of\_iterations = 1$ *to* N_i
  $(x, y, \mathrm{NAPs}) = Estimated \quad vehicle\_position$
    $(filtered\_image, search\_area)$;
  $Compute\_T_{\mathrm{AP}}(x)$;
  if $(\mathrm{NAPs} > T_{\mathrm{AP}})$
    $PositionX = x$;
    $PositionY = y$;
    $Search\_area = (x -$
      $number\_of\_image\_columns/8)$:
      (number_of_image_columns);
  else
    End_loop;

In our experiments, a number of iterations N_i between 1 and 4 performed well. As stated above, a good tradeoff was experimentally determined for three iterations.

After applying all these operations, the final saliency map contains a limited set of active and reliable pixels, which are used for centroid computation. Their position is then marked as a checkered square in the image (cf. Fig. 4, containing a qualitative example with some frames of an overtaking sequence). The car labeled (1) is tracked (frame a). Once it overtakes another vehicle, the vehicle estimation searches for a new vehicle (frame b). The car labeled (2) is found and tracked (frames c and d). When this vehicle has overtaken another, the system looks for the next car and finds the one labeled (3) (frame e).

Note that, due to perspective deformation, closer vehicles have large NAPs, meaning that in a multiple overtaking scenario, the estimation of the position of the overtaking vehicle (which is computed as a centroid of the active pixels) is not placed in the middle of the vehicle position but biased to the right-hand side of the image. The main consequence of this is that an iterative tracking not using active pixels at the left-hand side of the estimated position is able to track the closest vehicle, regardless of the number of overtaking vehicles on the scene or their position in the image. This procedure has been outlined in the iterative tracking pseudocode above.

D. Confidence Measurement Estimator Discussion

After the optical-flow filtering step, the resulting image only contains reliable points (active pixels) for centroid computation. If there are only a few points remaining, no reliable information can be obtained, and no estimation can be arrived at. The NAPs threshold (T_{AP}) also varies with the evolution of the system, as discussed in Section II-A. Closer vehicles appear larger in the image due to perspective, so NAPs must increase with the estimated vehicle position moving rightward. Nevertheless, with a small T_{AP}, we can obtain a stable vehicle position and a stability signal that indicates that our system confides in the acquired data. On one hand, we should bear in mind that the main goal of our system must always be to detect overtaking vehicles; true negatives are an unacceptable system response, whereas on the other hand, we should also avoid false positives. Sometimes the optical flow is very noisy (on a bumpy road, for example), and despite the number of thresholds imposed to compensate for them, some mistakes appear (such as the isolated points shown in Fig. 5). These spurious estimations should not be allowed to trigger the alarm signal, or they would upset the driver's confidence in the monitoring system. Although the number of errors presented as isolated dots in Fig. 5 is very small (less than 2%), from a psychological point of view, false-positive alarms significantly affect the driver's confidence. Therefore, our system only triggers the alarm in the shaded areas in Fig. 5. The system can also benefit from methods for monitoring the driver's state of vigilance or fatigue (cf. [36] and [37]), and information about this can efficiently be used to either inhibit or increase the generation of the alarm signal. This is important in reducing the number of unnecessary

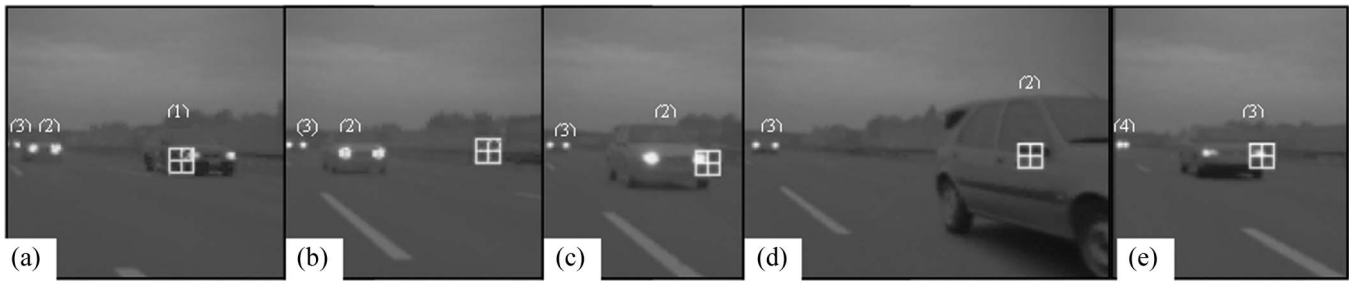


Fig. 4. Multiple cars overtaking on a highway on a cloudy day [sequence frames from (a)–(e)]. Number labels have been added at the top of the vehicles to clarify the process. The reliable position is automatically marked by the system in the figures with a checkered square.

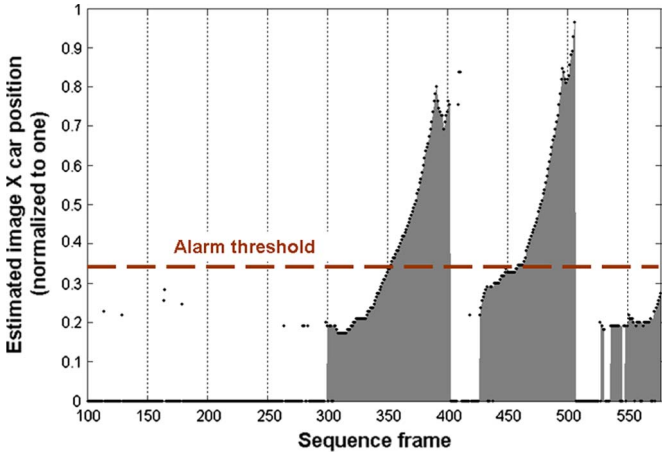


Fig. 5. Example of confidence vehicle-tracking estimators for an overtaking sequence of three cars. The vertical axis represents the estimated horizontal car position. The values have been normalized by dividing each value by the number of image columns to allow data representation, regardless of the image resolution. The horizontal axis corresponds to the time dimension (image sequence frame). The target vehicle closing speed is about 15 m/s. The black dots represent the estimation of the car's position using centroid computation from optical-flow data and Kalman filtering (a value zero of the vertical axis is used for nonreliable car position estimations). There are still some isolated errors. The filled-in areas represent high confidence frames in which the car's position is reliably estimated (i.e., exceeding the confidence thresholds), and the isolated points represent unreliable data. The horizontal line represents the limit at which the system must generate the alarm signal if necessary (due to a steering maneuver, for instance).

alarms and raises an interesting topic to be explored in future studies.

To solve this situation, a very simple hardware-friendly scheme was adopted. Using a temporal memory window of T_{MW} frames (with typically $T_{\mathrm{MW}} = 7$) and median filtering of the confidence value C described in (5), we finally achieved high confidence, high stability, and high reliability in deciding whether we are in a critical situation. This is represented in Fig. 1 by the block labeled *Temporal coherence*. The temporal persistence of the stimulus allows us to reject noisy inputs, and thanks to the large detection distance (provided by subpixel optical-flow configuration), we can use this median filtering without any loss of performance.

The final system output is represented by the grey areas in Fig. 5, corresponding to three different overtaking cars. These areas represent the alarm signal that a driver will see if he/she tries to steer toward the overtaking lane.

To conclude, the final system is able to robustly detect overtaking vehicles, but multiple operations are required. For the sake of clarity, a summary with a list of the system's different system typical ranges and values is set out in the Appendix.

III. System Performance Evaluation

Evaluating the accuracy and efficiency of the system for real-image sequences is not easy. A visual inspection of the results gives us some "quality hints" to evaluate the performance, but this is not a valid "quality evaluation procedure."

For our application, we used a camera mounted on the side-view mirror. We have tested the algorithm in different overtaking sequences provided by Hella KGaA Hueck and Company [31] with different vehicles and weather conditions (different light conditions, cloudy, foggy, sunny days, diverse road types, and so on). Thus, we obtained 20 sequences composed of more than 9000 frames. Our goals were the following:

1) to detect the overtaking car as soon as possible;
2) to reliably track it.

This is a complex task because if we use a very sensitive system, continuous false alerts can render the approach useless and make the driver lose confidence in the system. The next section describes the system benchmark procedure. It should be remembered here that although there are some commercial initiatives under development toward similar systems [8], [14], [15], no performance evaluation or scientific benchmarking methodology seems to have been applied to date. This makes it impossible to compare the different approaches and estimate their applicability.

A. Benchmark Methodology and System Description

The idea that the vehicle in question is the closest to us and, therefore, must be in the right-hand area of the image has important implications for our test. We are interested in detecting the vehicle as soon as we can and not losing track of it, particularly when it is close to us. We measure the distance at which reliable tracking starts to evaluate the quality of the system.

For benchmarking, special test sequences were recorded by Hella KGaA Hueck and Company [31] according to the preliminary version of the ISO standard (ISO/TC204/WG14/N40.27). Three systems are considered based on the areas they cover [see Fig. 6(a)].

1) Type I: *Blind-Spot Warning*. This system is intended to only warn about target vehicles in the adjacent zones (the zones on the left and right of the subject vehicle).

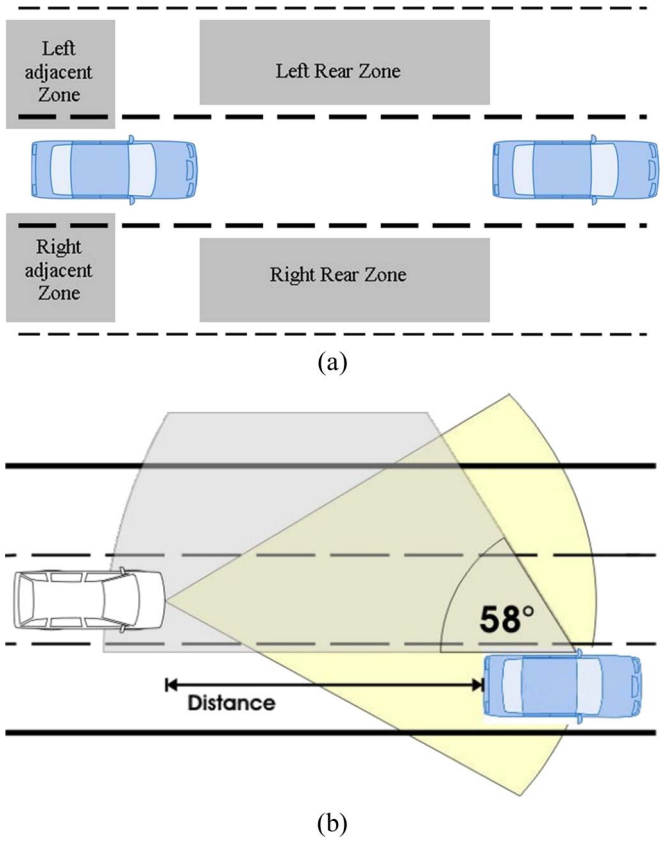


Fig. 6. (a) Car for device type classification. (b) Intercar distances using the LIDAR sensor and camera view angle to cover the blind-spot areas.

It is not required to provide warnings of target vehicles approaching the subject vehicle from the rear.

2) Type II: *Closing Vehicle Warning*. This system is intended to warn about target vehicles that are approaching the subject vehicle from the rear.
3) Type III: *Lane-Change Warning*. This type combines the blind-spot and closing-vehicle functions, i.e., a Type III system is a system that fulfils the Type I and Type II requirements.

The Type II specifications consider different closing speeds: A → 5–10 m/s; B → 15–20 m/s; C → 25–30 m/s. For the evaluation, we used two instrumented test cars: the *Target Car (TC)*, which is the overtaking car, and the *Subject Car (SC)*, which is equipped with the camera and the system described for tracking the target car. *TC* has a LIDAR sensor installed in front [38] [illustrated in Fig. 6(b)] to measure the distance between the two vehicles. Both instrumented data-acquisition systems are synchronized to match the recorded frames of *SC* with the LIDAR information of *TC* at any time. An onboard computer stores this information for offline analysis.

In the test scenario, it is possible to get image data from *SC* and, as a reference, the value of the distance between both vehicles from the LIDAR sensor of the overtaking car. Most of the recorded video streams have corresponding LIDAR-measured distances. Nevertheless, due to technical problems, only a limited number of cross-validated sequences were recorded. Day and night scenarios were tested, but we only include here the results from the day scenarios, which present the most difficult situation, since at night, the headlights of overtaking vehicles facilitate the tracking task.

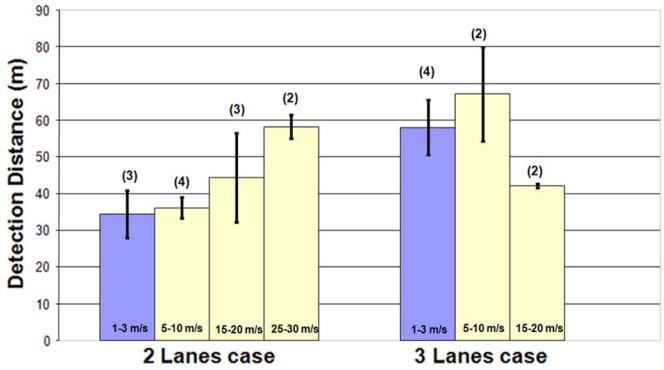


Fig. 7. System evaluation results. The average detection distances (in meters) with their typical deviation are depicted. Two different cases are considered: two-lane motorways and three-lane motorways. At the top of the bars, we include the number of cross-validated sequences used in each case.

B. Results of the Evaluation

The results from our 20 cross-validated test sequences (Fig. 7) indicate the distance between vehicles as measured using the LIDAR sensor. For each sequence, only one overtaking maneuver has been evaluated (the one done by the vehicle with the LIDAR sensor). For the remaining vehicle-overtaking maneuvers, which take place in the sequence (typically one to three or more), only qualitative tracking information is available, and therefore, we cannot use them in our benchmarking scheme. Basically, we have two different kinds of recorded sequences: one for the Type I system test (dark bars) and another for the Type II system test (light bars). In this case, three different approaching speeds are possible according to our preliminary standard (except for three lanes and 25–30 m/s, where no distance information is available due to technical problems).

Fig. 7 shows that vehicles approaching faster are reliably detected at longer distances (in the two-lane bars). This is highly desirable since *the Time-To-Contact (TTC)* is shorter in these situations, which is possible because *TC* is approaching faster, and the motion cues become significant even when *TC* is still far away.

The sequences were taken under different visibility and weather conditions. This also affects system performance (significantly in the third bar of the three-lane case in Fig. 7).

These results show the high potential for a possible application within the framework of a driver-assistance system, although further experiments with a larger number of overtaking sequences are required. With these data, we have been able to evaluate and classify the system, using the ISO draft, as nearly fulfilling the requirements for a Type-III system (lane-change warning), because we passed Type I and Type II tests, with the subtype C (relative velocities of up to 20 m/s).

IV. Conclusion

In this paper, we have described a system to track overtaking vehicles using the side-view-mirror perspective. Basically, it is implemented in two steps. First, we compute the optical

TABLE I
SYSTEM PROCESSING STAGES AND LIST OF PARAMETERS. FROM LEFT TO RIGHT, WE SHOW THE METHODS OF THE PROCESSING STAGES AND THEIR PURPOSE, THE METHOD EQUATION/ALGORITHM OF THE STAGE, THE PARAMETERS TO DETERMINE, THEIR DEFINED RANGE, AND THE VALUES USED IN OUR EXPERIMENTS

Method	Function	Description	Parameters	Parameters range	Value used
Velocity selection rule	Reject objects moving backwards	$V_x > V_{min}$ $\lvert V_y \rvert < V_x$.	V_{min}	0-1.5	0.25
Motion-filtering templates	Rigid body motion detection and perspective correction	See Equations (1), (2) and (3)	T_{x0}, T_{y0} N α	1-5 pixels 10-50 0.2-0.7	2 20 0.5
Active Pixel threshold	Fix minimum number of active pixels needed to estimate vehicle position	See Equations (4) and (5)	T_{AP0}	0.1-2% of the number of pixels of the image	0.25 % of the number of pixels of the image
Kalman filtering	Predict vehicle position to reduce latency error. Estimation stabilization	See Equations (6) and (7)	g_m g_v σ^2_Q σ^2_R	0.25-0.75 0.25-0.75 1-10 1-10	0.75 0.5 5 5
Iterative centroid computation	Closest vehicle detection for multiple overtaking scenarios	See algorithm code at section 2.3.	N_i	1-4	3
Temporal coherence filter	Reject spurious wrong measurements using the temporal persistence of the vehicle in the scenario	C(t)' = median(C, T_{MW})	T_{MW}	3-11	7

flow, and then, after a filtering stage, this motion-saliency map reliably represents vehicle points that are used to compute the overtaking vehicle's position. We implemented a hardware/software hybrid approach using a customized DSP for optical-flow computation combined with a standard processor for the tracking stages and generation of the alarm. Finally, we also applied a benchmark trial with a wide set of diverse overtaking sequences to evaluate the system's performance. Although further experiments with more overtaking sequences are required, the results shown are very promising because the system is very reliable and stable, even for very difficult image sequences in poor visibility (for some qualitative examples, cf. [26]).

From Fig. 7, we can also compute the *TTC* when the alarm signal is generated. Using the fastest velocity of each interval and taking into account the lowest detection distance (average value minus typical deviation), the worst case is 1.61 s, which was presented in the bar corresponding to two lanes and a closing speed of 15–20 m/s. Based on driver behavior studies [39], the worst reaction time for a driver is 1.5 s for braking (less if we consider that the lane-change maneuver just implies a steering action, which is more than 0.15 s faster than braking, and therefore, the reaction time becomes 1.35 s). Therefore, we believe that our system can effectively alert the driver and leave him/her enough time to react.

However, there are still some open issues for future work. 1) On three-lane roads, an overtaking vehicle in the outside lane should not generate a warning signal. This implies a distinction between overtaking maneuvers in the other two lanes. 2) In inverse overtaking scenarios, when the *SC* is overtaking the *TC*, the warning signal should be generated, since lane changing would also cause a dangerous situation. 3) A smart warning strategy (human–machine interface field) is necessary. Future work will cover these points and test the whole system on the vehicle.

APPENDIX

For the sake of clarity and to facilitate the replication of the presented approach, we have summarized the model stages and parameters in Table I, which presents the purpose (function) of the different methods applied, indicating the values of the parameters used for our study.

To determine the different values of the parameters, we made a heuristic multiparametric search. The parameter space scanning process was done via a feedforward procedure (sequentially adjusting the parameters that affect the first stages of the system) to arrive at the best system configuration sequentially throughout the different processing stages. The list of parameters shown above proved to be adequate for the complete set of experiments described in Section III. Furthermore, we found that the system performs quite stably and operates satisfactorily for most of the parameter values within the proposed ranges (Table I).

ACKNOWLEDGMENT

The authors would like to thank A. L. Tate for revising their English text.

REFERENCES

[1] J. Zhou, D. Gao, and D. Zhang, "Moving vehicle detection for automatic traffic monitoring," *IEEE Trans. Veh. Technol.*, vol. 56, no. 1, pp. 51–59, Jan. 2007.

[2] C. Demonceaux, A. Potelle, and D. Kachi-Akkouche, "Obstacle detection in a road scene based on motion analysis," *IEEE Trans. Veh. Technol.*, vol. 53, no. 6, pp. 1649–1656, Nov. 2004.

[3] M. Bertozzi, A. Broggi, A. Fascioli, T. Graf, and M.-M. Meinecke, "Pedestrian detection for driver assistance using multiresolution infrared vision," *IEEE Trans. Veh. Technol.*, vol. 53, no. 6, pp. 1666–1678, Nov. 2004.

[4] Y. Abramson and Y. Freund, "Active learning for visual object detection," Univ. California San Diego, San Diego, CA, CS2006-0871 Tech. Rep., Nov. 19, 2006. [Online]. Available. http://www.cse.ucsd.edu/Dienst/UI/2.0/Describe/ncstrl.ucsd_cse/CS2006-0871

[5] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos, "A vision-based approach to collision prediction at traffic intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 4, pp. 416–423, Dec. 2005.

[6] "Official Journal of the European Union, L 25/1-45," *On the approximation of the laws of the Member States relating to the type-approval of devices for indirect vision and of vehicles equipped with these devices, amending Directive 70/156/EEC and repealing Directive 71/127/EEC*, Nov. 10, 2003 (accessed Jul. 23, 2007). [Online]. Available: http://europa.eu.int/rapid/pressReleasesAction.do?reference=IP/04/193&format=HTML&aged=0&language=EN&guiLanguage=en

[7] *Hitachi: Image Processing Camera*. (accessed Jul. 23, 2007). [Online]. Available: http://www.hitachi.co.jp/en/products/its/product/platform/2004602_12387.html

[8] N. V. Mobileye, *Blind Spot Detection and Lane Change Assist (BSD/LCA)* (accessed Jul. 23, 2007). [Online]. Available: http://www.mobileye-vision.com/default.asp?PageID=226

[9] L. Xia and K. Fujimura, "Pedestrian detection using stereo night vision," *IEEE Trans. Veh. Technol.*, vol. 53, no. 6, pp. 1657–1665, Nov. 2004.

[10] R. Labayrade and D. Aubert, "In-vehicle obstacles detection and characterisation by stereovision," in *Proc. IEEE In-Vehicle Cognitive Comput. Vis. Syst.*, Graz, Austria, Apr. 1–3, 2003, pp. 13–19.

[11] J. Mar and L. Hung-Ta, "The car-following and lane-changing collision prevention system based on the cascaded fuzzy inference system," *IEEE Trans. Veh. Technol.*, vol. 54, no. 3, pp. 910–924, May 2005.

[12] T. Kato, Y. Ninomiya, and I. Masaki, "An obstacle detection method by fusion of radar and motion stereo," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 3, pp. 182–188, Sep. 2002.

[13] R. Labayrade, C. Royere, D. Gruyer, and D. Aubert, "Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner," *Auton. Robots*, vol. 19, no. 2, pp. 117–140, Sep. 2005.

[14] *Volvo BLIS System*. (accessed Jul. 23, 2007). [Online]. Available: http://www.mynrma.com.au/cps/rde/xchg/SID-3F5768EC-A548335E/mynrma/hs.xsl/mc_babyboomer.htm

[15] *Ficosa Digital Blind Spot Detector* (accessed Jul. 23, 2007). [Online]. Available: http://www.ficosa.com/eng/home_noticiaseventos.htm

[16] *Hella KGaA Hueck & Co. Lane Change Assistant System* (accessed Jul. 23, 2007). [Online]. Available: http://www.hella.com/produktion/HellaCOM/WebSite/Channels/AutoIndustry/Electronics/DA_Systems/Lane_Change_Assistant.jsp

[17] E. Dagan, O. Mano, G. P. Stein, and A. Shashua, "Forward collision warning with a single camera," in *Proc. IEEE Intell. Veh. Symp.*, Parma, Italy, Jun. 14–17, 2004, pp. 37–42.

[18] X. Desurmont, A. Bastide, C. Chaudy, C. Parisot, J. F. Delaigle, and B. Macq, "Image analysis architectures and techniques for intelligent surveillance systems," *Proc. Inst. Electr. Eng.—Vision Image Signal Process*, vol. 152, no. 2, pp. 224–231, Apr. 2005.

[19] N. Apostoloff and A. Zelinsky, "Vision in and out of vehicles: Integrated driver and road scene monitoring," *Int. J. Robot. Res.*, vol. 23, no. 4/5, pp. 513–538, 2004.

[20] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: Survey, system, and evaluation," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 20–37, Mar. 2006.

[21] S. Tan, J. Dale, A. Anderson, and A. Johnston, "Inverse perspective mapping and optic flow: A calibration method and a quantitative analysis," *Image Vis. Comput.*, vol. 24, no. 2, pp. 153–165, Feb. 2006.

[22] J. Díaz, E. Ros, E. M. Ortigosa, and S. Mota, "FPGA based real-time optical-flow system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 2, pp. 274–279, Feb. 2006.

[23] B. Lucas and T. Kanade, "An iterative image registration technique with applications to stereo vision," in *Proc. DARPA Image Understanding Workshop*, 1981, pp. 121–130.

[24] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vis.*, vol. 12, no. 1, pp. 43–77, Feb. 1994.

[25] *Handel-C Language Reference Manual, Version 3.1*, Celoxica, Oxfordshire, U.K., 2003. Doc. No. RM-1003-3.0.

[26] J. Díaz, E. Ros, S. Mota, and R. Agis, "Real-time embedded system for rear-view mirror overtaking car monitoring," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, vol. 4017. New York: Springer-Verlag, 2006, pp. 385–394.

[27] J. Weber and J. Malik, "Rigid body segmentation and shape description from dense optical flow under weak perspective," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 2, pp. 139–143, Feb. 1997.

[28] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, and A. Prieto, "Motion-driven segmentation by competitive neural processing," *Neural Process. Lett.*, vol. 22, no. 2, pp. 125–147, Oct. 2005.

[29] S. Mota, E. Ros, J. Díaz, S. Tan, J. Dale, and A. Johnston, "Detection and tracking of overtaking cars for driving assistance," presented at the Early Cognitive Vision Workshop, Isle of Skye, U.K., May 28–Jun. 1, 2004 (accessed Jul. 23, 2007). [Online]. Available: http://www.cn.stir.ac.uk/ecovision-ws/pdf/55.pdf

[30] S. Mota, E. Ros, J. Díaz, R. Rodriguez, and R. Carrillo, "A space variant mapping architecture for reliable car segmentation," in *Reconfigurable Computing: Architectures, Tools and Applications*, vol. 4419. New York: Springer-Verlag, 2007, pp. 337–342.

[31] Lippstadt, Germany: Dept. Predevelopment EE-11, Hella KGaA Hueck & Co. [Online]. Available: www.hella.de

[32] F. Dellaert and C. Thorpe, "Robust car tracking using Kalman filtering and Bayesian templates," in *Proc. SPIE—Intell. Transp. Syst.*, Pittsburgh, PA, Oct. 1997, vol. 3207, pp. 72–83.

[33] J. Gao, A. Kosaka, and A. C. Kak, "A multi-Kalman filtering approach for video tracking of human-delineated objects in cluttered environments," *Comput. Vis. Image Underst.*, vol. 99, no. 1, pp. 1–57, Jul. 2005.

[34] P. N. Pathirana, A. E. K. Lim, A. V. Savkin, and P. D. Hodgson, "Robust video/ultrasonic fusion-based estimation for automotive applications," *IEEE Trans. Veh. Technol.*, vol. 56, no. 4, pp. 1631–1639, Jul. 2007.

[35] G. Welch and G. Bishop, "An introduction to the Kalman filter," Dept. Comput. Sci., Univ. North Carolina Chapel Hill, Chapel Hill, NC, Tech. Rep. TR 95-041, 2002.

[36] L. M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez, "Real-time system for monitoring driver vigilante," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 63–77, Mar. 2006.

[37] J. Qiang, Z. Zhiwei, and P. Lan, "Real-time nonintrusive monitoring and prediction of driver fatigue," *IEEE Trans. Veh. Technol.*, vol. 53, no. 4, pp. 1052–1068, Jul. 2004.

[38] Hella KGaA Hueck & Co Press text, *Proximity Control Creates Safety* (accessed Jul. 23, 2007). [Online]. Available: http://www.hella-press.de/search_detail.php?text_id=433&archiv=0&language=e&newdir=eng

[39] M. Green, "'How long does it take to stop?' Methodological analysis of driver perception-brake times," *Transp. Hum. Factors*, vol. 2, no. 3, pp. 195–216, Sep. 2000.

Javier Díaz Alonso received the M.S. degree in electronics engineering and the Ph.D. degree in electronics from the University of Granada, Granada, Spain, in 2002 and 2006, respectively.

He is currently an Assistant Lecturer with the Computer Architecture and Technology Department, University of Granada. His main research interests deal with cognitive vision systems, high-performance image-processing architectures, and embedded systems based on reconfigurable devices. He is also interested in spiking neurons, biomedical devices, and robotics.

Eduardo Ros Vidal received the Ph.D. degree from the University of Granada, Granada, Spain, in 1997.

He is currently an Associate Professor with the Computer Architecture and Technology Department, University of Granada. He is currently responsible for two European projects related to bio-inspired processing schemes. His research interests include simulation of biologically plausible processing schemes, hardware implementation of digital circuits for real-time processing in embedded systems, and high-performance computer vision.

Alexander Rotter received the Diploma degree in electrical engineering from Fachhochschule Südwestfalen, Iserlohn, Germany.

He has actively participated in European Union research grants such as ECOVISION and DRIVSCO in the field of advanced vehicular systems. He is currently with Advanced Development Systems and Products, Hella KGaA Hueck and Company, Lippstadt, Germany, where he is involved in the field of driver-assistance systems.

Martin Mühlenberg received the Diploma degree in electrical engineering from Universität-GH Paderborn, Paderborn, Germany.

He has actively participated in European Union research grants such as ECOVISION in the field of advanced vehicular systems. He is currently with Advanced Development Systems and Products, Hella KGaA Hueck and Company, Lippstadt, Germany, where he is involved in the field of driver-assistance systems.