Machine Vision and Applications
DOI 10.1007/s00138-011-0381-5

ORIGINAL PAPER

A computer vision framework for the analysis and interpretation of the cephalo-ocular behavior of drivers

S. Metari · F. Prel · T. Moszkowicz · D. Laurendeau · N. Teasdale · S. Beauchemin · M. Simoneau

Received: 9 February 2011 / Revised: 21 August 2011 / Accepted: 31 October 2011

Abstract In this paper, we introduce a computer vision system specially designed for the analysis and interpretation of the cephalo-ocular behavior of drivers. The system is composed of both hardware and software components and is described in three steps. The first step is devoted to the description of the driving simulator and the developed software. The second step deals with the identification of the driver's visual search actions using computer vision. The latter are related to specific driving events such as blind spot checking and rear-view/lateral mirror verification. Based on the simulator's open module, the third step is concerned with the identification of car/road events (overtaking, crossing an intersection) and the mapping of these events with the driver's behavior. The proposed system will be used by a kinesiology research group for the evaluation and improvement of driver performances in a safe environment (driving simulator). In addition to the controlled environment, a modified version of the system also deals with real driving contexts (i.e. driving in a real car). Experimental results confirm both the robustness and the effectiveness of the proposed cephalo-ocular analysis framework.

Keywords Cephalo-ocular behavior · Driving simulator · Visual search actions · Boosted classifiers · Haar-like features · Pyramidal Lucas–Kanade method

S. Metari (✉) · F. Prel · T. Moszkowicz · D. Laurendeau
LVSN, Université Laval, Québec, QC, G1V 0A6, Canada
e-mail: samy.metari.1@ulaval.ca

F. Prel
e-mail: florent.prel.1@ulaval.ca

T. Moszkowicz
e-mail: Thierry.Moszkowicz@gel.ulaval.ca

D. Laurendeau
e-mail: Denis.Laurendeau@gel.ulaval.ca

N. Teasdale · M. Simoneau
GRAME, Université Laval, Québec, QC, G1V 0A6, Canada
e-mail: Normand.Teasdale@kin.msp.ulaval.ca

M. Simoneau
e-mail: Martin.Simoneau@kin.msp.ulaval.ca

S. Beauchemin
Department of Computer Science, The University of Western Ontario, London, ON, N6A 5B7, Canada
e-mail: beau@csd.uwo.ca

1 Introduction

During the last decades, computer vision has been used to solve a wide variety of problems in several application domains, such as visual surveillance, industrial inspection, process control and medical applications. In our context, we call upon computer vision techniques to study the cephalo-ocular behavior of drivers. Driving is a very important activity for a large portion of the population, especially among the elderly. Epidemiological studies show that this category of drivers may sometimes adopt an unsafe driving behavior. This is due to the difficulties experienced by those drivers in demanding driving situations such as overtaking, changing lanes, crossing an intersection, etc. Such driving contexts involve complex cephalo-ocular behaviors and visual search actions, such as those related to blind spot checking and rear-view/lateral mirror verification. Evaluation and improvement of the driver performance in a safe environment (driving simulator) are of great importance for road safety. The main objective of our work is the elaboration of a new computer vision system for evaluating and improving driving skills of older drivers (age between 65 and 80 years). Although the system is currently used with older drivers, it generalizes to

Published online: 14 December 2011

other categories of drivers as well. This system implies the analysis and treatment of cephalo-ocular behavior and visual search actions of older drivers in a simulated driving context. More specifically, we focus on the visual search actions related to the verification of the blind spot when overtaking and changing lanes. Experiments run in our laboratory have shown that 80% of older drivers do not check the blind spots in these contexts. Thus, by providing a new system allowing the objective detection of driving errors in a safe environment (simulator), it is expected that retraining drivers will lead to safer driving in the long term and that training received in the simulator will transfer to on-road safer driving. In the literature, relatively few research work deal with this topic. Murphy-Chutorian and Trivedi [17] introduced a new method for static head-pose estimation and a new algorithm for visual 3-D tracking. These methods are integrated into a real-time system for measuring the position and orientation of a driver's head. Trivedi and Gandhi [29] introduced a computer-vision based system devoted for the elaboration of safer automobiles. The proposed system includes three main components, namely the environment, vehicle, and driver. For the analysis of driver's behavior, the authors proposed a new algorithm devoted to the driver's view generation. This algorithm is based on the estimation of head-pose that uses the principal component analysis (PCA) for template matching. In [34], Wu and Trivedi presented an integrated approach for robustly locating facial landmark for drivers. An AdaBoost-based cascade of probability learners is used to detect the face edge primitives. The facial landmark detection approach is applied on the segmented faces. Murphy-Chutorian et al. [27] introduced a new identity and lighting invariant system designed for the estimation of a driver's head pose. The proposed system is valid for both daytime and nighttime driving conditions. FaceLAB [http://www.seeingmachines.com/product/facelab/] developped by Seeing Machines Inc. is an industrial software designed for face and eye tracking. FaceLAB allows the study of human behavior in a wide range of operational conditions and research settings. Our system is presented in three steps. The first one is devoted to the description of the driving simulator and the software components that were developed for supporting driver behavior analysis. The second one is concerned with the identification of the most important visual search actions of drivers. Based on the simulator's open module which provides access to all scenario events and car parameters, the last step deals with the identification of the car/road events and with the established mapping between the identified visual search actions and driving events. This mapping enables the analysis and interpretation of the cephalo-ocular behavior and the visual search actions of drivers and the assessment of whether or not this behavior is adapted to the driving context.

In Sect. 2 we describe the driving simulator (cf. Fig. 1) and its software components (cf. Fig. 3). Section 3 is devoted to the identification of the driver's visual search actions. The mapping between these actions and the identified driving events in the simulator is detailed in Sect. 4. Note that images and videos used in this paper can be both in color and in gray-level format.

2 The driving simulator

2.1 The hardware components

The driving simulator uses a real car from which the cockpit was kept including a steering wheel, a driver seat, rear view mirrors, brake and accelerator-pedals, speedometer and flashers (cf. Fig. 1). The car roof is removed, allowing a good aeration of the zone surrounding the driver order to avoid simulator sickness that may sometimes affect some drivers [26]. A screen on which the driving scenario is projected is installed in front of the driver and gives a good level of immersion. The driving cockpit is modified in such a way that realistic driving conditions are provided to the driver. The car velocity is indicated by the speedometer which is situated on the car dashboard as in a real driving context. The simulator software is provided by STISIM [http://www.systemstech.com/]. We have access to the simulator open module that allows us to have access to all events during the driving process. Driving scenarios are built in such a manner that the majority of driving situations faced by elderly drivers are considered. The acquisition of images of the driver is achieved using three cameras (cf. Fig. 2). These cameras are positioned in such a way as to observe the driver's head motion. A fourth camera records the scenario as it evolves on the screen and allows a visual mapping between the driving environment and the driver behavior. The four cameras are synchronized at a frame rate of 30 images per second. Since the cameras must observe the driver in low-light conditions (due to the constraint that driver immersion in the simulator must be kept at a maximum) they have good sensitivity in the "visible" infrared part of the spectrum. As we will

Fig. 1 The driving simulator

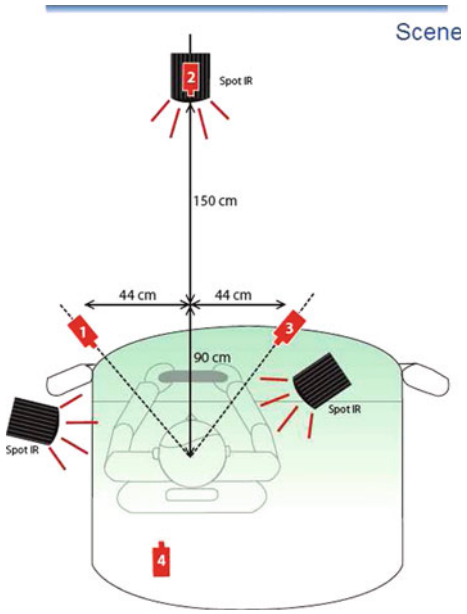


Fig. 2 Positions of the three cameras in *red* and infrared spots

see in the next sections, the central camera is very important since it covers the whole face of the driver in normal driving conditions (i.e. driving forward in a lane). The driver's face is illuminated using infrared lighting that provides good ambient illumination, avoids blinding effects, and does not affect immersion in the simulator.

2.2 The software components

The software developed for implementing the system allows the acquisition, processing and display of data in real-time as well as in playback mode. Without going through all the options, the interface is as simple as a standard video recorder (cf. Fig. 3). The main screen can display the cameras' video sequences or videos resulting from the application of processing modules in up to four areas of the interface window. Each processing module can also provide its own settings or display windows that can be opened separately to provide specific information on the ongoing experiment.

The software accepts the signal from the camera recording the scene, the three cameras recording the driver's head motion and is also interfaced to the driving simulator via a TCP/IP connection. The simulator open module has been developed specifically to act as a TCP/ IP server for the simulator's client communication modules, the "Data Compiler" module in our case (cf. Fig. 4). The client module connects to the open module and extracts from it relevant information on the car status (speed, position on the road, 3D position on the scene, etc.) and the scenario events (speed limits, signals

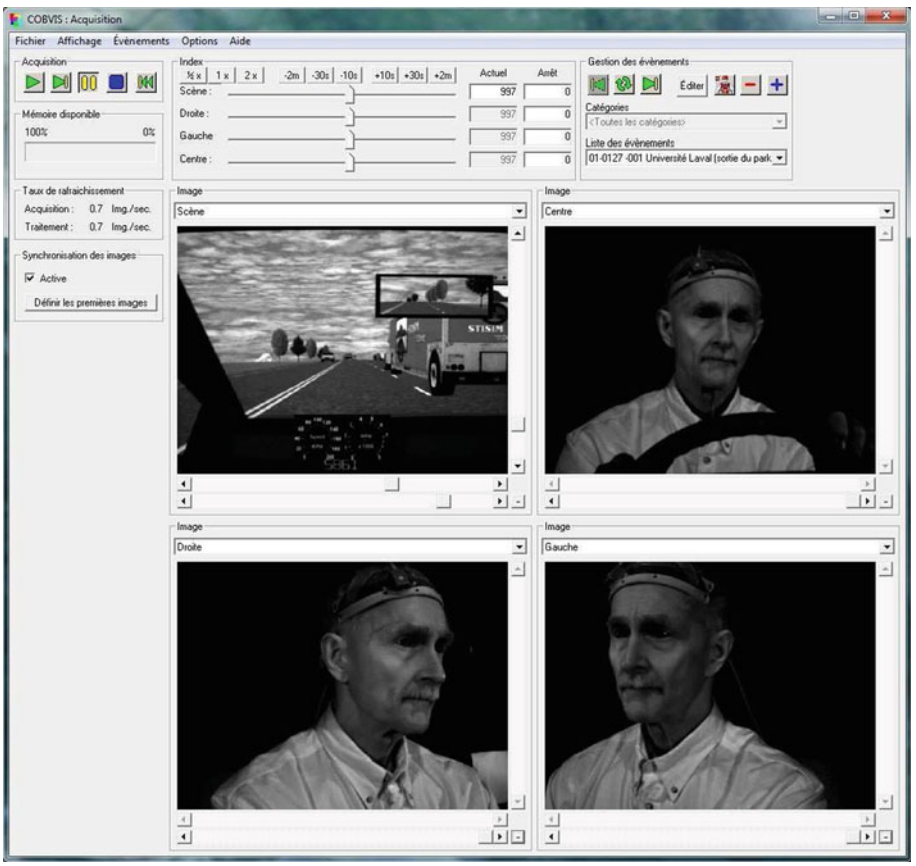


Fig. 3 Software interface (main screen)

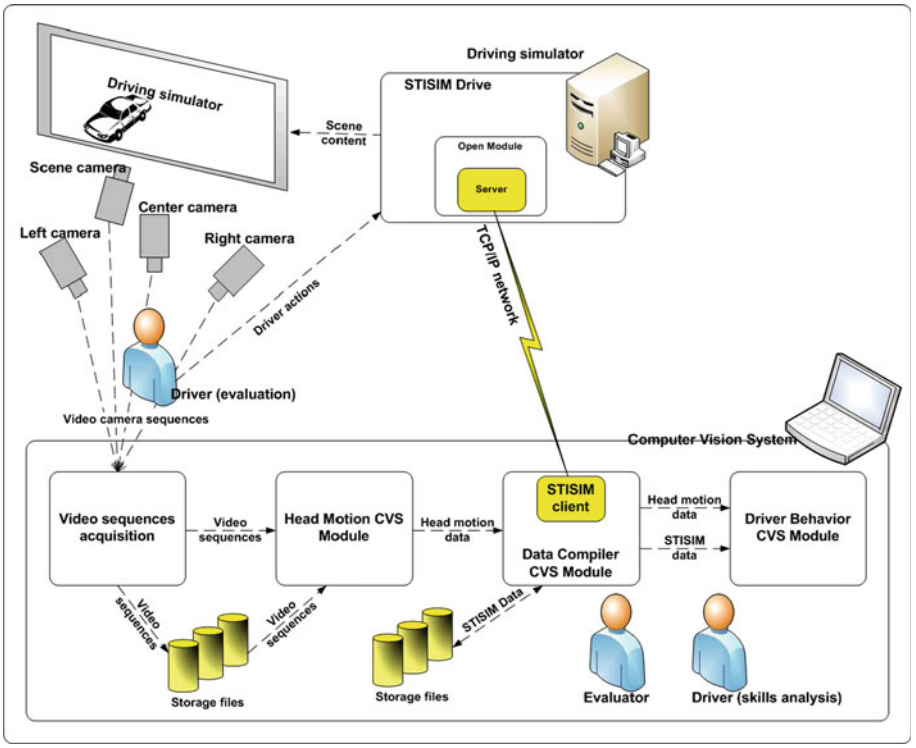


Fig. 4 Overall view of the computer vision system (CVS) connected to the STISIM drive (driving simulator)

or traffic lights, pedestrians, other vehicles, etc.) in synchronization with the video input captured by the four cameras. For each video frame, these data are stored (for future playback) and distributed sequentially to the processing software modules.

At the beginning of the processing chain, the "Head Motion" module uses the video sequences to identify the driver's visual search actions. At the end of the chain, the "Driver Behavior" module correlates the data generated by the previous modules to determine if the driver acts correctly during manoeuvers (for instance, did the driver check the blind spot before a lane change?). Finally, the module adds a bookmark to the interface for each important action detected to help driving monitors during driver training/re-training sessions following the experiment in the simulator.

3 Identification of visual search actions

The identification of the driver's visual search actions is achieved according to the following steps. We first start by detecting the most important facial features, namely the nose tip and the eyes. To achieve this detection, we introduce a new algorithm for eye detection and we call upon the cascade of boosted classifiers technique based on Haar-like features for detecting the nose tip. The next step consists in the tracking of these facial features using the pyramidal Lucas–Kanade method [13]. Finally, the identification of the visual search actions is based on the coordinates of facial features resulting from the detection and tracking steps. Note that these visual search actions are related to specific events such as the verification of blind spot and rear/lateral view mirror checking.

3.1 Detection of facial features

In pattern recognition, an important research field is concerned with the detection and localization of objects and patterns. In the literature, different techniques have been proposed including sliding window classifiers, pictorial structures [5], constellation models [6] and implicit shape models [12]. The sliding window classifiers [18,21,27,28,32,33] are based on the following approach: a sliding window is matched with image parts at different positions and scales. Each mapping reveals whether the sliding window contains the requested object or the background. This approach has been mainly used to detect rigid objects such as faces and cars [9,18,21,33]. Another use of this approach is to detect only parts of the object instead of the object as a whole [14,15]. Those detected parts need to be assembled to recognize the full object. Another set of approaches [2,7] is based on region mapping around extracted local interest points from the image, rather than performing operations on the whole image.

3.1.1 Eye detection

In what follows, we introduce a new method for eye detection. The main goal is to identify the two pupils of a person.

The proposed method is based on *a priori* knowledge of eye geometry, on the position of the eyes in the face and on their relative position (angle, distance, shape, etc). Note that the relative position is calculated empirically knowing that the distance separating drivers from cameras is relatively constant for all drivers. Additionally, using the method of Viola and Jones [33], we identify *a priori* the region of interest (ROI) of the eyes in the face. First, the method described in [33] is applied for face detetction. Second, based on the *a priori* knowledge of the face geometry, we identify the ROI of the eyes using the identified face's contour. The recognition of the two pupils reduces to the identification of a pair of blobs (connected set of pixels) with relatively round shapes and reasonable sizes. The proposed method is composed of the following steps:

Extraction of the blobs Blob extraction as shown in Fig. 7c is achieved according to the two following steps: first, a binarization process is applied to the eyes ROI followed by the application of a closing operator (dilatation and erosion). Second, an algorithm for connected components extraction is applied to obtain blobs. This algorithm finds 8-connected pixels belonging to a region in the input image. The OpenCV algorithm "cvBlobsLib" is used for this purpose. The resulting blobs are characterized by different sizes and shapes and only two of them correspond to the eyes. In what follows, a series of tests is applied to identify the pair of blobs corresponding to the eyes.

Subdivision of highly non-convex blobs The saturation process can generate highly non-convex blobs that may contain the requested blobs (i.e. the pupils). Note that in Euclidean space, an object is convex if for every pair of points within the object, every point on the straight line segment that joins them is also within the object. In vector space, a set S is said to be convex if, for all x and y in S and all t in the interval $[0, 1]$, the point $(1 - t)x + ty$ is in S. In other words, every point on the line segment connecting x and y is in S. Thus for a non-convex object, the above-mentioned property is violated. For example, if we apply the saturation process to an image of a driver wearing eyeglasses, the eye can merge with a part of the glasses (cf. Fig. 5). The shape of the blob containing the eye is thus highly non-convex and differs from the requested shape (eye geometry). One possible solution to this problematic situation is to subdivide the highly non-convex blob (cf. Fig. 6a) into several convex blobs (cf. Fig. 6b). To this end, we introduce the following algorithm:

- Calculation of the convex hull encompassing the considered blob using the algorithm proposed by Sklansky [25].
- Calculation of the difference between the convex hull of the blob and the blob itself. The OpenCV algorithm "CvConvexityDefect" used here returns the start and end points of the non-convexity, the coordinates of the deepest point on the contour and the depth of each non-convexity.
- Filtering: we only consider the strong non-convexities, i.e. a depth of the order of several pixels. Thus, for non-convex regions whose depth is greater than n pixels, the blob is separated into two parts.
- Calculation of the line separating the blob: the latter connects the deepest point with the midpoint between the extremity points (cf. Fig. 6a).
- Subdivision of the blob: the elements of the blobs on both sides of the line are grouped into two new blobs (cf. Fig. 6b).

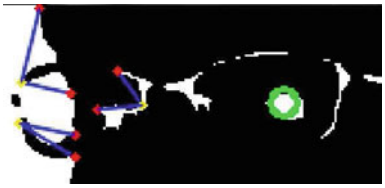


Fig. 5 Example of non-convex blob

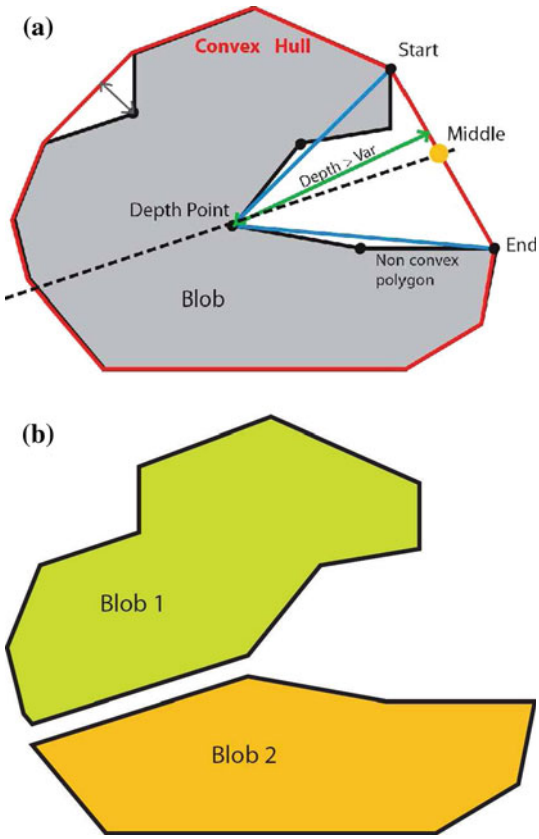


Fig. 6 The principle of blob separation

Selection of blobs Once the blobs are detected and divided whenever necessary, a first filter is applied to eliminate those blobs whose features are not compatible with the typical

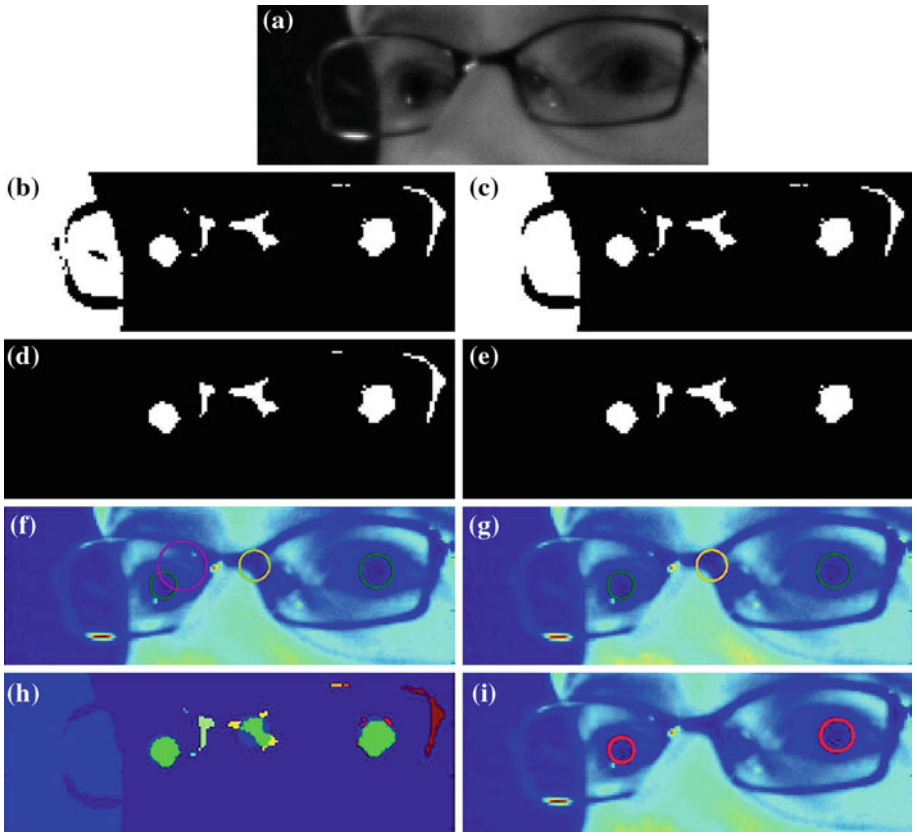


Fig. 7 Steps of the eye detection algorithm. **a** the region of interest, **b** after saturation of the image, **c** after closure, **d** after selection by size of blobs, **e** after selection by shape of blobs, **f** fitting circles on blobs, **g** fitting on the image of ROI and selection according to the radius of the fitted circle, **h** comparison between the area of the circles and the intersection of circle and blob, **i** final selection

shape of an eye. The selection criteria are very simple, but are discriminating enough to enable the identification of the requested blobs:

- *The number of pixels* must be within an acceptable range. Thus, a blob made up of less than 5 pixels is considered too small, but a blob with more than 200 pixels is too big (cf. Fig. 7d).
- *Dimensions* The width and height of the rectangle enclosing the blob are calculated (cf. Fig. 7e). The ratio width/height can reveal the shape of the blob. Each blob elongated in the vertical direction is eliminated (width/height « 1).
- *The shape* a circle C_{fit} is adjusted by a least squares method [8] to each of the blobs satisfying the previous tests (cf. Fig. 7f). The least squares method computes the circle for which the sum of the squares of the distances to the given points is minimum. Circles are represented algebraically i.e. by an implicit equation of the form $F(x) = 0$. If a point is on the curve then its coordinates x are zero of the function F.
  A first discrimination of the blobs is made according to the value of the radius of the circle.
  A second discrimination is based on the ratio R_A between the area covering the intersection between the blob and the circle and the area of the circle itself. This ratio reveals whether or not the blob is circular and is well registered in the best circle C_{fit} (cf. Fig. 7g).

$$R_A = \frac{A_{\text{blob}} \bigcap A_{\text{circle}}}{A_{\text{circle}}}, \tag{1}$$

  where A_{blob} and A_{circle} are respectively the area of the blob and the area of the circle in pixels.

The blobs that survived the above-mentioned criteria, are rated on a scale ranging from 0 to 1. To this end, a weighting approach based on the Standard Gaussian (SG) distribution is used to associate a weight with each blob, according to its characteristics that are compared to an ideal case. A weight of 1 corresponds to a blob corresponding to the ideal case.

$$\text{SG}(x) = \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{2}$$

with μ the mean, σ the standard deviation and x the possible values taken by the ratio R_A.

Parameter μ represents the reference value, i.e. the value corresponding to an ideal ratio. It would be 1 if we consider that the pupil is perfectly circular regardless of the viewpoints. In this case, the Gaussian is centered around 1. Parameter σ is set empirically and corresponds to the tolerance threshold on the ratio in (1). The distance between R_A and μ allows the selection of the weight for a blob according to (2).

Finally, each blob is assigned a weight associated with ratio R_A, to the radius of the circle C_{fit} and to the ratio width/height. The final weight of the blob is given by the average value of these three weights.

Selection per pairs of blobs At this step of the process, we selected the blobs that have a high probability of corresponding to the eyes. In what follows, blobs are not processed individually but in pairs. We identify the pairs of blobs that are arranged such as to correspond to both eyes. In our context (Driving simulator), the distance between the driver and the camera is roughly constant over time. Additionally, since the human morphology is relatively constant, we assume that the gap between the eyes varies slightly from one individual to another. Note that the default value of the gap is set to 80 pixels. These two assumptions are exploited thereafter and make it possible to effectively discriminate the blobs that can correspond to pairs of eyes.

We start by constructing, all possible combinations C of two blobs out of the group of n blobs that have passed the previous processing steps. Recall that

$$C = \overset{k=2}{\underset{n}{C}} = \frac{n!}{k!(n-k)!}. \tag{3}$$

Once all possible pairs are built, the next step consists in calculating the Euclidean distance d_{pair} (4) between the blob centers of each pair, as well as the angle α_{pair} (4) between the line connecting the centers of the two blobs of the considered pair and the horizontal of the image. These criteria allow a discrimination between more pairs of blobs.

$$\begin{aligned} d_{\text{pair}} &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}, \\ \alpha_{\text{pair}} &= \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right), \end{aligned} \tag{4}$$

with (x_1, y_1) and (x_2, y_2) the image coordinates of the blob centers. Note that we just keep the pairs of blobs for which $d_{\text{pair}} \in [60..100]$ and $\alpha_{\text{pair}} < 25°$.

For the remaining pairs, a weighting mechanism similar to the one used for the individual processing of the blobs is applied to compare the probability that a pair corresponds to the eyes. This weighting mechanism combines the information on both the pairs of blobs and the blobs themselves. Thus, if two pairs have similar values for d_{pair} and α_{pair}, the values characterizing the blobs will make the difference in the detection of the pair of blobs corresponding to the eyes. If by this process the weights assigned to each pair do not allow the selection of the requested pair of blobs, the final selection is reported to the matching step.

Experimental results of the above-mentioned approach are given in Fig. 8. Experiments conducted on many subjects have demonstrated that the above approach is reliable for finding the eyes and is also robust to variations of the position of the head in the image and also applies to subjects wearing glasses.

Fig. 8 Experimental results of the eye detection step

3.1.2 Nose detection

For nose detection we call upon the cascade of boosted classifiers based on Haar-like features. This machine learning approach for rapid object detection was first introduced by Viola and Jones [31] and then extended by Lienhart and Maydt [11] using a new set of rotated Haar-like features. Detection is achieved according to the following steps. First an AdaBoost-based classifier is trained from a set of positive and negative examples. Positive examples are target images and negative examples are arbitrary images not including the target (in this case the nose). Once the classifier is trained, the next step consists in target detection. For achieving this, a sliding window is applied at different positions in the requested image. For each position, the classifier decides whether or not the target is present in the window. Finally, the method returns the regions likely to contain the target. Further details about classifier-based detection can be found in [11,18,31,33]. In our case, the classifier was trained from a set of 500 images from which a set of 2,500 positive and 2,500 negative examples were constructed (cf. Figs. 9, 10). Positive examples are images of different noses extracted manually from our image database. Negative examples are images of different parts of the face not including the nose. Experimental results of nose detection are given in Fig. 11.

Fig. 9 A sample of positive examples for training the classifier for nose detection

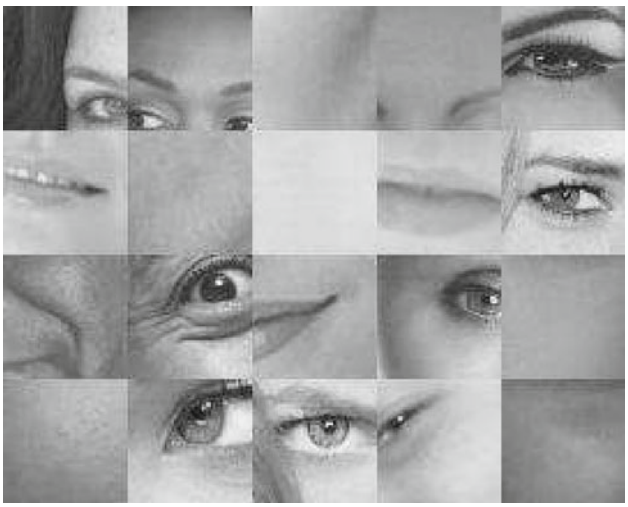


Fig. 10 A sample of negative examples for training the classifier for nose detection

In Fig. 11, results reveal that the detection of the nose tip is achieved successfully regardless of face orientation and skin color.

Note that the approach used for nose tip detection can be adapted to eye detection as well. This is achieved by the use of a new learning set based on eye models. Note that we use the learning set provided by the OpenCV library. In Fig. 12, we show the experimental results obtained with the joint detection of both eyes and nose tip using boosted classifiers. However, the eye detection method described in Sect. 3.1.1 is more accurate since it allows the detection of the pupils instead of the "average" region of each eye detected by the boosted classifiers. Notice that both nose and eye detection rate is almost equal to 100%.

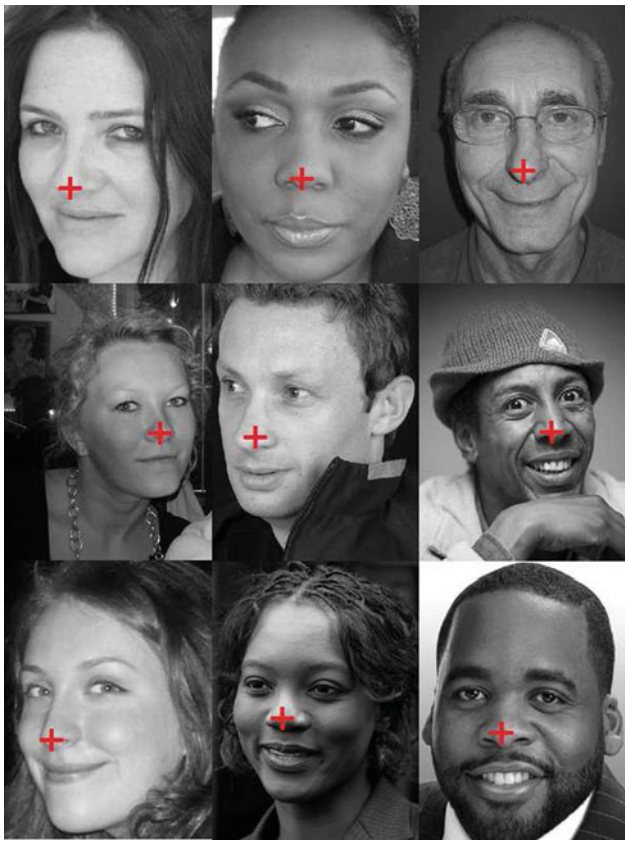


Fig. 11 Nose tip detection results

3.2 Tracking of facial features

Once the facial features (nose tip and eyes) are detected correctly in one image, the next step consists in tracking these features in video sequences. Object tracking is an important research field in the domain of computer vision. In the literature, three main families of approaches for object tracking can be found. The first one is point-based tracking [19,23,30]: at each frame, the requested objects are detected and represented by points. The correspondence of points is achieved according to the previous state (position and motion) of the object. The second one is kernel-based tracking [4,13,22,24]: the requested object is modeled as a geometric template (triangle, rectangle, ellipse, etc). Object tracking is achieved by calculating the kernel motion across the frames. This motion is often modeled as a parametric transformation such as an affine or a similarity transformation (translation, scale, rotation). The third approach is silhouette-based tracking [1,10,20] for which tracking is achieved by estimating the object region across the frames. The information encoded inside the object region is used to model the object. Tracking is performed by matching the silhouettes and object models from frame to frame.

Fig. 12 Joint detection of the eyes and the nose tip

Following an overview of the literature, we opted for the Lucas–Kanade (LK) method [13] which is one of the most reliable tracking methods. More specifically, we used the pyramidal implementation of this method [3]. A summary of the problem statement of the LK method is described in the following.

Brief summary of the LK method The LK algorithm is a two-frame differential method for optical-flow based motion estimation. This method is based on the assumption that the optical flow is constant at the local neighborhood of the considered pixel. Let us consider two gray-level images I and J of size $n_x \times n_y$, and consider a specific pixel $\mathbf{u}(u_x, u_y)$ from the first image I. The main goal of feature tracking is to find the location $\mathbf{v} = \mathbf{u} + \mathbf{d}$, on the second image J, such that $I(\mathbf{u}) \simeq J(\mathbf{v})$. Vector $\mathbf{d}$ corresponds to the image displacement. It is estimated by minimizing the residual function $\varepsilon(\mathbf{d})$, which is defined as follows [3]:

$$\varepsilon(\mathbf{d}) = \varepsilon(d_x, d_y) = \sum_{x=u_x-\omega_x}^{u_x+\omega_x} \sum_{y=u_y-\omega_y}^{u_y+\omega_y} [I(x, y) - J(x + d_x, y + d_y)]^2. \quad (5)$$

Note that the local neighborhood is of size $(2\omega_x + 1) \times (2\omega_y + 1)$. Typical values for ω_x and ω_y range from 2 to 7 pixels. We can attribute an arbitrary value belonging to the interval [2..7] to both ω_x and ω_y.

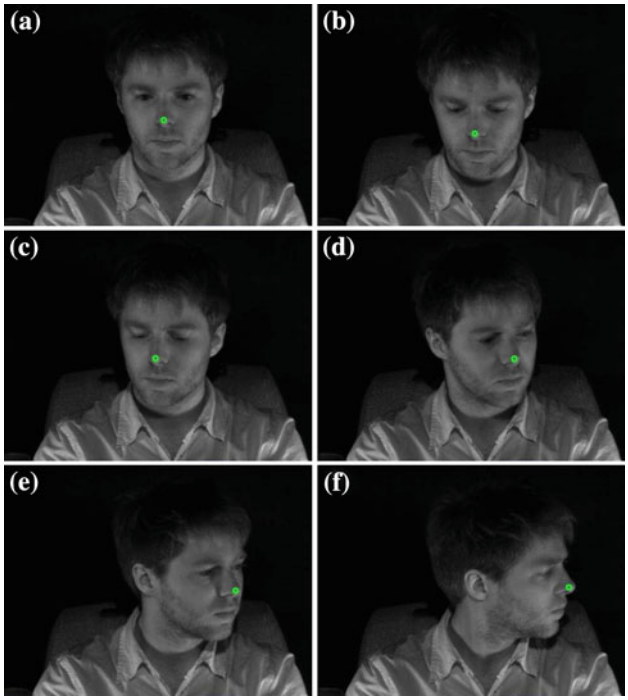


Fig. 13 Nose tip tracking. The above images are extracted from the same video at different times from **a** to **f**

Among well-known classical techniques, the least squares method is used the most for minimizing $\varepsilon(\mathbf{d})$. However, the pyramidal implementation [3] of the classical Lucas–Kanade algorithm remains the most powerful solution to this problem.

In Fig. 13, we show experimental results of nose tip tracking by selecting six different frames from a video sequence. Note that the facial feature (tip of nose) is tracked successfully throughout the sequence.

3.3 Identification of specific search actions

The main goal of our work is to study the cephalo-ocular behavior of drivers using computer vision. More specifically, we are interested in the study of the visual search actions related to the verification of the blind spot when changing lanes and overtaking. Recall that the blind spot is the space on each side of a car that is not covered by the driver's field of vision (including the field of vision due to rear-view and wing mirrors). To test whether the driver is checking the blind spots or not, the following algorithm was implemented:

1. Detection of the facial features (nose tip and eyes) using the techniques described in Sects. 3.1.1 and 3.1.2.
2. Tracking of the facial features using the method described in Sect. 3.2. The tracking process is accompanied by the

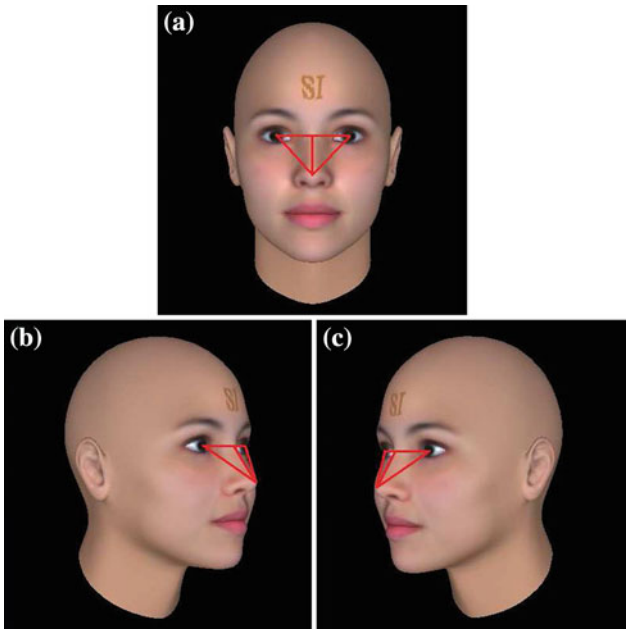


Fig. 14 The visual search actions related to the verification of the blind spots. **a** Initial position, **b** position due to the verification of the *left* blind spot, **c** position due to the verification of the *right* blind spot

calculation of different distances (cf. Fig. 14b, c) between the facial features. These distances allow the orientation of the driver's head to be estimated. Based on the head orientation, we can interpret the visual search actions of the driver in connection with events occurring in the scenario (nearby traffic, etc). A detailed technique for identifying the verification of blind spots by the driver is described next.

3. Return to step 1 of the algorithm when facial features are lost.

The verification of the blind spot is accompanied by a rotation of the driver's head in the direction of the considered path (cf. Fig. 14b, c). The angle of rotation adopted by the driver's head is inversely proportional to the distance separating the two eyes. When the driver verifies the blind spot, the angle of rotation of his head reaches its maximum value that corresponds to a minimum distance separating the two eyes. Additionally, the coordinates of the two eyes allow us to know in which direction the driver is currently looking. Based on these observations, we can accurately identify the visual search action related to the verification of the blind spot. Note that two additional events allow the verification actions of the blind spot to be identified. The first event is the loss of the left eye and/or the nose tip when verifying the left blind spot. The second event is the loss of the right eye and/or the nose tip when verifying the right blind spot. The loss of the nose tip is due to its confusion with the background when

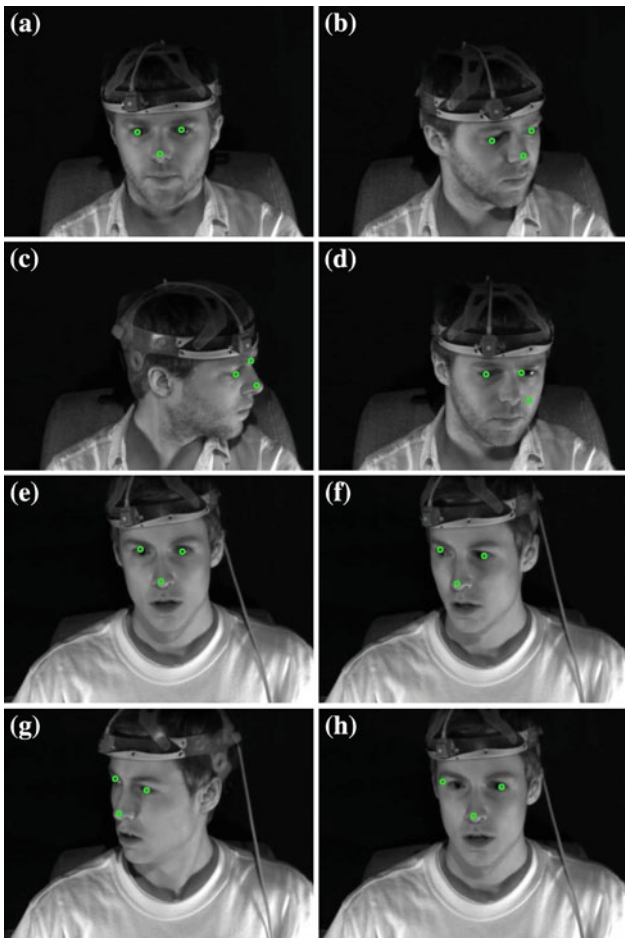


Fig. 15 The verification of the blind spot is accompanied by the loss of facial features (nose tip and/or eyes). **a**, **e** First detection of facial features. **b**,**c**,**f**,**g** Different steps of the tracking process. **d** Loss of nose tip and *left* eye. **h** Loss of the *right* eye

verifying the blind spot, while the loss of the eye is due to its partial or total occlusion in the video frame.

In Fig. 15, we show test results obtained using a cascade of boosted classifier and the pyramidal Lucas–Kanade method for facial features detection and tracking. The loss of nose tip and left eye due to the verification of the left blind spot is shown in Fig. 15d. The loss of the right eye due to the verification of the right blind spot is shown in Fig. 15h.

In Fig. 16, we show test results obtained using the proposed system. Following the loss of facial features, the system triggers the detection process and generates a new event for reporting that the driver is probably verifying the blind spot. Additionally to the identification of the driver's visual search action while verifying the blind spot, the system is able to support other events, such as the visual verification of rear-view and wing mirrors. Events resulting from the identification and analysis (using our system) of the cephalo-ocular behavior of drivers will be used by a Kinesiology research group to retrain older drivers in a safe-driving context.

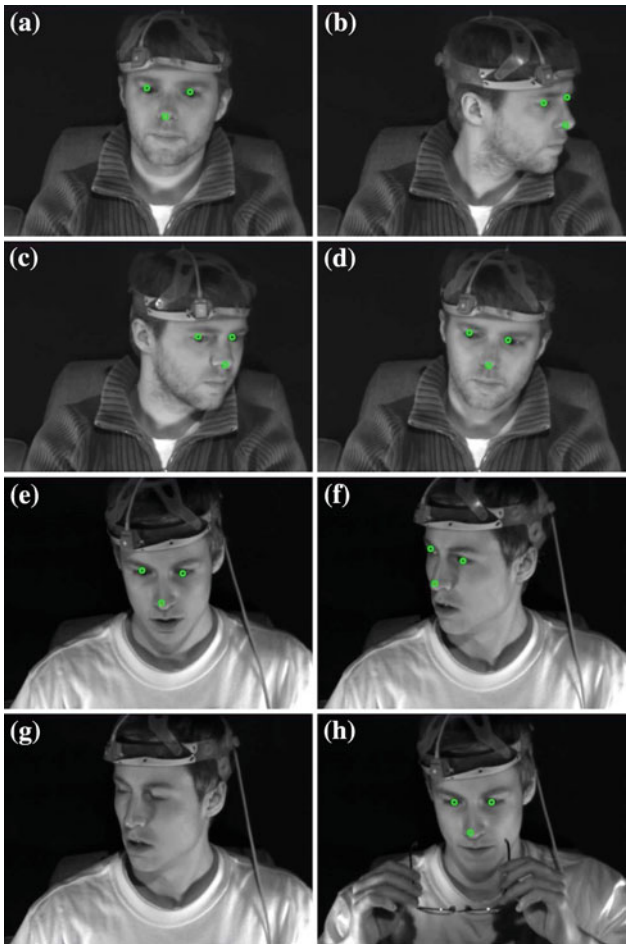


Fig. 16 Test results of the introduced framework. **a**,**e** First detection of facial features. **b**,**c**,**d**,**f** Different steps of the tracking process. **g** The loss of facial features triggers the detection process. **h** Tracking of facial features following the second detection

4 Analysis and interpretation of the cephalo-ocular behavior of drivers

This section is devoted to the analysis and interpretation of the cephalo-ocular behavior of drivers in a simulated driving context. The execution of this task calls upon two different modules. The output of the first module is the driver's visual search actions related to blind spot checking, normal driving, etc. The output of the second module is the identified car/road events extracted from the driving simulator. The analysis of the driver behavior reduces to the study of the correspondences between outputs of these two modules, i.e. each frame of the simulator scenario is correlated with each frame of the video sequences of the driver's head to check whether or not proper cephalo-ocular behavior is adopted with respect to the driving context. The correlation between these frame is stored in a log report that can be used by monitors for retraining older drivers in a safe driving context. We start by describing how the specific car/road events (overtaking, changing lanes, the presence of an intersection, etc.) are modeled. Note that the simulator scenario and the driver's head video are synchronized.

4.1 Identification of car/road events

The identification of car/road events from the driving simulator is achieved by modeling each event as a procedure that is a function of the environmental variables that are extracted from the simulator open module. More specifically we are interested in the identification of specific events, such as overtaking, stopping the car, changing lanes, etc. The particularity of these events lies in the accompanied visual search actions. For instance, we can mention the visual search actions related to blind spot checking when overtaking and changing lane which consist in rotating the head. Using the simulator open module we can have access to a set of environmental variables related to both the car and the road. For example, we can extract the lateral/longitudinal velocity and acceleration of the car, the car's current lane position with respect to the roadway's centerline, the width of the right/left side of the road, the car heading angle relative to roadway centerline, etc. We use these environmental variables to model car events such as overtaking, changing lanes, stopping at an intersection, etc. Since we are interested in the verification of blind spot by older drivers, we deal with car events requiring blind spot checking.

A lane change is identified as follows: based on the width of the left/right side of the road, we model a region of lane change. Note that each side of the road can have one or more lanes. The modeling of lane change regions is only considered for roads with two or more lanes. Once the registered car position crosses this region, a potential lane change is signaled. For the overtaking action, two cases can occur: First, overtaking of a car through the same driving lane. Second overtaking of a car by means of the inverse lane (lane of oncoming traffic). In the first case, we identify an overtaking action if two successive and opposite lane changes are identified during a given time-lapse. In the second case (overtaking through the inverse lane), we use the car position to identify the crossing between normal and opposite traffic lanes. Once again if two successive and opposite passages are established during a given time-lapse a potential overtaking action is signaled. Thus, the modeling of car events is based on the values of environmental variables that are extracted via the simulator open module.

The identification of car/road events is carried out at each frame of the simulator scenario according to the following process: each event is modeled as a c++ function using the values of the extracted environmental variables. Each function returns a score, 0 if the event is not identified and 1 if it is identified.

Table 1 Identification of car/road events

Frame	State
0	Driving
1	Driving
2	Driving
3	Driving
325	Lane change
326	Lane change
327	Lane change
328	Lane change
329	Lane change
340	Driving
341	Driving
342	Driving
1763	Overtaking
1764	Overtaking
1765	Overtaking
1766	Overtaking
1767	Overtaking
2407	Intersection
2408	Intersection
2409	Intersection
3017	Car stop
3018	Car stop
3019	Car stop
3020	Car stop
3021	Car stop
3022	Driving
3023	Driving
4132	Car crash
4133	Car crash
4134	Car crash

Additionally, a priority between events is set according to the following order: driving < car stop < lane change < overtaking < intersection < car crash. This order allows us to select the most important event between two or more identified events. Thus, if "driving" and "overtaking" events are identified, the overtaking event is selected because it has a higher priority. An example of experimental results of car/road events identification is given in Table 1: frame 2 corresponds to "driving" state, frame 328 corresponds to "lane change", etc.

4.2 Mapping between the driver's visual actions and the car/road events

In Sect. 3.3, the identification of driver's visual search actions was modeled using computer vision. At each frame of the driver's head video a given visual action is suggested. Since the driver's head video and the simulator scenario are synchronized, it is easy to establish, at each frame, a mapping between the identified car/road events and the driver's visual search actions. An example of this mapping process is given in Table 2. The mapping state informs the monitor if the visual search action is adequate (cf. frame 1765) for the identified car/road events or not (cf. frame 3323). The established mapping log serves as a reference to the monitor for retraining of older drivers.

Table 2 Identification of car/road events and corresponding driver's visual search actions

Frame	Car/road event	Visual action state	Mapping state
0	Driving	Driving	Ok
1	Driving	Driving	Ok
2	Driving	Driving	Ok
3	Driving	Driving	Ok
325	Lane change	Blind spot checking	Ok
326	Lane change	Blind spot checking	Ok
327	Lane change	Blind spot checking	Ok
328	Lane change	Blind spot checking	Ok
329	Lane change	Blind spot checking	Ok
340	Driving	Driving	Ok
341	Driving	Driving	Ok
342	Driving	Driving	Ok
1763	Overtaking	Blind spot checking	Ok
1764	Overtaking	Blind spot checking	Ok
1765	Overtaking	Blind spot checking	Ok
1766	Overtaking	Blind spot checking	Ok
1767	Overtaking	Blind spot checking	Ok
3319	Lane change	Driving	Error
3320	Lane change	Driving	Error
3321	Lane change	Driving	Error
3322	Lane change	Driving	Error
3323	Lane change	Driving	Error
3324	Lane change	Driving	Error
3325	Lane change	Driving	Error
5222	Driving	Driving	Ok
5233	Driving	Driving	Ok
5244	Driving	Driving	Ok

4.3 Retraining of older drivers

The above described system is devoted to the analysis of driver performances in a safe driving context. If we consider the category of older drivers, driving retraining class may offer an opportunity to attenuate some of the aging manifestation that alter driving skills. Unfortunately, few effective opportunities are offered to older drivers who want to maintain or increase their driving performance. Figure 17 shows

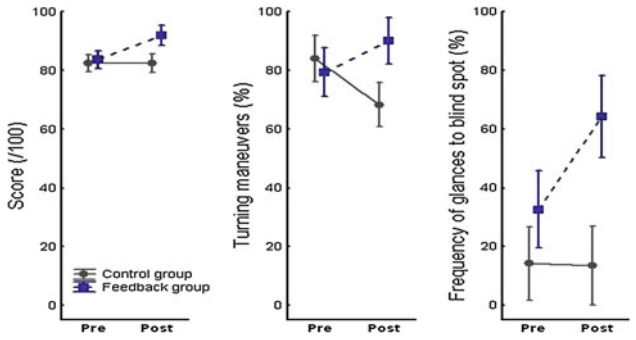


Fig. 17 On-road score (*left*), successful turning manoeuvres (*middle*) and glance at blind spot prior to lane change (*right*)

experimental results related to older drivers' performances following a driving retraining classes. We deal with two groups of older drivers, each driver of the first group receives a driving-specific feedback on his own performance. Drivers of the second group do not receive any driving feedback. Experimental results show that the driving performances of the first group (Fig. 17 blue plots) are enhanced compared to the driving performances of the second group (Fig. 17 gray plots). More specifically, the performances of *on-road* driving for successful left-turn manoeuvres and for blind spot checking prior to lane change are enhanced for drivers of the first group who have received *in-simulator* training. This suggests that training received in the simulator can be transfered to safer driving in real on-road conditions. Note that data analysis for these experiments was conducted manually by a kinesiology research group. The main goal of our research work is to automate this process of analyzing the correspondence between scenario events and visual search behavior of the driver. Notice that the accuracy of the proposed system depends on the accuracies of the used methods and approaches (eye/nose detection, tracking of facial features, visual action and car/road event identification). Since accuracies of these methods and approaches are almost equal to 100%, we can conclude that the entire accuracy of this system is almost equal to 100%.

5 Conclusion

In this paper, we introduced a computer vision system specially designed to the analysis and interpretation of the cephalo-ocular behavior of a driver. The system is described in three main steps. The first step is devoted to the description of both hardware and software components of the system. The second step deals with the identification of the driver's visual search actions. The last step is concerned with the established mapping between the identified visual search actions and the identified car/road events. All of the experiments confirm both the accuracy of the proposed system and its usefulness for automatic analysis of driver's cephalo-ocular behavior in a simulator. Future work will consist in testing the system on a large group of subjects as well as implementing the head tracking module in a real on-road driving context.

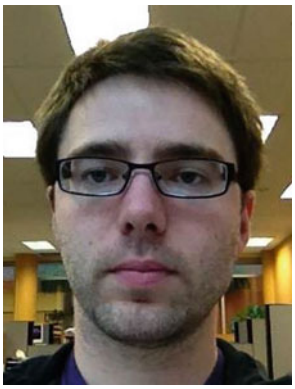
References

1. Bertalmio, M., Sapiro, G., Randall, G.: Morphing active contours. IEEE Trans. Pattern Anal. Mach. Intell. **22**(7), 733–737 (2000)
2. Bouchard, G., Triggs, B.: A hierarchical part-based model for visual object categorization. In: IEEE Conference on Computer Vision and Pattern Recognition (2005)
3. Bouguet, J.Y.: Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm. Intel Corporation Microprocessor Research Labs (2000)
4. Comaniciu, D., Ramesh, V., Andmeer, P.: Kernel-based object tracking. IEEE Trans. Pattern Anal. Mach. Intell. **25**, 564–575 (2003)
5. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV **61**(1), 55–79 (2005)
6. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR. vol. 2, pp. 264–271 (2003)
7. Fergus, R., Perona, P., Zisserman, A.: A sparse object category model for efficient learning and exhaustive recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (2005)
8. Gander, W., Golub, G., Strebel, R.: Least-squares fitting of circles and ellipses. BIT Numerical Mathematics. Springer, Berlin (1994)
9. Rowley, H.A., Baluja, S., Kanade, T.: Human face detection in visual scenes. In: Advances in Neural Information Processing Systems, vol. 8 (1995)
10. Kang, J., Cohen, I., Medioni, G.: Object reacquisition using geometric invariant appearance model. In: International Conference on Pattern Recongnition, pp. 759–762 (2004)
11. Lienhart, R., Maydt, J.: An extended set of haar-like features for rapid object detection. In: IEEE International Conference on Image Processing, pp. 900–903 (2002)
12. Leibe, B., Leonardis, A., Schiele, B.: Combined object categorization and segmentation with an implicit shape model. In: ECCV workshop on statistical learning in computer vision, pp. 17–32 (2004)
13. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: International Joint Conference on Artificial Intelligence (1981)
14. Mikolajczyk, K., Schmid, C., Zisserman, A.: Human detection based on a probabilistic assembly of robust part detectors. In: Proceedings of the 8th European Conference on Computer Vision, May 2004
15. Mohan, A., Papageorgiou, C., Poggio, T.: Examplebased object detection in images by components. IEEE Trans. Pattern Anal. Mach. Intell. **23**(4), 349–361 (2001)
16. Murphy-Chutorian, E., Doshi, A., Trivedi, M.M.: Head pose estimation for driver assistance Systems: a robust algorithm and experimental evaluation. In: IEEE Intelligent Transportation Systems Conference, pp. 709–714 (2007)
17. Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation and augmented reality tracking: an integrated system and evaluation for monitoring driver awareness. IEEE Trans. Intell. Transp. Syst. **11**(2), 300–311 (2010)
18. Papageorgiou, C., Poggio, T.: A trainable system for object detection. Int. J. Comput. Vis. **38**(1), 15–33 (2000)

19. Salari, V., Sethi, I.K.: Feature point correspondence in the presence of occlusion. IEEE Trans. Pattern Anal. Mach. Intell. **12**(1), 87–91 (1990)
20. Sato, K., Aggarwal, J.: Temporal spatio-velocity transform and its application to tracking and interaction. Comput. Vis. Image Underst. **96**(2), 100–128 (2004)
21. Schneiderman, H., Kanade, T.: A statistical model for 3D object detection applied to faces and cars. In: IEEE Conference on Computer Vision and Pattern Recognition (2000)
22. Schweitzer, H., Bell, J.W., Wu, F.: Very fast template matching. In: European Conference on Computer Vision, pp. 358–372 (2002)
23. Shafique, K., Shah, M.: A non-iterative greedy algorithm for multi-frame point correspondence. In: IEEE International Conference on Computer Vision, pp. 110–115 (2003)
24. Shi, J., Tomasi, C.: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 593–600 (1994)
25. Sklansky, J.: Finding the convex hull of a simple polygon. Pattern Recogn. Lett. **1**, 79–83 (1982)
26. Teasdale, N., Lavallière, M., Tremblay, M., Laurendeau, D., Simoneau, M.: Multiple exposition to a driving simulator reduces simulator symptoms for elderly drivers. In: Proceedings of the Fifth International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design. Big Sky, USA (2009)
27. Murphy, K.P., Torralba, A., Eaton, D., Freeman, W.T.: Object detection and localization using local and global features. In: Proceedings of Toward Category-Level Object Recognition, pp. 382–400 (2006)
28. Torralba, A., Murphy, K.P., Freeman, W.T.: Using the forest to see the trees: exploiting context for visual object detection and localization. Commun. ACM **53**(3), 107–114 (2010)
29. Trivedi, M.M., Gandhi, T., McCall, J.C.: Looking-in and looking-out of a vehicle: computer-vision-based enhanced vehicle safety. IEEE Trans. Intell. Transp. Syst. **8**(1), 108–120 (2007)
30. Veeman, C., Reinders, M., Backer, E.: Resolving motion correspondence for densely moving points. IEEE Trans. Pattern Anal. Mach. Intell. **23**(1), 54–72 (2001)
31. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Conference on Computer Vision and Pattern Recognition (2001)
32. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: IEEE Conference on Computer Vision and Pattern Recognition (2003)
33. Viola, P., Jones, M.: Robust real-time object detection. Int. J. Comput. Vis. **57**(2), 137–154 (2004)
34. Wu, J., Trivedi, M.M.: An integrated two-stage framework for robust head pose estimation. In: AMFG, pp. 321–335 (2005)

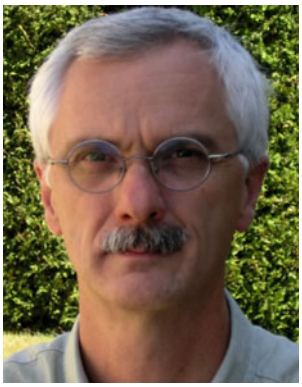
Author Biographies

Samy Metari received his B.Eng. degree in computer science from the University of Constantine, Algeria (2001), the M.Sc. degree in computer science from the University of Poitiers, France (2003), and Ph.D. degree in computer science from the University of Sherbrooke, QC, Canada (2009). He is currently a postdoctoral researcher at Laval University and an associate RRC member at MITACS. His research interests include computer vision, image processing, and pattern recognition.



Florent Prel graduated from Optronic Engineering school in 2007 (Ecole Nationale Supérieure de Sciences Appliquées et de Technologie) and has a Master in Electrical Engineering (Laboratoire de Vision et Sytèmes Numériques at Laval University, directed by Denis Laurendeau). He works at ABB as system Engineer, particularly on the development of Infrared Hyperspectral Spectroradiometers and electro-optics systems.

Thierry Moszkowicz holds a M.Sc. degree in electronic engineering (1989) from the University Paris-Sud (XI), France and a M.Sc. degree in electrical engineering specialization in computer science (1995) from Laval University, Quebec, Canada. He worked as computer science engineer in France, and as analyst in Canada. Since 2007 he works as a research professional at Laval University, Quebec, Canada.



Denis Laurendeau holds a Bachelor's degree in Engineering Physics (1981) and M.Sc. (1983) and Ph.D. degrees (1986) in Electrical Engineering from Laval University, Québec, Canada. In 1987 he was a visiting scientist at Hydro-Quebec Research Institute (IREQ) where he developed computer vision applications for live-line maintenance at IREQ's Robotics Division. In 2001, he was a visiting researcher at ABB-Bomem where he worked as project leader in the field of Fourier Transform Spectrometry. He is currently an invited scientist at Centre de Recherche Informatique de Montréal (CRIM). Dr. Laurendeau's research interests include 3-D modeling for Virtual Reality, simulation in VR, object tracking, and biomedical applications of computer vision. Dr. Laurendeau holds research grants from NSERC, Auto21-NCE, and FQRNT. He participates in university-industry partnerships with CAA-Quebec, Systems Technology Inc., GM-Canada, RDDC-Valcartier and Creaform. Dr. Laurendeau is Director of the REPARTI research center and Head of the Computer Vision and Systems Laboratory at Laval University. He is currently president of the International Association for Pattern Recognition (IAPR).

Normand Teasdale is a professor in kinesiology at the Faculty of medicine at Laval University. He received his Ph.D. in Kinesiology from UCLA in 1987. He has published extensively about how aging modifies the control of movement and driving a vehicle.

Steven Beauchemin is an Associate Professor of Computer Science at the University of Western Ontario, London, Canada. Dr. Beauchemin's research interests revolve around computer vision techniques for the automotive industry. As part of past and present research projects, Dr. Beauchemin has been involved in the training and re-training of elderly drivers in simulators and in instrumented vehicles. These projects are funded in part by the Auto21 centre of excellence (www.auto21.ca). Dr. Beauchemin leads a team of graduate researchers and students in the use of vision technologies for intelligent, Advanced Driving Assistance Systems (i-ADAS).

Martin Simoneau received the M.S. and Ph.D. degrees in kinesiology from Laval University, Quebec City, Canada, in 1996 and 1999, respectively. He was a postdoctoral researcher at the Department of Physical Therapy and Human Movement Sciences at Northwestern University Medical School, Chicago, IL, from 1999 to 2001. He is Professor of Kinesiology at Université Laval. His current research interests include assessing the neuromechanics mechanisms involved in balance and movements control and the role of sensorimotor integration during motor adaption.