A Reduced Feature Set for Driver Head Pose Estimation

Katerine Diaz-Chito[a,b,*], Aura Hernández-Sabaté[a,b], Antonio M. López[a,b]

[a] *Centre de Visió per Computador*
[b] *Universitat Autònoma de Barcelona*

Abstract

Evaluation of driving performance is of utmost importance in order to reduce road accident rate. Since driving ability includes visual-spatial and operational attention, among others, head pose estimation of the driver is a crucial indicator of driving performance. This paper proposes a new automatic method for coarse and fine head's yaw angle estimation of the driver. We rely on a set of geometric features computed from just three representative facial keypoints, namely the center of the eyes and the nose tip. With these geometric features, our method combines two manifold embedding methods and a linear regression one. In addition, the method has a confidence mechanism to decide if the classification of a sample is not reliable. The approach has been tested using the CMU-PIE dataset and our own driver dataset. Despite the very few facial keypoints required, the results are comparable to the state-of-the-art techniques. The low computational cost of the method and its robustness makes feasible to integrate it in massive consume devices as a real time application.

Keywords: Head pose estimation, driving performance evaluation, subspace based methods, linear regression

1. Introduction

Driver fatigue/drowsiness and distraction are known to be behind a large amount of traffic accidents. Accordingly, different systems have been developed to detect such situations [1, 2, 3, 4]. Distractions are specially challenging because many times are difficult to predict in advance since they may be due to sudden events in the environment or in the cabin. Indeed, a more general challenge including distractions is driving performance. Evaluation of driving performance is of utmost importance in order to reduce road accident rate. Behavior analysis while driving generally points out the abilities of the driver, which include cognitive (attention, executive functions, and memory) and (visual-spatial) perception skills, as well as, their fatigue levels or attention capability [5]. These abilities can be analyzed from several points of view, either measuring non-visual features like heart rate variability [6], or analyzing visual features such as eye blinking behavior [7], gaze direction estimation [8] or analysis of motion of the hands [9] or the feet [10]. In particular, head pose is a crucial indicator of driver's field of view and his/her attention focus [11] and, like most of the indicators named above, it deserves further consideration.

Head pose estimation is a challenging problem in itself due to the variability introduced by factors such as driver's identity and expression, cabin and outdoor illumination, etc. [12]. In fact, during the last decade there has been an increasing interest in developing methods to estimate head pose [13] for different applications such as security and surveillance systems [14], meeting rooms [15], intelligent wheelchair systems [16], and driving monitoring [1, 17]. In the particular case of driving performance, when drivers are paying attention to the road ahead, their facial direction is within approximately $\pm 15^\circ$ from the straight normal [18]. Thus, the yaw angle of the driver could contribute to determine if he/she perceives road elements such as traffic lights, roundabouts, crosswalks and the attention he/she devotes. Accordingly, in this paper we focus on the computation of such angle from still images. Such a method can be very useful as part of a multi-cue system [19, 20] for early detection of abnormal driving performance in common situations. For example, if the driver does not pay attention to the correct direction in a roundabout, or if he/she attends in a crosswalk but does not see a pedestrian crossing on, an Advanced Driver Assistance

*Corresponding author
Email addresses: kdiaz@cvc.uab.es (Katerine Diaz-Chito), aura@cvc.uab.cat (Aura Hernández-Sabaté), antonio@cvc.uab.cat (Antonio M. López)

System (ADAS) combining driving monitoring and exterior hazard assessment, can decide to elevate the warning level or even braking the car.

1.1. Related work and contribution

Murphy-Chutorian and Trivedi [13] divide head pose estimation methods in 8 categories. Three of them are regression methods, geometric methods and manifold embedding ones.

Regression methods apply a regression model on a training set in one or more directions (angles). The regression model usually is a non-linear one, such as, Support Vector Regressors [21], Sparse Bayesian Regression [22] or Neural Networks [23]. One drawback of these methods is that they take into account the whole face image, so that its high dimensionality decreases the efficiency. In some cases, the dimensionality can be reduced, like when the face is well localized. Still, this high dimensionality makes not clear if an specific regression tool is capable of learning the proper curve modeling the directions arch.

Geometric methods are an alternative to directly explode most influencing properties on human head pose estimation, which are usually based on human perception. These features can be divided on two types, those based on face appearance, such as, orientation information of head images [24, 25], skin color histogram [26] or facial contours [27], and those relying on a set of (usually 5-6) local facial keypoints [28]. In the first case, computational cost is still high, since they need to analyze the whole face image. In the second case, facial features detection needs to be highly precise and accurate. To overcome the limitations of a single category method, manifold embedding methods are usually combined with them, gaining in accuracy [12, 29, 30].

In the same fashion, this paper combines the three methods explained above to estimate the continuous angle of head pose of a driver. Roughly, given an image of the driver's head, we rely on a small set of geometric features computed from just three representative facial keypoints, namely the center of the eyes and the nose tip. Our method is based on a combination of subspace projections, as Fisher's Linear Discriminant (FLD) and Principal Component Analysis (PCA), as well as multiple linear regression adjusted for each pose interval. Figures 1 and 2 sketch the main steps of the method, split in training the system and testing new samples. For the training (Fig.1), from a set of samples, we extract the facial keypoints to compute a geometric feature vector. A projection of the samples on a FLD allows the system to suppress some samples not useful to train. Then, the new set of samples is projected on another subspace

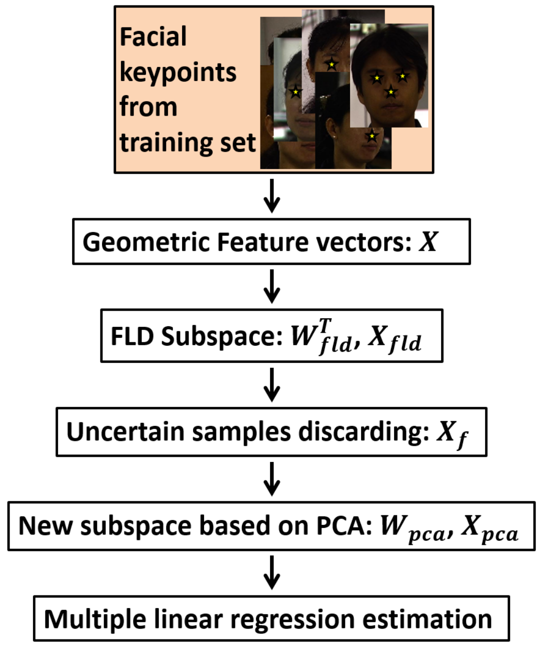
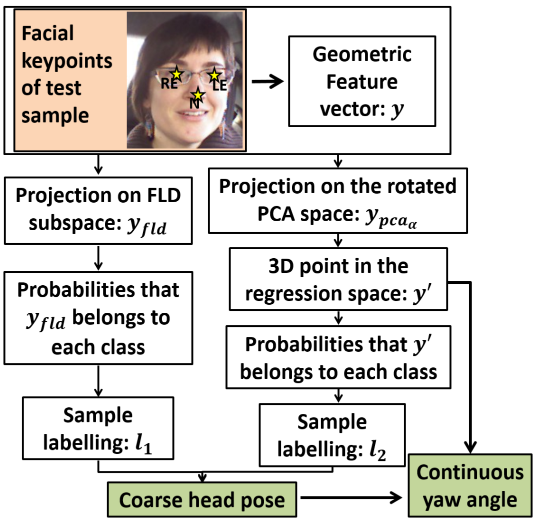

Facial keypoints from training set
Geometric Feature vectors: X
FLD Subspace: W_{fld}^T, X_{fld}
Uncertain samples discarding: X_f
New subspace based on PCA: W_{pca}, X_{pca}
Multiple linear regression estimation

Figure 1: Workflow of the system training

based on PCA and a multiple linear regression is computed to estimate the regression parameters. When a new sample is given (Fig.2), it is projected and then classified on the FLD subspace and on the one based on PCA. A combination of both classifications gives us the final coarse yaw angle estimation while the regression parameters serve to compute the continuous yaw angle. Besides, the method integrates a mechanism to self-evaluate the likelihood of the generated hypothesis and discard non likely poses by comparing the discrete angle obtained from FLD and the continuous angle from the regression.

The analysis of the results assessing the reliability of the method shows that, although the very few facial keypoints required, the approach has high accuracy and precision, which makes it comparable to the methods present in the literature. Besides, the computational cost is as low as it can run in real time, making easy to integrate it in massive consume devices such as tablets or mobiles and be part of a multi-cue system for driving performance evaluation. As well, the robustness of the method against noise in the facial keypoint detection is proven.

The remains of the paper are organized as follows. Section 2 describes the mathematical tools involved in the driver's yaw angle estimation. Section 3 describes the detection of the facial keypoints and the geometric features derived from them while section 4 presents the



Facial keypoints of test sample
RE
LE
N
Geometric Feature vector: y
Projection on FLD subspace: y_{fld}
Projection on the rotated PCA space: y_{pca_α}
3D point in the regression space: y'
Probabilities that y_{fld} belongs to each class
Probabilities that y' belongs to each class
Sample labelling: l_1
Sample labelling: l_2
Coarse head pose
Continuous yaw angle

Figure 2: Workflow of the test step

workflow of the method. The experimental setting and the measures used to assess the reliability of the method are detailed in section 5. Results and their analysis, are shown in section 6, while the method is compared with the ones in the literature in section 7. Last section, 8, is devoted to conclude the paper with some final remarks.

2. Mathematical Tools

In this section, we explain the mathematical tools used along the method.

2.1. Subspace based methods

Statistical methods based on subspaces have been broadly used in computer vision and related fields for recognition and classification tasks due to their appealing capabilities and good practical behavior. Among the most popular methods we find FLD and PCA.

FLD computes a linear transformation that minimizes the scatter of the samples within each class, while maximizing the scatter between classes. The main goal of FLD is to find a projection matrix, W_{fld}, of the linear subspace W that maximizes the Fisher's criterion:

$$\mathrm{W}_{fld} = \underset{\mathrm{W}}{arg\ max} \frac{|\mathrm{W}^T S_B \mathrm{W}|}{|\mathrm{W}^T S_W \mathrm{W}|} \tag{1}$$

where $S_B = \sum_{j=1}^{c}(\overline{\mathrm{x}}^j - \overline{\mathrm{x}})(\overline{\mathrm{x}}^j - \overline{\mathrm{x}})^T$ is the between-classes scatter matrix and $S_W = \sum_{j=1}^{c}\sum_{i=1}^{m_j}(\mathrm{x}_i^j - \overline{\mathrm{x}}^j)(\mathrm{x}_i^j - \overline{\mathrm{x}}^j)$ is the within-class scatter matrix. c is the number of classes, m_j the number of samples of the jth class and the total number of samples is $M = \sum_{j=1}^{c} m_j$. x_i^j is the ith sample of the jth class, $\overline{\mathrm{x}}^j$ is the mean vector of the samples of the jth class, and $\overline{\mathrm{x}}$ is the mean vector of all the samples.

If S_W is not singular, the Fisher's criterion is maximum when the vectors in W_{fld} are the eigenvectors associated at the non-zero eigenvalues of $(S_W^{-1} S_B)$. The new vectors (projected samples) are linear combinations of the original ones:

$$\mathrm{x}_{fld}^i = \mathrm{W}_{fld}^T\, \mathrm{x}_i \qquad i = 1, \ldots, M \tag{2}$$

PCA is used for dimensionality reduction, compression and feature extraction, which preserves the maximum variability in the original sample space (vectors). A high correlation of input vectors involves redundant information. The PCA method reduces this redundancy by uncorrelating these vectors. PCA seeks to maximize the total scatter of the projected vectors by the following criteria:

$$\mathrm{W}_{pca} = \underset{\|\mathrm{w}\|=1}{arg\ max} |\mathrm{W}^T S_T \mathrm{W}| \tag{3}$$

where $S_T = \sum_{i=1}^{M}(\mathrm{x}_i - \overline{\mathrm{x}})(\mathrm{x}_i - \overline{\mathrm{x}})^T$ is the total scatter matrix. $\mathrm{W}_{pca} \in \mathbb{R}^{d\times r}$ is a matrix with r orthogonal columns, $r < d$.

The above criteria is maximum if the column vectors that form W_{pca} are the r first eigenvectors associated to the r first eigenvalues of S_T, ordered by descending value.

The new vectors (transformed samples) are linear combinations of the original ones and are constructed in the order of importance given by the total scatter of the vectors, as:

$$\mathrm{x}_{pca}^i = \mathrm{W}_{pca}^T\, \mathrm{x}_i \qquad i = 1, \ldots, M \tag{4}$$

2.2. Regression methods

Regression models describe the relationship between a dependent variable, Y, and one or more independent (explanatory) variables. In the particular case of multiple linear regression, the general model can be written as follows:

$$Y = \mathbf{X}\boldsymbol{\beta} + \varepsilon \tag{5}$$

which is equivalent to:

$$\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_d \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

and

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_d x_{id} + \varepsilon_i$$

is the ith response, $i = 1,\ldots,n$, β_k is the kth regression coefficient, $k = 1,\ldots,d$, x_{ik} is the ith observation on the independent variable x_k, and ε_i is the ith noise term, which models the random error.

The regression parameters can be estimated as:

$$\beta = (\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T\,Y \tag{6}$$

3. Features

The facial keypoints we use are the left eye center (LE), right eye center (RE), and nose tip (N) as shown in Fig.3(a). These keypoints are enough to calculate 10 geometric features consisting of Euclidean distances, ratios, differences and angles. Figure 3(b)-(d) shows the

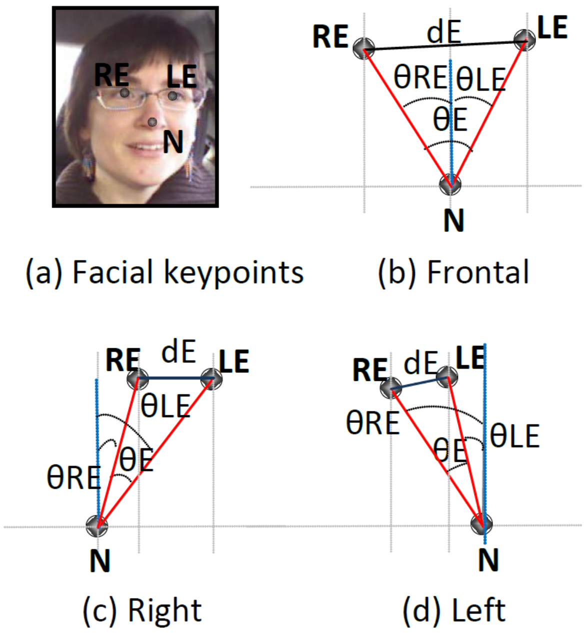


Figure 3: (a) Facial keypoint set used for head pose estimation. (b)-(d) Features directly computed using RE, LE and N facial keypoints, (b) right view, (c) frontal view and (d) left view.

features that can be directly extracted from RE, LE and N detection in the three different views, right, frontal and left. Let $\mathrm{x} = [$d(RE, N), θE, θRE, θLE, r(dE, θE), r(θE, θRE), r(θE, θLE), r(θRE, θLE), r(dRE, dRL, θE), s(θLE, θLR)$]^T \in \mathbb{R}^{10}$ be a 10-dimensional vector with the geometric features described in Table 1.

4. Head Pose Estimation

The method presented in this paper for estimating head yaw angle is split in two main steps, one for training the system and another one for computing the angle of the head in a new image.

4.1. Model Training

Let $\mathrm{X} = (\mathrm{x}_1,\ldots,\mathrm{x}_M) \in \mathbb{R}^{d\times M}$ be a matrix of M d-dimensional vectors (samples) with the geometric features proposed above, partitioned according to c possible classes associated to c discrete yaw angles of the head. That is:

$$\mathrm{X} = (\mathrm{x}_1^1,\ldots,\mathrm{x}_{m_1}^1,\ldots,\mathrm{x}_1^c,\ldots,\mathrm{x}_{m_c}^c) \in \mathbb{R}^{d\times M}$$

Briefly, our method proceeds as follows. The samples are projected into a subspace easier to classify by means of FLD and, there, *uncertain* samples are removed by k-NN. Afterwards, dimensionality reduction is performed by PCA. The resulting subspace is further aligned (rotated) and a new dimension is generated in the transformed subspace. Finally, the piecewise multiple linear regressors that output yaw angles are computed given the transformed (FLD-PCA-Rotated) geometric features. The details are described below and Figure 4 shows the main space transformations and projections.

First, the geometric feature vectors of the training set are transformed into a FLD subspace following equation (1) for finding the projection matrix, W_{fld}:

$$\mathrm{X}_{fld} = \mathrm{W}_{fld}^T\mathrm{X} \ \in \mathbb{R}^{d\times(c-1)}$$

The transformed vectors keep their class label, but this allows us to remove uncertain transformed ones. In particular, for each transformed vector we apply a k-NN classifier, and if the classification result does not match its class label we consider it as uncertain and, thus, it is no used for the rest of the process. Figure 4(a) shows the first two dimensions of our sample set in the FLD subspace, partitioned in 5 classes. Uncertain samples (outliers) are marked in green.

The resulting matrix, $\mathrm{X}_f = (\mathrm{x}_1,\ldots,\mathrm{x}_{M_f}) \in \mathbb{R}^{d\times M_f}$, $M_f \leq M$, is then transformed into its PCA subspace, following equation (4):

$$\mathrm{X}_{pca} = \mathrm{W}_{pca}^T\mathrm{X}_f \ \in \mathbb{R}^{r\times M_f}$$

where r is the new dimension of the reduced subspace so that $r \leqslant d$. In order to keep the 90% of the cumulative energy, we experimentally found that only the two first components are needed. Therefore, from now on we can assume $r = 2$.

In order to distribute the samples as symmetrical as possible with respect to the second principal component, and orthogonal to the first one, the samples in X_{pca}

Table 1: Description of geometric features

Geometric Feature	Description
d(RE, N)	normalized distance between RE and N
θE	angle between RE, LE and N
θRE	angle between RE and the vertical axis from N
θLE	angle between LE and the vertical axis from N
r(a1, a2)	ratio between values a1 and a2
s(θLE, θLR)	the result of subtracting θLE to θLR
dE	Euclidean distance between RE and LE
r(dRE, dRL, θE)	(1 - dRE/dRL) + θE
dRE	normalized distance between RE and N
dLE	normalized distance between LE and N

are rotated an angle α from the first principal component. Thus, α is calculated from the *arctan* (in degrees) of the slope of the line joining the center of the clusters that are farthest away. That is, if $C_1 = (pc_{1,1}, pc_{1,2})$ and $C_2 = (pc_{2,1}, pc_{2,2})$ are the centroids of two clusters containing the samples that are farthest away, α is:

$$\alpha = arctan\left(\frac{pc_{2,2} - pc_{2,1}}{pc_{1,2} - pc_{1,1}}\right) \quad (7)$$

Those clusters are computed using a Fuzzy C-means clustering [31] into the PCA subspace. In this way the rotated samples are:

$$\begin{aligned} \mathrm{X}_{pca_\alpha} &= \mathrm{R}_\alpha \mathrm{X}_{pca} \quad (8) \\ &= \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} \mathrm{X}_{pca} \end{aligned}$$

where R_α is the rotation matrix. Figure 4(b) shows the samples transformed in the PCA subspace before rotating them. As well, the centroids of the two clusters that are farthest away are marked in green and the slope of the line joining them shows the angle α to rotate the samples.

The new feature representation lie on a nonlinear manifold in a 2-dimensional subspace that are still a bit superposed. Thus, in order to discriminate the samples and obtain a better feature representation, a new dimension is generated and added to the X_{pca_α} plane.

This discriminative dimension is computed by means of a nonlinear function with the same sign of the first component (that is, positive when $\mathrm{X}^{(1)}_{pca_\alpha} > 0$ and negative when $\mathrm{X}^{(1)}_{pca_\alpha} < 0$). In particular, the function chosen is the four-quadrant inverse tangent $atan2(y,x)$, which computes the angle between the positive x-axis of a plane and the point given by the coordinates (x,y). This function is positive for the upper half-plane, $y > 0$, and negative for the lower half-plane, $y < 0$. Unlike the single *arctan* function, it returns the appropriate quadrant of the computed angle. Then, the new dimension for each sample is computed as:

$$\mathrm{f}_i = atan2(\mathrm{X}_{pca_\alpha}(1,i), \mathrm{X}_{pca_\alpha}(2,i)) \quad (9)$$

and $\mathrm{F} = (\mathrm{f}_1, \ldots, \mathrm{f}_{M_f}) \in \mathbb{R}^{M_f}$. So the new representation of samples is:

$$\mathrm{X}' = \begin{pmatrix} \mathrm{X}_{pca_\alpha} \\ \mathrm{F} \end{pmatrix} \in \mathbb{R}^{3\times M_f}$$

Figure 4(c) shows the samples set in the new representation space. A hyperplane is adjusted for each class following the linear regression model explained in subsection 2.2 and, accordingly to equation (6), regression parameters are estimated for each class, obtaining the following matrix:

$$\mathrm{B} = (\boldsymbol{\beta^1}, \ldots, \boldsymbol{\beta^c}) \in \mathbb{R}^{3\times c} \quad (10)$$

In this case, we force that the independent term for each class be the own angle of the class.

The result of all this training process is the following: a W_{fld} projection matrix on FLD subspace, a W_{pca} projection matrix on PCA subspace, a R_α rotation matrix for the two principal components, a new samples set X' and the regression coefficients B. The steps of the whole process are shown in Algorithm 1, where the input of the algorithm is the set of training samples $\mathrm{X} \in \mathbb{R}^{10\times M}$ with their corresponding discrete angle $\boldsymbol{\Phi} \in \mathbb{R}^M$.

4.2. *Test process*

In the test process, the head pose of new samples is classified, and the yaw angle is estimated as well. Following the work flow shown in figure 2, firstly the geometric feature vector, $\mathrm{y} \in \mathbb{R}^d$, is computed. The label of the test sample is predicted as a coarse head pose by

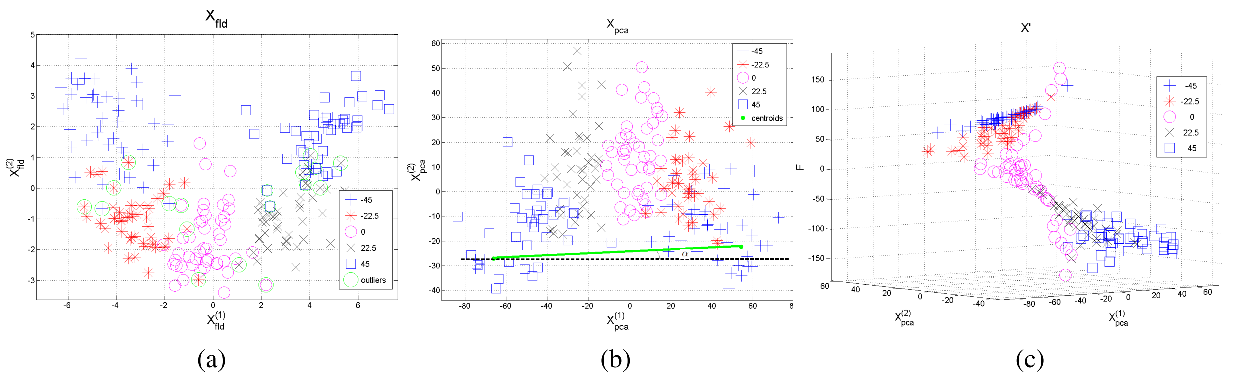


Figure 4: (a) First two dimensions of the geometric feature vectors into the FLD subspace. (b) Non-uncertain samples in the PCA subspace, centroids of the two clusters that are farthest away (green mark), and α rotation angle. (c) Final samples used in the regression.

```
Algorithm 1 Model Training
Input: $\mathrm{X} \in \mathbb{R}^{d\times M}$, $\boldsymbol{\Phi} \in \mathbb{R}^{M}$.
Output: $\mathrm{W}_{fld} \in \mathbb{R}^{d\times(c-1)}$, $\mathrm{W}_{pca} \in \mathbb{R}^{d\times 2}$, $\mathrm{R}_{\alpha} \in \mathbb{R}^{2\times 2}$, $\mathrm{B} = (\boldsymbol{\beta}^{1},\dots,\boldsymbol{\beta}^{c}) \in \mathbb{R}^{3\times c}$.

1: Transform the space containing the original sample set X into a FLD subspace, obtaining the projection mapping $\mathrm{W}_{fld} \in \mathbb{R}^{d\times(c-1)}$ and the transformed sample set $\mathrm{X}_{fld} = \mathrm{W}_{fld}^{T}\,\mathrm{X}$.
2: Remove outliers into the FLD subspace by means of $k$-NN to obtain $\mathrm{X}_{f} \in \mathbb{R}^{d\times M_f}$, $M_f \leq M$.
3: Transform the space containing the reduced sample set $\mathrm{X}_f$ into a PCA subspace, obtaining the projection mapping $\mathrm{W}_{pca} \in \mathbb{R}^{d\times 2}$ and the transformed sample set $\mathrm{X}_{pca} = \mathrm{W}_{pca}^{T}\,\mathrm{X}_f$.
4: Find the centroids $C_1 = (pc_{1,1}, pc_{1,2})$ and $C_2 = (pc_{2,1}, pc_{2,2})$ (using a Fuzzy C-means clustering) of the samples that are farthest away into $\mathrm{X}_{pca}$ .
5: Calculate the rotation angle, $\alpha$, from the $arctan$ in degrees of the slope of the line joining the centroids, $\alpha = arctan(pc_{2,2} - pc_{2,1}/pc_{1,2} - pc_{1,1})$
6: Rotate the transformed samples $\mathrm{X}_{pca}$,
   $$\mathrm{X}_{pca_\alpha} = \mathrm{R}_{\alpha}\,\mathrm{X}_{pca} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} \mathrm{X}_{pca}$$
7: Generate the transformed training set,
   $$\mathrm{X}' = \begin{pmatrix} \mathrm{X}_{pca_\alpha} \\ \mathrm{F} \end{pmatrix} \in \mathbb{R}^{3\times M_f}$$
   where $\mathrm{F} = (\mathrm{f}_1,\dots,\mathrm{f}_{M_f}) \in \mathbb{R}^{M_f}$ and $\mathrm{f}_i = atan2(\mathrm{X}_{pca_\alpha}(1,i), \mathrm{X}_{pca_\alpha}(2,i))$.
8: For each class, compute the regression coefficients by Eq.(6) to obtain $\mathrm{B} = (\boldsymbol{\beta}^{1},\dots,\boldsymbol{\beta}^{c}) \in \mathbb{R}^{3\times c}$ .
9: Return $\mathrm{W}_{fld}$, $\mathrm{W}_{pca}$, $\mathrm{R}_{\alpha}$, B
```

means of the combination of two probabilities of belonging to a class, the one obtained after projecting the vector on the FLD subspace and the one obtained after projecting it on the rotated and extended PCA subspace.

That is, on the one hand, the feature vector y is projected on the FLD subspace,

$$\mathrm{y}_{fld} = \mathrm{W}_{fld}^{T}\,\mathrm{y}$$

Using a k-NN classifier, the probabilities that y_{fld} belongs to each class c are computed, and their maximum is chosen:

$$m_1 = \max_{c}(P(\mathrm{y}_{fld} \in c))$$

The class label corresponding to m_1 is denoted as l_1.

On the other hand, the feature vector y is projected on the regression space in two steps. First, it is projected on the rotated PCA

$$\mathrm{y}_{pca_\alpha} = \mathrm{R}_{\alpha}\,\mathrm{W}_{pca}^{T}\mathrm{y}$$

Second, as in the training process, a third coordinate is added to y_{pca_α}:

$$\mathrm{f}_{\mathrm{y}} = atan2(\,\mathrm{y}_{pca_\alpha}(1,1),\ \mathrm{y}_{pca_\alpha}(2,1))$$

and the 3D point in the regression space is

$$\mathrm{y}' = \begin{pmatrix} \mathrm{y}_{pca_\alpha} \\ \mathrm{f}_{\mathrm{y}} \end{pmatrix}$$

The maximum of the probabilities that y' belongs to each class c is computed by means of k-NN:

$$m_2 = \max_{c}(P(\mathrm{y}' \in c))$$

and a new label, l_2, is obtained.

Finally, both labels, l_1 and l_2, are compared and the discrete angle, φ_d is computed. If l_1 and l_2 do not coincide, the most likely class among both cases decides the classification. If both classifiers assign the same probability to the sample but for two different classes, the sample is labeled as uncertain. The result can be summarized as:

$$\varphi_d = \begin{cases} \varphi(l_1), & l_1 = l_2 \\ \varphi(l_1), & l_1 \neq l_2, m_1 > m_2 \\ \varphi(l_2), & l_1 \neq l_2, m_1 < m_2 \\ uncertain, & \text{otherwise.} \end{cases} \quad (11)$$

where $\varphi(l_i), i = 1,2$ is the angle assigned to the jth chosen class from l_i.

The estimation of the continuous head's angle is computed by means of the regression coefficients of the jth class to which the sample belongs to:

$$\varphi_{cont} = (\beta^j)^T \mathrm{y}' \quad (12)$$

Algorithm 2 shows the steps to follow in the test phase, where the input is a test sample, $\mathrm{y} \in \mathbb{R}^d$, and the outputs, the corresponding discrete and continuous angles, φ_d and φ_{cont}, respectively.

Algorithm 2 Yaw computation of test samples

```
Input: $\mathrm{y} \in \mathbb{R}^d$.
Given: $\mathrm{W}_{fld}$, $\mathrm{W}_{pca}$, $\mathrm{R}_\alpha$, B
Output: $\varphi_d$, $\varphi_{cont}$

1: Project y into the FLD subspace, $\mathrm{y}_{fld} = \mathrm{W}_{fld}^T \mathrm{y}$.
2: Calculate the maximum of the probabilities of belonging
   to each class in the FLD subspace:
       $m_1 = \max_c (P(\mathrm{y}_{fld} \in c))$
   and associate it to the corresponding class label $l_1$.
3: Project y into the rotated PCA subspace,
   $\mathrm{y}_{pca_\alpha} = \mathrm{R}_\alpha \mathrm{W}_{pca}^T \mathrm{y}$.
4: Generate the new 3D point as $\mathrm{y}' = \begin{pmatrix} \mathrm{y}_{pca_\alpha} \\ \mathrm{f_y} \end{pmatrix}$,
   $\mathrm{f_y} = atan2(\mathrm{y}_{pca_\alpha}(1,1), \mathrm{y}_{pca_\alpha}(2,1))$.
5: Calculate the maximum of the probabilities of belonging
   to each class in the regressed space:
       $m_2 = \max_c (P(\mathrm{y}' \in c))$
   and associate it to the corresponding class label $l_2$.
6: Compute the discrete angle $\varphi_d$ by (11).
7: Calculate the continuous angle $\varphi_{cont}$ by (12).
8: Return $\varphi_d$ and $\varphi_{cont}$
```

5. Validation protocol

Our method has been tested on two **sets of data**, one based on images from a controlled scenario and the other based on images acquired while driving[1]. For the first dataset, we have considered the CMU-Pose, Illumination and Expression database [32], which contains 13 images of 68 persons that present head pose changes in the horizontal axis of $[-135° : 22.5° : 135°]$ for a total of 884 images of 640×480 pixels each. In our particular case, since the final application only requires poses within the angular area useful to drive, we only take into account 5 poses with the angles $[-45° \; -22.5° \; 0° \; 22.5° \; 45°]$, for a total of 340 images. Figure 5 illustrates the database by showing the images used for one of the subjects.

As second dataset, we have used our own one acquired while driving in real scenarios. It is composed of 606 samples of 640×480 pixels each, acquired in different days from 4 drivers (2 women and 2 men) with several facial features like glasses and beard, classified in 3 possible classes [18, 33]. The first class is the "looking-right" class and contains the head angles between $-45°$ and $-30°$. The second one is the "frontal" class and contains the head angles between $-15°$ and $15°$. The last one is the "looking-left" class and contains the head angles between $30°$ and $45°$. Figure 6 shows a driver of our dataset.

The experiments carried on to evaluate the validity of our feature set and our overall head's yaw angle estimation explore two aspects of the method: its reliability and its robustness against facial keypoint detection.

The **reliability of the method** is assessed by means of the absolute error for the continuous head pose estimation and the accuracy for the coarse one. Since we aim at evaluating our yaw estimation method independently of the procedure for detecting the required keypoints, we run the experiments using manual ground truth and using an automatic detection based on deep convolutional network cascade [34]. This method cascades three convolutional networks to make coarse-to-fine prediction. In the first level they use three individual networks to detect the whole face, eyes and nose, and nose and mouth. In the second and third levels, they refine the prediction by taking local patches centered at the previous predicted positions. It is worth to mention that this method is an external detector off-the-shelf and can effectively predict and locate our keypoints with

[1]The driver's database is available in `http://adas.cvc.uab.es/site/index.php/datasets/`

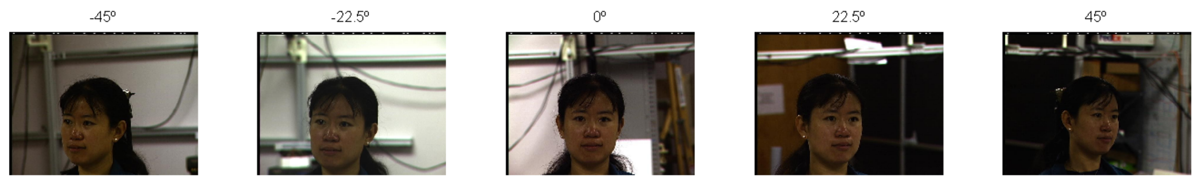


Figure 5: Images of CMU-PIE dataset for a subject with a head yaw angle of −45°, −22.5°, 0°, 22.5°, and 45°.

Figure 6: Images of a driver from our own database.

high accuracy, even when low-level features from local regions are ambiguous or corrupted.

The Absolute Error (AE) between the continuous angle estimated by our approach, φ_{cont}, and the discrete angle from the ground truth, φ, is computed as:

$$AE = \|\varphi_{cont} - \varphi\|$$

The Accuracy (Acc) of the coarse head pose estimation is computed on both datasets as:

$$Acc = \frac{TP}{N}$$

where N is the total number of test samples and TP the number of samples correctly classified.

Besides, in order to visualize the reliability of the method and the classes of missclassified samples the confusion matrix between ground truth and the estimated angle is computed for both datasets.

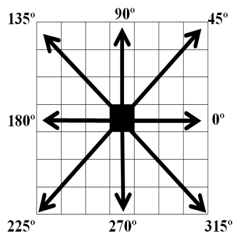


Figure 7: Principal directions along where we have added noise to each facial keypoint

The **robustness of the method** is assessed by means of the reliability of the method with added Gaussian noise to the facial keypoint detection. That is, we have moved each facial keypoint, independently of the others, along eight principal directions [0°:45°:315°] as Figure 7 shows, up to 20 pixels and we have computed the accuracy of the method. It is worth to note that the added noise is independent from the head image size, but it implicitly includes noise in the 10-dimension feature vector, since the computed geometric features are normalized.

6. Experimental Results

The system has been trained and evaluated with those samples where facial keypoints were detected, which represents a 98.81% of the total samples. This sample set has been randomly divided in two groups, one for training with 70% of samples and the remaining 30% for testing. The training set consists of 47 samples for classes at −45°, 0°, 22.5°, 45°, and 46 samples for the class at −22.5°. The test set consists of 20 samples for classes at −45° and 0°, and 19 samples for classes at −22.5°, 22.5°, and 45°. Since the results of the evaluation may be significantly different depending on how the division of the set is made [35], to reduce variability and average over different sources of randomness we repeat the process ten times for different random partitions and report the ranges (mean ± std) of the measures explained in the above section. The k-NN classifier uses $k = 3$ nearest neighbors and the Euclidean distance.

Table 2 shows the Absolute Error rates, in degrees, for the continuous yaw angle estimation reliability with manual (left) and automatic (right) detection. Notice that the standard deviation is below 1.64° in all cases, indicating that the method is stable among all the classes.

Table 2: Absolute Error rates using manual and automatic detection

Angle	Manual	Automatic
$-45°$	4.63 ± 0.80	3.68 ± 1.12
$-22.5°$	4.35 ± 1.27	4.28 ± 1.44
$0°$	2.70 ± 1.64	2.63 ± 1.39
$22.5°$	4.14 ± 1.22	3.66 ± 1.06
$45°$	2.02 ± 0.97	2.28 ± 1.11
Global	3.57 ± 1.56	3.31 ± 1.40

Boxplots in Figure 8 show a general view of the accuracy rate (in percentage) of our approach for discrete yaw angles estimation for each class, while Table 3 summarizes this accuracy. Both results are shown with manual and automatic detection of the facial keypoints. A global average accuracy over 92% with a standard deviation around 6% makes our method comparable to the state-of-the-art techniques. As before, accuracy does not substantially differ among classes.

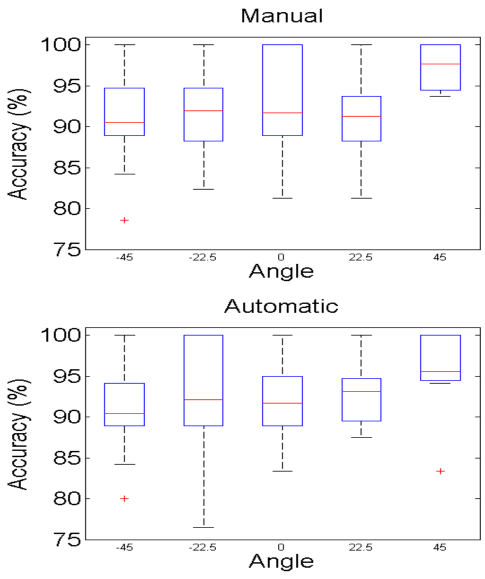


Figure 8: Accuracy boxplots for discrete angles estimation with manual (top) and automatic (bottom) facial keypoint detections.

In order to deepen on the error font, Figure 9 shows the confusion matrix between ground truth and estimated poses for the worst performance iteration. As well, to be able to compare our method with others, the classification results for the best performance iteration

Table 3: Accuracy rates using manual and automatic detection

Angle	Manual	Automatic
$-45°$	90.53 ± 6.06	90.41 ± 5.72
$-22.5°$	91.98 ± 6.05	92.09 ± 7.74
$0°$	91.69 ± 6.72	91.67 ± 5.71
$22.5°$	91.30 ± 5.93	93.14 ± 4.43
$45°$	97.71 ± 2.97	95.61 ± 5.10
Global	92.64 ± 6.05	92.58 ± 5.87

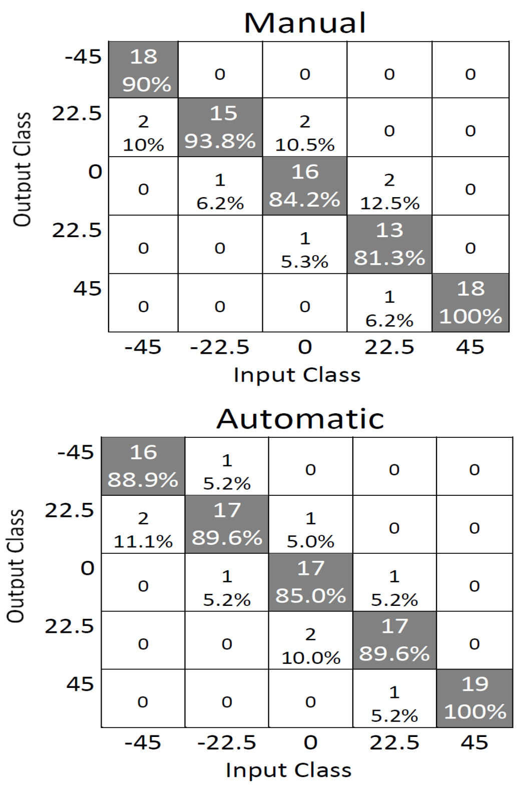
are summarized in the confusion matrix shown in Figure 10 and the summary for the real dataset is shown in Figure 11. Again, all the results are reported for manual (top plots) and automatic (bottom plots) detection of the facial keypoints. Note that the number of samples appearing in the confusion matrices are different since there are some uncertain samples per class, which may vary across the iterations and processes. The ranges of the number of samples in which both methods do not agree for each class and all the iterations are summarized in Table 4.

We can observe that all the misclassified samples are correlated to the first nearest neighbor, which could mean that the classes are not totally separated but there can be problems only in the border areas of each class. As well, samples labeled with a positive angle are not assigned to a negative one, which means that our geometric features and methodology used are robust. Still, the accuracy is above 81% for all classes in the worst performance iteration and reaches a 100% in all but one class for the best one, both in the controlled scenario and the driving dataset.

Table 4: Samples labeled as uncertain for each class and all iterations

Angle	Manual	Automatic
$-45°$	1.40 ± 1.78	1.40 ± 1.17
$-22.5°$	2.00 ± 1.70	1.40 ± 0.70
$0°$	1.90 ± 1.20	0.80 ± 1.14
$22.5°$	3.40 ± 1.51	1.60 ± 1.78
$45°$	1.00 ± 0.82	0.60 ± 0.70

Regarding the comparison between manual and automatic detection, a Student's t-test is performed for finding significative differences with the null hypothesis that both methods have equal means. The test cannot reject the null hypothesis at a significant level of 5%, with the p-value $p = 0.95$ and the confidence interval is $(-2.69, 2.87)$, so that we can ensure that both methods perform equally.


Manual
Output Class
Input Class
Automatic
Output Class
Input Class

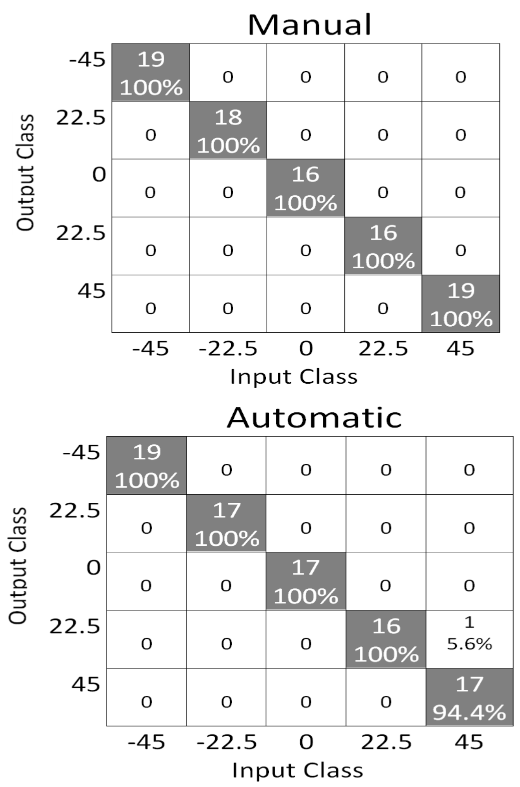
Manual

	-45	-22.5	0	22.5	45
-45	18 90%	0	0	0	0
22.5	2 10%	15 93.8%	2 10.5%	0	0
0	0	1 6.2%	16 84.2%	2 12.5%	0
22.5	0	0	1 5.3%	13 81.3%	0
45	0	0	0	1 6.2%	18 100%

Automatic

	-45	-22.5	0	22.5	45
-45	16 88.9%	1 5.2%	0	0	0
22.5	2 11.1%	17 89.6%	1 5.0%	0	0
0	0	1 5.2%	17 85.0%	1 5.2%	0
22.5	0	0	2 10.0%	17 89.6%	0
45	0	0	0	1 5.2%	19 100%

Figure 9: Confusion matrix for CMU-PIE dataset with manual (top) and automatic (down) facial keypoint detection for the worst iteration.


Manual
Output Class
Input Class
Automatic
Output Class
Input Class

Manual

	-45	-22.5	0	22.5	45
-45	19 100%	0	0	0	0
22.5	0	18 100%	0	0	0
0	0	0	16 100%	0	0
22.5	0	0	0	16 100%	0
45	0	0	0	0	19 100%

Automatic

	-45	-22.5	0	22.5	45
-45	19 100%	0	0	0	0
22.5	0	17 100%	0	0	0
0	0	0	17 100%	0	0
22.5	0	0	0	16 100%	1 5.6%
45	0	0	0	0	17 94.4%

Figure 10: Confusion matrix for CMU-PIE dataset with manual (top) and automatic (down) facial keypoint detection for the best performance iteration.

Finally, it is worth to note that the elapsed time for performing the angle is approximately 2ms on a 2.67GHz Intel(R) Xeon(R) CPU and a non-optimized MatLab code. Thus, it is a negligible time compared to image acquisition and eyes and nose detection.

Figure 12 illustrates the robustness of the method. Each plot corresponds to the average accuracy of the method by adding noise to the detection. The top plot corresponds to the right eye, the middle one to the left eye and the bottom one to the nose. The legend of the principal directions, which are shown in Figure 7, are depicted in the bottom of the figure. The distances range in the x-axis is [1,20] pixels.

Notice that plots show that up to 2 pixel noise do not affect the accuracy of the method, which keeps above the 80% until 4 pixel error in all cases.

7. Comparison to existing methods

In order to compare our method with the current state of the art, Table 5 summarizes the results obtained by the methods in the literature that use CMU-PIE. For each method, we report the accuracy rate (Acc), the number of iterations they run the experiments (Iter.), its application to continuous environments (φ_{cont}) and if the method is fully automatic or not (Aut.). In case the experiments of a method have been run more than once, we report the mean and the standard deviation as long as they are reported in the corresponding paper. The number of classes considered in the literature are generally 9, within the range from $[-90°, 90°]$, although in [36] the authors consider 5 classes, within the range $[-45°, 45°]$. In both cases the increments are given by the dataset, which is 22.5°.

First of all, it is worth to note that accuracy rates are not totally comparable. Most of the methods [12, 24, 37, 38] use a hold-out procedure but they execute the experiments once and, thus, the evaluation may be significantly different depending on which data points are in the training set and which are in the test set. In case they use a k-fold cross-validation [36, 39], they do not report any standard deviation, so we cannot know the dispersion of the data in relation to the average. For

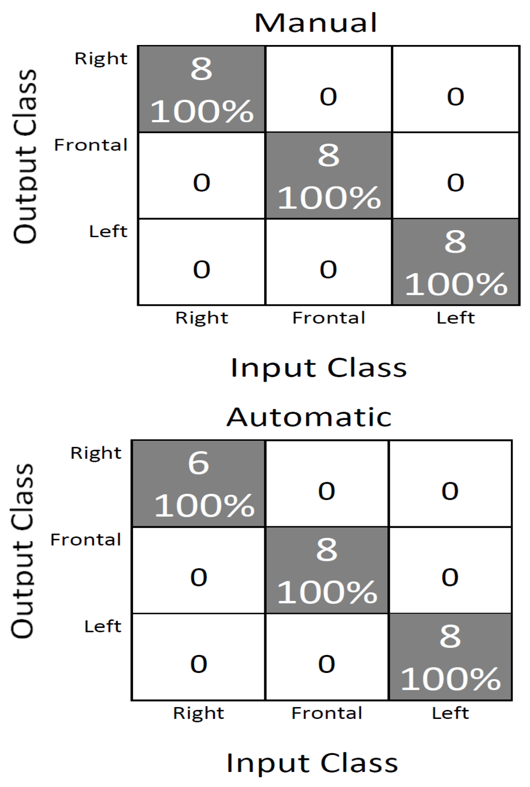


Figure 11: Confusion matrix for our dataset with manual (top) and automatic (down) facial keypoint detection for the best performance iteration.

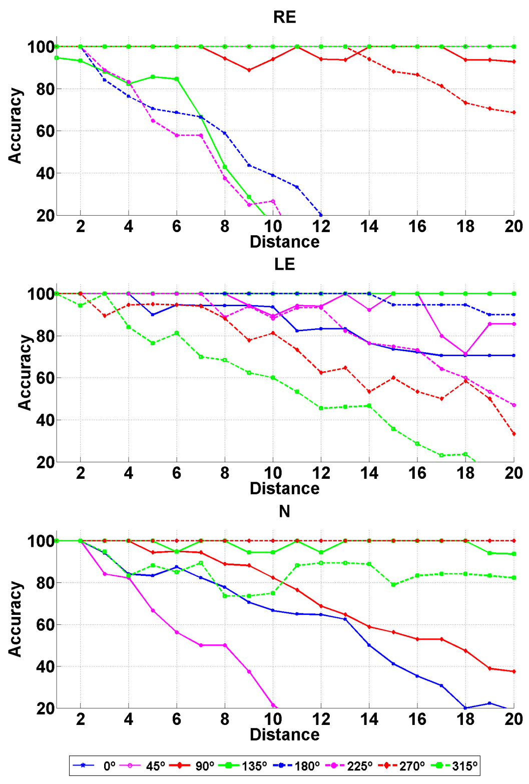


Figure 12: Accuracy of the method according to the distance of facial keypoint detections to the real one

Table 5: Comparison to the-state-of-the-art methods that use CMU-PIE dataset

Method	Acc (%)	Iter.	φ_{cont}	Aut.
Brown [37]	91.00	1	×	×
Ba [24]	94.80	1	✓	-
Saquib [38]	84.06	1	×	×
Hu [12]	98.70	1	×	×
Dong [39]	96.02 ± -	4	×	✓
Dahmane [36]	82.26 ± -	6	✓	✓
Ours	92.58 ± 5.87	10	✓	✓

example, the accuracy rate for our worst performance iteration in the automatic case is 90.53%, while the best performance iteration is 98.85%, which would be the best result in Table 5.

Brown et al. [37] present a comparative study between two coarse pose estimation schemes, a neural network approach and a probabilistic model. Although in both cases the best performance is 91% the main disadvantages of these approaches are: 1) the performance and the complexity of the system depend on the image resolution and the head localization error; 2) it is not an automatic process, since the head bounding box is extracted semiautomatically; 3) the head pose estimation is discrete.

Ba et al. [24] present a probabilistic framework for joint head tracking and pose estimation by incorporating head pose estimated by means of a Naive Bayesian classifier into a mixed-state particle filter framework. Although the global accuracy is 94.80%, head tracking requires a good initialization and head pose estimation needs to learn the same number of Gaussian Mixture Models as poses it has. As well, the framework has high computational complexity due to the combination of both methods in a particle filter framework.

Saquib et al. [38] process full facial images using local energy based shape histogram as feature space. This approach drops its accuracy to 84.06%. The confusion matrix of their results shows that some missclassifications are not correlated to the first nearest neighbor but to the second and third nearest neighbor. For example, the samples of the class 0° are misclassified as ±22.5°, ±45° and −67.5°, proving that the feature space generated is not robust enough to discriminate among the different poses.

The approach presented by Hu et al. [12] is based on the Lie Algebrized Gaussians feature as a subspace where the within-class covariance normalization based Support Vector Machine (SVM) is used as classifier. Although it has a global accuracy of 98.7%, this semiautomatically approach needs to crop the faces with an empirical value of head size that depends on the image characteristics. As well, there are too many parameters that need to be tuned and decided, which are important for a proper head pose estimation. These include the number of Gaussian components, the type of kernel for dimensionality reduction and its parameters, among others, so the complexity and storage cost is increased and the probability of overfitting to a particular scenario is high. Finally, the confusion matrix of their results shows that the method cannot present continuous poses characterization since some poses that belong to the classes at 0° and 45° are misclassified. In the case of class at 0° there are poses classified as −67.5° and 90°, while in the case of class 45° there are poses at −45°.

Dong et al. [39] propose two image representations to describe head pose variations and they combine them in two linear subspace methods and classify the pose by means of the nearest centroid of the classes. The best combination (HOG+LDA) achieves a 96.02% of average accuracy, with 99% of the subspace energy. The performance of this approach is seriously affected by the dimension reduction: the greater the reduction is (not less than 55% of energy), the worse the classification results (≈ 87%). Although the final projection dimension is $c-1$, the dimension of the original space is 32×32, while the dimension of the space we propose is 10, which we reduce until 2 dimensions. In addition, this approach is limited to discrete environments.

Dahmane et al. [36] use a decision tree as classification system. The images of the faces are obtained automatically and processed to extract symmetrical areas and their corresponding features. The best result reported is a global average accuracy of 82.26%.

In contrast to these methods, we present an automatic real time approach to estimate both coarse and continuous yaw angle. The exhaustive validation of the method shows its high accuracy and precision as well as its robustness against noise in the facial keypoint detection, independently of the image resolution and the head localization error, as well as its application in real time.

8. Discussion and Conclusions

In this paper, we have introduced a new methodology for driver coarse and fine head's yaw angle estimation by using a feature set generated from a reduced set of facial keypoints. The approach is based on a combination of subspace methods, as PCA and FLD, and multiple linear regression. As well, it integrates a mechanism to self-evaluate the likelihood of the generated hypothesis and discard uncertain poses by comparing pose label from FLD and PCA.

The reliability of the method has been assessed in two different datasets, a controlled scenario (CMU-PIE) and in real driving (our own database). The global average accuracy for CMU-PIE dataset shows that the method performs like other state-of-the-art strategies. The best performance of our method reaches an accuracy of 100% for manual detection of facial keypoints and a 98.85% for automatic one. In the case of the driving dataset, a continuous angle is computed with an absolute average error below 5.13°. As well, for the coarse estimation, the misclassified samples keep a high correlation (first near neighbor) with the real angle.

Thus, we can conclude that 3 facial keypoints, corresponding to the center of both eyes and the nose tip, are enough to extract 10 geometric features based on angles and Euclidean distances and obtain accurate and precise results for both coarse and fine head pose estimation. This is probably due to the fact that nose tip and eyes are fixed parts of the face so that their degrees of freedom are closely linked to the same degrees of freedom as the face. In this case, adding more facial keypoints, like the edges of the mouth could add more angles and distances that might be redundant or even hinder a reliable angle, since the degrees of freedom of the mouth are independent from driver's face. Indeed, mouth state information could serve to detect driver fatigue [4], but only the edges might not be enough to report a robust state of the mouth and detect events like yawning.

Besides, our method has proven its robustness, maintaining a high accuracy in noisy detections up to 4 pixels for all directions. Several factors such illumination changes can hinder this robustness, misleading proper outputs in some frames. In order to avoid that errors and other induced from the acquisition devices, it would be interesting to explore the improvements of adding some kind of temporal continuity. However, this temporal continuity could be incompatible with the detection of abrupt changes corresponding to drivers behavior abnormalities or distractions. Consequently, to detect abnormal driving performance, abrupt changes of drivers head pose should be carefully modeled and distinguished from those actions implying a short and fast movement of the head. For that aim, the gradient of the head motion would be interesting to incorporate in the model, although it requires a further analysis.

Finally, the low computational cost of the overall method allows it to be integrated in a tablet or mobile for a real time application, which will be done in mid term.

ACKNOWLEDGMENT

This work was supported by the Spanish MINECO project TRA2014-57088-C2-1-R and the Secretaria d'Universitats i Recerca del Departament d'Economia i Coneixement de la Generalitat de Catalunya (2014-SGR-1506).

[1] B.-G. Lee, W.-Y. Chung, Driver alertness monitoring using fusion of facial features and bio-signals, Sensors 12 (7) (2012) 2416–2422.

[2] R. Mbouna, S. Kong, M.-G. Chun, Visual analysis of eye state and head pose for driver alertness monitoring, IEEE T-ITS 14 (3) (2013) 1462–1469.

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

[4] T. Azim, M. A. Jaffar, A. M. Mirza, Fully automated real time fatigue detection of drivers through fuzzy expert systems, Applied Soft Computing 18 (2014) 25–38.

[5] C. H. Zhao, B. L. Zhang, X. Z. Zhang, S. Q. Zhao, H. X. Li, Recognition of driving postures by combined features and random subspace ensemble of multilayer perceptron classifiers, Neural Computing and Applications 22 (1) (2013) 175–184.

[6] M. Patel, S. Lal, D. Kavanagh, P. Rossiter, Applying neural network analysis on heart rate variability data to assess driver fatigue, Expert systems with Applications 38 (6) (2011) 7235–7242.

[7] H. Jo, M. Lee, In-attention state monitoring for a driver based on head pose and eye blinking detection using one class support vector machine, in: Neural Information Processing, Springer, 2014, pp. 110–117.

[8] F. Vicente, Z. Huang, X. Xiong, F. De la Torre, W. Zhang, D. Levi, Driver gaze tracking and eyes off the road detection system, Intelligent Transportation Systems, IEEE Transactions on 16 (4) (2015) 2014–2027.

[9] S. Martin, E. Ohn-Bar, A. Tawari, M. M. Trivedi, Understanding head and hand activities and coordination in naturalistic driving videos, in: Intelligent Vehicles Symposium Proceedings, 2014 IEEE, IEEE, 2014, pp. 884–889.

[10] C. Tran, A. Doshi, M. M. Trivedi, Modeling and prediction of driver behavior by foot gesture analysis, Computer Vision and Image Understanding 116 (3) (2012) 435–445.

[11] E. Murphy-Chutorian, A. Doshi, M. M. Trivedi, Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation, in: Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE, IEEE, 2007, pp. 709–714.

[12] C. Hu, L. Gong, T. Wang, F. Liu, Q. Feng, An effective head pose estimation approach using lie algebrized gaussians based face representation, Multimedia Tools and Applications 73 (3) (2014) 1863–1884.

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

[14] Y. Tian, L. Brown, J. Connell, S. Pankanti, A. Hampapur, A. Senior, R. Bolle, Absolute head pose estimation from overhead wide-angle cameras, in: AMFG, 2003, pp. 92–99.

[15] R. Stiefelhagen, Tracking focus of attention in meetings, in: ICMI, 2002, pp. 273–280.

[16] G. C. Lee, C. K. Loo, L. Chockalingam, An integrated approach for head gesture based interface, Applied Soft Computing 12 (3) (2012) 1101–1114.

[17] D. Mikio, K. Matti, K. JuhaM., Measurement of driver's visual attention capabilities using real-time ufov method, International Journal of Intelligent Transportation Systems Research 9 (3) (2011) 115–127.

[18] J. Jaeik, L. Sung, J. Ho, P. Kang, K. Jaihie, Vision-based method for detecting driver drowsiness and distraction in driver monitoring system, Optical Engineering 50 (12) (2011) 127202–24.

[19] M. Rezaei, R. Klette, Look at the driver, look at the road: No distraction! no accident!, in: Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, IEEE, 2014, pp. 129–136.

[20] E. Ohn-Bar, A. Tawari, S. Martin, M. Trivedi, Vision on wheels: Looking at driver, vehicle, and surround for on-road maneuver analysis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014, pp. 185–190.

[21] Y. Li, S. Gong, J. Sherrah, H. Liddell, Support vector machine based multi-view face detection and recognition, Image and Vision Computing 22 (5) (2004) 413–427.

[22] Y. Ma, Y. Konishi, K. Kinoshita, S. Lao, M. Kawade, Sparse bayesian regression for head pose estimation, in: Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, Vol. 3, IEEE, 2006, pp. 507–510.

[23] M. Voit, K. Nickel, R. Stiefelhagen, Neural network-based head pose estimation and multi-view fusion, in: Multimodal technologies for perception of humans, Springer, 2007, pp. 291–298.

[24] S. Ba, J.-M. Odobez, A probabilistic framework for joint head tracking and pose estimation, in: ICPR, Vol. 4, IEEE, 2004, pp. 264–267.

[25] B. Ma, W. Zhang, S. Shan, X. Chen, W. Gao, Robust head pose estimation using lgbp, in: Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, Vol. 2, IEEE, 2006, pp. 512–515.

[26] J. Sherrah, S. Gong, E.-J. Ong, Face distributions in similarity space under varying head pose, Image and Vision Computing 19 (12) (2001) 807–819.

[27] V. N. Balasubramanian, J. Ye, S. Panchanathan, Biased manifold embedding: A framework for person-independent head pose estimation, in: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE, 2007, pp. 1–7.

[28] J.-G. Wang, E. Sung, Em enhancement of 3d head pose estimated by point at infinity, Image and Vision Computing 25 (12) (2007) 1864–1874.

[29] K. S. Huang, M. M. Trivedi, Robust real-time detection, tracking, and pose estimation of faces in video streams, in: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, Vol. 3, IEEE, 2004, pp. 965–968.

[30] Y. Hu, L. Chen, Y. Zhou, H. Zhang, Estimating face pose by facial asymmetry and geometry, in: Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on, IEEE, 2004, pp. 651–656.

[31] J. Wu, H. Xiong, C. Liu, J. Chen, A generalization of distance functions for fuzzy C-Means clustering with centroids of arithmetic means, IEEE T-Fuzzy Systems 20 (3) (2012) 557–571.

[32] S. Terence, B. Simon, B. Maan, The CMU pose, illumination, and expression (PIE) database, in: IEEE IC-FG, 2002.

[33] K. Ohue, Y. Yamada, S. Uozumi, S. Tokoro, A. Hattori, T. Hayashi, Development of a new pre-crash safety system, in: SAE Tech. Paper 2006-01-1461, 2006.

[34] Y. Sun, X. Wang, X. Tang, Deep convolutional network cascade for facial point detection, in: CVPR, 2013, pp. 3476–3483.

[35] J. D. Kelleher, B. Mac Namee, A. D'Arcy, Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies, MIT Press, 2015.

[36] A. Dahmane, L. Slimane, I. Bilasco, C. Djeraba, Head pose estimation based on face symmetry analysis, Signal, Image and Video Processing 9 (8) (2015) 1871–1880.

[37] L. Brown, Y. Tian, Comparative study of coarse head pose estimation, in: Motion and Video Computing Workshop on, 2002, pp. 125–130.

[38] M. Saquib, O. Hellwich, Head pose estimation in face recognition across pose scenarios, in: VISAPP, 2008, pp. 235–242.

[39] L. Dong, L. Tao, G. Xu, P. Oliver, A study of two image representations for head pose estimation, in: ICIG, 2009, pp. 963–968.