sensors

MDPI

Article

An Adaptive Face Tracker with Application in Yawning Detection

Aasim Khurshid [1,2,*] and Jacob Scharcanski [2]

[1] Sidia Instituto de Ciencia e tecnologia, Amazonas, Manaus 69055-035, Brazil
[2] Instituto de Informatica, UFRGS, Porto Alegre 9500, Brazil; jacobs@inf.ufrgs.br
* Correspondence: aasim.khurshid@sidia.com

Received: 29 January 2020; Accepted: 24 February 2020; Published: 9 March 2020

Abstract: In this work, we propose an adaptive face tracking scheme that compensates for possible face tracking errors during its operation. The proposed scheme is equipped with a tracking divergence estimate, which allows to detect early and minimize the face tracking errors, so the tracked face is not missed indefinitely. When the estimated face tracking error increases, a resyncing mechanism based on Constrained Local Models (CLM) is activated to reduce the tracking errors by re-estimating the tracked facial features' locations (e.g., facial landmarks). To improve the Constrained Local Model (CLM) feature search mechanism, a Weighted-CLM (W-CLM) is proposed and used in resyncing. The performance of the proposed face tracking method is evaluated in the challenging context of driver monitoring using yawning detection and talking video datasets. Furthermore, an improvement in a yawning detection scheme is proposed. Experiments suggest that our proposed face tracking scheme can obtain a better performance than comparable state-of-the-art face tracking methods and can be successfully applied in yawning detection.

Keywords: face tracking; error prediction; features resyncing; online learning; incremental PCA; yawning detection; feature extraction for emotion analysis

1. Introduction

Object visual tracking essentially deals with locating, identifying, and determining the dynamics of moving (possibly deformable) target objects in various areas such as car tracking [1], face detection [2], and driver monitoring [3]. Representational methods are applied successfully for dimensionality reduction and improve discriminative ability in classification problems [4]. Some visual object tracking methods applied representational based methods with pre-computed fixed appearance models [5]; however, the visual appearance of the tracked target object may change along the time and for this reason they may interrupt tracking the target object after a period of time when the tracking conditions change (e.g., the scene illumination changes, occlusions). Some authors proposed to use the data generated during the tracking process to accommodate possible target appearance changes, such as in online learning [6], incremental learning for visual tracking (ivt) [7], patch based approach with online representation of samples [8], and in online feature learning techniques based on dictionaries [1]. Often, online visual tracking methods tend to miss the target object in complex scenarios, such as when the head pose changes while tracking faces, or in cluttered backgrounds and/or in object occlusions [9]. The reasons for this behaviour include the inability to access the tracking error and to update the object appearance at runtime. To approach these issues, Kim et al. [10] utilized a constrained generative approach to generate generic face poses in particle filtering framework, and a pre-trained SVM classifier to discard poorly aligned targets. Furthermore, Correlation filters based methods have become popular in visual object tracking [11]. Li et al. proposed a multi-view model for visual tracking via correlation filters (MCVFT), which fuses multiple features

and selects the discriminative features among them [11]. Similarly, Danelljan et al. proposed to use Spatially Regularized Discriminative Correlation Filters (SRDCF) to track visual objects [12]. Furthermore, Discriminant Correlation Filters (DCFs) are also used to train lightweight network architecture to learn the convolutional features for object tracking, such as in DCFNet [13]. Similarly, object tracking by reconstruction based on the online 3D construction of the target to learn DCFs proved to be efficient to track target face with various pose [14]. Moreover, Sanchez et al. [15] proposed an Incremental Cascaded Continuous Regression (iCCR) method for face tracking. The iCCR method is a new formulation for the Cascaded Continuous Regression (CCR) approach, which is adaptive and can be utilized in incremental learning.

On the other hand, geometric shape and appearance models such as Active Appearance Models (AAM) [9], Active Shape Models (ASM) [16] and Constrained Local Models (CLM) [17,18] can capture robust features even in cluttered or fast changing scenarios, improving the robustness of the tracking process. These methods often are based on local shape matching, and try to minimize the difference between a tracked target object and the learned target appearance (i.e., to maximize the shape matching). Unfortunately, most shape and appearance model based methods are not easily applicable to real time tracking due to their complexity. Nevertheless, combining online learning with shape and appearance models can increase the online learning efficiency (e.g., by using appearance models to correct the tracking process and reduce tracking failures).

The proposed approach is applied to face tracking and improves on a well-known object tracking method based on the incremental PCA [7]. The proposed scheme learns from the data generated during face tracking, and corrects the estimated tracking mistakes with a resyncing mechanism. A dynamic tracking error predictor is proposed to estimate how accurately the target face is being tracked. The tracking error predictor adapts itself in time and tends to be consistent in long video sequences (see Section 2.4). If the tracking error is estimated to be increasing, the tracking process is corrected by a resyncing mechanism based on CLM. Furthermore proposed is an improvement of the typical CLM named Weighted CLM (W-CLM) that assigns a weight to each landmark (feature point) based on its consistency in a temporal window (see Section 2.2). In this work, the proposed tracking method is applied to face and facial landmarks tracking, and CLM or W-CLM are used to re-adjust the facial features locations (landmarks) and avoid tracking failures. The proposed Adaptive Face Tracker with Resyncing Mechanism (AFTRM) optimizes the CLM search process without using the landmarks weights. Whereas, another proposed method named Adaptive Face Tracker with Resyncing Mechanism with Weights (AFTRM-W) applies a landmark weight (calculated during the W-CLM training phase) to improve the facial landmark search process. Face tracking based on facial features can provide a low cost solution for a number of measurement applications, such as yawning detection, expression analysis, fatigue detection and vigilance [19]. In this work, the tracked facial landmarks are evaluated in the context of face tracking and in a driving scenario application (i.e, yawning detection).

The major contributions of the paper include:

- Face tracker that can track face and facial landmarks in challenging conditions.
- The proposed tracking scheme utilizes the tracked target face samples collected during tracking to update the appearance model online to adapt to the shape and appearance changes in the tracked face along the time.
- A dynamic error prediction scheme to evaluate the correctness of the tracking process during face tracking.
- Utilization of a resyncing mechanism based on the Constrained Local Models (CLM), when the error predictor indicates high error.
- An improvement in the classical CLM approach, namely Weighted CLM (W-CLM) to improve the facial landmark localization.
- An improvement in a yawning detection scheme by using facial landmarks and imposing multiple conditions to avoid false positives.

The remaining of this paper is organized as follows. The proposed methodology is described in Section 2, followed by our experimental results in Section 3. Finally, Section 4 gives our conclusions and the future prospects of this work.

2. Proposed Adaptive Face Tracking Method

Figure 1 shows the block diagram of the proposed face tracking method, and the blocks functions are explained below:

- Block 1: In the first video frame, the initial target face, its affine parameters and the landmarks are localized using W-CLM (for details on W-CLM, see Section 2.2).
- Block 2: In order to track the target face in the subsequent video frames, new affine parameters values are drawn around the affine parameters values of the initial/tracked target face in the previous video frames (see details in Section 2.3).
- Block 3: The affine parameters previously computed are used to warp the current video frame candidate target face samples of size $u \times u$.
- Block 4: If a specific number (τ) of new target face samples have been gathered, the eigenbases are built.
- Block 5: If the condition in block 4 is satisfied, the candidate target face samples are decomposed into patches ($v \times v$ and $v \leq u$), because the eigenbases are built using patches (see Section 2.1).
- Block 6: The tracked target face is found among the candidate target face samples by maximizing the likelihood function in Equation (10) (see details in Section 2.3).
- Block 7: If the condition in block 4 is not satisfied, the tracked target face is estimated by the mean of the previously tracked target face samples. (see Equation (8)).
- Block 8: The proposed error predictor checks if resyncing of the tracked target face is required to correct the tracking process (see details in Section 2.4).
- Blocks 9-10: If resyncing of the tracked target face is not required, the eigenbases are updated if a sufficient new tracked target face samples τ have been accumulated (see details in Section 2.1).
- Blocks 12-13: In case the tracking error is higher than a threshold Γ_T, W-CLM is used to re-locate the tracked target face landmarks and correct the tracking process (see details in Section 2.2).
- Block 14: The yawning is detected (see details in Section 3.4).
- Block 15: The tracked target face and its affine parameters are used as seeds to keep tracking the tracked target face in the next video frame if there are more frames to process.

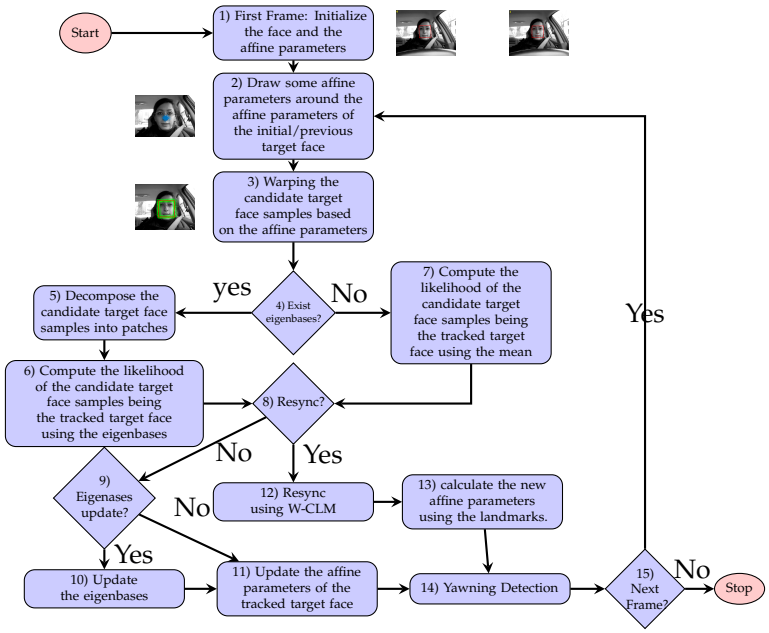


Figure 1. Block diagram of the proposed face tracking approach.

The proposed tracking algorithm is able to track non-rigid objects such as faces and detect early potential tracking deviations from the tracked target object. The incremental update of the tracking process parameters is inspired on the incremental PCA approach [7]; however, the proposed method uses local texture information (patches of size $v \times v$) rather than global information (the target object as a whole) to build the eigenbases, as explained in Section 2.1. A description of the W-CLM scheme and how it is used as a resyncing mechanism is in Section 2.2. How the proposed tracking method is applied to the face and facial landmark tracking is explained in Sections 2.3 and 2.4.

2.1. Incremental Update of the Eigenbases and the Mean

Let $A = \{\mathbf{I}(1), ..., \mathbf{I}(n)\}$ be a data matrix of $d \times n$ dimensions, where each one of the n columns $\mathbf{I}(t)$ contains a patch of the tracked target face samples represented by a column vector of d dimensions, and n is the number of samples in A (each patch is a target face observation or sample). Let $A \stackrel{SVD}{=} UCV^T$ be the Singular Value Decomposition (SVD) of A, where C is a diagonal matrix with the singular values (i.e., square root of non-zero eigenvalues), and U and V are the left and right orthonormal eigenvectors of A. Let $B = \{\mathbf{I}(1), \ldots, \mathbf{I}(m)\}$ of dimensions $d \times m$ be the new samples received over time, where m is the number of new samples received. Now the goal is to efficiently compute the SVD of the combination of the old data A and the new data B: $[A\ B] \stackrel{SVD}{=} U^{'}C^{'}V^{'T}$. Computing the SVD every time that new data is received is time-consuming (and impractical) for applications such as object tracking, and the incremental updates of the eigenbases tends to be more interesting. The concatenation of A and B can be expressed in a partitioned form in a way to utilize the previously computed SVD of A as follows [7]:

$$\begin{bmatrix} A & B \end{bmatrix} = \begin{bmatrix} U & \widetilde{B} \end{bmatrix} \begin{bmatrix} C & U^T B \\ 0 & \widetilde{B}^T B \end{bmatrix} \begin{bmatrix} V^T & 0 \\ 0 & I \end{bmatrix}, \tag{1}$$

where $\widetilde{B}$ represents the new eigenbases associated to the newly received data matrix B, which are orthogonal to the eigenbases in U and is unknown at this stage. Let $R = \begin{bmatrix} C & U^T B \\ 0 & \widetilde{B}^T B \end{bmatrix}$, then $R \stackrel{SVD}{=} \widetilde{U}\widetilde{C}\widetilde{V}^T$ can be computed in constant time regardless the initial data size in A. Now, the SVD of $[A\ B]$ can be expressed more conveniently as [7]:

$$\begin{bmatrix} A & B \end{bmatrix} \stackrel{SVD}{=} \left(\begin{bmatrix} U & \widetilde{B} \end{bmatrix} \widetilde{U} \right) \widetilde{C} \left(\widetilde{V}^T \begin{bmatrix} V^T & 0 \\ 0 & I \end{bmatrix} \right), \tag{2}$$

where I is the identity matrix. Finally, $U^{'} = [U\ \widetilde{B}]\ \widetilde{U}$ and $C^{'} = \widetilde{C}$ are the new eigenvectors (eigenbases) and singular values, respectively, which considers the new data in B. Since only $U^{'}$ and $C^{'}$ will be utilized in the proposed tracking scheme, $V^{'}$ is disregarded from now on. Furthermore, only the desired number of eigenvectors (γ) associated with non-zero singular values will be further processed, while other eigenvectors and singular values that exceed the γ ranked singular values will be disregarded. While updating the eigenbases, it is necessary to down-weight the older observations since the more recent observations are more informative about the current appearance of the tracked target face. Therefore, a forgetting factor f ($\in [0, 1]$) is multiplied by the singular values in $C^{'}$ [7], since $\mu(t)$ plays a key role in the detection of the tracked target face. Consequently, the mean $\mu(t)$ at time t is calculated incrementally as follows:

$$\mu(t) = \frac{f \cdot n \cdot \mu_n + m \cdot \mu_m}{m + f \cdot n}, \tag{3}$$

where μ_n represents the mean of the data matrix A with n face samples, and μ_m is the mean of the newly added observations B, and $t = m + n$. An important benefit of having the forgetting factor f is that the mean $\mu(t)$ at time t can change in response to new observations, even if the total number of older observations in A is large.

2.2. Weighted Constrained Local Model (W-CLM) as the Feature Detector Used for Resyncing

The Constrained Local Model method (CLM) tends to be an accurate facial feature detector, but it tends to converge slowly, making its use in tracking problems challenging. Nevertheless, if CLM is used less often in comparison with other components of the tracking process, the CLM based tracking system could be viable for real-time operation. In this work, the proposed tracking scheme is applied to face tracking, and a modified CLM method, namely, the Weighted Constrained Local Model (W-CLM) is utilized to resync important facial features and avoid tracking failure, and also for the initialization of the tracking process. Consequently, the proposed method potentially is self-driven and self-corrected in real-time.

Weights Computation: The proposed W-CLM method utilizes CLM training data to evaluate the landmarks consistency by assigning higher weights to more consistent landmarks during the CLM search process. Multivariate Mutual Information (MMI) evaluates the mutual dependence between two or more random variables [20], and is utilized here to evaluate the consistency of each facial landmark. Firstly, MMI is computed independently for the feature vector of each facial landmark within a temporal window. Each feature vector x_i represents the texture information in a window of size $\sqrt{l} \times \sqrt{l}$ around a facial landmark location in a given video frame. MMI is used to evaluate the differences of the co-occurrence probabilities of n random variables describing the local texture, and indicates how consistent is the texture information around a particular landmark in the training images, and is used as a weight $\hat{w}_i \in [0, 1]$ of a facial landmark:

$$\hat{w}_i(x_1, x_2, , .., x_N) = log \frac{p(x_1, x_2, .., x_N)}{\prod_{i=1}^{n} p(x_i)}, \tag{4}$$

where, x_i is a column vector of size l, containing texture information around a particular landmark $i = 1, 2, \ldots, Z$ in a video frame at time t. The weights $\hat{w}_i$ of the landmarks are combined in a diagonal matrix to be used in the W-CLM search process. In practice, the CLM consists of two stages (modules): (1) CLM model building; (2) CLM search [18], that are discussed next:

2.2.1. CLM Model Building

CLM uses two models: (a) a shape model that deals with shape information, and (b) a patch model that considers local patch information. Both models are combined to represent the target object (i.e., face). Images of the cropped faces and a set of facial feature points (landmarks) are used as the training data to build the CLM face model.

In order to build the CLM shape model, all the shapes are aligned with the first (initial) shape of the training set using procrustes analysis [21], which attenuates the adverse effects of shape variations in terms of scale, translation and rotation, leaving only the intrinsic variations of the face shape S_r. On these aligned faces, the PCA is performed to capture the face shape variations (eigenvectors) in the training data, and to obtain an indication of the total face variation by the eigenvalue of each eigenvector [22]. Therefore, each shape can be written as a linear combination of the eigenvectors P and the mean shape ($\overline{S}$) as $S_r = \overline{S} + PH_r$, where $H_r = P^T \hat{S}_r$ is a column vector containing the coefficients of the corresponding eigenvectors P for representing the face shape S_r in M.

In order to build a patch model for each facial landmark, a linear Support Vector Machine (SVM) [23] is trained with positive and negative samples as a patch classifier. The positive examples are the patches (feature templates) captured from the target face only, around the facial landmarks available in the training dataset. The negative examples are the randomly selected patches captured elsewhere in the training images (i.e., excluding the face). Suppose there are k training sample vectors $v^{(1)}, v^{(2)}, \ldots v^{(k)}$, and each training sample vector is a column vector of l dimensions $v^{(c)} = [v_1^{(c)}, v_2^{(c)}, \cdots, v_l^{(c)}]^T$. An input value $u^{(c)} = \{-1, 1\}$ must be assigned to each training sample $c = \{1, 2, \ldots k\}$ of the positive/negative classes which is used as a label of each training sample. The SVM classifier output is written as a linear combination of the input support vectors as $u^{(c)} = \Omega^T v^{(c)} + \Theta$,

where $\Omega = [\Omega_1, \Omega_2, \ldots, \Omega_Z]$ are the weights of each dimension of the input support vectors, and Θ is a constant acting as a bias to prevent overfitting. The goal of the SVM training is to search for the right values of weights Ω. For details on training CLM, please refer to [18].

2.2.2. Weighted CLM Search Method

Given a set of initial facial landmarks, a cropped patch around the position of each landmark is classified by the patch model, while preserving the shape constraints, using the following objective function:

$$f(S_t) = \sum_{i=1}^{Z} \hat{w}_i \imath_i(x_i, y_i) - \beta \sum_{j=1}^{o} \frac{-h_j^2}{\lambda_j}, \tag{5}$$

where, $\imath_i(x_i, y_i)$ is the patch of size $\sqrt{l} \times \sqrt{l}$ classified by the patch model in the $g \times g$ neighborhood of the location of the landmark i ($g = 8$ and $l = 100$ in our experiments) and $\hat{w}_i$ is the weight that describes the impact of the landmark i in the optimization process. The term $\sum_{i=1}^{Z} \hat{w}_i \imath_i(x_i, y_i)$ in Equation (5) is the patch model response and it is optimized using the quadratic programming and can be readily solved using the Matlab *quadprog* function. The term $\sum_{j=1}^{o} \frac{-h_j^2}{\lambda_j}$ is the shape constraint, where $h_j \in H_r$ is the corresponding eigenvector coefficient in the eigenvectors representation $H_r = P^T \hat{S}$ of the current shape S, and λ_j is the eigenvalue corresponding to the eigenvector in P and o is the number of eigenvectors in P, whereas the parameter $\beta \in [0, 1]$ establishes a compromise between the patch and shape models.

For each landmark, the patch model is used to find a response patch at each landmark location in the local region, and the response patch is used to fit a quadratic function. Then, the best landmark positions are obtained by optimizing the function in Equation (5), created by combining the quadratic functions from the patch model and shape constraints from the CLM shape model. Then, each landmark is moved to its new position, and the process is repeated to obtain the optimum landmarks locations (i.e., face shape), or until the maximum number of iterations is reached (see details in [18]). For other promising fitting strategies, please look at the generative shape regularization model for robust face alignment [24] and unified embedding for face recognition and clustering [25]. In case of a new video sequence, the mean face shape $\overline{S}$ is used for the landmarks initialization, but in the subsequent frames the previous frame face landmarks are used to initialize the landmarks.

2.3. The Proposed Tracking Method Applied to Human Faces

As mentioned before, in the current work, the proposed tracking method is applied to face tracking. For face tracking, the state at time t is described by the affine parameters vector $\chi(t)$= ($x(t)$, $y(t)$, $s(t)$, $\theta(t)$, $\alpha(t)$, $\phi(t)$), where $x(t)$ and $y(t)$ represent the translation of the tracked target face with respect to the origin of the image, $s(t) = M/u$ is the scale of the tracked target face w.r.t the size of the image $(M \times N)$ which contains the tracked target face $(u \times u)$, whereas $\theta(t)$, $\alpha(t)$ and $\phi(t)$ are the rotation angle w.r.t the horizontal axis, the aspect ratio, and the skew direction, respectively, at time t. The aspect ratio $\alpha(t)$ and the scale $s(t)$ are used to keep the tracked target face in the xy image space (see details in [7]). The dynamics of each parameter in $\chi(t)$ are independently modeled by a Gaussian distribution $\mathcal{N}(.)$ centered at $\chi(t-1)$, and going from $\chi(t-1)$ to $\chi(t)$ is given by:

$$p(\chi(t)|\chi(t-1)) = \mathcal{N}(\chi(t); \chi(t-1), \psi(t)), \tag{6}$$

where $\psi(t)$ is a diagonal matrix with each main diagonal element representing the variance of the corresponding affine parameter. Equation (6) is referred as the motion model, because it models the motion of the tracked target face from one frame to the next frame.

Figure 2 shows an example of the working of the motion model. The affine parameters $\chi(t)$ are represented by a point in affine parameter space; the affine parameter space is a six-dimensional space, and only three dimensions are shown in Figure 2. The red point in the Figure 2 represent the affine

parameters of the tracked target face in the previous frame. Numerous affine parameters are computed using the Gaussian distribution centered around the affine parameters associated with the tracked target face in the previous frame using Equation (6), and these affine parameters are shown as blue points in Figure 2. Furthermore, these affine parameters are used to warp the candidate target face samples which may contain the tracked target face in the current frame, shown in green color faces in Figure 2 to check if they correspond to the tracked target face $\mathbf{I}(t)$.

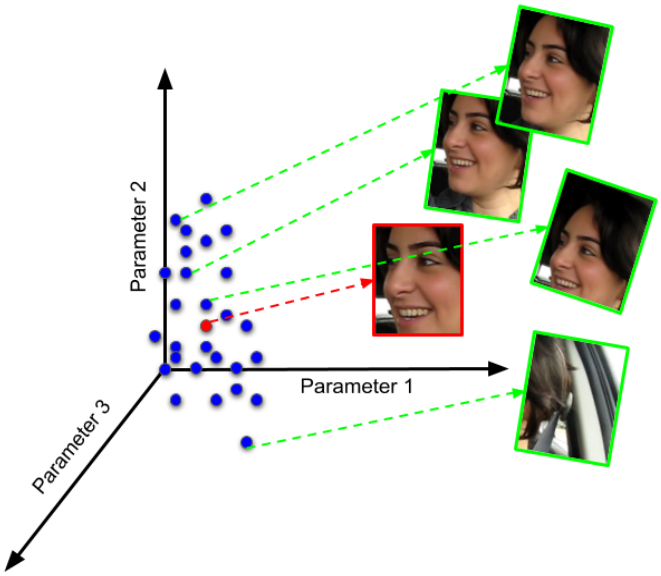


Figure 2. Motion model example ($p(\chi(t)|\chi(t-1))$) in image space; Affine parameter space, each point in affine parameter space is warped into a bounding box in image space.

In order to find the tracked target face in the current frame, every candidate target face sample is represented within the space of the tracked target face $\mathbf{I}(t)$ that is spanned by the eigenbases U and centered at the mean $\mu(t)$, where U is obtained incrementally using the method explained in Section 2.1 [7]. The likelihood $p(\mathbf{I}(t)|\chi(t))$ that the candidate target face sample is the tracked target face $\mathbf{I}(t)$ is inversely proportional to the distance δ of the candidate target face sample to a reference point in the space (i.e., mean $\mu(t)$ projected in the space spanned by U). This distance is comprised by the sample distance to the space (δt) and the within space distance (δw) of the projected sample to the reference point $\mu(t)$. The likelihood ($p_{\delta t}$) that a candidate target face sample projected in the space spanned by U corresponds to the tracked target face is approximated by the negative exponential value of δt:

$$p_{\delta t}(\mathbf{I}(t)|\chi(t)) = \mathcal{N}(\mathbf{I}(t); UU^T + \varsigma I) \approx exp(-\delta t), \tag{7}$$

where, $\delta t = ||(\mathbf{I}(t) - \mu(t)) - UU^T(\mathbf{I}(t) - \mu(t))||^2$, ςI is the noise in the observation process, and I is the identity matrix and ideally $\varsigma \to 0$. It is worth mentioning that in the initialization, the eigenbases are not available yet, because the eigenbases only are build after a specific number τ of tracked target face samples are observed, then in the initialization U=0 and the mean $\mu(t)$ are used to estimate the likelihood $p(\mathbf{I}(t)|\chi(t))$ that $\mathbf{I}(t)$ contains the tracked target face, and Equation (7) is simplified to:

$$p(\mathbf{I}(t)|\chi(t)) = exp(-||(\mathbf{I}(t) - \mu(t))||). \tag{8}$$

Similarly, the likelihood ($p_{\delta w}$) that $\mathbf{I}(t)$ contains the tracked target face is given by the negative exponential of the Mahalanobis distance δw:

$$p_{\delta w}(\mathbf{I}(t)|\chi(t)) = \mathcal{N}(\mathbf{I}(t); \mu(t), UC^{-2}U^T) \approx exp(-\delta w), \tag{9}$$

where, $\delta w = ||(\mathbf{I}(t) - \mu(t))^T UC^{-2}U^T(\mathbf{I}(t) - \mu(t))||^2$. Finally, the likelihood of a candidate target face sample $\mathbf{I}(t)$ being the tracked target face is given by the combined likelihoods $p_{\delta w}$ and $p_{\delta t}$ to ensure a more reliable decision score as follows:

$$p(\mathbf{I}(t)|\chi(t)) = p_{\delta t}(\mathbf{I}(t)|\chi(t))p_{\delta w}(\mathbf{I}(t)|\chi(t)). \tag{10}$$

The candidate target face sample with the highest likelihood to be the tracked target face in Equation (10) is selected. Furthermore, the affine parameters $\chi(t)$ associated with the tracked target face are used to estimate the tracking landmarks locations (facial landmarks) as shown below:

$$\Lambda_T(t) = \chi(t)[\Lambda(1); \vec{1}], \tag{11}$$

where, $\Lambda(1)$ are the facial landmarks locations in the initial target face and $\vec{1}$ is an unitary vector of length Z (total number of landmarks). The pseudo code of the above described procedure is given in Algorithm 1.

Algorithm 1 Incremental Learning For Face Tracking Algorithm (ILFT).

procedure ILFT($I(t)$, $\Lambda_T(t-1)$, $\chi(t-1)$, C, U, μ_o, $\mathbf{I}(t-1)$, $flag$,Y) ▷ $I(t)$ is the current frame, $\mathbf{I}(t-1)$, $\Lambda_T(t-1)$,$\chi(t-1)$, are the tracked target face sample, facial landmarks and the affine parameters of the previous frame, C and U is the singular values and eigenvectors respectively, C and U are empty matrix in the start and are computed and updated after each τ frames, $flag$ is the counter for number of frames for batch size and Y is 1, if there is at least one more frame to process, otherwise Y is 0.

    while (Y $= 1$) **do**

        $flag \leftarrow flag + 1$;

        Draw a finite number of affine parameters centered at $\chi(t-1)$ using Equation (6);

        Warp the candidate target face samples from $I(t)$ using these affine parameters;

        Compute the probability of every candidate target face being the tracked target face using Equation (10);

        Select the candidate target face sample with highest likelihood as the tracked target face sample $\mathbf{I}(t)$;

        Estimate the facial landmarks using Equation (11);

        if ($flag \geq \tau$) **then**

            $flag \leftarrow 0$;

            Calculate $C^{'}$ and $U^{'}$ (see details in Section 2.1).

            Update the mean ($\mu(t)$) using Equation (3);

        end if

        if (nextframe == Null) **then**

            Y $\leftarrow 0$

        end if

    end while

    return $\Lambda_T(t)$, $\chi(t)$,$C^{'}$, $U^{'}$,$\mu(t)$, $\mathbf{I}(t)$;

end procedure

2.4. Tracking Error Prediction and Resyncing Mechanism

Visual tracking is prone to failure if the object changes, moves quickly or changes its appearance. If the tracking methods fails, the tracking error may keep on increasing and the facial tracking process may fail. Most of the methods available do not provide a self assessment of tracking process correctness [5–7,26,27]. The proposed method is based on an error predictor that estimates the tracking error $\varepsilon(t)$ at runtime. It was found experimentally that a relevant measure to predict the tracking error

is the tracking difference of the facial landmarks locations in consecutive frames, which is represented by $\Delta(t)$ at time t, and its adequacy can be verified by observing the correlation ρ of $\Delta(t)$ with the tracking error $\varepsilon(t)$, where $\Delta(t)$ at time t is given by:

$$\Delta(t) = \frac{1}{Z}\sum_{i=1}^{Z}||\Lambda_T^{(i)}(t) - \Lambda_T^{(i)}(t-1)||^2, \tag{12}$$

where, $\Lambda_T^{(i)}(t)$ is the location (x_i, y_i) of the facial landmark i at time t estimated by the proposed method. To further improve the tracking error prediction, a median filter can be applied to the $\Delta(t)$ noisy estimates (see details in Section 3).

The next stage of the face tracking process is to predict the potential tracking failures, and if a resyncing is required. This is done by checking if the value of $\Delta(t)$ in Equation (12) is higher than a threshold Γ_T. A constant threshold value is not suitable for real applications because $\Delta(t)$ may vary from one person to another due to different face sizes, closeness to the camera, and/or the number of facial landmarks used. For this reason, the median value ($\Gamma_T = Median(\Delta(T))$) is used as a dynamic threshold instead:

$$\Psi(t) = \begin{cases} 1, & \text{if } \Delta(t) \geq \Gamma_T, \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

where $\Delta(T) = \{\Delta(1), ..., \Delta(t)\}$, and $\Psi(t)$ is used to indicate if resyncing is required. Moreover, the proposed error predictor is highly correlated with the actual tracking error (see Section 3). When the tracking predictor indicates a substantial error, i.e., $\Psi(t) = 1$, the W-CLM features are used for correcting (resyncing) the tracking process by re-adjusting the tracked landmarks $\Lambda(t)$.

Algorithm 2 provides the pseudo code of the proposed method applied to human faces. In the first frame, the face and the facial features are initialized using the W-CLM search method. In the other frames, Algorithm 1 is used to track the face and the facial features until the estimated tracking error increases. When the tracking predictor indicates a substantial error, W-CLM is used to resync the tracking process, which re-locates the facial landmarks $\Lambda_T(t)$ to the correct locations. This error prediction and correction scheme helps the proposed face tracker to adapt to the facial shape and appearance changes of the target along time and the target is not missed indefinitely. Furthermore, the detected facial landmarks $\Lambda_T(t)$ are then used for further processing such as computing the new affine parameters $\chi(t)$, and locating the tracked target face based on the candidate samples $\mathbf{I}(t)$ in the current frame. Moreover, new eigenbases are created starting from the resynced frame, and the old data is discarded because it is not relevant anymore.

Algorithm 2 Adaptive Face Tracker with Resyncing Mechanism using W-CLM (AFTRM-W).

procedure AFTRM($I(t)$, $\Lambda_T(t-1)$, $\mathbf{I}(t-1)$,Y) ▷ $I(t)$ is the current frame, $\mathbf{I}(t-1)$ and $\Lambda_T(t-1)$ are the tracked target face sample and facial landmarks of the previous frame, respectively and Y indicates if there is atleast one more frame to process, and $Y = 1$ in the first frame.
  while $Y = 1$ **do**
    Track $\mathbf{I}(t)$ and estimate $\Lambda_T(t)$ using Algorithm 1;
    Compute tracking points difference ($\Delta(t)$) using Equation (12);
    Calculate $\Psi(t)$ using Equation (13); ▷ $\Psi(t)$ results a binary value and checks if resyncing is required.
    if $\Psi(t) = 1$ **then**
      Update $\mathbf{I}(t)$ and $\Lambda_T(t)$ using W-CLM; ▷ see Section 2.2.
      Re-Initialize the eigenbases U and the mean $\mu(t)$;
      Calculate the new affine parameters $\chi(t)$;
    end if
  end while
  return $\Lambda_T(t)$, $\chi(t)$, $\mathbf{I}(t)$;
end procedure

3. Experimental Results and Discussion

The YawDD [28] and Talking Face Video [29] datasets are used in the experimental evaluation. YawDD dataset contains videos of drivers performing various tasks such as talking/singing and yawning. The camera was installed on the dash or under the car front mirror. The videos were taken under various illumination conditions. YawDD dataset contains the total of 119 participants from different age groups with the minimum age of sixteen years are involved. The videos from 29 participants are recorded using camera installed on the dash, and for other 90 participants, the camera is installed under the front mirror. On the other hand, the Talking Face video consist of 5000 frames obtained from a video of a person engaged in conversation with various face movements [29].

Table 1 shows the important parameters used in the proposed tracking method, their range and optimal values are presented, which are chosen empirically. For building the eigenbases U, the candidate/tracked target face sample is resized to $u \times u$ ($u = 32$) for computational efficiency, the number of eigenvectors γ = 16, the patch size is set to $v \times v$ ($v = 8$) and the eigenbases are updated every five frames ($\tau = 5$), with a forgetting factor f = 0.95.

Table 1. Parameters used and their ranges.

Param	min	max	Optimal
Batch Size τ	1	16	5
Face size u	16	64	32
Patch size v	4	32	8
Forgetting factor f	0.5	1.0	0.95

The proposed face tracking algorithm is quantitatively evaluated using Center Location Error (CLE), that measures the distance between center locations of the tracked target face with the manually labeled center location of the target face that is used as the groundtruth. Furthermore, for detailed evaluation on YawDD dataset, six videos have been annotated manually, which includes the target face and landmarks ($Z = 68$) on the face, nose and the eyes. These videos contain different background and varied illumination. Additionally, person-specific characteristics, such as face changes, head motion, and glasses are also included. The proposed face tracking method is tested to verify if they can track the facial landmarks consistently on these videos. Hence, the error was measured by the root

mean squared error (RMSE) between the estimated landmark locations (Λ_T) and the manually-labeled groundtruth (Λ_G) locations of the landmarks as follows:

$$\varepsilon(t) = \frac{1}{Z}\sum_{i=1}^{Z}||\Lambda_G^{(i)}(t) - \Lambda_T^{(i)}(t)||_2, \tag{14}$$

where $\varepsilon(t)$ represents the tracking error in the video frame at time t, whereas $\Lambda_G^{(i)}$ represents the ground truth location (x_i, y_i) of the landmark i.

3.1. Choice of Batch Size

In the object tracking methods that learn the appearance of the tracked target object incrementally, the batch size plays an important role. Batch size describes that after how many frames the appearance model is updated. Different batch sizes have been tested to optimize the performance of the proposed tracking method. The phenomenon of batch size τ with the average RMSE tracking error ε_M and number of resyncs r is shown in Figure 3 for different batch sizes ($1 \leq \tau \leq 16$). The size of the triangle indicates the batch size, which means after how many frames the resync of the features is performed (larger the size of the triangle, bigger batch size). A larger batch size (big triangles in Figure 3) requires a lower number of resyncs, but it confers higher errors and vice versa. Contrarily, small triangles tend to lie on the upper left (upper for a large number of resyncs and left confers to smaller error) of the plot, which shows that more resyncs are required, and the error is low. However, frequent resyncs and updates may slow down the number of frames processed per second, as shown in Table 2.

Figure 3 indicates that frequent updates (small batch size) on the basis of the proposed method has a lower tracking error than for large batch size. The reason for this behavior is that it updates the most recent appearance of the face and also the resync (if required) of the features are performed after a specific number of frames. The optimal trade-off is the batch size that minimizes both the number of resyncs (r) and the tracking error ε, which is defined as:

$$\arg\min \forall_{\tau=1}^{n}\, c(r_\tau, \varepsilon_\tau), s.t.\;\; c(r_\tau, \varepsilon_\tau) = (1-\kappa)r_\tau + \kappa\varepsilon_\tau, \tag{15}$$

where c indicates a cost function, n is the total number of batch sizes (n = 16 in the current experiments), and κ is a bias between the tracking error ε and number of resyncs r (κ = 0.5 in the current experiments). Figure 4 shows an example graph of the batch size and the cost function. The objective is to minimize the cost function to achieve an optimal batch size (τ) and in this example the error function attains minimum value (green circle) when τ is 6.

Table 2. Average frames per second (fps) and number of times resync is activated for different batch sizes τ in AFTRM and AFTRM-W total of 2000 frames.

Method	**Batch Size (τ)**									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
AFTRM	0.09	0.21	0.25	0.31	0.29	0.38	0.53	0.62	0.79	0.98
AFTRM-W	0.07	0.23	0.20	0.23	0.33	0.45	0.56	0.54	0.83	1.10
N^o of resync	1002	474	279	231	194	171	161	146	*117*	**108**

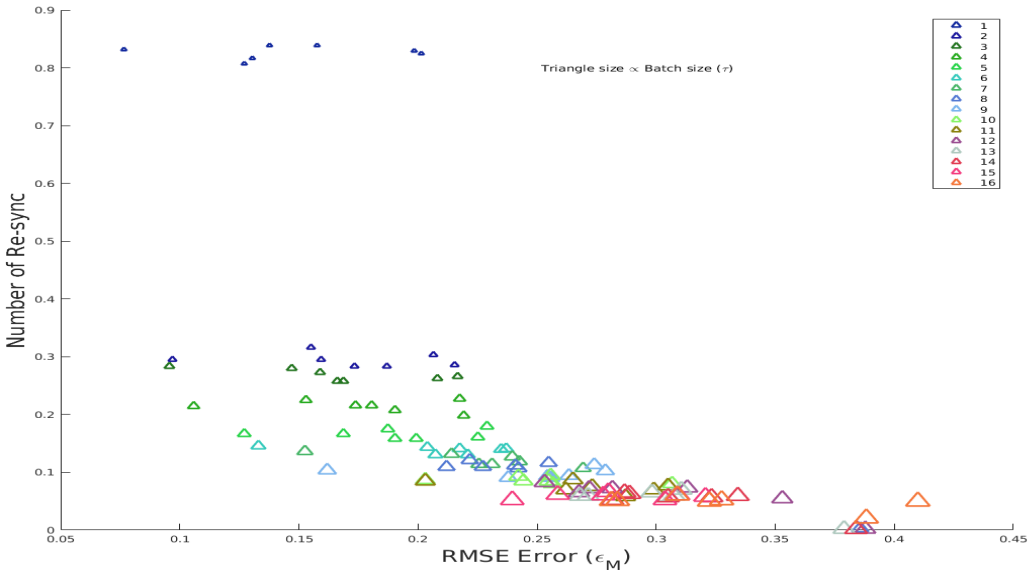


Figure 3. Batch size effect on error (ε) and number of resyncs (r) in multiple videos (normalized to [0,1]).

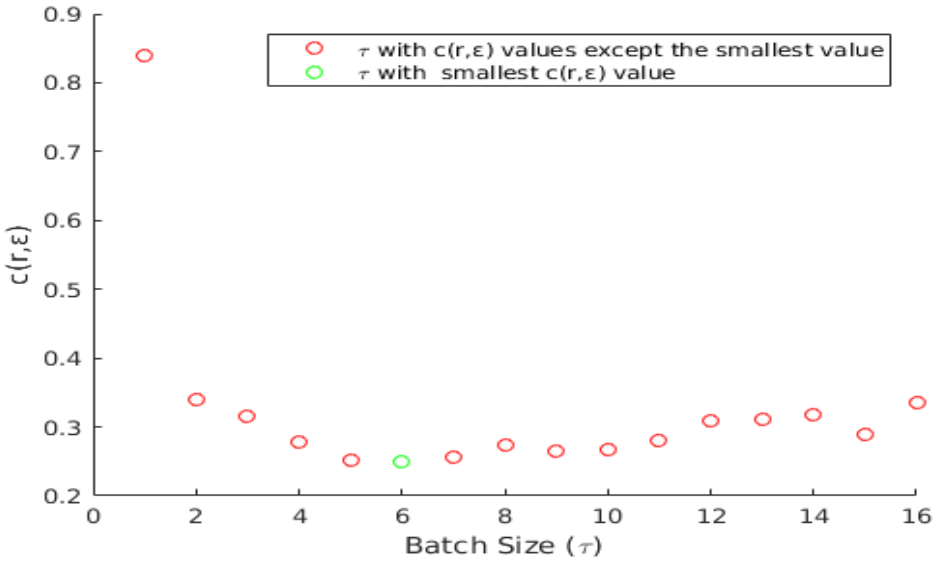


Figure 4. An example plot of cost function $c(r_\tau, \varepsilon_\tau)$ and batch size τ (τ = [1 16]).

3.2. Discussion on Error Prediction and Resyncing

Figure 5 shows some examples of the tracking errors of the proposed ILFT method without the error prediction and the resyncing procedure, illustrating some video frames with the tracked target face enclosed in a bounding box and the tracked facial landmarks plotted in red, whereas the yellow facial landmarks show the ground-truth landmarks. Figure 5a shows the effect of a tilted face on the tracking process, and Figure 5b shows that bad lighting also affects the tracking process, which tends to decrease its performance when the lighting conditions are changed during tracking. When the face deformation is not detected correctly, it is difficult to do facial expression analysis as shown in Figure 5c. The illumination changes may cause the tracked target face to be confused with the background, resulting in the tracking process permanent failure as can be seen in Figure 5d. Often, the tracking process fails in complex scenarios, since the eigenbases are then built using slightly incorrect tracked target face samples.

(**a**) (**b**) (**c**) (**d**)

Figure 5. Failed results in challenging conditions using the ILFT method: (**a**) tracking failure due to face tilting; (**b**) bad lighting leading to error; (**c**) face expression leading to error (**d**) tracking failure due to similarity with the background.

Nevertheless, this tracking failure can be avoided if the tracker has an estimate of the tracking error. The proposed method addresses this problem using an error predictor and a resyncing scheme. Figure 6 shows the plots of the proposed error predictor $\Delta(t)$ computed using Equation (12) and the actual tracking error $\varepsilon(t)$ of the tracked facial landmarks. The plots in Figure 6 suggest some correlation ρ between $\Delta(t)$ and actual tracking error $\varepsilon(t)$, but the data is noisy and the correlation is low. Due to the noisy nature of $\Delta(t)$ and the actual tracking error $\varepsilon(t)$, a one dimensional median filter of fifth order is applied on a sliding window of τ frames to smooth consistently $\Delta(t)$ (i.e., $\overline{\Delta}(t)$=$\{\Delta(t)-\tau$, $\Delta(t)-\tau+1,\ldots,\Delta(t)\}$), increasing the correlation between $\bar{\varepsilon}(t)$ and $\overline{\Delta}(t)$, as shown in Figure 7. It can be seen that the filtered $\overline{\Delta}(t)$ and $\bar{\varepsilon}(t)$ have higher correlation because the data is smoothed and has fewer spikes. To further improve the tracking error prediction, a median filter of fifth order is applied over a sliding window of τ previous values of $\overline{\Delta}(t)$ (i.e., $\hat{\Delta}(t)$=$\{\overline{\Delta}(t)-\tau$, $\overline{\Delta}(t)-\tau+1,\ldots,$ $\overline{\Delta}(t)\}$), and the correlation between $\hat{\Delta}(t)$ and $\hat{\varepsilon}(t)$ is improved, as can be seen in Figure 8. Using the proposed error predictor, the tracking quality can be evaluated and the re-estimation of the tracking landmarks locations uses W-CLM when $\Psi(t)$=1 (see Equation (13)). Some results obtained using this error prediction and resyncing based face tracking scheme are shown in Figure 9. The proposed tracking process tends to adapt to the changes in the tracked target face and work correctly in long video sequences, even if there is a tilt in the face (see Figure 9a), bad lighting (see Figure 9b), changes in face expression (see Figure 9c), or if the tracked face is similar to the background and under varied face expressions (see Figure 9d).

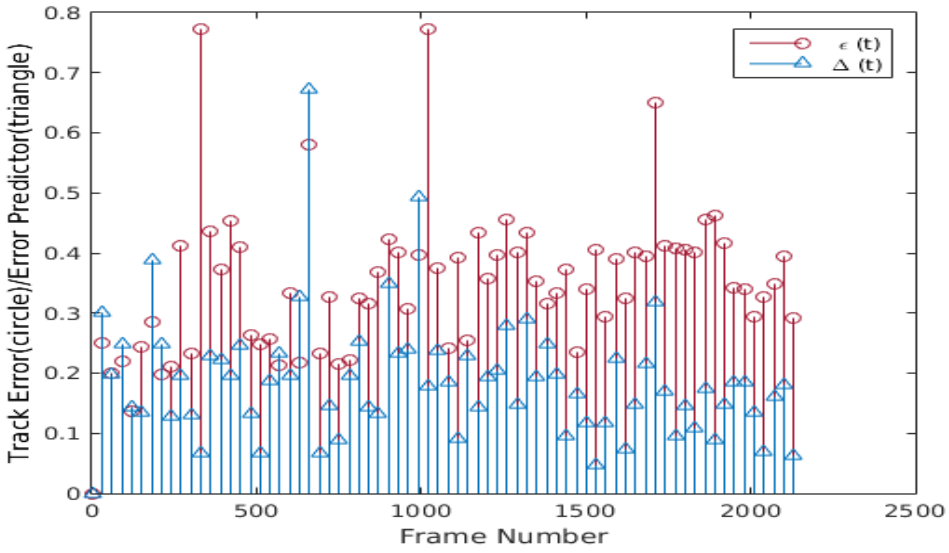


Figure 6. Plot of $\Delta(t)$ and $\varepsilon(t)$ (ρ = 0.21658).

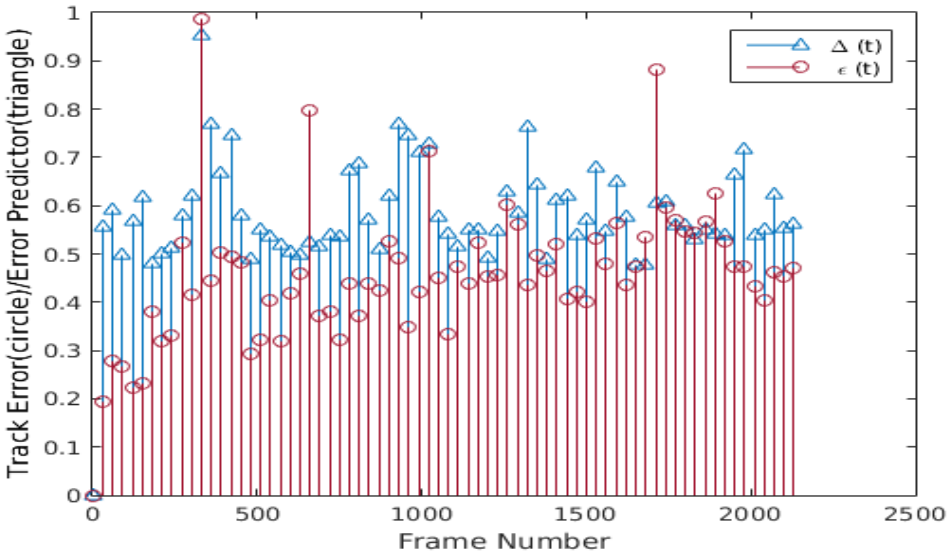


Figure 7. Plot of $\overline{\Delta}(t)$ and $\overline{\varepsilon}(t)$, the median of $\Delta(t)$ and $\varepsilon(t)$ (ρ = 0.68328).

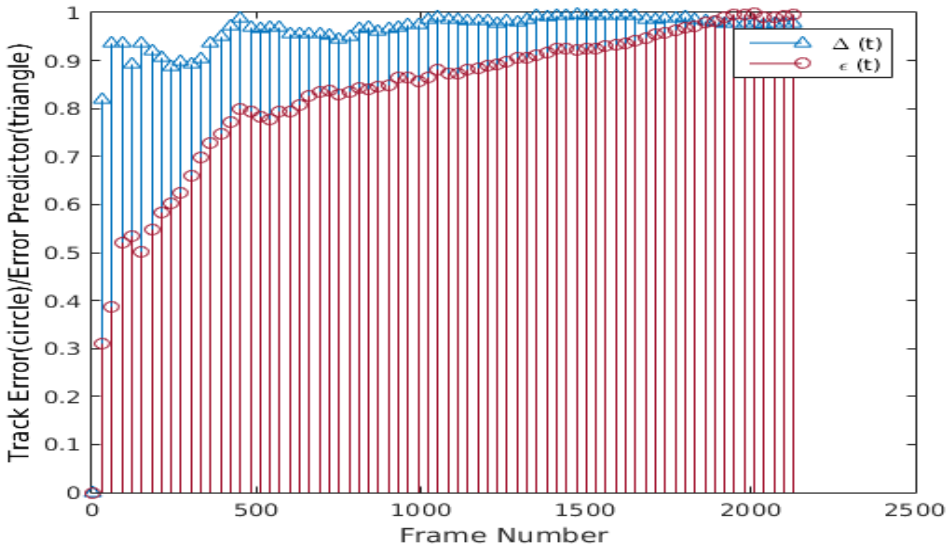
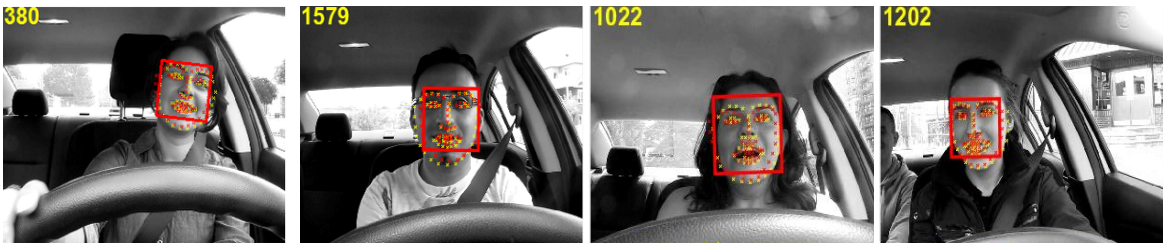


Figure 8. Plot of $\hat{\Delta}(t)$ and $\hat{\varepsilon}(t)$, the median of $\overline{\Delta}(t)$ and $\overline{\varepsilon}(t)$ (ρ = 0.91037).



(**a**) (**b**) (**c**) (**d**)

Figure 9. Illustration of the results obtained by the Adaptive Face Tracker with Resyncing Mechanism using the W-CLM (AFTRM-W) method: (**a**) tilted face; (**b**) bad lighting; (**c**) face expression change; and (**d**) similar background.

3.3. Quantitative Evaluation of the Proposed Face Tracking Method

Next is presented a quantitative comparison of the proposed AFTRM and AFTRM-W with the following methods: Incremental Learning Tracking Based on Independent Component Analysis (ILICA) [5], Incremental Learning for Robust Visual Tracking (IVT) [7], Incremental Cascaded Continuous Regression (iCCR) [15], Approximate Structured Output Learning for CLM [30], MCVFT [11], DCFNet [13], and MMDL-FT and MMDL-FTU [31].

Table 3 shows the RMSE in tracking of the facial landmarks of the proposed AFTRM, AFTRM-W, and of the comparative methods. Each column indicates the average RMSE tracking error ε_m for the whole video sequence using the method specified in the first column. The last column illustrates the average tracking error obtained for all the tested videos. For the comparative methods, the parameters (if required) are set to the default values as proposed by their respective authors. Furthermore, the initialization for Terissi et al. [5], Ross et al. [7], Wang et al. [13], Li et al. [11], MMDL-FT, MMDL-FTU [31], AFTRM and AFTRM-W is done by using the W-CLM search method. Furthermore, Table 4 compares the CLE of the proposed AFTRM-W with the state-of-the-art methods based on all the videos of YawDD dataset with the camera installed on dash. Tables 3 and 4 show that the proposed AFTRM and AFTRM-W tend to outperform the other methods, whereas AFTRM-W has an improved performance in comparison with AFTRM. This is due to the weighting scheme, as consistent landmarks receive higher weights, improving the quality of the resyncing mechanism. The methods proposed by Zheng et al. [30], Sanchez et al. [15], Wang et al. [13] and our previous MMDL-FTU method [31] perform similarly to the proposed AFTRM method, whereas AFTRM-W has performed better than all the other tested methods and has a smaller tracking error. In our view, the higher tracking error presented by the comparative methods occur because once the tracking error is introduced, it keeps on increasing and eventually the tracking process fails. On the other hand, we solve this problem by estimating the tracking error during tracking and resyncing the facial landmarks if the tracking error tends to increase. For this reason, the proposed method can adapt to the challenging conditions and avoid to miss the tracked target indefinitely. Consequently, the proposed method can be used for consistent face tracking and facial features tracking in long video sequences, which can be used to detect different facial expressions, such as yawning, talking, fatigue, and so on.

Table 3. Comparison of facial feature tracking methods in terms of RMSE on YawDD dataset (the best results are in bold).

Video	1	2	3	4	5	6	Average
Terissi et al. [5]	38.43	26.93	50.38	66.44	66.12	16.75	34.24
Ross et al. [7]	21.43	10.56	183.7	30.12	6.23	12.17	44.04
Zheng et al. [30]	33.93	11.46	12.41	17.05	12.26	14.02	16.86
Sanchez et al. [15]	16.42	11.48	10.33	22.07	14.49	9.84	14.10
Li et al. [11]	23.14	11.20	15.58	21.46	14.97	9.61	14.74
Wang et al. [13]	20.32	11.35	**7.51**	10.84	15.33	18.07	13.57
MMDL-FT [31]	10.12	7.19	7.63	22.02	8.06	30.75	14.29
MMDL-FTU [31]	*9.73*	*6.50*	*7.76*	16.62	7.76	19.37	11.29
AFTRM	15.01	9.22	13.78	15.31	5.91	7.53	11.12
AFTRM-W	**6.54**	**3.56**	10.65	**5.27**	**4.85**	**3.62**	**5.65**

To compliment the experiments, the proposed method is tested on the Talking Face video [29]. Table 5 compares the proposed method with the comparative methods using CLE and RMSE measures. The experimental results show similar trend. The proposed and the comparative methods perform well on the talking face video [29]. Ross et al. [7] has performed much better on the talking face video because of its effectiveness in static background conditions. Furthermore, AFTRM-W performs better than all the comparative methods. This proves the efficiency of the proposed method and its effectiveness in the face tracking.

Face and facial landmarks can be used as a cue to many facial analysis applications, such as yawning detection, talking, facial expression detection and so on. In this paper, we evaluate the effectiveness of the proposed face and facial landmarks tracking in the context of yawning detection, which is explained next.

Table 4. Center Location Error (CLE) comparison of AFTRM and AFTRM-W with comparative methods on YawDD dataset (the best results are in bold).

Video	Male Videos	Female Videos	Average
Terissi et al. [5]	25.92	18.37	22.15
Ross et al. [7]	14.74	11.33	13.03
Zheng et al. [30]	13.02	10.14	11.58
Sanchez et al. [15]	14.11	10.17	12.14
Li et al. [11]	17.23	14.93	16.08
Wang et al. [13]	15.52	13.58	14.55
MMDL-FT [31]	10.61	8.70	9.65
MMDL-FTU [31]	10.36	8.68	9.52
AFTRM	*8.81*	*7.54*	*8.18*
AFTRM-W	**5.31**	**4.24**	**4.78**

Table 5. Center Location Error (CLE) and RMSE comparison of AFTRM and AFTRM-W with comparative methods on Talking Face Video datasets (the best results are in bold).

Dataset Name	Talking Face Video	
Method	**CLE**	**RMSE**
Terissi et al. [5]	27.79	26.43
Ross et al. [7]	12.05	10.09
Zheng et al. [30]	11.31	11.15
Sanchez et al. [15]	10.42	10.51
Li et al. [11]	9.27	11.39
Wang et al. [13]	11.63	13.46
MMDL-FT [31]	16.45	15.91
MMDL-FTU [31]	13.93	13.48
AFTRM	*8.92*	*8.98*
AFTRM-W	**6.81**	**6.62**

3.4. Evaluation of the Proposed Face Tracking Method in Yawning Detection

The accurate detection of facial landmarks is requirement fro many facial analysis applications such as human emotion analysis, fatigue detection and so on. To prove the effectiveness of the facial landmark features detected using the proposed tracker in a facial analysis application, i.e., a yawning detection scheme. An accurate yawning detection is requirement for many facial analysis applications. One of the most common usage of yawning is in driver fatigue detection, which is one important factor among others to detect fatigue in drivers [3]. Yawning detection is used as a case study to evaluate proposed tracking method in a practical facial tracking problem, where the local face appearance is changing. The proposed method takes an inspiration from the Omidyeganeh et al. [3] yawning detection approach, which is based on the backprojection theory and detects yawning based on the pixels counts in the binary mouth blocks of the current and the reference frames. To convert into a binary image, the pixel values greater than a threshold Γ_0 receive a value of 1 (named 'white pixels'), and 0 (named 'black pixels') otherwise. The proposed method improves on the method proposed by Omidyeganeh et al. [3] in two ways. Firstly, the proposed method uses only the pixels which are in the lips to measure the mouth openness in a binary image (see in Figure 10 that only the pixels inside the white region are used), as compared to [3] which uses a rectangular mouth block and includes some pixels outside the lips to detect yawning. Secondly, yawning is detected in each video frame if the

following three conditions are satisfied: (1) the ratio of the number of black pixels in the current frame (NBC) and the reference frame (NBR) is greater than Γ_1 (i.e., $\frac{NBC}{NBR} > \Gamma_1$); (2) the ratio of the number of black and white pixels (NWC) in the current frame is greater than Γ_2 (i.e., $\frac{NBC}{NWC} > \Gamma_2$); and (3) the ratio of a vertical distance between the midpoints (VD) and the distance between the corner points (HD) of the mouth is greater than Γ_3 (i.e., $\frac{VD}{HD} > \Gamma_3$). The first frame is used as a reference in the proposed scheme and is assumed to contain a closed mouth. Using the pixels of the reference frame tends to minimize scale issues when using conditions 2 and 3, that use only the pixels within the mouth region of the current frame. The proposed yawning detection scheme is evaluated in terms of; (1) True Positive Rate (TPR), which is the rate of True Positives (TP) detected as yawning, i.e., $TPR = \frac{TP}{TP+FN}$; True Negative Rate (TNR), which is the rate of True Negatives (TN) correctly detected as non-yawning, i.e., $TNR = \frac{FP}{FP+TN}$; False Positive Rate (FPR) is the rate of yawning falsely detected as non-yawning, False Negative Rate (FNR) is the rate of non-yawning falsely detected as yawning, and the Correct Detection Rate (CDR) is defined as $CDR = \frac{TPR+TNR}{TPR+TNR+FPR+FNR}$. Table 6 shows a comparison of the proposed method using data provided by AFTRM and AFTRM-W, with state of the art methods in yawning detection, including Chiang et al. [32], Bouvier et al. [33] and Omidyeganeh et al. [3]. The proposed method tends to outperform the comparative methods on the YawDD dataset [28]. Furthermore, the proposed method has a higher TPR, which indicates the effectiveness of the proposed method. The threshold values for Γ_1, Γ_2 and Γ_3 are set to 1, 0.5 and 2.5, respectively.

(**a**) (**b**) (**c**) (**d**) (**e**) (**f**) (**g**) (**h**)

Figure 10. Illustration of a closed mouth and yawning sequence.

Table 6. Yawning detection results on the YawDD dataset [28].

Method	TPR	TNR	FPR	FNR	CDR
Chiang et al. [32]	0.3990	0.4562	0.6010	0.5438	0.4276
Bouvier et al. [33]	0.6764	0.5437	0.3236	0.4563	0.6101
Omidyeganeh et al. [3]	0.6578	0.7733	0.3419	0.2266	0.7155
AFTRM (Proposed)	0.8120	0.7222	0.1879	0.2777	0.76703
AFTRM-W (Proposed)	0.9307	0.7551	0.0693	0.2449	0.8429

4. Conclusions

A new adaptive face tracking scheme has been proposed, which tends to reduce the face tracking errors by using a tracking divergence estimate and a resyncing mechanism. This resyncing mechanism locates adaptively the tracked facial features (e.g., facial landmarks), which tends to reduce the tracking errors and to avoid missing the tracked face indefinitely. The proposed Weighted Constrained Local Model (W-CLM) method improves the CLM feature search mechanism by assigning higher weights to more robust facial landmarks, and is used in resyncing.

The performance of the proposed face tracking method was evaluated in the drivers video sequences of the YawDD and on Talking video datasets. Both the datasets contain significant changes in illumination and head positioning. Our experiments suggest that the proposed face tracking scheme potentially can perform better than comparable state-of-the-art methods, and can be applied in yawning detection while obtaining higher Correct Detection Rates (CDRs) and True Positive Rates (TPRs) than comparable methods available in the literature. In the future, we intend to extend our work to develop a tracker for a more general class of non-rigid objects.

Author Contributions: This research article is the result of the contribution of A.K. and J.S.. A.K. has contributed in preperation of the research methodolgy, fromal analysis, writing the original draft, software and review. J.S. has the role of supervisor and helped in preparing the methodolgy of the research, formal analysis; writing the original draft, reviweing, and project administration. All authors have read and agreed to the published version of the manuscript.

Funding: CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brazil) and Sidia Instituto de Ciencia e Tecnologia provided the financial support for this project.

Acknowledgments: The authors would like to thank CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brazil) and Sidia Instituto de Ciencia e Tecnologia for support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, H.; Li, S.; Fang, L. Robust Object Tracking Based on Principal Component Analysis and Local Sparse Representation. *IEEE Trans. Instrum. Meas.* **2015**, *64*, 2863–2875. [CrossRef]
2. Chrysos, G.G.; Antonakos, E.; Snape, P.; Asthana, A.; Zafeiriou, S. A comprehensive performance evaluation of deformable face tracking "In-the-Wild". *Int. J. Comput. Vis.* **2018**, *126*, 198–232. [CrossRef] [PubMed]
3. Omidyeganeh, M.; Shirmohammadi, S.; Abtahi, S.; Khurshid, A.; Farhan, M.; Scharcanski, J.; Hariri, B.; Laroche, D.; Martel, L. Yawning Detection Using Embedded Smart Cameras. *IEEE Trans. Instrum. Meas.* **2016**, *65*, 570–582. [CrossRef]
4. Liu, S.; Wang, Y.; Wu, X.; Li, J.; Lei, T. Discriminative dictionary learning algorithm based on sample diversity and locality of atoms for face recognition. *J Vis. Commun. Image Represent.* **2020**, 102763. doi:10.1016/j.jvcir.2020.102763. [CrossRef]
5. Terissi, L.D.; Gomez, J.C. Facial motion tracking and animation: An ICA-based approach. In Proceedings of the Signal Processing Conference, Poznan, Poland, 3–7 September 2007.
6. Babenko, B.; Yang, M.H.; Belongie, S. Visual tracking with online multiple instance learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, USA, 22–24 June 2009; pp. 983–990.
7. Ross, D.A.; Lim, J.; Lin, R.S.; Yang, M.H. Incremental Learning for Robust Visual Tracking. *Int. J. Comput. Vis.* **2008**, *77*, 125–141. [CrossRef]
8. Ou, W.; Yuan, D.; Li, D.; Liu, B.; Xia, D.; Zeng, W. Patch-based visual tracking with online representative sample selection. *J. Electron. Imaging* **2017**, *26*, 1–12. [CrossRef]
9. Cootes, T.; Edwards, G.J.; Taylor, C. Active appearance models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2001**, *23*, 681–685. [CrossRef]
10. Kim, M.; Kumar, S.; Pavlovic, V.; Rowley, H. Face tracking and recognition with visual constraints in real-world videos. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 24–26 June 2008; pp. 1–8.
11. Li, X.; Liu, Q.; He, Z.; Wang, H.; Zhang, C.; Chen, W.S. A multi-view model for visual tracking via correlation filters. *Knowl.-Based Syst.* **2016**, *113*, 88–99. [CrossRef]
12. Danelljan, M.; Häger, G.; Khan, F.S.; Felsberg, M. Learning Spatially Regularized Correlation Filters for Visual Tracking. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Las Condes, Chile, 11–18 December 2015; pp. 4310–4318.
13. Qiang Wang, Jin Gao, J.X.M.Z.W.H. DCFNet: Discriminant Correlation Filters Network for Visual Tracking. *arXiv* **2017**, arXiv:1704.04057.
14. Kart, U.; Lukežič, A.; Kristan, M.; Kämäräinen, J.K.; Matas, J. Object tracking by reconstruction with view-specific discriminative correlation filters. In Proceedings of the Object Tracking by Reconstruction with View-Specific Discriminative Correlation Filters, Long Beach, CA, USA, 16–20 June 2019.
15. Sanchez L, E.; Martinez, B.; Tzimiropoulos, G.; Valstar, M. *Cascaded Continuous Regression for Real-Time Incremental Face Tracking*; Springer: Amsterdam, The Netherlands, 11–14 October 2016; pp. 645–661.
16. Cootes, T.F.; Taylor, C.J.; Cooper, D.H.; Graham, J. Active shape models-their training and application. *Comput. Vision Image Underst.* **1995**, *61*, 38–59. [CrossRef]
17. Cristinacce, D.; Cootes, T.F. Feature Detection and Tracking with Constrained Local Models. In Proceedings of the British Machine Vision Conference (BMVC), Edinburgh, Scotland, 4–7 September 2006; p. 3.
18. Lucey, S.; Wang, Y.; Cox, M.; Sridharan, S.; Cohn, J.F. Efficient constrained local model fitting for non-rigid face alignment. *Image Vis. Comput.* **2009**, *27*, 1804–1813. [CrossRef] [PubMed]
19. Shirmohammadi, S.; Ferrero, A. Camera as the instrument: the rising trend of vision based measurement. *IEEE Instrum. Meas. Mag.* **2014**, *17*, 41–47. [CrossRef]

20. Van de Cruys, T. Two multivariate generalizations of pointwise mutual information. In Proceedings of the Workshop on Distributional Semantics and Compositionality. Association for Computational Linguistics, Portland, Oregon, USA, 11–13 June 2011; pp. 16–20.
21. Dryden, I.L. Shape analysis. *Wiley StatsRef Stat. Ref. Online* **2014**. [CrossRef]
22. Tipping, M.E.; Bishop, C.M. Probabilistic principal component analysis. *J. R. Stat. Soc. Ser. B.* **1999**, *61*, 611–622. [CrossRef]
23. Cortes, C.; Vapnik, V. Support vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
24. Gu, L.; Kanade, T. *A Generative Shape Regularization Model for Robust Face Alignment*; Springer: Berlin, Germany, 2008; pp. 413–426.
25. Schroff, F.; Kalenichenko, D.; Philbin, J. FaceNet: A unified embedding for face recognition and clustering. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, CA, USA, 7–12 June 2015; pp. 815–823.
26. Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, HI, USA, 8–14 December 2001.
27. Yuan, Y.; Emmanuel, S.; Lin, W.; Fang, Y. Visual object tracking based on appearance model selection. In Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW), San Jose, CA, USA, 15–19 July 2013; pp. 1–4.
28. Abtahi, S.; Omidyeganeh, M.; Shirmohammadi, S.; Hariri, B. YawDD: A Yawning Detection Dataset. In Proceedings of the 5th ACM Multimedia Systems Conference, Singapore, 19–21 March 2014. doi:10.1145/2557642.2563678. [CrossRef]
29. Talking Face Video. Available online: http://www-prima.inrialpes.fr/FGnet/data/01-TalkingFace/talking_face.html (accessed on 20 August 2019).
30. Zheng, S.; Sturgess, P.; Torr, P.H. Approximate structured output learning for constrained local models with application to real-time facial feature detection and tracking on low-power devices. In Proceedings of the 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), Shanghai, China, 22–26 April 2013; pp. 1–8.
31. Khurshid, A.; Scharcanski, J. Incremental multi-model dictionary learning for face tracking. In Proceedings of the 2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Houston, TX, USA, 14–17 May 2018, pp. 1–6. doi:10.1109/I2MTC.2018.8409614. [CrossRef]
32. Chiang, C.; Tai, W.; Yang, M.; Huang, Y.; Huang, C. A novel method for detecting lips, eyes and faces in real time. *Real-Time Imaging* **2003**, *9*, 277–287. [CrossRef]
33. Bouvier, C.; Benoit, A.; Caplier, A.; Coulon, P.Y. Open or closed mouth state detection: Static supervised classification based on log-polar signature. In Proceedings of the International Conference on Advanced Concepts for Intelligent Vision Systems, Juan-les-Pins, France, 20–24 October 2008; pp. 1093–1102.