Image and Vision Computing 31 (2013) 322–340

Contents lists available at SciVerse ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

# Hierarchical On-line Appearance-Based Tracking for 3D head pose, eyebrows, lips, eyelids and irises ☆

Javier Orozco [a,*], Ognjen Rudovic [a], Jordi Gonzàlez [c], Maja Pantic [a,b]

[a] Imperial College, Department of Computing, London, UK
[b] University of Twente, EEMCS, Twente, Netherlands
[c] Computer Vision Center, Campus UAB, Barcelona, Spain

ARTICLE INFO

*Article history:*
Received 5 January 2012
Received in revised form 11 September 2012
Accepted 4 February 2013

*Keywords:*
On-line appearance models
Levenberg–Marquardt algorithm
Line-search optimization
3D face tracking
Facial action tracking
Eyelid tracking
Iris tracking

ABSTRACT

In this paper, we propose an On-line Appearance-Based Tracker (OABT) for simultaneous tracking of 3D head pose, lips, eyebrows, eyelids and irises in monocular video sequences. In contrast to previously proposed tracking approaches, which deal with face and gaze tracking separately, our OABT can also be used for eyelid and iris tracking, as well as 3D head pose, lips and eyebrows facial actions tracking. Furthermore, our approach applies an on-line learning of changes in the appearance of the tracked target. Hence, the prior training of appearance models, which usually requires a large amount of labeled facial images, is avoided. Moreover, the proposed method is built upon a hierarchical combination of three OABTs, which are optimized using a Levenberg–Marquardt Algorithm (LMA) enhanced with line-search procedures. This, in turn, makes the proposed method robust to changes in lighting conditions, occlusions and translucent textures, as evidenced by our experiments. Finally, the proposed method achieves head and facial actions tracking in real-time.

© 2013 Elsevier B.V. All rights reserved.

## 1. Introduction

For the last two decades vision-based investigations of non-verbal communication, in particular head and facial actions have caused a surge of interest by CVPR community [1]. Tracking human faces in video sequences is useful for a number of applications such as security and human–machine interaction. Faces have a key role in human–computer interaction systems, because they represent a rich source of information; they are the main gateway to express our feelings and emotional states. The interpretation of user's intentions may be possible if we are able to describe 3D face pose and facial feature location in real-time.

An approach to tackling this problem is to develop a vision-based tracking system since such a solution would be non-invasive. However, building robust and real-time marker-less trackers for head and facial features is a difficult task due to the high variability of the face and the facial features in videos. One of the most challenging tasks is the simultaneous tracking of head and facial features, which is a combination of rigid and non-rigid movements. This requires accurate estimation of subtle facial movements, robustness to occlusions and illumination changes.

The tracking of head and facial features has been accurately solved by adopting Feature-Based Trackers (FBT) [2,3]. In [2] a two-stage approach was developed for 3D tracking of head pose and facial deformations in monocular image sequences. A stable facial tracking is obtained by learning possible deformations of 3D faces from stereo data and using optical flow representation associated with the tracked features. This FBT is accurate for simultaneous head and facial feature tracking but inherits the drawbacks of stereo vision and optical flow computation; namely, this system is restricted to controlled illumination, it requires pre-calibration and it is sensitive to large variations in head pose and facial feature position. Instead, [3] proposes a statistical method based on a set of linear predictors modeling intensity information for accurate and real-time tracking of facial features.

Active Shape Models (ASM) [4] are an alternative to FBT. ASMs use a point distribution model to capture the shape variations while local appearances are modeled for a set of landmarks by using pixel intensity gradient distributions. The shape parameters are iteratively updated by locally finding the best nearby match for each landmark point. ASMs may be improved by using state-of-the-art texture-based features on expense of additional computational loads. Also, ASMs are sensitive to occlusions and illumination changes due to their reduced texture information. However, in contrast to the proposed OABT, the main limitation of ASMs is that they require a large amount of annotated training data in order to learn the shape models.

☆ This paper has been recommended for acceptance by Qiang Ji.

* Corresponding author. Tel.: +44 20 7594 8336; fax: +44 20 7581 8024.
*E-mail address:* forozcoc@imperial.ac.uk (J. Orozco).
*URL:* http://www.ibug.doc.ic.ac.uk/people/jorozco (J. Orozco).

0262-8856/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.imavis.2013.02.001

Appearance changes have been tackled by adopting statistical facial texture-based models. Active Appearance Models (AAM) have been proposed as a powerful contribution to the state-of-the-art for analyzing facial images [5]. Deterministic and stochastic Appearance-Based Tracking (ABT) methods have been proposed [6–8]. These methods can successfully address the image variability and drifting problems by using deterministic or statistical models for the global appearance of a rigid object class: the face. Few approaches attempt to track both the head and the facial features in real-time, e.g., [6,8]. These works have addressed the combined head and facial feature tracking using the AAM principles. However, [6,8] require exhaustive learning stages of orthogonal Eigen-spaces assumed to span all forthcoming images otherwise retraining is required.

To overcome the problems of ill-trained AAMs and drifting problems due to challenging upcoming faces, some authors have proposed adaptive and on-line trained AAMs [9–11]. Empirical evidences showed that person specific AAMs have better performance modeling facial movements than generic AAMs, see [12]. In [9], authors achieved significant reduction on the convergence residuals by applying incremental PCA to build On-line Appearance Models (OAM). A similar method is to update the AAM template to avoid drifting problems while correcting for illumination variations [13]. In [11], authors proposed an automatic construction of AAMs by using an off-line trained shape model. In [10], a linear combination of texture models learned on-line and off-line is applied. The on-line model fits a logistic regression function to be later combined with the typical off-line trained AAMs. The problem of this approach also relates to time-consuming training of AAMs.

Many applications such as drowsiness detection and interfaces for handicapped individuals require tracking of the eyelids and the irises. For applications such as driver awareness systems, one needs to do more than tracking the locations of the person's eyes in order to obtain their detailed description needed to reason about staring patterns and micro sleeps. Head, face and gaze tracking with AAM has been already treated in [14]. The authors use the AAM method presented in [12] to track head and face. And the gaze position is inferred from fitting a generic AAM. Accurate eyelid and iris trackings are challenging issues in the AAM framework that have not been addressed properly so far. We aim to address this issue in this work.

Detecting and tracking the eye and its features (eye corners, irises, and eyelids) have been addressed by many researchers [15–20]. However, most of the proposed approaches rely on intensity edges and are time consuming. In [19], detecting the state of the eye is based on the iris detection in the sense that the iris detection results will directly decide the state of the eye. This work constructed detailed texture templates and the head pose is estimated by using a cylindrical face model combined with image stabilization to compensate for appearance changes. This gaze estimation system has been recently improved in [20]; a saliency framework is used to adjust the resulting gaze estimations based on information about the scene. In [16], the eyelid state is inferred from the relative distance between the eyelid apex and the iris center. The authors reported that when the eyes were fully or partially open, the eyelids were successfully located and tracked 90% of the time. On the other hand, feature-based approaches [18,17] have been applied to iris and eyelids detection too. These methods depend heavily on the accuracy of the extracted intensity edges. Moreover, they require high-resolution images depicting an essentially frontal face. However, real environments offer challenging conditions and large variations in head pose and facial expressions are often observed. In our study, we do not use any edges and there is no assumption made regarding the head pose. In our work, the eyelid and iris motion are inferred at the same time with the 3D head pose and other facial actions, that is, the gaze tracking does not rely on the detection results obtained for other features such as the eye corners and irises.

We have previously proposed an On-line Appearance-Based Tracker (OABT) for 3D head pose and facial action tracking [21]. This OABT uses the AAM representation as baseline. Namely, a deformable shape model was used to drive an image warping process that produces the appearance texture. In contrast to FBTs and ASMs, the OABT and AAMs benefit from the modeling of the entire face texture while including global and local texture variations. However, unlike the AAMs, the OABT does not require prior learning of either facial texture or shape models. The OABT incrementally learns the texture model on-line from the previously tracked frames. The OABT estimates the shape model deformation parameters using a State Transition Process and the minimization by the LMA.

In [21], we adopted a single non-occluded shape-free appearance texture excluding the inner eye region. Excluding the eye region proved to be beneficial for estimation of a more stable 3D head pose. New tracks were estimated by applying a Vanilla Gradient Descent Method [22].

In contrast to feature-based gaze trackers, in [23], we proposed an improved gaze tracker method capable of inferring the position of eyelids and irises in real-time based on on-line learned appearance models. The method implemented two non-occluded OABTs using two independent deformable models. Accurate estimates were obtained by combining a generic Gauss–Newton Iterative (GNI) algorithm with backtracking procedures.

Simultaneous tracking of 3D head pose and facial actions is not a straightforward task. The challenges are as follows: First, 3D head pose variations highly affect facial feature positions and the facial appearance: Second, the upper eyelid is a highly deformable facial feature since it has a great freedom of motion: Third, the eyelid can completely occlude the iris and sclera, that is, a facial texture model will have two different appearances at the same locations: Finally, eyelid and iris movements are very fast, especially eyelid blinks and iris saccades, which are involuntary movements. A holistic tracking of 3D head pose and facial actions must provide estimates of facial actions, shape and textures varying at different rates.

In this paper, we combine and extend our previous works, presented in [21] and [23], in order to address the above-mentioned challenges and obtain robust, accurate, simultaneous 3D head pose, face, and facial action tracking. Specifically, an accurate OABT excluding the eye region is used to estimate 3D head pose, lips and eyebrows movements. Two non-occluded OABTs for eyelids and irises robustly estimate gaze movements. Thus, we extend our previous works in two directions. First, we perform a holistic and simultaneous tracking of 3D head pose, lips, eyebrows, eyelids and irises in monocular video sequences. A hierarchy of three non-occluded OABTs allows both tracking of the movements that vary at different rates and tracking of gaze movements in non-frontal faces. Second, we optimize the appearance estimation by applying a Levenberg–Marquardt Algorithm (LMA) enhanced with line-search procedures [24,25]. The previously used GNI algorithm requires more iterations to converge. Thereby, GNI is not very suitable for simultaneous head and facial action tracking in real-time. LMA provides faster convergence, accuracy and robustness while reducing the number of iterations per frame.

Our OABT requires manual initialization at the first frame to ensure the best tracking performance. This is attained by manually fitting the 3D face Candide model [26] to the first frame in the test sequence. Namely, animation and deformation parameters of the Candide model are manually chosen. However, if automatic initialization is required, an approach to semi-automatic initialization can be adopted. Given a set of images, forty facial landmarks can be obtained for each image by applying a face alignment algorithm from [27]. Next, the Candide model can be manually fitted to each image to obtain the tracking initialization parameters. Then, two regressors could be trained with the initialized facial landmarks, the animation, and deformation parameters calculated for each image. These regressors can be used subsequently to estimate the animation and deformation tracking parameters of a test image given the estimated positions of the facial landmarks. The work proposed in [27] cannot be applied

as is since it is inaccurate in case of non-neutral and non-frontal faces. The same is the case for almost all state-of-the-art facial point detectors (e.g. [28–30]); they are accurate for frontal faces and less accurate for non-frontal and expressive faces. Consequently, manual tuning is required after applying face alignment.

The paper is organized as follows. Section 2 describes the 3D deformable models and their composition to build appearance-based trackers. A facial model excluding the eye-region is used to compose a generic tracker for 3D head pose, eyebrows and lips facial actions. Two non-occluded deformable models are defined to track eyelids and iris separately. Section 3 presents a generic OABT for real-time 3D head pose and facial action tracking. An observation process defines the on-line learning of appearance textures while a transition process estimates the facial actions based on an optimized LMA. Section 4 explains how to assemble three OABTs to solve the problem of simultaneous tracking of 3D head pose, eyebrows, lips, eyelids and irises. We introduce backtracking procedures to explore the entire domain of facial actions seeking for global convergence while avoiding local minima. Section 5 compares the accuracy of this method with partial estimates of using our previous approaches [21,23]. In addition, we present a variety of results showing the stability, accuracy and robustness of the hierarchical OABT for simultaneous tracking of 3D head pose, lips, eyebrows, eyelids and irises. The method is tested with several challenges such as illumination changes, occlusions, translucent surfaces and real-time performance. Finally, in Section 6, we present conclusions and discuss future research related to optimization methods, face tracking, and facial image understanding.

## 2. Face modeling

### 2.1. Face representation

A human face can be represented as a 3D elastic surface with non-linear deformations caused by head rotations and facial movements, which make face modeling a significant challenge. In the context of face and facial tracking, two issues are crucial: image registration and motion extraction. We address them by the means of a 3D deformable model. To this end, we use the 3D face Candide Model [26], which is a wire-frame specifically developed for model-based face coding. We use it as a template for image registration and as a model for facial actions tracking. In what follows, the *shape* model is denoted by **S**, and it is composed of 113 vertices and 183 triangles, as shown in Fig. 1.

The above-mentioned facial deformations are directly related to face biometry and facial expressions. Therefore, a shape model is defined as a linear combination of deformations due to the biometry and facial expressions as follows:

$$\mathbf{S} = \mathbf{S}_0 + \mathbf{D}\,\vec{\beta} + \mathbf{A}\,\vec{\gamma}, \tag{1}$$

where $\mathbf{S}_0$ contains the position of the vertices of the initial shape. The matrix **D** denotes the biometric parameters and the vector $\vec{\beta}$ controls the biometric facial deformation.[1] The matrix **A** encodes the non-rigid facial actionsrelated to facial expressions that are controlled by the parameters stored in the vector $\vec{\gamma}$. Both the biometric deformations, $\vec{\beta}$, and the facial actions, $\vec{\gamma}$, are encoded according to the Facial Animation Parameters (FAPs) for MPEG-4, which are continuous variables in the range $[-1.0,1.0]$.[2]

[1] The vector $\vec{\beta}$ contains 21 FAPs that encode biometric deformations, while the matrix **D** encodes the possible deformations of the shape model, stored in a matrix of dimension $113\times3\times21$.

[2] The vector $\vec{\gamma}$ encodes 9 FAPs related to AUs. The matrix **A** encodes the possible deformations of the shape model due to the facial expressions. Thus, **A** is arranged as a matrix of dimension $113\times3\times9$.

To capture 3D head motions of the face, we adopt a weak perspective projection given the small depth of the face [31]. Furthermore, the 3D mesh is projected onto the image plane by applying an affine transform to obtain the appearance shape template in 2D. Specifically, let $\mathbf{R} = \left[\vec{r}_1, \vec{r}_2, \vec{r}_3\right]$ and $\mathbf{T} = [t_x, t_y, t_z]$ represent the rotation and translation between the coordinate systems of the 3D face model and the camera. Consequently, the 3D shape (described by Eq. (1)) is projected onto the image plane to obtain the corresponding 2D shape:

$$\mathbf{S}'(u,v) = \begin{bmatrix} s\,\vec{r}_1^{\,T} & t_x + u_c \\ s\,\vec{r}_2^{\,T} & t_y + v_c \end{bmatrix} \begin{bmatrix} \mathbf{S}(x,y,z) \\ 1, \end{bmatrix} \tag{2}$$

where $\mathbf{S}'$ is the projected 2D shape, $s$ is a scaling factor, and $\vec{r}_1$ and $\vec{r}_2$ are the first two rows of the rotation matrix **R**. Finally, $(u_c, v_c)$ is the center of the camera coordinate system.

Fitting the shape model to a face requires computing first the position of the vertices that combine the initial shape model, the biometric deformations and facial actions (see Eq. (1)). Next, the 2D shape is obtained by applying the affine transformation in Eq. (2).

Note that the biometric deformations, $\vec{\beta}$ in Eq. (1), are person dependent. Therefore, $\vec{\beta}$ remains constant during the tracking process. On the other hand, the facial actions, $\vec{\gamma}$, are generic animation factors related to facial muscular contractions, and are person independent. Hence, the goal of the tracking process is to estimate the deformation of the 3D wire-frame due to the changes in head pose and facial actions, encoded by the vector $\vec{g} = \left[\vec{\rho}\,, \vec{\gamma}\right]$ that contains the head pose and facial actions parameters, respectively. The vector $\vec{\rho} = [\theta_x, \theta_y, \theta_z, t_x, t_y, s]$ contains the global parameters describing the head rotation, translation and scale. The vector $\vec{\gamma} = [\gamma_0, \ldots, \gamma_8]$ contains the parameters describing the facial actions for eyebrows, lips, eyelids and irises.

### 2.2. Appearance texture

AAMs are statistical models combining information from facial texture and shape [5]. The shape model plays the role of template to register facial images and construct the appearance texture, $\Psi\left(\mathbf{I}, \vec{g}\right)$, which is obtained by applying a piecewise-affine warping function to an input image, Fig. 2.(a). This function maps the pixels of each triangle of the 3D shape, Fig. 2.(a), onto the corresponding triangle of the 2D mask template, Fig. 2.(b), as follows:

$$\begin{aligned} \triangle\mathbf{S}(x,y,z) &\overset{\Psi(\mathbf{I},\vec{g})}{\Rightarrow} \triangle\mathbf{S}'(u,v) \\ \Psi\left(\mathbf{I}, \vec{g}\right) &= \vec{\chi}, \end{aligned} \tag{3}$$

where $\triangle\mathbf{S}(x,y,z)$ corresponds to the triangles of the 3D mesh and $\triangle\mathbf{S}(u,v)$ corresponds to the triangles of the 2D reference appearance shape. Furthermore, the function $\Psi\left(\mathbf{I}, \vec{g}\right)$ is a linear combination of barycenters used to map the corresponding pixels between two poses of the shape model [32] (see Fig. 2). Moreover, the reference shape-free texture (see Fig. 2,(c)) is pose- and expression-normalized since the reference shape model is always neutral and with the frontal pose.

Given the resolution of the shape and appearance texture templates, the complete image warping is implemented as follows: (*i*) Adapt the shape model **S** to the image **I**, see Fig. 2,(a). (*ii*) Construct the appearance texture $\vec{\chi}$ using the warping function, $\Psi\left(\mathbf{I}, \vec{g}\right)$, Eq. (3). (*iii*) Perform the normalization, $\vec{\chi} \sim N(0,1)$, of the obtained appearance texture to partially compensate for the contrast variations caused by photometric transformations. For the sake of clarity, from now on, all the texture appearances are

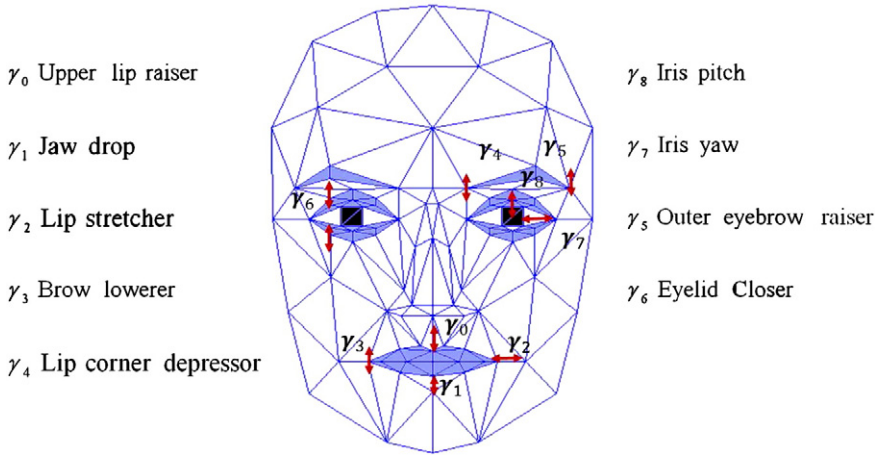


**Fig. 1.** 3D face model. The geometric vector $\vec{g}=\left[\vec{\rho},\vec{\gamma}\right]\in\mathbf{R}^{6+9}$ contains the face model parameters. Three Euler angles, translation and scale encode the head pose parameters, $\vec{\rho}=[\theta_x,\theta_y,\theta_z,t_x,t_y,s]$, while the facial actions, $\gamma_i=[-1.0, 1.0]$, are encoded according MPEG-4 FAP parameters. A face tracker aims to estimate both $\vec{\rho}$ and $\vec{\gamma}$ vectors.

assumed to be normalized images with fixed resolution (see Fig. 2(b)). Note, however, that the tracking accuracy increases with the image resolution.

*2.2.1. Head, eyebrows and lips texture*

We showed in our previous work in [23] that the estimations of the 3D head pose, eyebrows and lips are more accurate when the eye region is excluded from the corresponding appearance texture. Here, we adopt a similar approach, i.e.,we use the OAM without the eye regions, which is further used as the basis for the proposed hierarchical tracking system. Fig. 3 illustrates two appearance textures for a given input image Fig. 3(a). By adopting the appearance texture as shown in Fig. 3(c), we can obtain more stable estimates of the 3D head pose, eyebrows and lips.

*2.2.2. Eyelids' texture*

To independently track eyelid and iris movements from the rest of the face, we adopt a particular shape model for the eyelids, which excludes the iris and sclera region. Thus, only pixels from the eyelid area on the face are warped onto the corresponding appearance texture (see Fig. 4(a)). The advantage of such an OAM is that the pixels from the iris area are considered as outliers. Consequently, when the eyes are open in the input image, the eyelid appearance texture must not contain iris region pixels.

*2.2.3. Irises' texture*

The original Candide model [33] contains both eyelid and iris regions, which are self-occluded. Due to the difference in speed of the iris movements and the rest of the face, the former require different learning coefficients, gradient computations and backtracking procedures. Therefore, the irises must be modeled with an independent OAM. Fig. 4(b) depicts the corresponding shape model and appearance texture of the iris region. This OAM considers eyelid pixels as outliers, so the iris appearance texture must not contain eyelid region pixels.

## 3. On-line Appearance-Based Tracking (OABT)

In this section, we describe the problem of simultaneous tracking of the head and the facial actions by using the proposed On-line Appearance-Based Trackers (OABT). Formally, given a sequence of facial images, our goal is to estimate the 3D head pose, face location and the facial movements of lips, eyebrows, eyelids and irises. To this end, we employ a particle-filter-like approach consisting of an *observation* and a *transition* process. First, all images are projected onto an appearance texture space where they are modeled using Multivariate Gaussian Distribution (MGD). The *observation* process builds a likelihood function based on the MGD, which is updated on-line using a recursive filtering technique. At the same time, the state *transition* process estimates the geometric parameters of the shape model, used to warp incoming images as appearance textures.

*3.1. Observation process*

Given an image sequence, where each image $\mathbf{I}$ has $n$ pixels, a sequence of appearance textures, $\mathbf{X}_{l,t}$, is obtained by applying the piecewise-affine warping function $\Psi\left(\mathbf{I},\vec{g}\right):\mathbf{R}^n\rightarrow\mathbf{R}^l$. The columns of $\mathbf{X}$ are $l$-pixels appearance vectors representing faces at each time frame $t$:

$$\mathbf{X}_{l,t}=\begin{bmatrix}\chi_{0,0} & \cdots & \chi_{0,t}\\ \vdots & \ddots & \vdots\\ \chi_{l,0} & \cdots & \chi_{l,t}\end{bmatrix}=\left[\vec{\chi}_0^T,\ldots,\vec{\chi}_t^T\right] \tag{4}$$

Next, let us assume that a set of pixels at the same position ($\chi_{i,0}$, …,$\chi_{i,t}$) (rows in $\mathbf{X}$) are random variables (r.v.) following a Gaussian Distribution over time, $\chi_i\sim N(\mu_i,\sigma_i)$. Similarly, each appearance vector $\vec{\chi}$ follows a Gaussian Distribution independent of the time.[3] Furthermore, we normalize appearances to partially compensate for contrast variations while obtaining $\chi^*\sim N(0,1)$. The resulting appearances are jointly modeled using the MGD, where $\vec{\chi}\sim N_l\left(\vec{\mu},\Sigma\right)$ is an r.v. with $\vec{\mu}$ being an $l$-dimensional vector containing the means of each r.v. $\chi_i$ and $\Sigma$ is the corresponding covariance matrix.

Because the pixel variations are independent, the covariance matrix is diagonal, i.e., $\sigma_{ij}=\sigma_{ji}=0$ iff $i\neq j$, and $\Sigma=\sigma^2*\mathbf{I}$, where $\sigma^2$ is an $l$-dimensional vector containing the individual variances of each r.v. $\chi_i$. Then, an appearance $\vec{\chi}$ is an r.v. following MGD, $\vec{\chi}\sim N_l\left(\vec{\mu},\sigma^2\right)$:

$$\begin{aligned}\vec{\mu}&=[\mu_0,\ldots,\mu_l]^T\\ \vec{\sigma}&=[\sigma_0,\ldots,\sigma_l]^T.\end{aligned} \tag{5}$$

This results in the observation likelihood given by:

$$p\left(\vec{\chi}_t|\vec{g}_t\right)=\prod_{i=0}^{l}\frac{e^{-(\chi_i-\mu_i)^2/2\sigma_i^2}}{\sigma_i\sqrt{2\pi}}. \tag{6}$$

Assuming an initialized frame,[4] the observation model starts collecting appearance vectors until the appearance sequence $\mathbf{X}$ is long

[3] In [34], Matthews and Baker showed that an appearance regularization lessens the presence of outliers. They applied an *L2*-Euclidean regularization.

[4] There are two ways to initialize the tracker: Manual and Semi-Automatic. In the latter case, the manually labeled training data have to be provided. Note, however, that the tracker is still person independent.

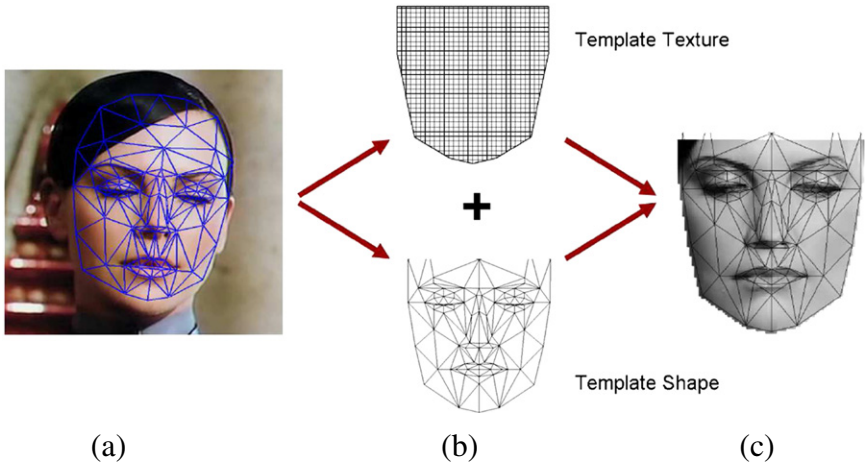


**Fig. 2.** (c) A shape-free appearance texture is obtained by applying the $\Psi\left(\mathbf{I},\vec{g}\right)$ piecewise-affine warping function. (a) The input image $\mathbf{I}$ is mapped onto the template texture pixels (b) based on the correspondence of both shape models in (a) and (b).

enough (approx. 50 to 100 frames) to be approximated by the MGD. Hence, the likelihood function can be used to obtain expected values of the model parameters. Rather than using the Bayesian approach to obtain the a posteriori Gaussian parameters, a time-efficient linear recursive approach is adopted, where $\vec{\mu}_t$ summarizes past observations under an exponential envelope with a degradation factor $\alpha$. Consequently, given the previous appearance estimation, $\hat{\vec{\chi}}_t$, the expected appearance is computed and used as the observation model [35] to obtain an appearance estimate for the next frame as follows:

$$\begin{aligned}\vec{\mu}_{t+1} &= \alpha\vec{\mu}_t + (1-\alpha)\hat{\vec{\chi}}_t \\ \vec{\sigma}^{\,t+12} &= \alpha\vec{\sigma}_t^2 + (1-\alpha)\left(\hat{\vec{\chi}}_t - \vec{\mu}_t\right)^2,\end{aligned} \tag{7}$$

where $\vec{\mu}$ and $\vec{\sigma}$ ($\mu_i$ and $\sigma_i$ for each pixel in the Eq. (5)) are initialized with the first appearance $\vec{\chi}_0$ and a const. value for $\vec{\sigma}$. We empirically found that the aforementioned approximation works well as soon as 50 frames are collected. Similarly, to lessen the presence of outliers and error propagation, the degradation rate $\alpha$ is set to $1/t$ until the 50th frame and a fixed value for the forthcoming frames.

### 3.2. State transition process

To estimate appearance variations between consecutive frames, i.e., $t$ and $t+1$, we adopt an adaptive velocity model, which is defined as follows:

$$\vec{g}_{t+1} = \vec{g}_t + \Delta\hat{\vec{g}}_t, \tag{8}$$

where $\Delta\hat{\vec{g}}_t$ is an estimated increment vector based on the previous frame. The quality of the transition estimation depends on the increment vector, which is used to minimize the distance between the expected $\vec{\mu}$ and the estimated $\hat{\vec{\chi}}$ appearances.

Let us now consider the minimization problem of the error function $\xi\left(\vec{g}\right) : \mathbf{R}^l \rightarrow \mathbf{R}$, which is convex and twice continuously differentiable:

$$\xi\left(\vec{g}\right) = \frac{1}{2}\sum_{j=0}^{l}\frac{\left(\hat{\chi}_j - \mu_j\right)^2}{\sigma_j^2} = \frac{1}{2}\|\vec{r}\left(\vec{g}\right)\|^2, \tag{9}$$

where $\vec{r}\left(\vec{g}\right) = [r_0,\ldots,r_l]^T$ is the *residual vector* between the expected $\vec{\mu}$ and estimated appearances, $\hat{\vec{\chi}}$. The vector $\vec{r}\left(\vec{g}\right)$ depends on the shape parameters, $\vec{g}$, as $\vec{u}$ and $\hat{\vec{\chi}}$ are obtained from Eqs. (7) and (3), respectively. Notice that this is the Mahalanobis normalization rather than the common *L2-Euclidean* normalization proposed in [8].

The condition for a vector $\vec{g}^*$ to be an optimal solution for Eq. (9) is $\nabla\xi\left(\vec{g}^*\right) = 0$. This problem is usually solved by an iterative algorithm that generates a sequence of solutions $\left(\vec{g}_0,\ldots,\vec{g}_k\right) \in dom\ \xi$, for which $\xi\left(\vec{g}_k\right) \rightarrow \xi^*$ as $k \rightarrow \infty$. There are many proposed solutions to this minimization problem based on the iterative first-order linear approximation, which is equivalent to the Gauss–Newton method [24,36,25,22]. Notice that minimizing the above criterion is equivalent to maximizing the likelihood function in Eq. (6).

The Vanilla gradient descent method is the simplest technique to find the optimal solution of the problems given by Eqs. (8) and

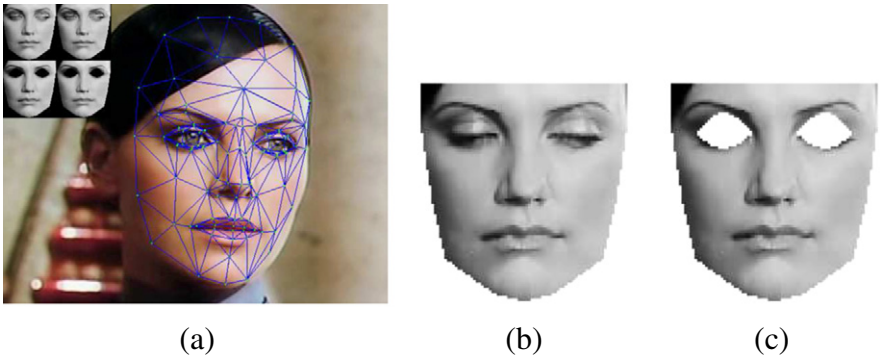


**Fig. 3.** (a) An input image with correct adaptation. (b) The corresponding appearance texture using a full shape model. (c) The appearance texture using a shape model without eye regions. Both appearance textures are shown at the top-left of input image.

(9) [36], but it suffers from the convergence problems around local minima. Moreover, finding the optimal λ according to the slope is time-consuming. A commonly used approximation is the Newton's Method (NM) [22], which provides significantly faster convergence to an optimal solution by combining curvature and gradient information. Yet, it is sensitive to the parameter initialization. To address this, Levenberg [24] provided an algorithm based on the Newton's quadratic assumption, where $\mathbf{H}\left(\vec{g}\right)\approx\mathbf{J}\left(\vec{g}\right)^T\mathbf{J}\left(\vec{g}\right)$ with **H** and **J** being the Hessian and Jacobian matrices, respectively, to improve the NM. The Levenberg Algorithm (LA) is more robust than NM, and it converges faster even when the current appearance and the corresponding vector $\vec{g}$ are far from the next estimation. However, LA does not use the Hessian if the steepest descend value, λ, is large.

As an extension of LA, Marquardt [25] proposed to scale each component of the gradient according to the curvature. The resulting Levenberg Marquardt Algorithm (LMA) makes larger movements along directions where the gradient is smaller, such as the classical *error valley*. Consequently, the Eq. (8) is modified according to the LMA as follows:

$$\vec{g}_{t+1}=\vec{g}_t-\left[\mathbf{H}\left(\vec{g}_t\right)+\lambda diag\mathbf{H}\left(\vec{g}_t\right)\right]^{-1}\nabla\xi\left(\vec{g}_t\right), \tag{10}$$

where $\mathbf{H}\left(\vec{g}_t\right)=\mathbf{J}\left(\vec{g}_t\right)^T\mathbf{J}\left(\vec{g}_t\right)$ and $\nabla\xi\left(\vec{g}_t\right)=\mathbf{J}\left(\vec{g}_t\right)^T\vec{r}\left(\vec{g}\right)$. For facial actions such as eyelids and irises, linearity has low probability. Nevertheless, we avoid local minima using backtracking procedures [totally out of context. make a connection with the previous paragraph].

The Jacobian matrix, $\mathbf{J}\left(\vec{g}_t\right)$, is computed using the approximate differences as follows:

$$\mathbf{J}=\frac{\partial\Psi\left(\mathbf{I}_t,\vec{g}_t\right)}{\partial\vec{g}_t}=\frac{\partial\vec{\chi}_t}{\partial\vec{g}_t}. \tag{11}$$

The *j*th column of **J**, for $j=\left\{1,\ldots,||\vec{g}||\right\}$, is estimated using the differences:

$$J_j\approx\frac{\Psi\left(\mathbf{I}_t,\vec{g}_t\right)-\Psi\left(\mathbf{I}_t,\vec{g}_t+\delta\vec{g}_j\right)}{\delta}, \tag{12}$$

where $\delta$ is the step size, which depends on the domain scale for each component of the vector $\vec{g}$ and the vector $\vec{g}_j$ is a vector with all elements zero except the *j*th element, which is one. The *j*th column of **J** is estimated using several steps around the current value $g_j$, and then by averaging over all these steps. Finally, $J_j$ is obtained by Eq. (13) as:

$$J_j=\frac{1}{k}\sum_{-k/2,k\neq0}^{k/2}\frac{\Psi\left(\mathbf{I}_t,\vec{g}_t\right)-\Psi\left(\mathbf{I}_t,\vec{g}_t+k\delta_j\vec{g}_t\right)}{k\delta_j}, \tag{13}$$

where $\delta$ is the smallest perturbation associated with the $g_j$ and *k* is the number of the steps.

### 3.3. Stability to outliers

In uncontrolled environments (e.g. CCTV systems), a tracking process can be affected by the illumination changes, occlusions, perturbing objects and fast movements. Drifting problems arise when outlier pixels are projected as bias error into the MGD and the LMA, Eqs. (6) and (10), respectively. To deal with outlier pixels, we constrain both the observation and transition processes with a Huber's function [37,38]. By this means, the appearance variations can be learnt on-line in unconstrained environments by combining both processes with the outlier filtering function:

$$\eta(\chi_i)=\begin{cases}\dfrac{\chi_i^2}{2} & if\ |\chi_i|\leq c\\ c|\chi_i|-\dfrac{c^2}{2} & if\ |\chi_i|>c,\end{cases} \tag{14}$$

where $\chi_i$ is the normalized pixel value in the appearance $\vec{\chi}$ and *c* is a constant outlier threshold, which is set to $3\vec{\sigma}$. Thus, the pixel $\chi_i$ is considered an outlier when $|\chi_i|>c$. The function $\eta\left(\vec{\chi}\right)$ constrains both the texture learning and the gradient descent computation, while improving the appearance expectations and estimations, $\vec{\mu}$ and $\hat{\chi}$, respectively. The *observation* process is made more robust to the influence of outlier pixels by combining Eqs. (6) and (14) as follows:

$$P\left(\vec{\chi}_t|\vec{g}_t\right)=\prod_{i=0}^{I}\frac{e^{-\eta(\chi_i)(\chi_i-\mu_i)^2/2\sigma_i^2}}{\sigma_i\sqrt{2\pi}} \tag{15}$$

Likewise, the state *transition* process is improved to down-weight the influence of outlier pixels by restricting the LMA to use the diagonal matrix $\Theta\left(\vec{\chi}\right)$, whose terms are:

$$\Theta(\chi_i)=\frac{1}{\chi_i}\frac{\partial\eta(\chi_i)}{\partial\chi_i}=\begin{cases}1 & if\ |\chi_i|\leq c\\ \dfrac{c}{|\chi_i|} & if\ |\chi_i|>c\end{cases} \tag{16}$$

Consequently, we combine Eqs. (10) and (16) to rewrite the LMA as:

$$\vec{g}_{t+1}=\vec{g}_t-\left[\mathbf{H}\left(\vec{g}_t\right)+\lambda diag\mathbf{H}\left(\vec{g}_t\right)\right]^{-1}\Theta\left(\vec{\chi}_t\right)\nabla\xi\left(\vec{g}_t\right) \tag{17}$$

## 4. Hierarchical tracking

In this section, we perform tracking of 3D head movements and facial actions simultaneously. As in our previous work [21], the parameters associated with the 3D head pose, and facial actions related to the eyebrow and lip movements are estimated together by means of the OAM introduced in Section 2.2.1. Facial actions are local movements of non-rigid surfaces caused by muscular contractions such as eyebrows and lips, which involve more muscles than eyelid and irises [39]. On the other hand, eyelid and iris movements involve lighter muscle activations, resulting in more spontaneous and faster movements. We showed in [23] that eyelids and irises can be tracked simultaneously by adopting sequential tracking approach to avoid self-occluded facial actions.

The tracking of the 3D head pose, eyebrows, lips, eyelids and irises is performed by simultaneously using three different OABTs. The first OABT tracks the 3D head pose, eyebrows and lips, while ignoring eye region motion. The second OABT tracks the eyelids, with iris region considered to be an outlier. Finally, the third OABT tracks irises by estimating yaw and pitch iris motion. These three OABTS are hierarchically combined, ensuring a global minimum error for all estimations.

**Algorithm 1**. Head, eyebrows and lips OABT.

1. Construct the appearance $\Psi\left(\mathbf{I}_t,\vec{q}_t\right)=\vec{\hat{\chi}}_t$, Eqs. (1)–(3).
2. Normalize the appearance, $\vec{\hat{\chi}}_t\sim N(0,1)$.
3. Obtain Gaussian parameters $\vec{\mu}_{t+1}$ and $\vec{\sigma}_{t+1}$ 2, Eq. (7).
4. Compute the Jacobian $\vec{J}\left(\vec{q}_t\right)=\frac{\partial\vec{r}}{\partial\vec{q}_t}$
5. Compute the Hessian $\vec{H}\left(\vec{q}_t\right)=\frac{\partial^2\vec{r}}{\partial\vec{q}_t}^2=\mathbf{J}\left(\vec{q}_t\right)^T\mathbf{J}\left(\vec{q}_t\right)$
6. Compute the diagonal matrix $\Theta\left(\vec{\chi}_t\right)$, Eq. (16).

**For** (*n-Iterations*)

7. Compute the steepest descent factor, $\nabla\xi\left(\vec{q}_t\right)=\mathbf{J}\left(\vec{q}_t\right)^T\vec{r}\left(\vec{q}_t\right)$
8. Compute the search direction, $\delta_{t+1}=\left[\mathbf{H}\left(\vec{q}_t\right)+\lambda diag\mathbf{H}\left(\vec{q}_t\right)\right]-1\Theta\left(\vec{\chi}_t\right)\nabla\xi\left(\vec{q}_t\right)$
9. Update the geometrical vector, $\vec{q}_{t+1}=\vec{q}_t-\left[\mathbf{H}\left(\vec{q}_t\right)+\lambda diag\mathbf{H}\left(\vec{q}_t\right)\right]^{-1}\Theta\left(\vec{\chi}_t\right)\nabla\xi\left(\vec{q}_t\right)$, Eq. (17).
10. Update the estimated appearance $\vec{\chi}_{t+1}=\Psi\left(\mathbf{I}_{t+1},\vec{q}_{t+1}\right)$, Eqs. (1)–(3) and (8).
5. Compute the residual image, $\xi\left(\vec{q}_{t+1}\right)=\frac{1}{2}\sum\frac{(\hat{\chi}_j-\mu_j)^2}{\sigma_j^2}$, Eq. (9).

**EndFor**

### 4.1. Head, eyebrows and lips tracker

The OABT for the 3D head pose, eyebrows and lips uses the OAM without the eye region, as described in Section 2.2.1. Formally, let us consider the geometric vector $\vec{q}=\left[\vec{\rho},\gamma_0,\ldots,\gamma_5\right]$ that models the OAM of this tracker. This vector includes six head pose parameters, $\vec{\rho}$ and the FAPs controlling the eyebrows and the lips. This vector controls the deformation of the shape model corresponding to the tracker $\vec{T}_{\vec{q}}$.

The tracking process starts by setting $\vec{q}=\vec{q}_t$. Then, the error vector $\Psi\left(\mathbf{I}_{t+1},\vec{q}_t\right)-\vec{\mu}_t$ and the corresponding Mahalanobis distance, $\xi\left(\vec{q}\right)$, is computed. The solution is obtained by the LMA Eq. (10), using the same number of differentials to compute the Jacobian and Hessian matrices for head, eyebrows and lips. Thus, we find the shift $\Delta\vec{q}$ to update the estimated geometric vector $\vec{q}_{t+1}$ using Eq. (8). This process is repeated iteratively until convergence, and it is summarized in Algorithm 1.

### 4.2. Eyelid tracker

Both eyelids and irises have smooth and spontaneous movements that are difficult to track using standard statistical deformable models for two reasons [21]. Firstly, the images depicting the eye region usually have low resolution if recorded using monocular cameras, which is often the case. This results in poor appearance models that are unable to recover subtle facial motions. Secondly, the eyelid and iris facial movements are non-linearly coupled since the iris motion deforms the eyelid surface, and vice versa. As a consequence, the OABTs will require additional computations to accurately estimate the eyelid position. By the same token, the eyelid blinking occludes the iris region and the iris normally moves during the blinking. Since the iris movements may be large or short and involuntary, this increases the challenge to recover the correct iris position after occlusions.

In this paper, we perform tracking of the eyelids and irises using two independent trackers. Firstly, the appearance model $\vec{\chi}\left(\vec{w}\right)$ for eyelid tracking is built by excluding the iris FAPs from the shape model (see Fig. 4,(a)), which is controlled by the vector $\vec{w}=\left[\vec{q},\gamma_6\right]$. Thus, iris pixels are excluded from the warping process of the OAM. Secondly, the appearance model $\vec{\chi}\left(\vec{g}\right)$ for iris tracking includes eyelid and iris pixels, $\vec{g}=[cw,\gamma_7,\gamma_8]$ (see Fig. 4(b)). However, the fitting process of this tracker mainly consists of the iris FAP estimation, since the eyelid facial action has been previous estimated. Therefore, once the eyelid tracker has converged, the iris tracker continues estimating the iris movements while refining the previous eyelid position.

There are two main steps in the estimation of the eyelids movements: firstly, the eyelid parameter is differentiated along the whole domain of FAPs to include possible blinking. Secondly, a damping factor, $\delta$, is computed by using backtracking procedures. This reduces the optimization time. The result is an efficient eyelid tracker, $\vec{T}_{\vec{w}}$ (see Algorithm 2).

**Algorithm 2**. Eyelid tracker.

7. This and the above steps up to 1 are the same as in Algorithm 1.
   **For** (*k-Iterations*)
8. Compute the search direction, $\delta_k(\lambda)=-\left[\mathbf{H}\left(\vec{w}_t\right)+\lambda diag\mathbf{H}\left(\vec{w}_t\right)\right]^{-1}\Theta\left(\vec{\chi}_t(\vec{w}_t)\right)\nabla\xi\left(\vec{w}_t\right)$, Eq. (17).
   Choose the damping factor λ via backtracking line-search procedure:
   - **i.** Consider the search direction and the starting vector $\vec{w}_k=\vec{w}_t\in dom\ \xi$.
     Set $\lambda_k=\sum_0^k\frac{(-1)^k}{k}$
   - **ii.** While $\xi\left(\vec{w}_k+\delta_k(\lambda_k)\right)>\xi\left(\vec{w}_k\right)+\delta_k(\lambda_k)$.
     Armijo Condition [22].
   - **iii.** Set $\lambda=\lambda_k$.
9. Update variables, $\vec{w}_{k+1}=\vec{w}_k+\delta_k(\lambda)$, Eq. (8)
10. Test convergence for stopping iterations, otherwise, consider $k=k+1$.
11. This and further steps as in Algorithm 1.

**EndFor**

The OAM of the eyelid tracker $\vec{T}_{\vec{w}}$ is learnt using Eq. (7), such that $\vec{\mu}_{t+1}=\vec{\mu}_{t+1}\left(\vec{w}\right)$ and $\vec{\sigma}_{t+1}=\vec{\sigma}_{t+1}\left(\vec{w}\right)$ correspond to the parameters of the MGD of $\vec{T}_{\vec{w}}$. To be able to estimate the eyelids correctly, steps (4) and (5) from Algorithm 2. consider the whole FAP range in the gradients computation. Steps (8)–(10) calculate the damping factor, λ, by using backtracking line-search procedures. Therefore, the eyelid tracker provides a space of solutions including both opened and closed eyelids for faster convergence.

### 4.3. Iris tracker

Although iris movements are not as fast as eyelid blink, their tracking is challenging due to the movements during eyelid occlusions. Specifically, the eyelid blinks can be as short as 0.2 s [40], while similar involuntary

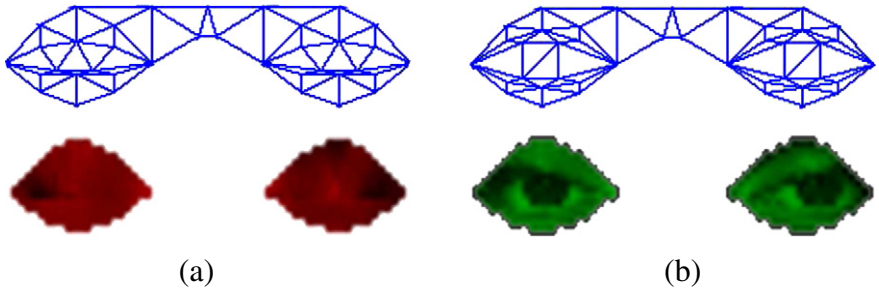


**Fig. 4.** (a) The eyelid texture is obtained based on shape model excluding inner eye region. (b) The iris texture is obtained with a shape model including both eyelid and iris regions. These OAMs consider each other's pixels as outliers.

iris movements, saccade or saccadic movements, change the gaze direction in 210 degrees per second [41]. Therefore, assuming that a video sequence is recorded at 20 frames per second (fps), the eyelid appears closed from neutral position in 2 frames whereas the iris saccade movement can change the gaze direction in 21 degrees in the same 2 frames. Saccade movements are particular involuntary movements that change the direction of the eyeball spontaneously aiming to correct the image perception against blurring by relocating it after eye closure.

Tracking eyelids and irises with a single OABT requires registering both eyelid and iris onto the same template appearance texture and consequently estimating their state transition with a single MGD. The transition process combined with an error function for outliers make the registration of eyelid and iris textures mutually exclusive due to self-occlusions. Consequently, the eyelid appearance changes will not be learnt.

Instead, it is possible to learn the appearance variation of both eyelids and irises by modeling the motion of these facial features with two different OABTs. Thus, eyelid and iris texture remain mutually exclusive but registered and learnt separately.

**Algorithm 3**. Iris tracker.

**7.** This and the above steps up to 1 are the same as in Algorithm 1.
**For** (*m-Iterations*)
**8.** Compute the search direction, $\delta_m(\lambda) = -\left[\mathbf{H}\left(\vec{g}_t\right) + \lambda diag\mathbf{H}\left(\vec{g}_t\right)\right]^{-1}\Theta\left(\vec{\chi}_t\left(\vec{g}_t\right)\right)\nabla\xi\left(\vec{g}_t\right)$, Eq. (17).
Choose the damping factor λ via backtracking line-search procedure:
**i.** Consider the search direction and the starting vector $\vec{g}_m = \vec{g}_t \in dom\ \xi$.
Set $\lambda_m = \frac{\lambda_{m-1}}{\upsilon}$, for $\upsilon > 1$
**ii.** While $\xi\left(\vec{g}_m + \delta_m(\lambda_m)\right) > \xi\left(\vec{g}_m\right) + \delta_m(\lambda_m)$
Armijo Condition [22].
**iii.** Set $\lambda = \lambda_m$.
**9.** Update variables, $\vec{g}_{m+1} = \vec{g}_m + \delta_m(\lambda)$, Eq. (8)
**10.** Test convergence for stopping iterations, otherwise, consider $m = m + 1$.
**11.** This and further steps as in Algorithm 1.
**EndFor**

The iris movements cover a smaller region than those of the eyelids. Therefore, the Step (8.*i.*) from Algorithm 3 models the damping factor, $\lambda_m$, by a monotonically decreasing function in a small range.

The iris tracker, $\mathbf{T}_{\vec{g}}$, considers the inner eye region in the AAM by using the geometric vector $\vec{g} = \left[\vec{w}, \gamma_7, \gamma_8\right]$ to deform the shape model and perform image warping, $\Psi\left(\mathbf{I}, \vec{g}\right) = \vec{\chi}\left(\vec{g}\right)$ (see Fig. 4(b)). Subtle iris movements are modeled by smaller differentials on the iris FAP, ±0.5. Thus, the iris gradients can be estimated by differentiating the error function with respect to $\gamma_7$ and $\gamma_8$ within the range $[\gamma_i - 0.5, \gamma_i + 0.5]$. Algorithm 3 gives the details about the iris tracking. Note that the backtracking procedure can handle smooth transitions of the iris facial actions by decreasing the damping factor λ.

### 4.4. Head, eyebrows, lips, eyelids and iris tracking

In the previous sections, the three trackers have been described: Section 4.1 described the OABT, $\vec{T}_{\vec{q}}$, that estimates the head pose and facial actions corresponding to the eyebrows and lips (see Algorithm 1). Section 4.2 described the OABT, $\vec{T}_{\vec{w}}$, for tracking of the eyelid movements and blinks by applying a backtracking procedure (see Algorithm 2). Finally, Section 4.3 described the OABT, $\vec{T}_{\vec{g}}$, used to track the iris movements and spontaneous saccades (see Algorithm 3).

In order to achieve robust and accurate simultaneous tracking of the head and facial actions, the three trackers mentioned above are combined in a hierarchical fashion. Namely, the tracker for 3D head pose, eyebrows and lips, $\vec{T}_{\vec{q}}$, is used as the basis, as shown in Fig. 5,(a). The goal of this tracker $\vec{T}_{\vec{q}}$ is to estimate the shape vector $\vec{q}$ that provides the best 3D head pose, eyebrows and lips adaptations upon convergence $min_{arg}||\xi(vecq)|| = \vec{q}^*$. Next, both head and smooth facial actions are assembled with eyelids by the tracker $\vec{T}_{\vec{w}}$ to provide the best shape according to the vector $\vec{w} = \left[\vec{q}, \gamma_6\right]$. Once the eyelid tracker has converged, $min_{arg}||\xi\left(\vec{w}\right)|| = \vec{w}^*$, all three trackers are combined, $\vec{T}_{\vec{q}}$, $\vec{T}_{\vec{w}}$ and $\vec{T}_{\vec{g}}$, to estimate the vector $\vec{g} = \left[\vec{w}, \gamma_7, \gamma_8\right]$, which includes the iris FAPs (see Fig. 5(b)).

Thus, the trackers are combined hierarchically in such a way that the estimations from one tracker are propagated to another. Specifically, the eyelid tracker uses the vector $\vec{q}$ as the initial guess for both gradient computation and LMA iterative process towards the optimal solution, $\vec{w}$. Likewise, the eyelid tracker $\vec{T}_{\vec{w}}$, is used as starting point for the iris tracker, which estimates the iris position from near correct estimations of the eyelids, eyebrows, lips and head pose. Both eyelid and iris trackers are conditioned to improve precedent trackers, which is quantified by the average residual error at the convergence, i.e. $\left|\left|\bar{\xi}(\mathbf{q}^*)\right|\right| \geq \left|\left|\bar{\xi}(\mathbf{w}^*)\right|\right| \geq ||\bar{\xi}(\mathbf{g}^*)||$.

Consequently, the three trackers are efficiently connected by applying three times the LMA iterative minimization. Note that we take into account the whole face in both eyelid and iris trackers to make estimations according to the 3D head pose. The eyelid tracker is independent from iris estimation but forced to improve the face tracker. The iris tracker is led to the correct eyelid position and required to improve the eyelid convergence error.

The hierarchical tracking combines the strengths of the three OABTs. Specific shape models contribute to avoid the uncertainty in the eye region to track the 3D head pose, eyebrows and lips as proven in [21]. The high contrast between eyelids, sclera and irises is avoided by using two different OAMs. The space of solutions of eyelid and iris trackers is expanded to the sparse Jacobian and Hessian matrices. Furthermore, the eyelid and iris trackers estimate gradient damping factors based on specific backtracking procedures. Consequently, all AAM fitting processes

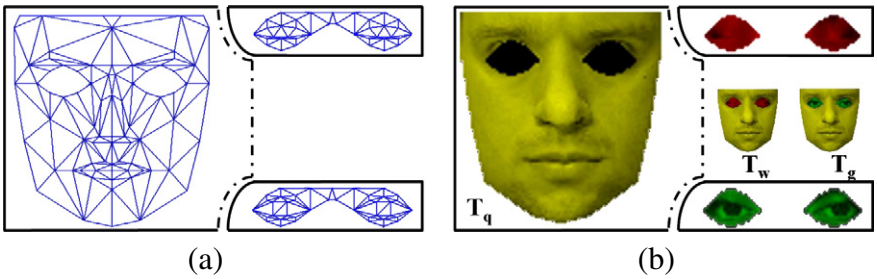


**Fig. 5.** (a) Three shape models are combined to simultaneously estimate 3D head pose, eyebrows, lips, eyelids and irises. (b) The corresponding appearance textures are combined into full face AAMs for eyelid and iris tracking, respectively.

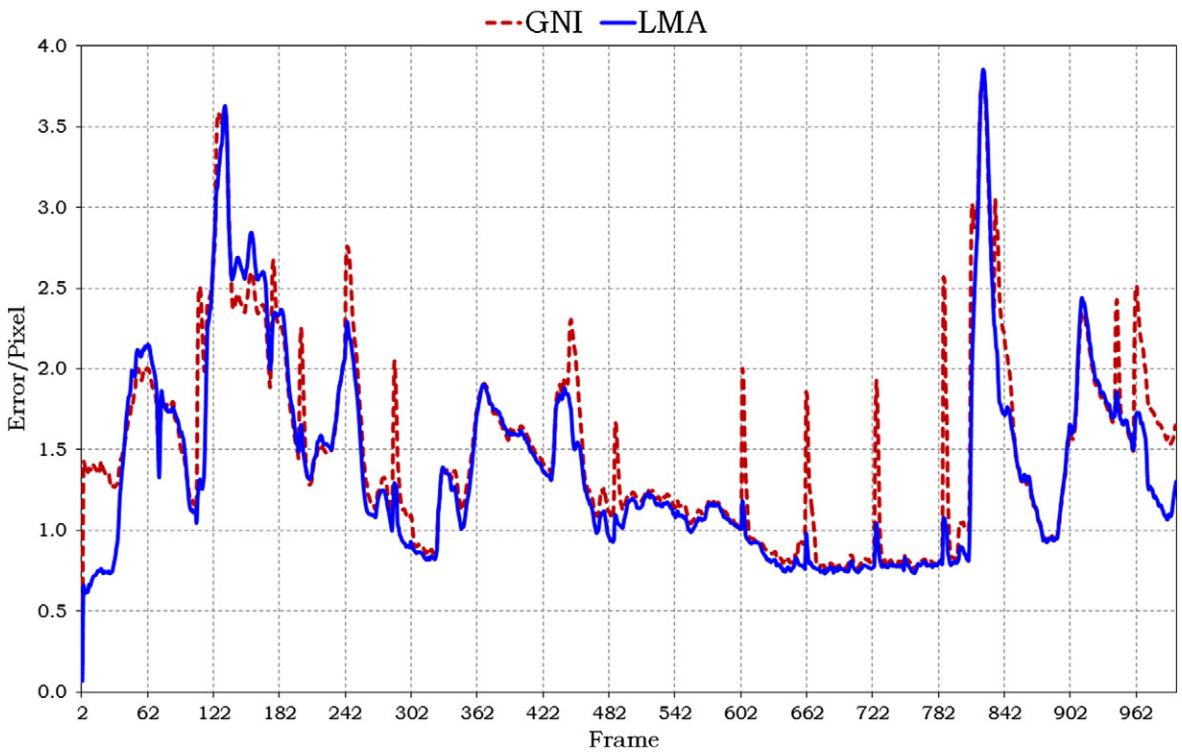


**Fig. 6.** Comparison of our previous GNI algorithm [21] and the current minimization via LMA Eq. (17). GNI algorithm (dash red line) is prone to divergence during gaze motion and large head poses. LMA (solid blue line) increases the accuracy of the head and facial feature tracking by reducing the fitting error and converging to a global minimum.

have been improved with a modified LMA and backtracking procedures. Therefore, the convergence to an optimal solution is ensured by simultaneous estimation of thegradient direction, curvature and particular damping factors.

## 5. Experimental results

In this section, we compare the results of simultaneous head and facial tracking using different testing video sequences. To measure the tracking performance, we compute the average error per pixel, which is computed using Mahalanobis distance between the expected ($\vec{\mu}$) and the estimated ($\hat{\vec{\chi}}$) appearance models (see Eq. (9)). This error measure is used to quantify the precision of the tracking system. We also compute Root-Mean-Square-Error (RMSE) between the estimated position of shape model vertices and the ground truth (i.e., manually annotated locations of the vertices). This error measure is used to quantify the accuracy of the shape model, being a part of the tracking system.

### 5.1. Efficiency of LMA

We test here if there is any gain in using the LMA-based optimization in the proposed hierarchical OABT (as explained in Algorithm 1), over the GNI-based optimization, as we previously proposed in [21]. In [21], the differential of the geometric vector $\vec{g}$ was computed as follows:

$$\Delta\vec{g} = -\left(\mathbf{G}_t^T\mathbf{G}_t\right)^{-1}\mathbf{G}_t^T\left(\Psi\left(\vec{\chi}_t,\vec{g}_{t-1}\right)-\vec{\mu}_t\right). \tag{18}$$

Eq. (17) to obtain the differential of the vector $\vec{g}$ (see the details of the generic implementation of the LMA in Algorithm 1).

A comparison between LMA and GNI is performed using a sequence from the FGnet database [42]. This database contains five sequences of 1000 frames of a talking person. Each image is of size $720\times576$ and manually annotated with 68 facial landmarks. Fig. 6 shows the error per pixel of the head and eyelid tracking when using GNI- and LMA-based optimization. We notice that the OABT based on GNI is sensitive to gaze motion and large head rotations, which is a consequence of its susceptibility to the local minima divergence. Conversely, an OABT based on LMA converges easily to global minima. As can be seen from Fig. 6, the OABT with LMA-based optimization outperforms our previous tracker [21] based on the GNI optimization, by increasing the precision and robustness while reducing the computational load.

This is attributed to the fact that, in contrast to the GNI-based optimization, the LMA-based optimization considers information about the curvature of the gradient which helps to avoid singularities of the pseudo-inverse of the gradient matrix. Also, the backtracking procedures of the LMA-based method help to avoid local minima. Finally, the improved tracking results are also due to the complete estimation of head and facial actions including eyelids and irises.

From Fig. 6, it can be seen that around frame numbers 122 and 842, the error per pixel increases from 1.0 to 3.5. The provided Ground Truth [42] contains some inaccurate annotations (outliers) for eyelids and irises.For example, during blinking, the upper and lower eyelids are not annotated at the same vertical position. After blinking, the iris position is annotated at the center of the eye. This causes a higher estimation error attained by the OABT, and it will be detailed in the following section.

### 5.2. Ground truth comparison

We tested the accuracy of the hierarchical OABTs against the ground truth of the FGnet database for face tracking [42]. The 68 annotated facial landmarks include the features tracked by the OABTs; eyebrows, lips, upper and lower eyelids, and iris center. The eyelid tracking accuracy is measured based on the vertical positions of upper and lower eyelids. It is worth mentioning that these positions may vary depending on the horizontal position of both the ground truth and the tracking result.

Iris tracking accuracy is assessed using the errors of four vertices of the 3D shape model and the corresponding points from the ground truth. Eyebrows and lip vertices are also measured against similar points on the ground truth.

The head-pose is more difficult to compare since the annotations are following the edges of the face without considering the 3D perspective. However, the average error for eyelids is 3.2 pixels per frame (see Fig. 7(b)). Similar results were obtained for the iris tracking where the average error is 2.2 pixels per frame and even better for the complete hierarchical tracking, i.e., 1.5 pixels per frame. Fig. 7(a) shows that higher errors coincide with those frames where eye blinkingor fast iris movements are occurring. Nonetheless, the error decreases as soon as the trackers improve their convergence in the following frames. Altogether, the accuracy and precision of the tracking estimations are comparable. This can also be verified with the correct adaptations, see Fig. 8. These results are also

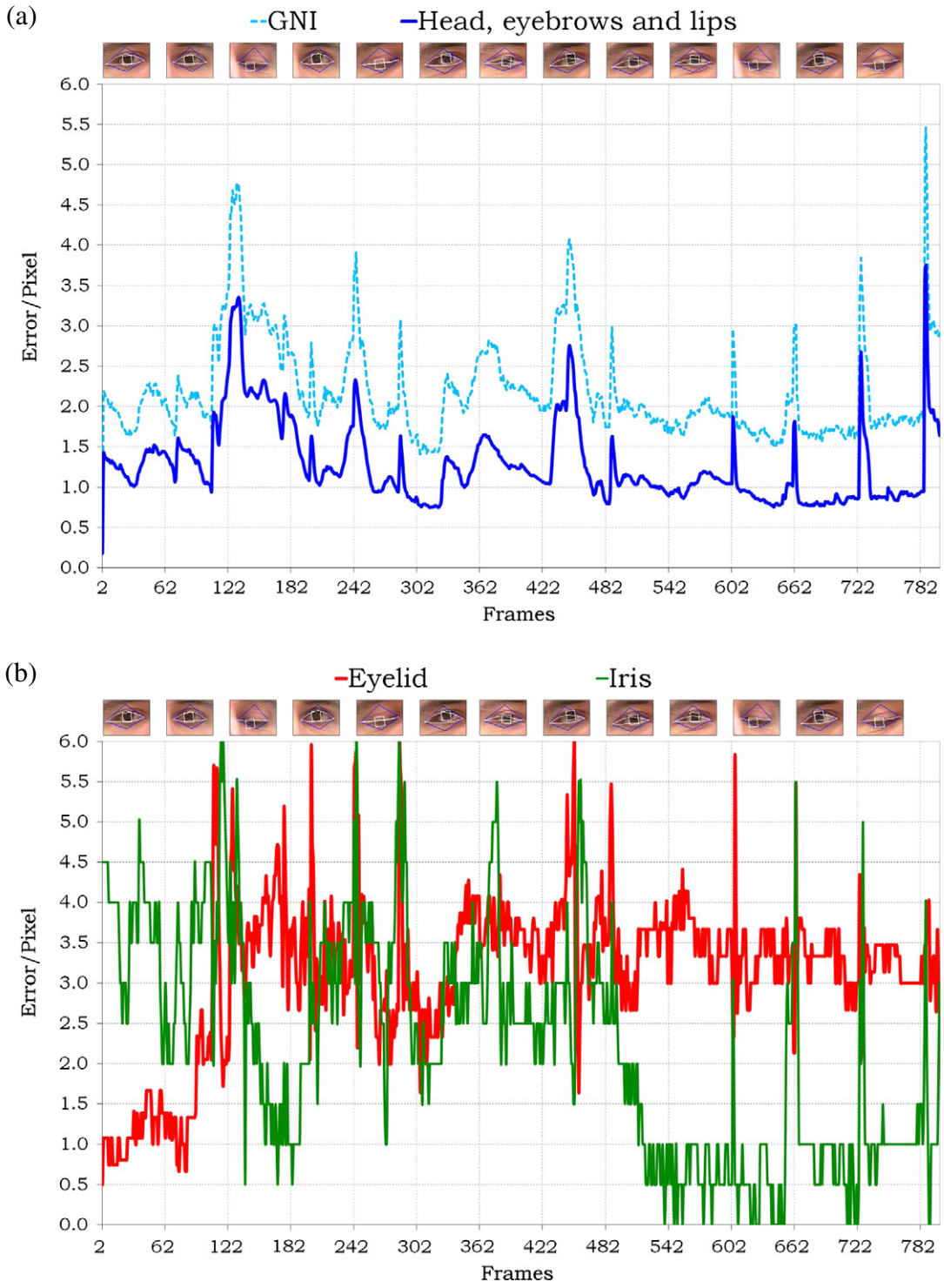


**Fig. 7.** The ground truth of FGnet talking face is compared to the OABT estimations. (a) The OABT using the previous GNI [21], estimates head, lips, eyebrows and eyelids with an average 2.5 error/pixel. Our hierarchical OABT achieves an average 1.5 error/pixel. (b) Using OABTs with the LMA for eyelids and irises, the average estimation error/pixel is 3.2 and 2.2 for irises, respectively. All tracking errors remain within constant bounds (no-drifting problems), but the OABT using LMA outperforms the one based on GNI algorithm.

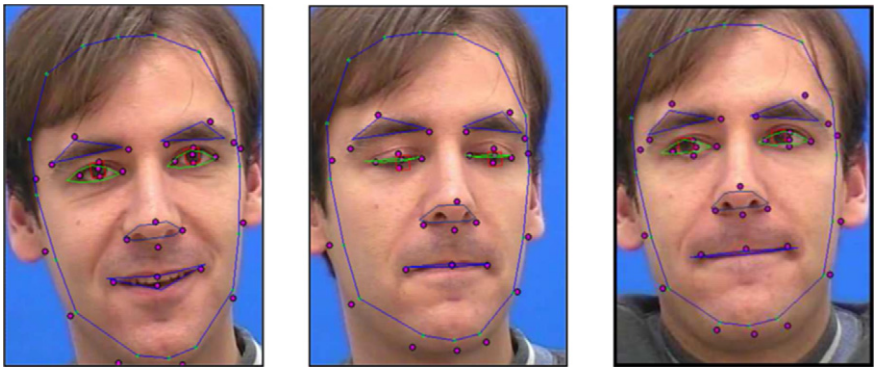


**Fig. 8.** Comparing tracking results and ground truth before, during and after the $122^{th}$ frame, (a), (b) and (c), respectively. The ground truth corresponds to the magenta filled circles and the tracking estimations are the contour lines blue, green and red. The ground truth has some wrong eyelid and iris annotations, (b) and (c), respectively. This reflects on a higher error/pixel specially during blinking.

compared with those attained by the previously proposed tracker based on the GNI optimization [21]. The distances between estimated facial landmarks and the FGnet ground truth are measured. Fig. 7(b) shows the GNI precision error w.r.t. the ground truth (dashed line), which is on average 2.5 when only head, lips, eyebrows and eyelids are estimated.

Caveat, the ground truth contains wrong annotations that can be easily verified using the eyelid tracker, $\vec{T}_{\vec{w}}$. Fig. 8 shows three frames of the FGnet talking face before, during and after the 122th frame. The magenta filled circles correspond to the 68 manually annotated landmarks. Tracking estimations are displayed by continuous solid lines in blue, green and red colors. Only about 40 points coincide with the vertices of the Candide shape model. Fig. 8(a) depicts a correct alignment of image and shape model. However, in Fig. 8(b), it is possible to see a correct detection of the blinking since the green solid line fits to the eyelid contour. On the other hand, the ground truth wrongly marks upper and lower eyelids at a different vertical position. Likewise, Fig. 8(c) shows the iris position marked with a point at the center of the eye whereas the red solid rectangle shows a correct fitting by the iris OABT.

The iris tracker estimates the vector $\vec{w}=\left[\vec{q},\gamma_6\right]$, where $\gamma_6=[-1.0, 1.0]$; $-1.0$ when eyes are closed and 1.0 when eyes are open. The hierarchical tracking estimates blinks at frames 124, 243 and 603, $\gamma_6=-1.0$, where the distance between upper and lower eyelids should be zero pixels. However, the distance between annotations of the ground truth are never zero, see Fig. 9. Similar results were

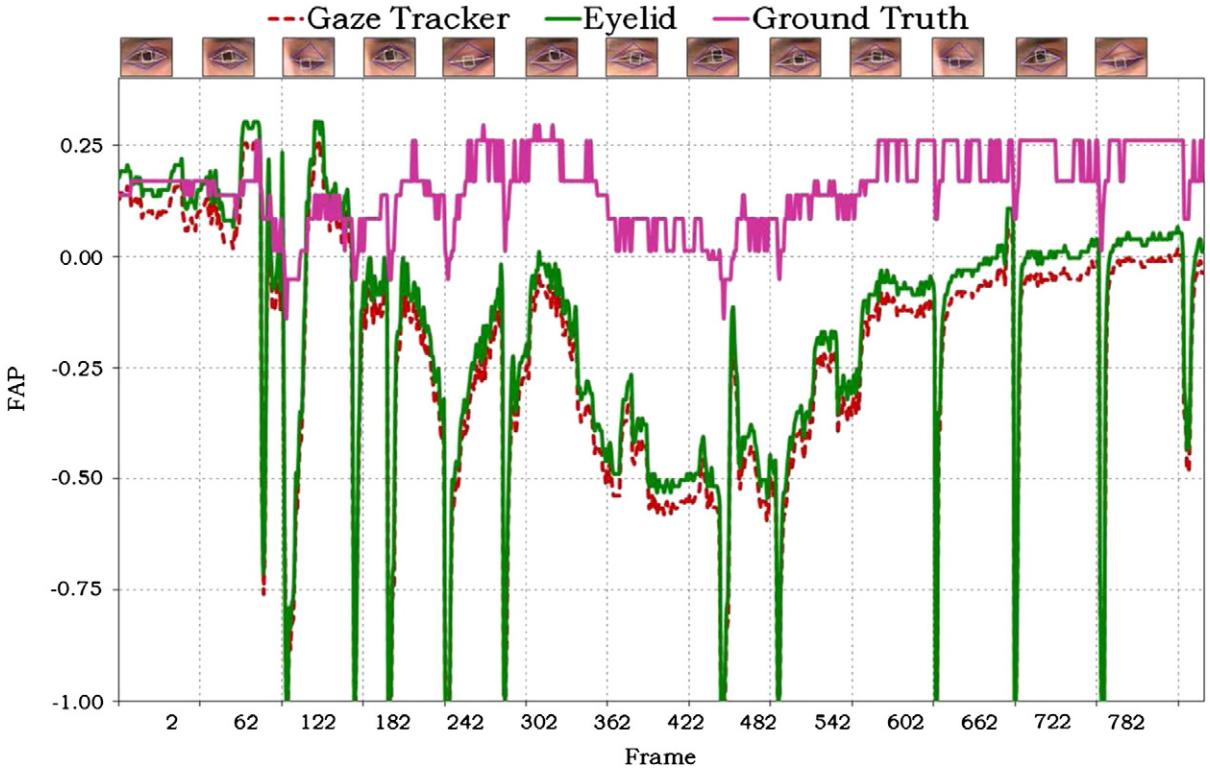


**Fig. 9.** The ground truth with the mislabeled eye blinks, but the eyelid tracker detects blinks as continuous FAP values within the range $[-1.0, 1.0]$ with a two decimal accuracy. The gaze tracker (dash line) presented in [23] can also estimate eyelid blinks with a lower accuracy than using an OABT with LMA. In addition, cropped images of the eyes are displayed at the top of the graph, which display the tracking fitting by white contour lines.

obtained when applying the gaze tracking presented in [23], which used a GNI algorithm for near frontal faces.

### 5.3. Eyelids tracking

Psychological studies addressed the importance of analyzing eyelid movements for emotion analysis, image encoding and Human Computer Interaction (HCI). These movements are characterized by the Facial Actions Coding System (FACS) [39].

The eyelids tracker, $\vec{T}_{\vec{w}}$, estimates the vector $\vec{w}=\left[\vec{q},\gamma_6\right]$, but the main task of this tracker is to estimate the eyelid facial action $\gamma_6\in[-1.0, 1.0]$, as a continuous variable. The lower bound of the FAP range corresponds to closed eyes whereas the upper bound corresponds to completely open eyes. The LMA minimization process in Algorithm 2 generates an appearance sub-space of $k$ possible solutions. The backtracking procedure estimates thedamping factor and the direction of the gradient based on a harmonic series. Consequently, the OABT can handle both the normal eye closure and the spontaneous blinks, which take on average two frames.

The eyelids tracker has been tested on an image sequence consisting of 700 frames, and recorded in the laboratory with a monocular camera and standard illumination. Each frame depicts a head and shoulders, and performance of extreme eyelid facial actions, which deform the eyelid surface in 3D. Fig. 10 shows the eyelid tracking estimations where the FAP values, $\gamma_6$, are continuous values in the range $[-1.0,1.0]$. Key frames have been highlighted by displaying cropped eye-regions at the top of the plot.

Note that even without using the edge detectors, the eyelid tracker is capable of tracking smooth eyelid movements like slit, closed and squint eyes. On the other hand, posed and spontaneous movements like raised eyelid, tightening, winks and blinks, are handled with high accuracy. Recall that eyelid position estimations are independent of the iris position because inner eye pixels are not warped onto the appearance texture of the eyelid tracker.

Fig. 10 shows the eyelid estimations by using the approaches proposed in [43,16], which provide estimates as discrete states. By contrast, the eyelid OABT provides estimates with two decimals of accuracy. Wide valleys or peaks correspond to normal raising-closing motion while sharp valleys and peaks are detected blinks. The similar results can be obtained when using the gaze tracker presented in [23] for near-frontal faces.

### 5.4. Iris tracking

The iris tracker, $\vec{T}_{\vec{g}}$, estimates the vector $\vec{g}=\left[\vec{w},\gamma_7,\gamma_8\right]$, i.e., it refines previous estimations of the head pose, eyebrows, lips and eyelids while estimating the iris pitch and yaw movements. Therefore, this tracker includes the whole vector $\vec{g}=\left[\vec{\rho},\vec{\gamma}\right]$ estimations. The

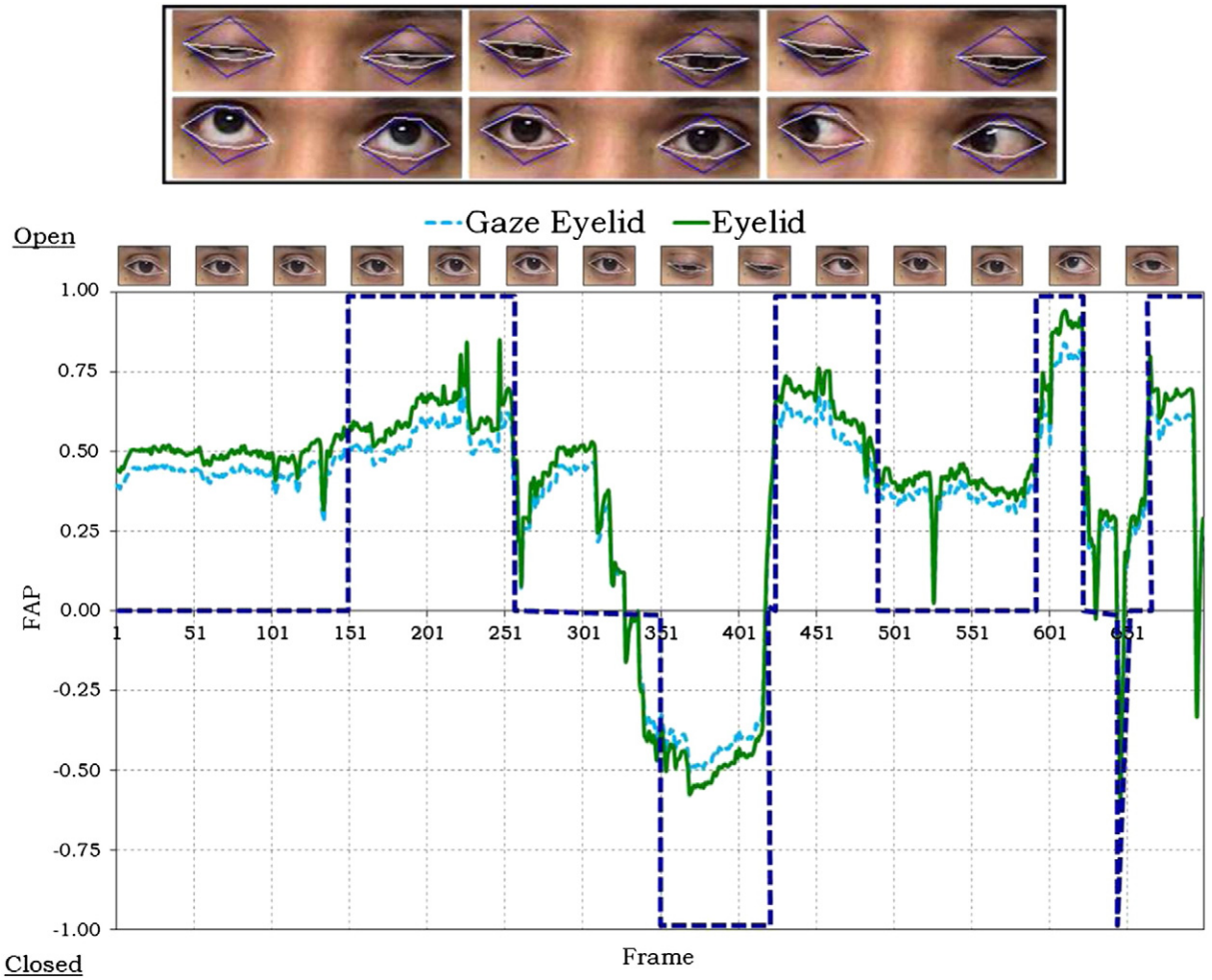


**Fig. 10.** The eyelid tracker, $\vec{T}_{\vec{w}}$, estimates eyelid positions as continuous variables rather than discrete states as shown by the dashed dark lines. Solid lines correspond to the OABT estimations within the FAP range $[-1.0, 1.0]$ with a two decimal accuracy. Similar results are obtained using the gaze tracker in [23] for near-frontal gaze movements. These results can also be verified by observing the eye cropped images displayed at the top of the plots. The drawn white lines correspond to the eyelid tracking fitting at each frame.

iris yaw and pitch parameters are evaluated as continuous variables in the same FAP range [−1.0,1.0], encoding movements from left to right and down to up, respectively.

The performance of this tracker has been tested on an image sequence of 500 frames of size 640×480 pixels. This video was recorded in a laboratory with a commercial photographic camera in VGA mode and standard illumination. The actor performs iris movements in all directions and looking askance, Fig. 11. The iris tracker encodes information related to four different FACS [39]; eyes turned up, down, left, right and extreme movements like askance where the iris appears partially occluded or distorted by the 3D perspective. However, involuntary movements such as iris saccades are commonly detected especially after eyelid occlusions.

Fig. 11 shows the iris tracker estimates for yaw and pitch movements. The dashed and light curves represent gaze-tracking results attained by using the discrete scale of the related approach [16]. Iris position estimations are displayed by drawing the fitted white box around the iris. The better the fitting the more accurate the iris tracking. The key frames are highlighted such as frames 101 and 151 where the iris pitch, $\gamma_8$, is estimated as about −0.5 when the subject is looking down. Likewise, at the frame 300 the iris yaw is estimated as $\gamma_7 = -1.0$ because the subject is looking askance to the left, whereas the frame 400 shows a $\gamma_7 = 1.0$ to indicate that the person is looking askance to the right. Fig. 11 shows the results for both eyes, including frames where the eyelids are occluding the irises. The gaze tracker presented in [23] can also be applied to obtain the continuous FAP estimations with similar accuracy. However, the gaze tracker loses its accuracy in non-frontal head poses.

### 5.5. Hierarchical tracking

We showed in our previous works (e.g., see [21]) that appearance based trackers should model shape and texture including only facial actions that are accurately fitted. Otherwise, the estimation error per pixel is propagated to the texture template. To this end, the eyelids and iris movements are estimated using two independent and non-occluded models, similarly as in [23]. The pitch and yaw iris movements are differentiated within shorter range and steepest descent than eyelid facial actions. Moreover, the backtracking procedure is also different for both trackers.

To test the ability of the iris tracker to accurately estimate the eyelids position, the FGnet sequence is first tracked with the iris tracker alone and then with the eyelid and iris trackers, applied hierarchically (see Fig. 12). The experiment is performed using the FGnet dataset [42]. The Gaze Tracker produces an average error per pixel of 3.5, whereas the iris and hierarchical tracker produce an average error

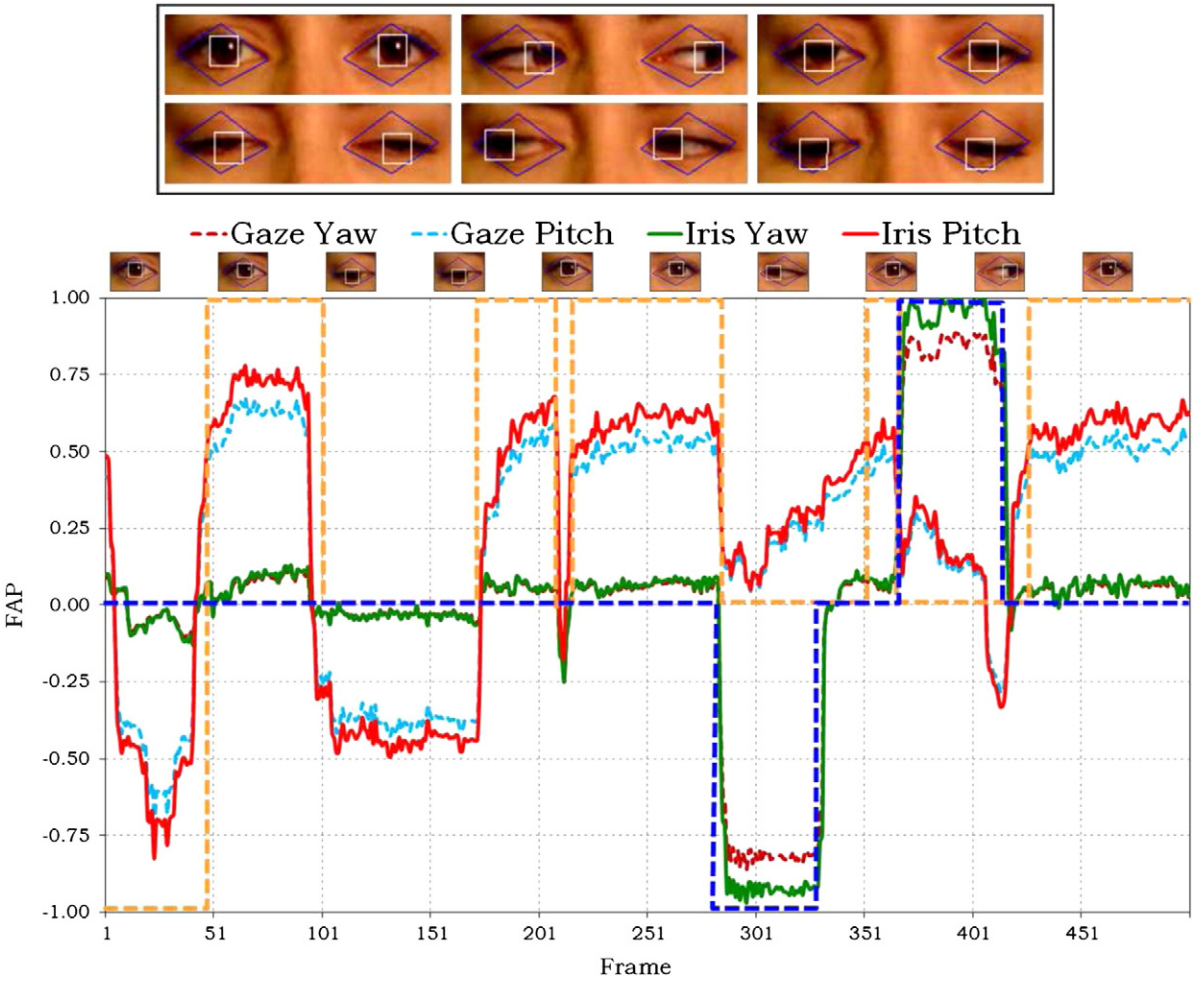


**Fig. 11.** The iris tracker, $\vec{T}_{\gamma}$, estimates pitch (orange solid line) and yaw (blue solid line) iris movements as continuous variables. Iris pitch corresponds to gaze directions up (+1.0) and down (−1.0) while iris yaw refers to right (+1.0) and left (−1.0) directions. Square dash lines correspond to discrete state estimations as previous related works. A gaze tracker [23] can be applied to obtain similar iris estimations but with lower accuracy. The eye region images (top) show the iris tracking estimations by drawing a white box.

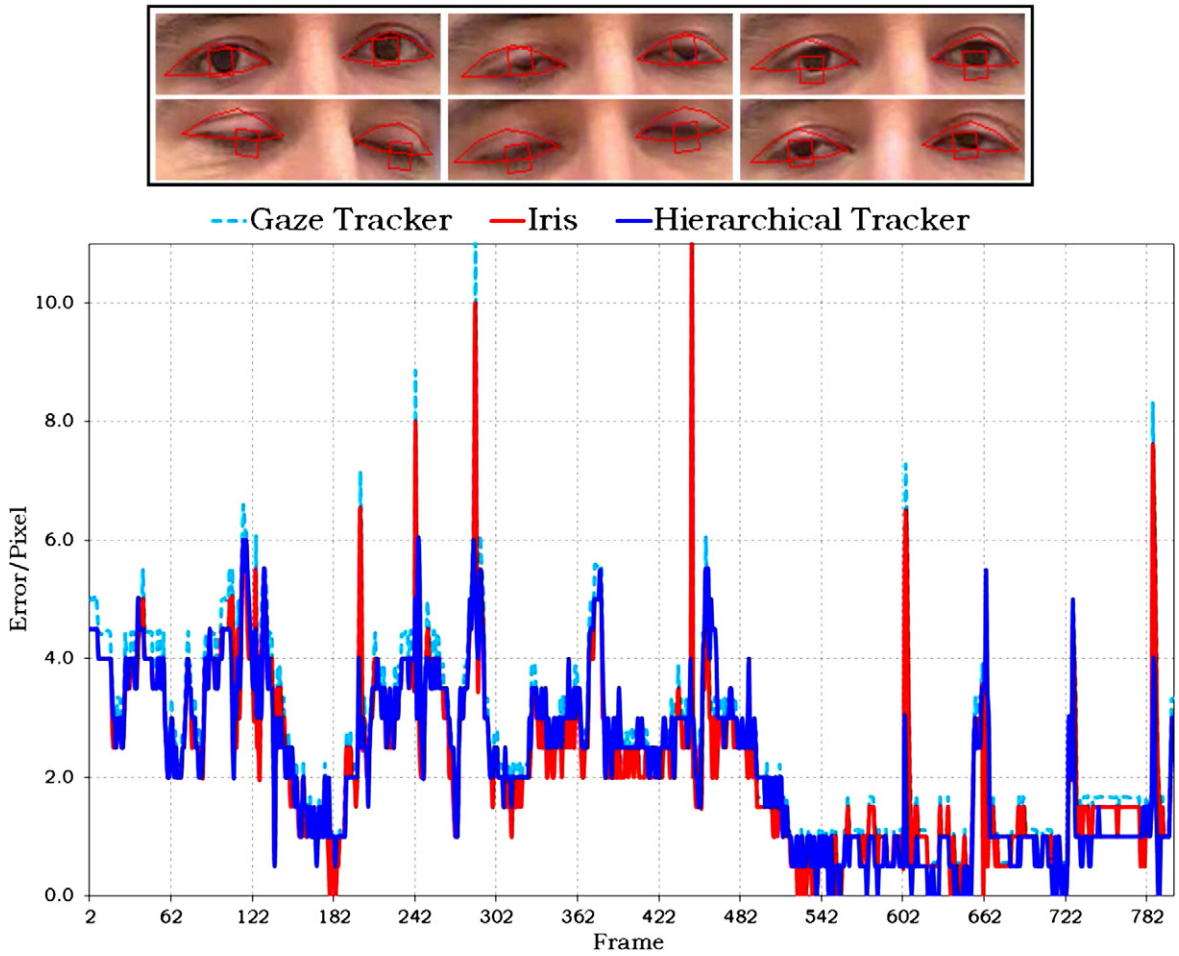


**Fig. 12.** The iris tracker is tested with and without previous use of the eyelid tracker. The top gallery shows cropped eye-region images with the tracking estimations drawn by red lines at the contours of eyelids and irises. The bottom plot shows the tracking performance in terms of error/pixel. The red line shows the error/pixel when using only the iris tracker whereas the blue line shows the error/pixel of the hierarchical tracker, $\vec{T}_{g^*}$, after applying $\vec{T}_{q^*}$ and $vecT_{w^*}$. The dashed line corresponds to the gaze tracker accuracy [23]. The Gaze Tracker produces an average error/pixel of 3.5, whereas the iris and hierarchical tracker report average error/pixel of 2.2 and 2.3, respectively.

per pixel of 2.2 and 2.3, respectively. The error per pixel increases for the iris tracker when eyes blinks occur whereas the hierarchical tracker reduces this error due to the correct eyelid estimations. The iris reference texture considers the occluding eyelid pixels as outliers and consequently these pixels are not warped onto the reference appearance texture. In conclusion, we confirm the results from [21] where it was shown that the iris tracker is not capable of estimating the eyelid movements. Thereby, all other movements such as head and facial actions are not accurately tracked by the iris tracker alone. The gaze tracker in [23] uses a sequential eyelid and iris OABT based on GNI, and it achieves similar accuracy to that shown in Fig. 12. Yet, it is highly sensitive to head pose variation.

The hierarchical tracking obtains first the estimates of the 3D head pose, eyebrows and lips, $\vec{T}_{q^*}$. Subsequently, it obtains the eyelid position from the eyelid tracker $\vec{T}_{w^*}$. Finally, the iris tracker $\vec{T}_{g^*}$ estimates the iris movements while improving estimations of the previous trackers (see Fig. 13). The eyelid estimation is more accurate because of the second iterative process that attains the best eyelid adaptation. Consequently, the estimation error decreases for both global head motion and local facial actions. These results are better than the gaze tracking results in [23], since the latter method is sensitive to non-frontal faces. Its average gaze tracking error per pixel for the eyelids is 3.5, while the eyelid and hierarchical trackers with LMA obtain an average error per pixel of 3.1 and 2.8, respectively.

### 5.6. Lighting conditions

Most appearance trackers suffer from drifting problems due to their sensitivity to illumination changes. To alleviate this, an exhaustive training of textures and shapes of AAMs is required. To demonstrate the ability of the proposed hierarchical tracker to successfully handle illumination changes, an image sequence of 800 frames is tested. Images of $352 \times 288$ pixels size were recorded with a web camera in an indoor scenario while a fluorescent lamp was intermittently illuminating the face in several positions (see Fig. 14). The camera is at the bottom of the image plane, which is an additional challenge for pre-calibrated vision systems.

Fig. 14 shows well fitted 3D shape models of the hierarchical tracking under changing illumination. The tracker has a controlled learning ability based on the model likelihood enhanced by Huber's function. Furthermore, each hierarchical tracker has a different learning rate (see Eq. (7)), so that they can cope with three sorts of kinematics: head–eyebrows–lips, eyelids and irises. For example, at frame 525, the light is turned off and the estimation error increases in 3.0 pixels, as shown in Fig. 15. The FAP plot shows how both eyelid

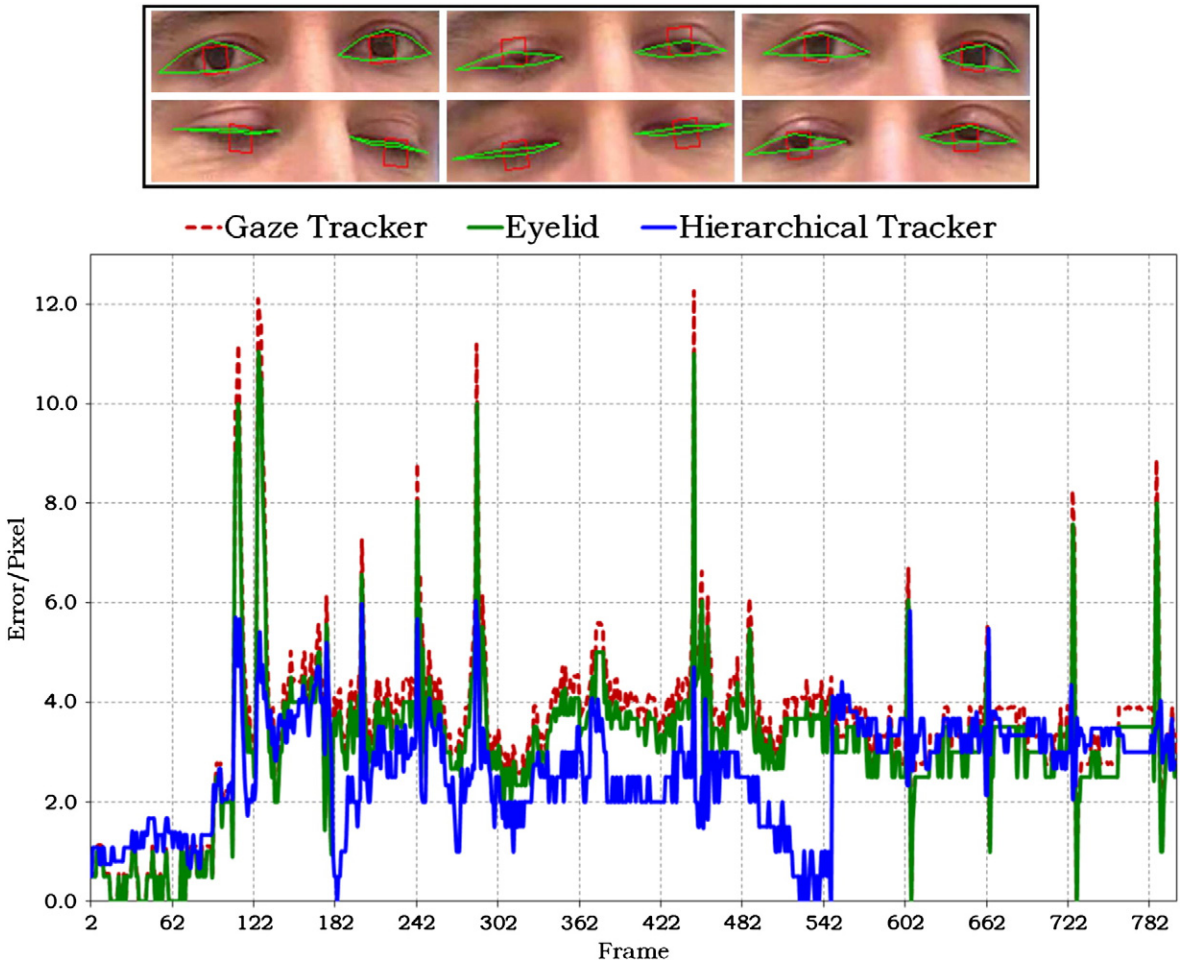


**Fig. 13.** The gaze tracker [23] accurately estimates eyelid and iris positions, but it is sensitive to non-frontal faces. Instead, the hierarchical tracker $\vec{T}_{g^*}$ improves the eyelid tracking estimations during the iris tracking with accurate head pose estimations. Eye-region images are also used to show the tracking fitting by drawing green and red polygons for eyelid and iris tracking, respectively. The Gaze Tracker produces an average error/pixel of 3.5, whereas the eyelid and hierarchical trackers obtained average error/pixel of 3.1 and 2.8, respectively.

and iris trackers need extra time to accommodate for the new environment illumination. See also Fig. 16 for details on gaze tracking estimations.

So far, the hierarchical tracking has shown higher accuracy and stability to illuminations changes than a gaze tracker based on a GNI and two eye region shape models [23] (see Figs. 15, 16). Assuming non-flashing lights, the number of iterations for the LMA can be the same in all trackers. As long as the new illumination conditions remain stable, the hierarchical tracking improves the estimations. Therefore, expected appearances have less information from previous illumination and the LMA algorithm becomes stable, as shown in Fig. 15, after the frame 541.

### 5.7. Occlusions and real-time

As expected, estimation error increases with occlusions and illumination changes. However, a right combination of learning rates allows the hierarchy of trackers to recover without propagating the error onto the observation models.

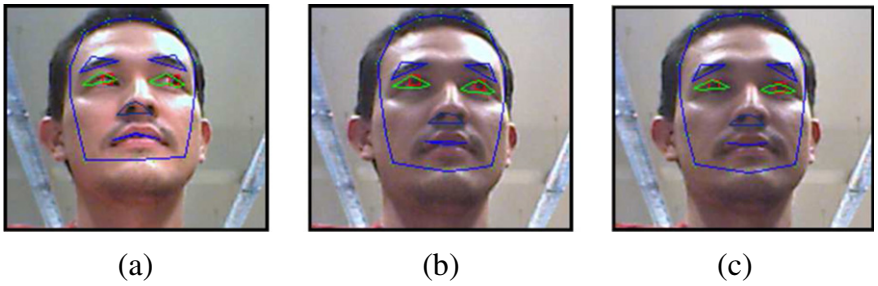


**Fig. 14.** Illumination changes pose a significant challenge for pre-trained AAMs. The hierarchical OABT can handle flashing lights due to the on-line appearance learning, a Huber's function and backtracking procedures. Thus, outlier pixels are rejected, and the AAM is only updated with the highly likely pixel variations. (a) Facial actions are correctly tracked before changing the illumination. (b) and (c) show the tracking results after the 541*th* frame. The error per pixel increases at these frames due to the illumination variation.

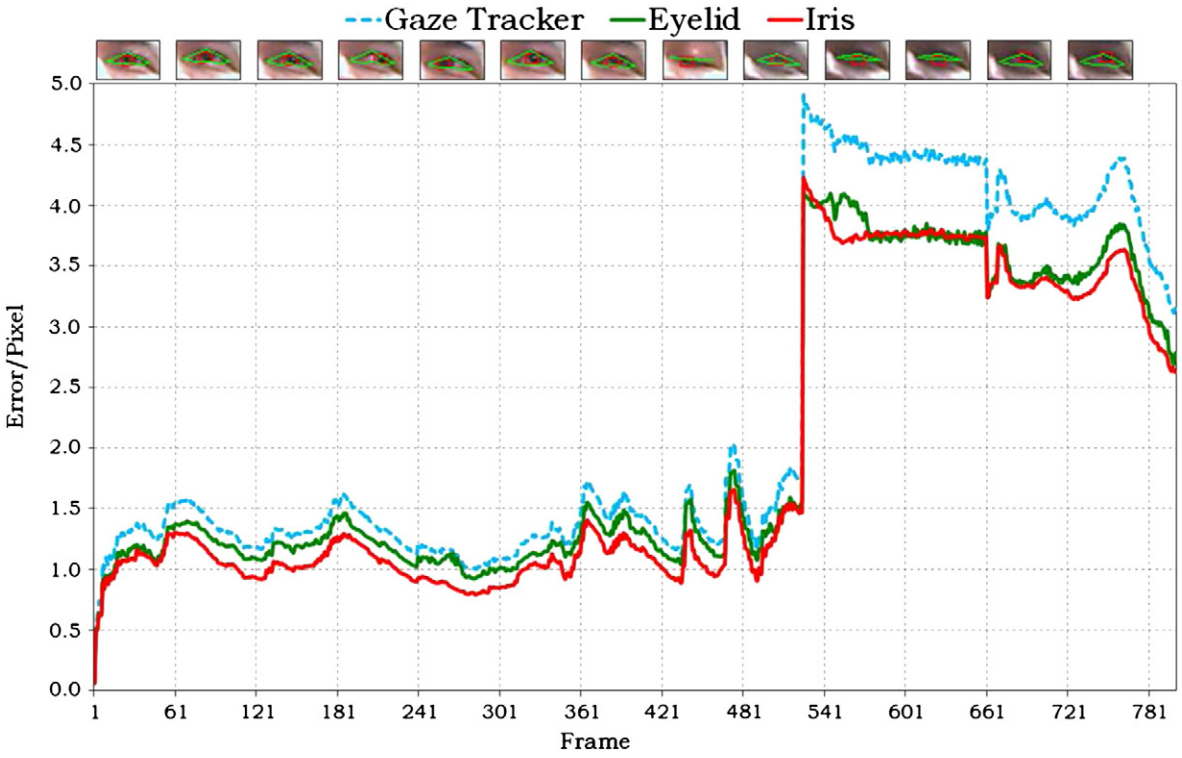


**Fig. 15.** Under different lighting conditions, the tracking error increases until the observation model is updated accordingly to the Huber's function. Once the error decreases the OABT gets stable, robust and accurate. The hierarchical tracker is more stable than using a gaze tracker as in [23].

To further test the robustness of the proposed hierarchical OABT, an image sequence of 600 frames with severe occlusions is used. This sequence was recorded indoors with a monocular camera. The actor performs head movements and exaggerated facial actions while the illumination is subtly changed. At one frame, the actor starts wearing eyeglasses, thereby, producing occlusions and intensity variations. Other local occlusions are also captured in this sequence, such as eyelid blinks, which occlude the iris region for three

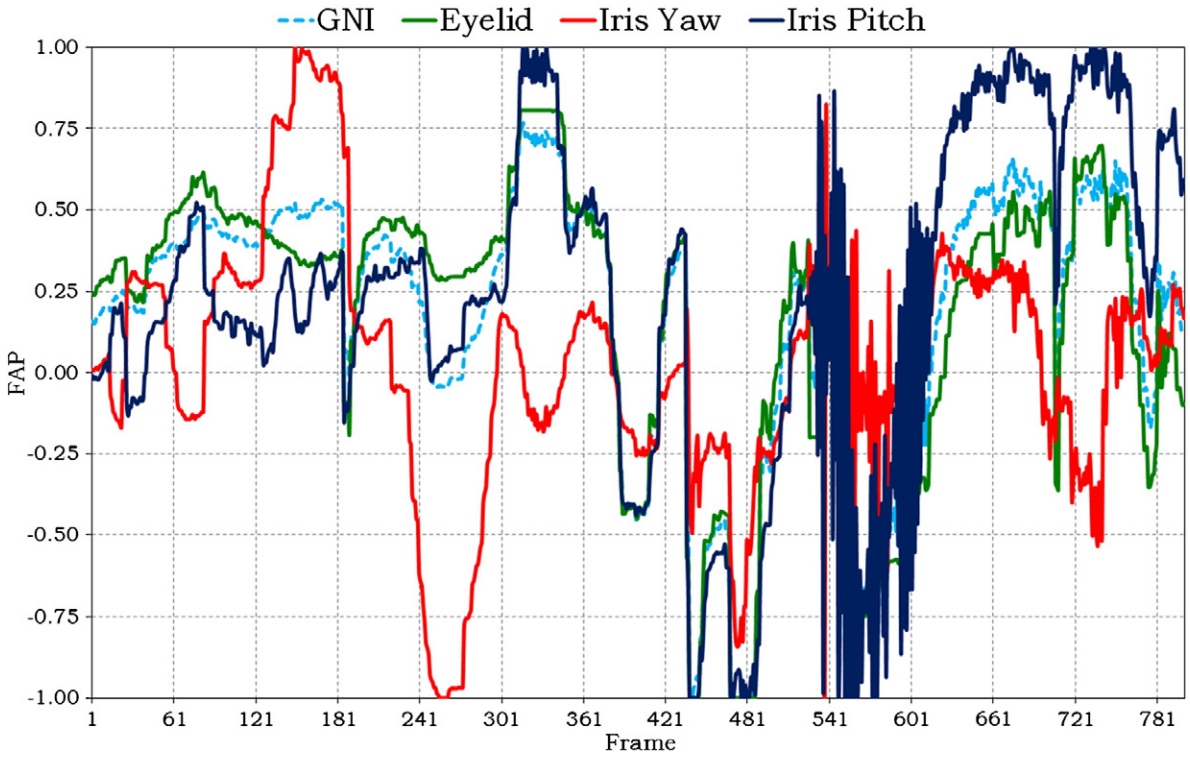


**Fig. 16.** The tracking recovers stability if illumination changes are not extreme, otherwise, the noise ends up filtering both observation and transition processes, hence drifting problems arise. Here, the hierarchical OABT also outperforms OABT with GNIin [21].

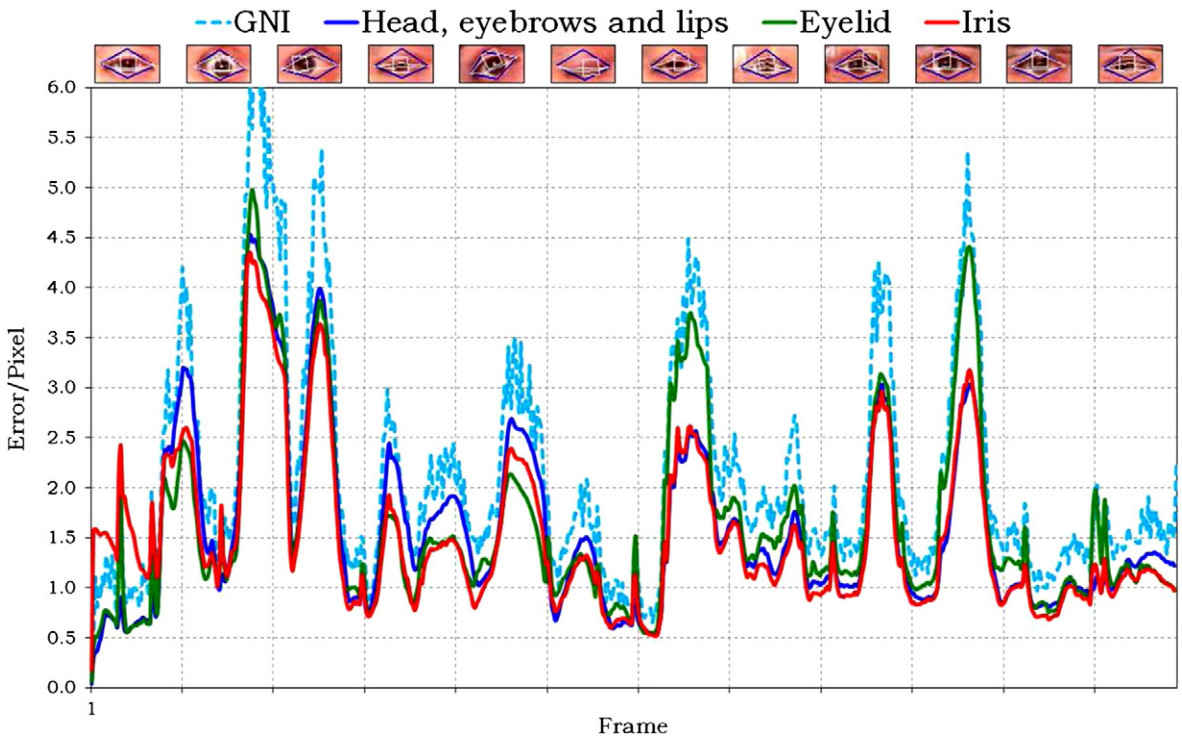


**Fig. 17.** This sequence exhibits both blinks and saccades. For example, we can see eyelid blinks at frames 17, 164, and 232 of the iris curve. We can also see iris saccadic movements at frames 96, 330 and 483 of the eyelid curve. The OABT with GNI [21] performs with higher error/pixels as it cannot track iris movements.

or more frames. Consequently, saccadic movements are expected since the iris has to recover. This re-adaptation can take one or two more frames while increasing the estimation error, for example, at frames 17, 164, and 232 in Fig. 17. This shows the influence of the iris movements on the estimation error of the eyelid tracker at frames 96, 330, and 483, which is evident according to the peaks of the eyelid curve. Notice how the error per pixel is higher when using the previous tracker based on GNI [21].

Next, we test the real-time performance of the proposed tracker. To this end, we implemented an ABT with small appearance reference texture, 1426 pixels (i.e. $40\times42$ pixels). This resulted in the tracker running real-time. Accuracy, effectiveness and robustness are tested by comparing the convergence error, output images and the spent time to find the correct adaptation. This is shown in Fig. 18.

The hierarchical tracking with a small appearance resolution produces results of an average 85% of correct adaptations and the performance of 32 frames per second (fps). On the other hand, when using the big appearance resolution, the tracker produces an average of 96% correct adaptations and performance of 1.1 fps. It is worth mentioning that iris has less pixels in the small resolution, $2\times3$ pixels, than $5\times6$ pixels for the big resolution.

### 5.8. Large head movements

Out-of-plane movements have a strong impact on the 3D head pose estimation and more than the 50% of pixels are considered outliers, which results in noisy appearances. However, the Huber's function provides stability to both observation and state transition processes.

An image sequence of 650 frames is used, where the subject performs large head movements and facial actions, to assess the effect of 3D head pose inaccuracies on the facial action tracking (see Fig. 19). The number of iterations was increased, thus extending the ratio of the line search stage without losing the real-time performance. Fig. 19(a) depicts extreme head rotations where the facial actions remain almost unaffected by the introduced noise. The 2D projection of out-of-plane rotated faces produces very small errors in the image plane, comparable to that of the alignment error between the shape and the regions of eyebrows, lips, eyelids and irises.

| Frame | 36 | 136 | 328 | 446 | 487 |
|---|---|---|---|---|---|
| Big $\chi$ | 1.1 fps | 1.1 fps | 1.1 fps | 1.1 fps | 1.2 fps |
| Small $\chi$ | 32.5 fps | 32.5 fps | 28.3 fps | 26.7 fps | 36.4 fps |

**Fig. 18.** Using an appearance template $\vec{\chi}$ 1426 the tracking achieves 32 fps while using a double template resolution the frame rate drops to 1.1 fps. About 85% of the frames are correctly adapted running at real-time frame rate while 96% of the correct adaptations are obtained using big appearance template.

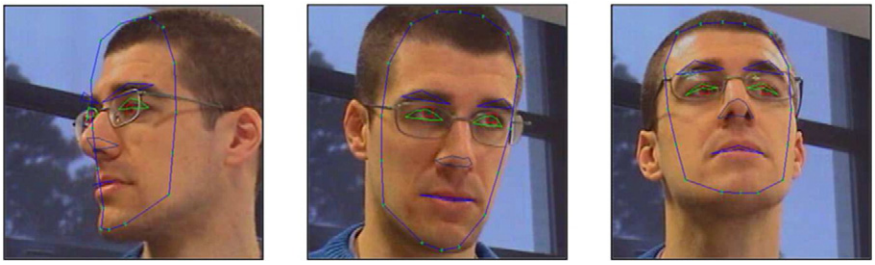

**Fig. 19.** The hierarchical OABT handles out-of-plane movements as outliers, retrieving the position over 50% of outlier pixels. Once the head returns nearby to previous locations the tracking recovers, otherwise re-initialization is required. This sequence was recorded with a monocular camera.

However, the tracking still accurately handles their independent movements.

## 6. Conclusions

We proposed an efficient method for simultaneous tracking of head, eyebrows, lips, eyelids and irises that can run real-time. We have extended our previous work on on-line appearance-based trackers to deal with eyelid and iris motions. By adopting three OAMs we decoupled rigid and non-rigid movements to avoid self-occluding facial actions. Three OABTs are built and coupled through Levenberg–Marquardt Algorithm optimized with backtracking procedures. This resulted in substantial improvement in accuracy of the gaze tracking and robustness of the simultaneous tracking of the head and facial actions.

The proposed hierarchical OABT holistically estimates the 3D head pose and facial actions related to lips, eyebrows, eyelids and irises. Compared with other gaze tracking techniques, the proposed approach avoids completely intensity edge segmentation.Tracking estimations are encoded as FAP parameters according to the MPEG-4 format, such that all the estimations are continuous variables of single precision in floating point.

The proposed tracking approach is based upon a robust transition process that estimates an optimal 3D shape model by the means of a modified LMA, which avoids local minima by differentiating each facial action within specific FAP ranges and differentialsteps. Due to the additional challenge of gaze tracking with deformable models, the LMA is combined with backtracking and line-search procedures to estimate the steepest descent factor for each facial action. Consequently, eyelids and iris OABTs result in efficient trackers capable of handling eye blinks and iris saccadic movements.

Experiments on the FGnet database, video sequences from the internet and our laboratory, showed the accuracy and robustness of our proposed method. The FGnet database for face tracking was tested in three experiments showing the accuracy with respect to a manually annotated ground truth. The experiments also show the tracking stability in presence of illumination changes, translucent surfaces on the eye region, occlusions and out-of-plane movements.

The proposed method is suitable for studies on human behavior analysis, facial expressions and facial emotions where the latest state-of-the-art addressed solutions are based on spatio-temporal interpretations of facial deformations. On the other hand,the real-time performance of this tracker allows it to be used in Human Computer Interaction applications, since it can work with standard monocular video cameras and image resolutions.

The proposed tracking approach can be extended in several directions. Firstly, automatic initialization of the first frame is still an open issue. Although we outlined a semi-automatic initialization of the tracker, a more sophisticated approach that can deal with various poses and facial expressions is needed. Secondly, a further investigation on appearance texture registration is required to improve the on-line learning of profile faces beyond pan rotation angles of $\pm 45°$. Thirdly, the self-occluded deformable models affect mainly the state transition process and, thereby, bring the necessity for using three different OABTs. Additional research on deformable models or different fitting methods is required to reduce the number of trackers to a single tracker with improved accuracy and robustness. Lastly, the application of other optimization methods [44], line-search direction and backtracking [45], aiming at extension of this tracking system to multiple faces in the same image.

## Acknowledgments

This work has been supported by the European Research Council under the ERC starting grant agreement no. ERC-2007-StG-203143 (MAHNOB). The work of Javier Orozco was also supported in part by the European Community's 7th Framework Programme [FP7/20072013] under grant agreement no. 231287 (SSPNet). The work of Jordi Gonzàlez was supported by the Consolider-Ingenio2010 MIPRCV-CSD200700018; Avanza I+D ViCoMo, TSI-020400-2009-133 and DiCoMa TSI-020400-2011-55; along with the Spanish projects TIN2009-14501-C02-01 and TIN2009-14501-C02-02.

## References

[1] Z. Zeng, M. Pantic, G. Roisman, T. Huang, A survey of affect recognition methods: audio, visual, and spontaneous expressions, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2009) 39–58.
[2] S. Gokturk, J. Bouguet, R. Grzeszczuk, A data-driven model for monocular face tracking, Eighth International Conference on Computer Vision, vol. 2, 2001, pp. 701–708.
[3] E.-J. Ong, R. Bowden, Robust facial feature tracking using shape-constrained multi-resolution selected linear predictors, IEEE Trans. Pattern Anal. Mach. Intell. 33 (2011) 1844–1859.
[4] T. Cootes, C. Taylor, D. Cooper, J. Graham, Active shape models — their training and application, Comput. Vis. Image Underst. 61 (1995) 39–59.
[5] T. Cootes, G. Edwards, C. Taylor, Active appearance models, Trans. Pattern Anal. Mach. Intell. 23 (2001) 681–684.
[6] J. Ahlberg, An active model for facial feature tracking, EURASIP J. Appl. Signal Process. 2002 (2002) 566–571.
[7] M.L. Cascia, S. Sclaroff, V. Athitsos, Fast, reliable head tracking under varying illumination: an approach based on registration of texture-mapped 3d models, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 322–336.
[8] I. Matthews, S. Baker, Active appearance models revisited, Int. J. Comput. Vis. 60 (2004) 135–164.
[9] J. Sung, D. Kim, Adaptive active appearance model with incremental learning, Pattern Recognit. Lett. 30 (2009) 359–367.
[10] X. Liu, Video-based face model fitting using adaptive active appearance model, Image Vis. Comput. 28 (2010) 1162–1172.
[11] J. Nuevo, L. Bergasa, D. Llorca, M. Ocana, Face tracking with automatic model construction, Image Vis. Comput. 29 (2011) 209–218.
[12] R. Gross, I. Matthews, S. Baker, Generic vs. person specific active appearance models, Image Vis. Comput. 23 (2005) 1080–1093.
[13] I. Matthews, T. Ishikawa, S. Baker, The template update problem, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 810–815.
[14] T. Ishikawa, S. Baker, I. Matthews, T. Kanade, Passive Driver Gaze Tracking with Active Appearance Models, Technical Report CMU-RI-TR-04-08, Robotics Institute, Pittsburgh, PA, 2004.
[15] H.Z.H. Liu, Y. Wu, Eye states detection from color facial image sequence, In proceedings of the 2nd International Conference of Image and Graphics, vol. 4875, 2002, pp. 693–698.
[16] A.R.S. Sirhey, Z. Duric, A method of detecting and tracking irises and eyelids in video, in: International Conference on Pattern Recognition, vol 35, 2002, pp. 1389–1401.
[17] H. Liu, Y. Wu, H. Zha, Eye states detection from color facial image sequence, in: SPIE International Conference on Image and Graphics, vol. 4875, 2002, pp. 693–698.

[18] Y. Wu, H. Liu, H. Zha, A new method of detecting human eyelids based on deformable templates, in: International Conference on Systems, Man and Cybernetics, vol. 1, 2004, pp. 604–609.
[19] T. Moriyama, J. Xiao, J. Cohn, T. Kanade, Meticulously detailed eye model and its application to analysis of facial image, IEEE Trans. Pattern Anal. Mach. Intell. 28 (2006) 738–752.
[20] R. Valenti, N. Sebe, T. Gevers, What are you looking at? Improving visual gaze estimation by saliency, Int. J. Comput. Vis. 98 (2012) 324–334.
[21] F. Dornaika, J. Orozco, Real time 3d face and facial feature tracking, J. Real-Time Image Proc. 2 (2007) 35–44.
[22] J. Nocedal, S. Wright, Numerical Optimization, Springer, 1999.
[23] J. Orozco, F. Roca, J. Gonzàlez, Real-time gaze tracking with appearance-based models, Mach. Vis. Appl. 20 (2009) 353–364.
[24] K. Levenberg, A method for the solution of certain problems in least squares, Q. Appl. Math. 2 (1944) 164–168.
[25] D. Marquardt, An algorithm for least-squares estimation of non-linear parameters, SIAM J. Appl. Math. 11 (1963) 431–441.
[26] M. Rydfalk, Candide, a parameterized face, Technical Report, Dept. of Electrical Engineering, Linkoping University, Sweden, 1987.
[27] S. Milborrow, F. Nicolls, Locating facial features with an extended active shape model, Proceedings of the 10th European Conference on Computer Vision: Part IV, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 504–513.
[28] M.F. Valstar, B. Martinez, X. Binefa, M. Pantic, Facial point detection using boosted regression and graph models, Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, San Francisco, USA, 2010, pp. 2729–2736.
[29] B. Martinez, M.F. Valstar, X. Binefa, M. Pantic, Local evidence aggregation for regression based facial point detection, IEEE Trans. Pattern Anal. Mach. Intell. 99 (2012) 1-1.
[30] X. Zhu, D. Ramanan, Face detection, pose estimation, and landmark localization in the wild, Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2012, pp. 2879–2886.
[31] Y. Aloimonos, Perspective approximations, Image Vis. Comput. 8 (1990) 177–192.
[32] T. Cootes, C.J. Taylor, Statistical Models of Appearance for Computer Vision, Imaging Science and Biomedical Engineering, University of Manchester, 2004.
[33] J. Ahlberg, Candide-3 — an updated parameterized face, Tech. Rep. LiTH-ISY-R-2326, department of Electrical Engineering, 2001.
[34] I. Matthews, S. Baker, Active appearance models revisited, Technical Report CMU-RI-TR-03-02, The Robotics Institute, Carnegie Mellon University, 2002.
[35] A. Jepson, D. Fleet, T. El-Maraghi, Robust on-line appearance models for visual tracking, IEEE Trans. Pattern Anal. Mach. Intell. 25 (2003) 1296–1311.
[36] A. Cauchy, Mèthodes gènèrales pour la rèsolution des systèmes dèquations simultanèes, C. R. Acad. Sci. Paris 25 (1847) 536–538.
[37] P. Huber, Robust estimation of a location parameter, Ann. Math. Stat. 35 (1964) 73–101.
[38] D. Clark, Comparing Huber's m-estimator function with the mean square error in backpropagation networks when the training data is noisy, The Information Science Discussion Paper Series, 19, 2000, pp. 1–12.
[39] P. Ekman, W.V. Friesen, Facial Action Coding System: A Technique for the Measurement of Facial Movement, Consulting Psychologists Press, Palo Alto, 1978.
[40] J. Taylor, J. Elsworth, M. Lawrence, J. Sladek, R. Roth, J. Redmond, Spontaneous blink rates correlate with dopamine levels in the caudate nucleus of MPTP-treated monkeys, Exp. Neurol. 158 (1999) 214–220.
[41] E. Castet, G. Masson, Motion perception during saccadic eye movements, Nat. Neurosci. 3 (2000) 177–183.
[42] T. Cootes, Talking face video, www-prima.inrialpes.fr/FGnet2002, (FGnet - IST-2000-26434).
[43] T. Moriyama, T. Kanade, J. Cohn, Z. Ambadar, J. Gao, H. Imamura, Automatic recognition of eye blinking in spontaneosly occurring behavior, Proceedings of the 16th International Conference on Pattern Recognition, vol. 4, IEEE Computer Society, Washington, DC, USA, 2002, pp. 78–81.
[44] J. Shewchuk, An Introduction to the Conjugate Gradient Method without the Agonizing Pain, School of Computer Science, Carnegie Mellon University, USA, 1994.
[45] N. Andrei, A New Gradient Descent Method with an Anticipative Scalar Approximation of Hessian for Unconstrained Optimization, Research Institute for Informatics, Romania, 2005.