Look at the Driver, Look at the Road: No Distraction! No Accident!

Mahdi Rezaei and Reinhard Klette
The University of Auckland
Private Bag 92019, Auckland, New Zealand
m.rezaei@auckland.ac.nz

Abstract

The paper proposes an advanced driver-assistance system that correlates the driver's head pose to road hazards by analyzing both simultaneously. In particular, we aim at the prevention of rear-end crashes due to driver fatigue or distraction. We contribute by three novel ideas: Asymmetric appearance-modeling, 2D to 3D pose estimation enhanced by the introduced Fermat-point transform, and adaptation of Global Haar (GHaar) classifiers for vehicle detection under challenging lighting conditions. The system defines the driver's direction of attention (in 6 degrees of freedom), yawning and head-nodding detection, as well as vehicle detection, and distance estimation. Having both road and driver's behaviour information, and implementing a fuzzy fusion system, we develop an integrated framework to cover all of the above subjects. We provide real-time performance analysis for real-world driving scenarios.

1. Introduction

Advanced driver-assistance systems (ADAS) are a current goal in computer vision, especially at centers of the automotive industry. A real-world ADAS needs to understand the driver's behavior (e.g., by analyzing facial features, or by steering-wheel motion analysis). Face detection is arguably still difficult for extreme head poses [2] or challenging lighting conditions. The system also needs to detect potential hazards on the road. Simultaneous "driver" and "road" monitoring requires object detection, pose tracking, and data fusion [24].

First, we propose a comprehensive solution for detecting a driver's direction of attention, yawning, and head nodding. The method is based on the novel ideas of *asymmetric appearance modelling* (ASAM), and the *Fermat-point transform*. Then we combine the introduced method for driver monitoring with road monitoring (i.e., vehicle detection and distance estimation). The system finally analyses the correlation between a driver's head pose with potential road hazards, in order to prevent imminent crashes at early stages.

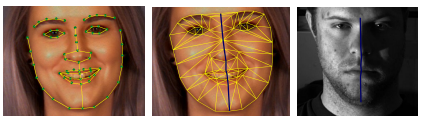

Figure 1. 64 keypoint landmarks (left). Symmetric Delauney triangulation (middle). Asymmetric intensity variations (right).

Using monocular vision only, we keep the system at low computational costs; yet we compete with the state-of-the-art. To the best of our knowledge, no previous research has jointly addressed all of the above mentioned subjects as one integrated real-time solution.

We provide techniques for two important challenges that have rarely been addressed so far: (A) Dealing with intensity asymmetry and unbiased illumination for the same object, such as a driver's face (Fig. 1, right), and (B) Mapping a generic rigid 3-D face model onto deformable faces.

The paper is organized as follows: Section 2 discusses related work. Section 3 proposes the ASAM to define a driver's face shape and appearance. Section 4 discusses driver-pose estimation via mapping of face appearance into a 3-D model, enhanced (for the first time) by the idea of a Fermat-point transform. Section 5 reviews our vehicle detection technique applicable for day, night, and other challenging conditions. Sections 6 and 7 focus on "Driver-Road" fuzzy fusion and experimental results. Section 8 provides concluding remarks.

2. Related Work

Xie et al. [4] propose driver-fatigue detection using the *active appearance model* (AAM), as introduced by Cootes [3], by fitting it to the eye region, followed by head-pose detection depending on the face-centroid. The method appears to be too basic to be applicable in highly-dynamic real-world scenarios.

Mosquera and Castro [21] use a recursive algorithm to improve convergence accuracy when modeling a driver's face. Results show improvements compared to Cootes' AAM method [3]; however, a driver's facial features are not yet taken into account.

Chutorian and Trivedi [22] monitor a driver's activity by

using an array of Haar-wavelet AdaBoost cascades for initial face detection, and applying localized gradient orientation (LGO) as input for supporting vector regressors. The method uses a rigid facial-mesh model to track the driver's head. There is a general weakness here as the tracking module may easily diverge for face shapes that are highly different to the given mesh model.

Visage Technologies® provides a state-of-the-art commercial head tracker [23] based on feature-point detection and tracking of the nose boundary and eye regions. Despite of accurate results under ideal conditions, this tracking system fails in the presence of noise and non-ideal conditions.

Krüger and Sommer used Gabor wavelet networks [10] for head-pose estimation. Claimed advantages cover independence to affine deformations, and high-precision of the algorithm for *any* desired input (the input may range from a coarse representation of a face to an almost photo-realistic subject). Nevertheless, the results are not backed-up by validation or comparison with other techniques.

3. Asymmetric Appearance Models

Appearance models (AM), as originally introduced by Cootes et al. [3], are widely used for object modeling, especially in the context of face processing. Many research work address it as an optimization problem to find an improved fitting algorithm and to reduce the matching errors.

3.1. Implementation

An AM combines a shape model and a texture model. In order to define a face-AM we need to train a variation of face shapes (as shape model) and face intensities (as texture model).

Considering 64 point-landmarks, as illustrated in Fig. 1, right, and using the MUCT face dataset [20], we create an annotated face dataset in order to train a generic face-shape model. Following the standard AM approach [3], and applying a uniform coordinate system, annotated faces are represented by a vector $f = [x_0, y_0, ..., x_i, y_i]^\top$. A face-shape model is defined as follows:

$$f = \bar{f} + P_s b_{s_i} \ , \qquad (1)$$

where $\bar{f}$ is the mean face shape applying *principal component analysis* (PCA) on the available face data, P_s is an orthogonal matrix of face-shape variations, and b_s is a vector of face-shape parameters (given in distance units). By applying a translation (t_x, t_y), and a rotation and scaling ($s_x = s \cdot \cos\theta - 1$, $s_y = s \cdot \sin\theta$), each sample face is warped into the mean shape model, thus creating a new face F. Let $F = S_t(f)$ be this warped image, where S_t is the warping function, and $t = [s_x, s_y, t_x, t_y]^\top$ is the pose parameter vector. Figure 2 illustrates the steps for creating the *appearance face model* based on only two sample faces. The second row of Fig. 2 shows examples of shape variations with different deformation parameters applied to each sample face. The blue face shows the obtained mean-shape.

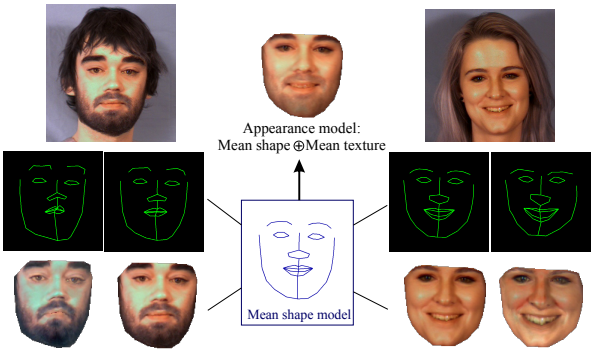

Figure 2. Conversion of face *shape* and face *texture models* of two sample faces into a mean *appearance model*.

To create a face texture (intensity) model, first a symmetric Delaunay triangulation is applied to shape-feature points, for each sample face (Fig. 1, middle). Considering g as a texture vector of a sample face image, similar to the shape-warping stage, we have a mapping $g \rightarrow g^*$, where g^* is generated after scaling and adding an offset to current intensity g. This way we create a shape-free "intensity patch" for each sample face given in the training dataset. This is done by raster scanning of the texture vector g, and a linear normalization of g for every half of the face as follows:

$$g_L^* = \frac{(g_L - \mu_L.\mathbf{1})}{\sigma_L} \quad , \quad g_R^* = \frac{(g_R - \mu_R.\mathbf{1})}{\sigma_R} \ , \qquad (2)$$

where μ_L, μ_R and σ_L^2, σ_R^2 are means and variances for the left and right part of the face-intensity patch, g_L, g_R are the left and right half of the g vector, g_L^*, g_R^* are normalized data, and $\mathbf{1}$ is a vector of ones. After the normalization, we have that ${g^*}^\top \cdot \mathbf{1} = 0$ and $|g^*| = 1$.

As part of *asymmetric appearance models* (ASAM), Eq. (2) considers individual asymmetric intensity normalization for each half of a face. This is a crucial step, and treating the face halves as two distinct objects can help us to prevent divergence of face-shape matching due to cumulative intensity-normalization errors. Figure 1, left, shows how face intensity can vary depending on light source location and due to the nose bridge. This is a common but neglected point in applications such as driving scenarios, where one side of the face is brighter than the other side.

Similarly, by applying a PCA to the normalized intensity data, a face intensity-model is estimated as follows:

$$g_L = \bar{g_L^*} + P_{g_L} b_{g_L} \quad , \quad g_R = \bar{g_R^*} + P_{g_R} b_{g_R} \ , \qquad (3)$$

where $\bar{g^*}$ is the mean vector of normalized gray-level or intensity data, P_g is an orthogonal matrix of texture-modes of variations, and b_g is a vector of intensity parameters in gray-level units (Fig. 2, third row). We apply this as individual processes for each half of the face.

Face-*shape* and *texture* can therefore be summarized as b_s and b_g. Since there could be some correlation between

intensity and shape data, a combined AM is considered. For each sample face, a concatenated vector b is defined:

$$\begin{bmatrix} W_s b_s \\ b_g \end{bmatrix} = \begin{bmatrix} W_s P_s^\top (f - \bar{f}) \\ P_g^\top (g - \bar{g^*}) \end{bmatrix} , \quad (4)$$

where W_s is a diagonal matrix which defines appropriate weights for the concatenation of b_s and b_g at places where they have different units (i.e. the distance unit versus the intensity unit). The RMS in g, per unit change in b_s, is considered to define appropriate weights W_s for Eq. 4. This makes b_s and b_g to be proportional. Applying another PCA to these vectors, the AM is given as

$$b = Qc , \quad (5)$$

where c is the vector of parameters for the *combined* AM which unifies shape and intensity models:

$$f = \bar{f} + P_s W_s Q_s c \quad , \quad g = \bar{g*} + P_g Q_g c . \quad (6)$$

An eigenvector Q is subdivided as follows:

$$Q = \begin{bmatrix} Q_s \\ Q_g \end{bmatrix} . \quad (7)$$

For the training stage, we used a set of 7,512 images from the MUCT face dataset [20], each one annotated as per the proposed 64 point-landmark approach, followed by 2,500 pixel intensity sampling from each half of a face.

3.2. Asymmetric AAM

Reviewing the Cootes et al. method [3], an *active appearance model* (AAM) refers to an active search and refinement process to adapt a previously trained face-AM into an unknown face; by *asymmetric AAM* (ASAAM) we process a face as an asymmetric object.

Accuracy and speed of model refinement is a crucial step, as it can directly affect the next step of our algorithm for head-pose estimation. Using our recently proposed *Global Haar* (GHaar) classifier [14], the classifier can return robust face detection and localization even under noisy and challenging lighting conditions.

Having model parameters c and shape-transforming parameters t, the rough position of the *trained* model points can be mapped on the image frame F, which also represents the initial shape of the face patch.

As part of the matching process, pixel samples g_{im} from the region of the image are taken and projected to the left or right texture model frame, $g_s = T_u^{-1}(g_{im})$. Given the current texture model as $g_m = \bar{g^*} + Q_g c$, the difference (residual) between texture of the current image frame and the current model is as follows:

$$r(p) = g_s - g_m , \quad (8)$$

where p is the parameter vector of the model:

$$p^\top = (c^\top | t^\top | u^\top) . \quad (9)$$

Applying RMS and measuring residual errors, the model parameters can be gradually refined. This can be seen as an optimization approach in which a few iterations lead to smaller residuals, thus to the best match of the model with the input face. Starting from a current estimation for appearance model parameters c, at position t, texture transformation u, and a face example with the current estimation as g_{im}, the iterative algorithm is summarized as Algorithm 1.

Algorithm 1 Iterative search and model refinement

1: Use $g_s = T_u^{-1}(g_{im})$ to map the texture sample frame into the texture model frame.
2: Calculate the current (initial) error, $E_0 = |r|^2 = |g_s - g_m|^2$
3: Evaluate the predicted displacements based on RMS method, $\delta p = -Rr(p)$, where R is the matrix of texture sample points and the model parameter is $R = \left(\frac{\delta r}{\delta p}^\top \frac{\delta r}{\delta p}\right)^{-1} \frac{\delta r}{\delta p}^\top$
4: Set $k = 1$.
5: Set $p = p + k\delta p$ to update the parameters of the model.
6: Calculate F' and g'_m as the new points, and new texture model frame, respectively.
7: Sample the face at F', so as obtain a new estimate of g'_{im}
8: Evaluate a new updated error-vector, $r' = T_{u'}^{-1}(g'_{im}) - g'_m$; therefore the updated error $E_1 = |r'|^2$.
9: **if** $E_1 < E_0$ **then**
10: Accept the last estimate,
11: **else**
12: Set $k = k/2$,
13: Goto Step 5; repeat until no further decrease for $|r'|^2$
14: **end if**

Experimental results show that after 3 to 5 iterations, the ASAAM method rapidly converges to the actual face image. Figure 3 shows an example of inaccurate model fitting by the standard AAM, and an improvement by ASAAM.

4. Driver's Head Pose and Gaze Estimation

In this section, we detect the driver's direction of attention, based on mapping 2-D feature-points into a 3-D face model, and a method which we call the *Fermat-point transform*. Pose-estimation is the preliminary step to analyse driver's attention in correlation with road hazards.

Different approaches have been developed e.g., pose detection from orthography and scaling (POS), or POS with iteration (POSIT) [5, 6, 7], 3-D morphable models [8], random decision forests [1], or multi-view based training [2].

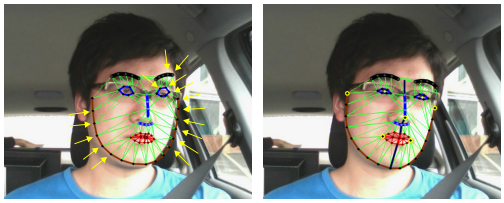

Figure 3. (Left) Inaccurate model fitting, especially for the right half of the face. (Right) ASAAM method leading to a proper model matching both halves of the face.

The above work, even the most recent one, only considers a generic 3-D model, or a set of convex polygons as a model-reference for pose detection. Regardless of the pose-detection methodology and 3-D model specification, the *matching error* of the model with a query object has not yet been addressed.

In the next two sub-sections we introduce a solution to minimize the 2-D to 3-D matching error, thus a more accurate pose estimation.

4.1. Optimized 2-D to 3-D Pose Modelling

Figure 4 shows the pinhole-camera model [7] with calibrated focal length f, center of projection O, axes Ox, Oy, and Oz, and unit vectors i, j, and k in camera coordinates. In the 3-D object plane, we have a face model with feature points F_1, F_2, ..., F_n. The coordinate frame for the face reference is centered at F_0, and it is specified by (F_0u, F_0v, F_0w).

We assume that the driver's face shape is already computed by ASAAM. This relates every feature-point F_n to coordinates (F_nu, F_nv, F_nw), and therefore projected points p_1, p_2,..., p_n to image plane coordinates as of (x_n, y_n). Only coordinates of (X_n, Y_n, Z_n) in the camera coordinate system are unknown. We find the driver's face-pose by calculating rotation matrix and translation vector $O \rightarrow F_0$. Combining all available information, we have:

$$R = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix}, \quad T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}, \quad (10)$$

$$C = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad P = \left[\begin{array}{c|c} R_{3x3} & T_{3x1} \\ \hline 0_{1x3} & 1 \end{array}\right], \quad (11)$$

where R is the rotation matrix, T is the translation matrix, C is the camera matrix, f is the focal length, (c_x, c_y) is the camera's principal point, and P is the pose matrix. Thus, the projection of a given object point (X_n, Y_n, Z_n) into camera coordinates can be represented as follows:

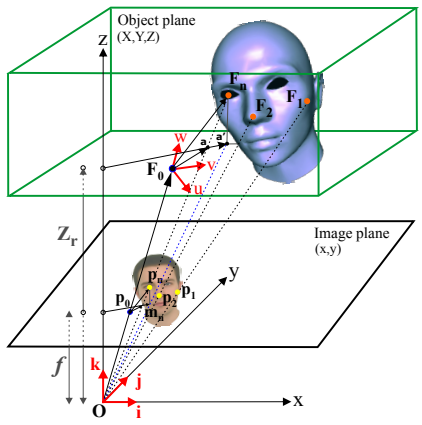

Figure 4. Projection of a 3D face model M into the image plane c.

$$\begin{bmatrix} i_n \\ j_n \\ k_n \end{bmatrix} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} i_u & i_v & i_w & T_x \\ j_u & j_v & j_w & T_y \\ k_u & k_v & k_w & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_n \\ Y_n \\ Z_n \\ 1 \end{bmatrix} . \quad (12)$$

For computing R, we only need i and j (k is simply the cross product $i \times j$). On the other hand, according to Fig. 4, the translation vector T is equal to vector OF_0. Having Op_0 and OF_0 aligned, it follows that

$$T = OF_0 = \frac{Z_0}{f} Op_0 , \quad (13)$$

Therefore, by calculating i, j, and Z_0 (the depth of point F_0), the pose of the face can be defined. Depth variation is small within a driver's facial feature points, compared to the distance between camera and face (i.e., $Oa \approx Oa'$). Thus we can approximate the depth of every point F_n as being equal to Z_r. Consequently, this simplifies the previous expression, and the projection of any point (X_n, Y_n, Z_n) from the object plane to the image plane can be expressed as

$$(x_n, y_n) = (f\frac{X_n}{Z_n}, f\frac{Y_n}{Z_n}) , \quad (14)$$

or as

$$(x_n, y_n) = (\frac{f}{1+\Delta z}\frac{X_n}{Z_r}, \frac{f}{1+\Delta z}\frac{Y_n}{Z_r}) , \quad (15)$$

with $1 + \Delta z = \frac{Z_n}{Z_r}$. Let Pr_1 and Pr_2 be the quadro-elements in the first and second row of the matrix P (in Eq. 11). Thus, we define 4-D vectors I and J as follows:

$$I = \frac{f}{T_r} Pr_1 \quad and \quad J = \frac{f}{T_r} Pr_2 . \quad (16)$$

Knowing the coordinates of vectors F_0F_n in the object plane, and knowing the x_n and y_n coordinates for points p_0 and p_n in the image plane, the fundamental equations are:

$$F_0F_n.I = x'_n \quad , \quad F_0F_n.J = x'_n , \quad (17)$$

$$x'_n = x_n(1 + \Delta z_n) \quad , \quad y'_n = y_n(1 + \Delta z_n) , \quad (18)$$

$$\Delta z_n = Pr_3.\frac{F_0F_n}{Z_r - 1} . \quad (19)$$

Any initial (approximated) value for ΔZ_n solves the above equations, thus a driver's face pose can be approximated.

We performed a statistical analysis on 84 3-D face models from the TurboSquid dataset [9], first, to select an optimum 3-D model with a best match with our face dataset, and, second, to assess mean and variance of depth in regions of eyes, nose tip, mouth, and ear tops. Knowing f

and Z_r (i.e., the distance of the camera to the face), and by applying the analysed statistical data, we derived an initial value $\Delta z_n = 0.082$. Once i and j are computed, the value of Δz_n can be refined and optimized after two or three iterations. This is much faster than a blind POSIT algorithm that needs four to ten iterations to refine Δz [6].

4.2. Face Registration by Fermat-transform

As part of our algorithm to obtain roll, yaw, and pitch angles of a driver's face, we use the state-of-the-art method of EPnP by Lepetit et al. [11]. However, we add an important pre-processing step to minimize the mismatch errors of 2-D to 3-D corresponding points. Work that uses the perspective-n-points (PnP) method, normally considers four points around the nose boundary [5, 12]. This may simplify the case into a planar problem, as sampled feature points around the nose boundary have almost the same depth in the camera-coordinate system. However, as a weakness, those feature points cover only a limited region of the face which might be affected by noise. This causes larger matching errors for registering the corresponding points of a 2-D face with our 3-D model.

Using five correspondences, we consider pose estimation of a driver's face as a P-5-P problem. Generally, the method estimates the pose of the camera from n 3-D to 2-D point correspondences. The idea is to define n 3-D points as a weighted sum of four pre-selected control-points as below:

$$F_n = \sum_{j=1}^{4} \alpha_{nj} C_j^F \quad , \quad p_n = \sum_{j=1}^{4} \alpha_{nj} C_j^p \quad \text{with} \quad \sum_{j=1}^{4} \alpha_{nj} = 1 \quad \text{and} \quad n = 1, ..., 5 \tag{20}$$

where C_j^F and C_j^p are control points of a 3-D model and the image coordinate system, respectively, and α_{nj} are homogeneous barycentric coordinates. Control points can be chosen arbitrarily or aligned with the principal direction of the data. Let $\{\mathbf{i}_n\}_{n=1,...,5}$ be the 2-D projection of reference points $\{\mathbf{F}_n\}_{n=1,...,5}$ in the 3-D model. Then we have that

$$\omega_n \begin{bmatrix} \mathbf{i}_n \\ 1 \end{bmatrix} = A p_n = A \sum_{j=1}^{4} \alpha_{nj} C_j^p \ , \tag{21}$$

or

$$\omega_n \begin{bmatrix} i_n \\ j_n \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{nj} \begin{bmatrix} x_j^p \\ y_j^p \\ z_j^p \end{bmatrix} \ , \tag{22}$$

where ω_i are scalar projective parameters. Although four pairs of points in both world- and image-coordinate systems are sufficient to solve a PnP problem, we use five points (ears' top, nose tip, and mouth corners) in order to maintain both redundancy and robustness towards image noise, or to reduce the impact of errors from the ASAAM stage. Furthermore, those five points cover a wider region of the face, also with different depth values. Before proceeding to the EP5P solution, we propose a new point-set transform

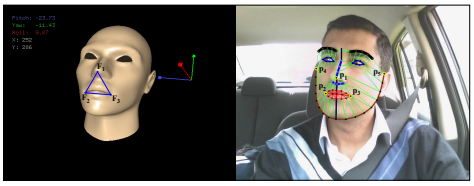

Figure 5. Determining roll, yaw, and pitch of a driver's face based on ASAAM, driver's face-shape normalization, and EPnP.

and normalization to minimize the 3-D model's *matching error* with the actual face shape. The final objective is to gain a more accurate pose estimation, and to avoid *model matching divergence* due to residual errors. After solving Eq. 12, we rescale a driver's face shape (obtained from the ASAAM stage) to match the points p_4 and p_5 to the known corresponding points F_4 and F_5 in the 3-D model (Fig. 5). However, due to face-shape variations, we can expect that the remaining triple points (p_1, p_2, and p_3) will not exactly match with the corresponding points in the 3-D model (F_1, F_2, and F_3).

This is especially a matter of concern for the nose tip (p_1), as the nose length may highly vary from face to face. As a novel contribution, we adapt the Fermat-Torricelli problem [13] from geometry into computer vision to minimize the model matching error.

After finding the Fermat points and the isogonic centers $\mathbf{P}_1$ and $\mathbf{P}_2$ for the triangles $\Delta p_1 p_2 p_3$ and $\Delta F_1 F_2 F_3$, we translate $p_1 p_2 p_3$ (with no rotation) such that $\mathbf{P}_1$ matches $\mathbf{P}_2$. Therefore we have that

$$\mathbf{P}_{x,y} = \arg \min_{x,y \in \mathbb{R}} \left\{ \sum_{n=1}^{3} \sqrt{(F_n^x - p_n^x)^2 + (F_n^y - p_n^y)^2} \right\} . \tag{23}$$

Thus, triple points p_1, p_2, p_3 are translated in an order such that they have the minimum possible distance to corresponding points F_1, F_2, F_3 in the 3-D model. Based on these new translated points, we wrap the other facial points accordingly. We name this process as *Fermat-transform*.

Since a vehicle's driver does not change during a driving course, we apply the same relative Fermat-transform and scaling to all the input faces and all the pre-calculated Delaunay triangles, with respect to the Fermat point $\mathbf{P}$. This guarantees that the face shape-model matches our 3-D model as close as possible, while we keep p_4 and p_5 unchanged at their original locations. Figure 5 shows a sample output of a driver's attention estimation based on the techniques proposed in Sections 3, 4.1, and 4.2.

5. Vehicle Detection and Distance Estimation

This section addresses *road-scene monitoring* as the second component of our driver-assistance system.

Road scenes are typically a highly dynamic environment with moving vehicles as potential hazards. Developing a vehicle classifier based on our recently proposed

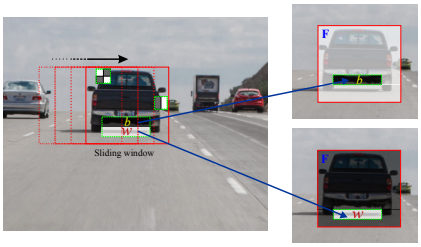

Figure 6. Comparison of standard Haar-like features and global Haar-features.

idea of *global Haar-like features* (GHaar) [14], and using an inverse-perspective mapping [15], we not only can detect the leading vehicles on the road but we can also estimate the distance to the vehicles using a monocular camera only.

For any given standard Haar feature [16], we define two global Haar-like features as follows:

$$F_b = S_F - S_{b_i} \quad \text{and} \quad F_w = S_F - S_{w_i} \ , \qquad (24)$$

where S_F is the integral value of the sliding window [14] (Fig. 6), and S_{b_i}, and S_{w_i} are the integral values of the black and white (b and w) patches, respectively.

We use global features in conjunction with the local ones, which provide global information about the query window as an added value to the local intensity information via a standard Haar feature. Integration of the standard Haar and global Haar features leads to a higher rate of true positives, less false alarms, and 30% faster performance [14].

6. Driver-Assistance by In-Out Fuzzy Fusion

In this section we discuss how to assess the *risk level* of a given driving scenario based on five inputs of driver's distraction monitoring (yaw, roll, pitch, yawning, head nodding), and two inputs about road situations (distance, and the angle of the detected vehicles to the ego vehicle).

Dealing and judgment for determining the crash risk based on all the above information could be a highly complex problem. In related work by Fletcher *et al*. [17] and Mori *et al*. [18], the authors judge based on multi-sensor information to detect road speed signs, obstacles on the road, and the time to collision *(TTC)*. An alarm is raised if the TTC is smaller than a given threshold, and the driver is looking into the opposite direction of the road.

Some of these warnings are false alarms due to inaccurate judgment, or some could be too late as the driver's reaction time could vary, depending on the driving situation, or the driver's level of awareness. In addition, dealing with all the above information in a strictly mathematical sense, could be complicated or non-verifiable.

Studying the driving behaviour of an expert driver, one can confirm that a driver neither thinks about accurate measurement of distance to obstacles, nor calculates the TTC. Instead, a driver uses linguistic expressions such as *very far, far, close, or very close* to interpret distance to hazards, or

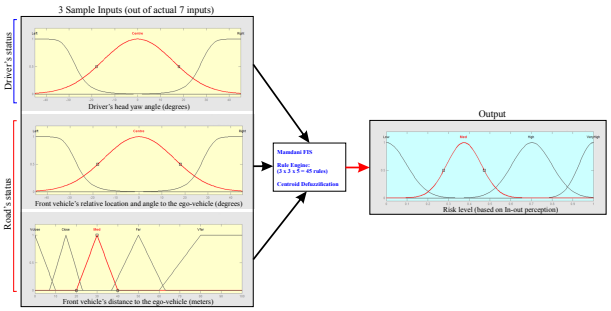

Figure 7. Three sample fuzzy inputs (yaw, distance, and angle), 11 fuzzy membership functions, Mamdani rule-engine, and 4 risk levels as the output of the proposed FIS.

may consider *very high, high, average,* or *low* to express a relative speed.

Based on such approximations, the driver decides on how much to push on the accelerator pedal, how much to push the brake pedal, or how to adjust the steering angle to escape a hazard. In other words, any judgment by an expert driver is based on such approximations concatenated with some simple **if-then** rules in the driver's mind. The results are sufficiently accurate to prevent a potential crash.

We suggest that the driving behaviour can be extrapolated by the concept of *fuzzy logic* [19]. Using Gaussian, trapezoid, and triangular membership functions, Mamdani rules, min/max norm operators, and centroid defuzzification, we modeled our fuzzy inference system (*FIS*) based on the seven existing inputs of *driver-road* information.

Figure 7 illustrates the overall structure of the module as a *decision-level* fusion, based on three sample inputs out of all the seven inputs.

7. Experimental Results

Using two monocular cameras, one facing toward the road, and another one facing toward the driver, Figs. 8 and 9 show experimental results for the techniques discussed in Sections 3, 4, and 5.

By considering the already detected feature points for a driver's face, we performed yawning detection as per Fig. 8, the 4^{th} row, based on measuring mouth openness over a continued period of time (τ = 1.5 sec):

$$f(n) = \begin{cases} t & \text{if } \frac{d(a,a')}{d(p_2,p_3)} \geq 0.6 \text{ and } \Delta t \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

were n is the frame number, and Δt is continuous time elapsed since the first detection of wide-mouth openness (i.e., $f(n) \neq 0$). A similar approach is used for "head nodding" detection using the relative length of the *nose* to the distance of *nose tip to upper lip*.

Figure 9 illustrates successful vehicle detection and distance estimation, even for challenging rainy night conditions, using the proposed hybrid GHaar classifier and inverse perspective mapping.

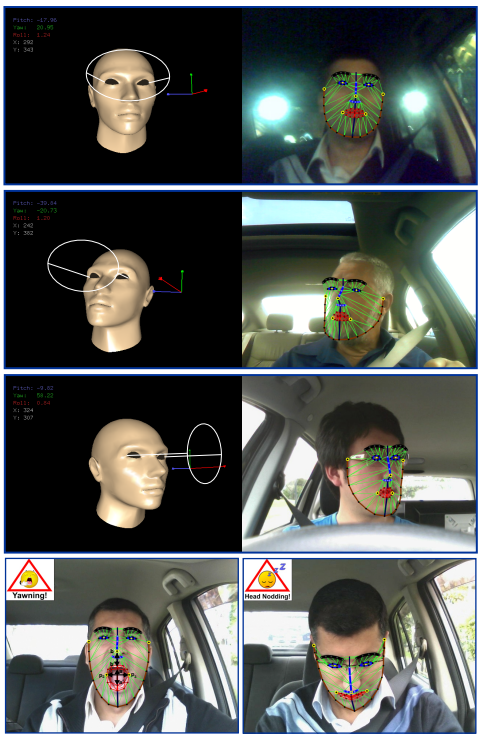

Figure 8. Pose estimation, yawning, and head-nodding detection.

As discussed in Sec. 6, we have seven inputs for the system (yaw, roll, pitch, yawning, head nodding, vehicle distance, and vehicle angle). In order to reduce the complexity of the discussion, Fig. 7 only illustrates three selected inputs of the system (driver's head yaw angle, distance to the closest vehicle, and the angle of ego-vehicle to the lead vehicle). Similarly, Fig. 10, shows the defuzzified output of the *risk-level* for only four dimensions out of our 8-D system (7 input, 1 output). Looking at any sample point within the defuzzified surface plots, it can be confirmed that the calculated *risk-level* is "quite rational", depending on the input parameters about road and driver's status. Applying fuzzy fusion for all the available inputs, the whole system can guide a distracted driver under various high-risk traffic conditions.

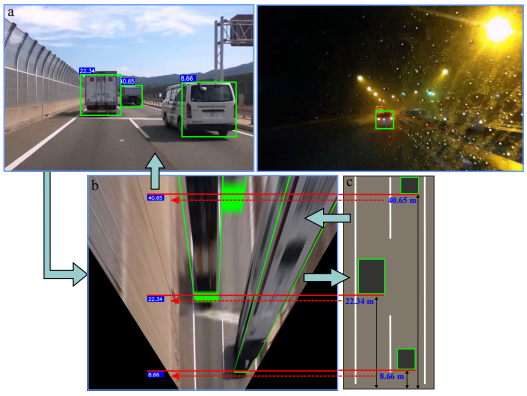

Figure 9. **(a)** Vehicle-detection based on *GHaar*. **(b)** Monocular *bird's eye view* image. **(c)** Distance estimation.

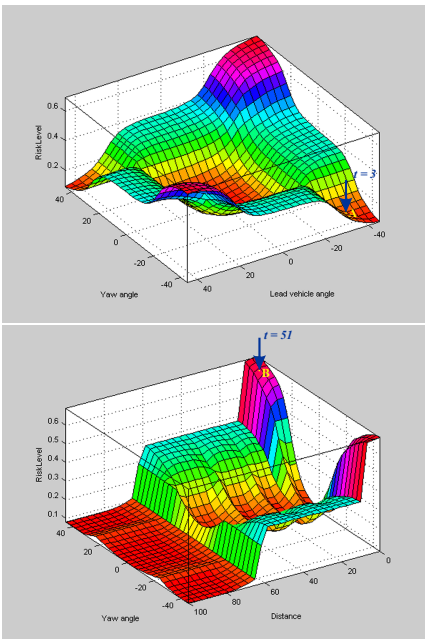

Figure 10. Defuzzified surface plot of the risk-level, based on driver's attention direction and road situation.

Figure 11 demonstrates the obtained graphs after processing two 60-second simultaneous video recordings from a driver's face and road conditions. We used the processed data as sample inputs for the proposed fuzzy fusion system.

In the assessed videos, the ego-vehicle was moving on the right lane of the road. When defining the angle of detected vehicles, we refer to the right lane of the road. Figure 11.*c*, shows three detected vehicles in front of the ego-vehicle with respect to their distance to the ego-vehicle.

As shown, within the seconds 0-4, the vehicle number 2 (V2) has a very close (apparently *high risk*) distance of $d \approx 5m$ to the ego-vehicle. At the same time (also 0-4), the driver also has a 2-second distraction toward the left-hand side of the road with $yaw = -30°$ (Fig. 11.*a*). However, the graph in Fig. 11.*b* shows an angle of approximately $42°$ which means V2 is not moving in the same lane as the ego-vehicle. Based on our empirical tests, vehicles driving in the same lane cannot have an angular deviation of more than $\pm 6°$. Similarly V1 also travels in the left lane (angle around $20°$). V3 with a small angle of about zero, is in the right lane (same lane as the ego vehicle). Figure 11.*c* confirms a distance of $d \approx 15m$, thus no high-risk warning is raised for V3 (point A in the defuzzified risk plot, Fig 10, top).

On the other hand, in t = 47 to 52, the distance of V3 sharply decreases (e.g., due to sudden braking); V3 is moving in the same lane of the ego-vehicle; also, at the same time the driver has an $\approx$ 3-second distraction to the right-hand side of the road ($yaw \approx +40°$). In such a case, the FIS fires one of the high-risk rules to warn the driver, to prevent an imminent crash (Fig. 10, point B).

Using a Core i7 system with 8GB RAM, the entire sys-

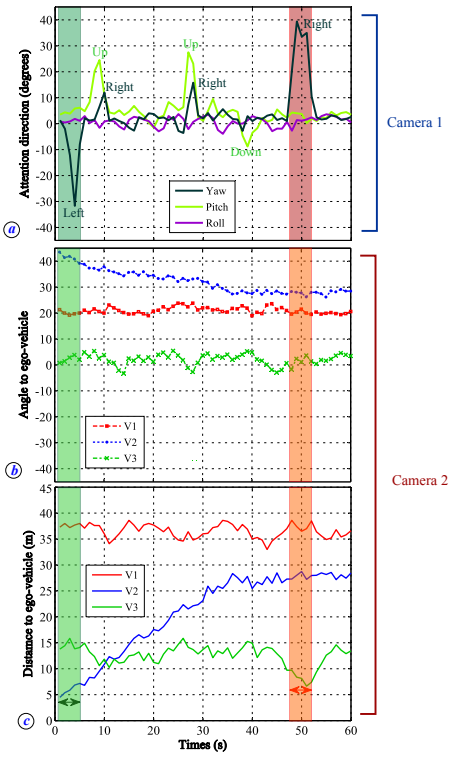

Figure 11. Processed data for simultaneous *driver* and *road* monitoring: (a) driver's head pose, (b) detected vehicles' angle to the ego-vehicle, and (c) detected vehicles' distance to the ego-vehicle.

tem was able to perform in real-time, at a speed of 21 fps.

8. Conclusion

The paper introduced accurate head-pose estimation by tackling three important weaknesses of previous related work: failure of face shape modeling due to asymmetric light variation, pose divergence due to residual errors of mismatching a 2D face to a generic 3D model, and slow operation of standard approaches for model refinement (due to a need of up to 10 iterations). We also proposed robust vehicle detection and distance estimation. All the acquired information from monocular cameras contributed simultaneously in a real-time fuzzy-fusion system to prevent road crashes. Experiments were conducted for different subjects, different driving scenarios, and various lighting conditions in day and night. The results are robust and promising. However, such a high-dimensional system may requires further evaluations.

References

[1] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake. Real-time human pose recognition in parts from single depth images. in: Proc. CVPR, 1297–1304, 2011.

[2] X. Zhu, D. Ramanan. Face detection, pose estimation, and landmarking localization in the wild. in: Proc. CVPR, 2879–2884, 2012.

[3] T. F. Cootes, G.J. Edwards, C. J. Taylor. Active appearance models. IEEE Trans. PAMI, 23(6), 681–685, 2001.

[4] J. F. Xie, M. Xie, M., W. Zhu. Driver fatigue detection based on head gesture and PERCLOS. in: Proc. IEEE ICCWAMTIP, 128–131, 2012.

[5] T. Gernoth, K. A. Martínez, A. Gooßen, R.-R. Grigat. Facial pose estimation using active appearance models and a generic face model. in Proc. VISAPP, 499–506, 2010.

[6] P. Martins, J. Batista. Monocular head pose estimation. in: Proc. ICIAR, 357–368, 2008.

[7] D. F. DeMenthon, L. S. Davis. Model-based object pose in 25 lines of code. IJCV, 15(1):123–142, 1995.

[8] J. Xiao, S. Baker, I. Matthews, T. Kanade. Real-time combined 2D+ 3D active appearance models. in: Proc. CVPR, 535–542, 2004.

[9] TurboSquid, online 3-D model database, in: `www.turbosquid.com/Search/3D-Models/face`.

[10] V. Krüger, G. Sommer. Gabor wavelet networks for efficient head pose estimation. JIVC, 20(9):665–672, 2002.

[11] V. Lepetit, F. Moreno-Noguer, P. Fua. EPnP: An accurate O(n) solution to the PnP problem. IJCV, 81(2), 155–166, 2009.

[12] E. Murphy-Chutorian, M. M. Trivedi. Head pose estimation in computer vision: A survey. IEEE Trans. PAMI, 31(4):607–626, 2009.

[13] M. Hazewinkel. Fermat-Torricelli problem, Encyclopedia of Mathematics, Springer, 2001.

[14] M. Rezaei, H. Ziaei Nafchi, S. Morales. Global Haar-like features: A new extension of classic Haar features for efficient face detection in noisy images, in: Proc. PSIVT, 302–313, 2013.

[15] R. Jiang, R. Klette, T. Vaudrey, S. Wang. New lane model and distance transform for lane detection and tracking. in: Proc. CAIP, 1044–1052, 2009.

[16] P. Viola, M. Jones. Robust real-time face detection. IJCV, 57(2):137–154, 2004.

[17] L. Fletcher, A. Zelinsky. Driver inattention detection based on eye gaze-road event correlation. Robotics Research, 28(6):774–801, 2009.

[18] M. Mori, C. Miyajima, P. Angkititrakul, T. Hirayama, L. Yiyang, N. Kitaoka, K. Takeda. Measuring driver awareness based on correlation between gaze behavior and risks of surrounding vehicles. in: Proc. ITSC, 644–647, 2012.

[19] L. A. Zadeh. Fuzzy Sets, Fuzzy Logic, Fuzzy Systems. World Scientific Press, 1996.

[20] MUCT face dataset, in `www.milbo.org/muct/`.

[21] L. Teijeiro-Mosquera, J. L. Alba-Castro. Recursive pose dependent AAM: Application to drivers' monitoring. in: Proc. IEEE IVS, 356–361, 2011.

[22] E. Murphy-Chutorian, M. M. Trivedi. Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness. IEEE Trans. ITS, 11(2):300-311, 2010.

[23] Visage Tech. `www.visagetechnologies.com/`.

[24] M. Rezaei, R. Klette. Simultaneous analysis of driver behaviour and road condition for driver distraction detection. IJDIF, 2(3):217-236, 2011.