Rigid and Non-rigid Face Motion Tracking by Aligning Texture Maps and Stereo-Based 3D Models*

Fadi Dornaika and Angel D. Sappa

Computer Vision Center
Campus UAB
08193 Bellaterra, Barcelona, Spain
{dornaika, sappa}@cvc.uab.es

Abstract. Accurate rigid and non-rigid tracking of faces is a challenging task in computer vision. Recently, appearance-based 3D face tracking methods have been proposed. These methods can successfully tackle the image variability and drift problems. However, they may fail to provide accurate out-of-plane face motions since they are not very sensitive to out-of-plane motion variations. In this paper, we present a framework for fast and accurate 3D face and facial action tracking. Our proposed framework retains the strengths of both appearance and 3D data-based trackers. We combine an adaptive appearance model with an online stereo-based 3D model. We provide experiments and performance evaluation which show the feasibility and usefulness of the proposed approach.

1 Introduction

The ability to detect and track human heads and faces in video sequences is useful in a great number of applications, such as human-computer interaction and gesture recognition. There are several commercial products capable of accurate and reliable 3D head position and orientation estimation (e.g., the acoustic tracker system Mouse [www.vrdepot.com/vrteclg.htm]). These are either based on magnetic sensors or on special markers placed on the face; both practices are encumbering, causing discomfort and limiting natural motion. Vision-based 3D head tracking provides an attractive alternative since vision sensors are not invasive and hence natural motions can be achieved [1]. However, detecting and tracking faces in video sequences is a challenging task due to the image variability caused by pose, expression, and illumination changes.

Recently, deterministic and statistical appearance-based 3D head tracking methods have been proposed and used by some researchers [2, 3, 4]. These methods can successfully tackle the image variability and drift problems by using deterministic or statistical models for the global appearance of a special object class: the face. However, appearance-based methods dedicated to full 3D head tracking may suffer from some inaccuracies since these methods are not very sensitive to out-of-plane motion variations. On the other hand, the use of dense 3D facial data provided by a stereo rig or a

* This work was supported by the MEC project TIN2005-09026 and The Ramón y Cajal Program.

J. Blanc-Talon et al. (Eds.): ACIVS 2006, LNCS 4179, pp. 675–686, 2006.

range sensor can provide very accurate 3D face motions. However, computing the 3D face motions from the stream of dense 3D facial data is not straightforward. Indeed, inferring the 3D face motion from the dense 3D data needs an additional process. This process can be the detection of some particular facial features in the range data/images from which the 3D head pose can be inferred. For example, in [5], the 3D nose ridge is detected and then used for computing the 3D head pose. Alternatively, one can perform a registration between 3D data obtained at different time instants in order to infer the relative 3D motions. The most common registration technique is the Iterative Closest Point (ICP) [6] algorithm. This algorithm and its variants can provide accurate 3D motions but their significant computational cost prohibits real-time performance.

The main contribution of this paper is a robust 3D face tracker that combines the advantages of both appearance-based trackers and 3D data-based trackers while keeping the CPU time very close to that required by real-time trackers. In our work, we use the deformable 3D model *Candide* [7] which is a simple model embedding non-rigid facial motion using the concept of facial actions. Our proposed framework for tracking faces in videos can be summarized as follows. First, the 3D head pose and some facial actions are estimated from the monocular image by registering the warped input texture with a shape-free facial texture map. Second, based on these current parameters the 2D locations of the mesh vertices are inferred by projecting the current mesh onto the current video frame. Then the 3D coordinates of these vertices are computed by stereo reconstruction. Third, the relative 3D face motion is then obtained using a robust 3D-to-3D registration technique between two meshes corresponding to the first video frame and the current video frame, respectively. Our framework attempts to reduce the number of outlier vertices by deforming the meshes according to the same current facial actions and by exploiting the symmetrical shape of the 3D mesh.

The resulting 3D face and facial action tracker is accurate, fast, and drift insensitive. Moreover, unlike many proposed frameworks (e.g., [8]), it does not require any learning stage since it is based on online facial appearances and online stereo 3D data.

The remainder of the paper proceeds as follows. Section 2 introduces our deformable 3D facial model. Section 3 states the problem we are focusing on, and describes the online adaptive appearance model. Section 4 summarizes the appearance-based monocular tracker that tracks in real-time the 3D head pose and some facial actions. It gives some evaluation results. Section 5 describes a robust 3D-to-3D registration that combines the monocular tracker's results and the stereo-based reconstructed vertices. Section 6 gives some experimental results.

2 Modeling Faces

In this section, we briefly describe our deformable face model and explain how to produce a shape-free facial texture map.

A Deformable 3D Model. As mentioned before, we use the 3D face model *Candide*. This 3D deformable wireframe model was first developed for the purpose of model-based image coding and computer animation. The 3D shape of this wireframe model is directly recorded in coordinate form. It is given by the coordinates of the 3D vertices

$\mathbf{P}_i, i = 1, \ldots, n$ where n is the number of vertices. Thus, the shape up to a global scale can be fully described by the $3n$-vector $\mathbf{g}$; the concatenation of the 3D coordinates of all vertices $\mathbf{P}_i$. The vector $\mathbf{g}$ is written as:

$$\mathbf{g} = \mathbf{g}_s + \mathbf{A}\,\boldsymbol{\tau}_\mathbf{a} \tag{1}$$

where $\mathbf{g}_s$ is the static shape of the model, $\boldsymbol{\tau}_\mathbf{a}$ the animation control vector, and the columns of $\mathbf{A}$ are the Animation Units. In this study, we use six modes for the facial Animation Units (AUs) matrix $\mathbf{A}$. Without loss of generality, we have chosen the six following AUs: lower lip depressor, lip stretcher, lip corner depressor, upper lip raiser, eyebrow lowerer and outer eyebrow raiser. These AUs are enough to cover most common facial animations (mouth and eyebrow movements). Moreover, they are essential for conveying emotions.

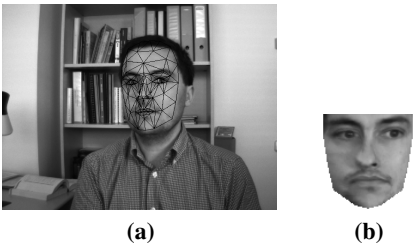
In equation (1), the 3D shape is expressed in a local coordinate system. However, one should relate the 3D coordinates to the image coordinate system. To this end, we adopt the weak perspective projection model. We neglect the perspective effects since the depth variation of the face can be considered as small compared to its absolute depth. Thus, the state of the 3D wireframe model is given by the 3D head pose parameters (three rotations and three translations) and the internal face animation control vector $\boldsymbol{\tau}_\mathbf{a}$. This is given by the 12-dimensional vector $\mathbf{b}$:

$$\mathbf{b} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z, \boldsymbol{\tau}_\mathbf{a}^T]^T \tag{2}$$

Shape-free Facial Texture Maps. A face texture is represented as a shape-free texture (geometrically normalized image). The geometry of this image is obtained by projecting the static shape $\mathbf{g}_s$ using a centered frontal 3D pose onto an image with a given resolution. The texture of this geometrically normalized image is obtained by texture mapping from the triangular 2D mesh in the input image (see figure 1) using a piece-wise affine transform, $\mathcal{W}$. The warping process applied to an input image $\mathbf{y}$ is denoted by:

$$\mathbf{x}(\mathbf{b}) = \mathcal{W}(\mathbf{y}, \mathbf{b}) \tag{3}$$

where $\mathbf{x}$ denotes the shape-free texture map and $\mathbf{b}$ denotes the geometrical parameters. Several resolution levels can be chosen for the shape-free textures. The reported results are obtained with a shape-free patch of 5392 pixels.



(a) **(b)**

Fig. 1. (a) an input image with correct adaptation. **(b)** the corresponding shape-free facial map.

3 Problem Formulation and Facial Texture Model

Given a monocular video sequence depicting a moving head/face, we would like to recover, for each frame, the 3D head pose and the facial actions encoded by the control vector $\boldsymbol{\tau}_{\mathbf{a}}$. In other words, we would like to estimate the vector $\mathbf{b}_t$ (equation 2) at time t given all the observed data until time t, denoted $\mathbf{y}_{1:t} \equiv \{\mathbf{y}_1, \ldots, \mathbf{y}_t\}$. In a tracking context, the model parameters associated with the current frame will be handed over to the next frame. For each input frame $\mathbf{y}_t$, the observation is simply the shape-free texture map associated with the geometric parameters $\mathbf{b}_t$. We use the HAT symbol for the tracked parameters and textures. For a given frame t, $\hat{\mathbf{b}}_t$ represents the computed geometric parameters and $\hat{\mathbf{x}}_t$ the corresponding shape-free texture map, that is,

$$\hat{\mathbf{x}}_t = \mathbf{x}(\hat{\mathbf{b}}_t) = \mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_t) \tag{4}$$

The estimation of $\hat{\mathbf{b}}_t$ from the sequence of images will be presented in the next Section.

By assuming that the pixels within the shape-free patch are independent, we can model the facial appearance using a multivariate Gaussian with a diagonal covariance matrix $\boldsymbol{\Sigma}$. The choice of a Gaussian distribution is motivated by the fact that this kind of distribution provides simple and general model for additive noises. In other words, this multivariate Gaussian is the distribution of the facial texture maps $\hat{\mathbf{x}}_t$. Let $\boldsymbol{\mu}$ be the Gaussian center and $\boldsymbol{\sigma}$ the vector containing the square root of the diagonal elements of the covariance matrix $\boldsymbol{\Sigma}$. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are d-vectors (d is the size of $\mathbf{x}$). Although the independence assumption may be violated, at least locally, we adopt it in our work in order to keep the problem tractable. In summary, the observation likelihood at time t is written as

$$p(\mathbf{y}_t|\mathbf{b}_t) = p(\mathbf{x}_t|\mathbf{b}_t) = \prod_{i=1}^{d} \mathbf{N}(x_i; \mu_i, \sigma_i)_t \tag{5}$$

where $\mathbf{N}(x_i; \mu_i, \sigma_i)$ is a normal density:

$$\mathbf{N}(x_i; \mu_i, \sigma_i) = (2\pi\sigma_i^2)^{-1/2} \exp\left[-\rho\left(\frac{x_i - \mu_i}{\sigma_i}\right)\right], \quad \rho(x) = \frac{1}{2}x^2 \tag{6}$$

We assume that the appearance model summarizes the past observations under an exponential envelope, that is, the past observations are exponentially forgotten with respect to the current texture. When the appearance is tracked for the current input image, *i.e.* the texture $\hat{\mathbf{x}}_t$ is available, we can compute the updated appearance and use it to track in the next frame.

When the appearance is tracked for the current input image, i.e. the texture $\hat{\mathbf{x}}_t$ is available, we can compute the updated appearance and use it to track in the next frame.

It can be shown that the appearance model parameters, i.e., the μ_i's and σ_i's can be updated from time t to time $(t+1)$ using the following equations (see [9] for more details on OAMs):

$$\mu_{i_{(t+1)}} = (1-\alpha)\,\mu_{i_{(t)}} + \alpha\,\hat{x}_{i_{(t)}} \tag{7}$$

$$\sigma^2_{i_{(t+1)}} = (1-\alpha)\,\sigma^2_{i_{(t)}} + \alpha\,(\hat{x}_{i_{(t)}} - \mu_{i_{(t)}})^2 \tag{8}$$

This technique, also called recursive filtering, is simple, time-efficient and therefore, suitable for real-time applications. The appearance parameters reflect the most recent observations within a roughly $L = 1/\alpha$ window with exponential decay. Note that $\boldsymbol{\mu}$ is initialized with the first patch $\hat{\mathbf{x}}_0$. In order to get stable values for the variances, equation (8) is not used until the number of frames reaches a given value (*e.g.*, the first 40 frames). For these frames, the classical variance is used, that is, equation (8) is used with α being set to $\frac{1}{t}$.

4 Tracking by Aligning Facial Texture Maps

We consider the state vector $\mathbf{b} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z, \boldsymbol{\tau}_{\mathbf{a}}^T]^T$ encapsulating the 3D head pose and the facial actions. In [10], we have developed a fast method to compute this state from the previous known state $\hat{\mathbf{b}}_{t-1}$ and the current input image $\mathbf{y}_t$. An overview of this method is presented here.

The sought geometrical parameters $\mathbf{b}_t$ at time t are estimated using a region-based registration technique that does not need any image feature extraction. For this purpose, we minimize the *Mahalanobis* distance between the warped texture and the current appearance mean - the curent Gaussian center $\boldsymbol{\mu}_t$

$$\min_{\mathbf{b}_t} D(\mathbf{x}(\mathbf{b}_t), \boldsymbol{\mu}_t) = \min_{\mathbf{b}_t} \sum_{i=1}^{d} \left(\frac{x_i - \mu_i}{\sigma_i} \right)^2 \tag{9}$$

The above criterion can be minimized using iterative first-order linear approximation which is equivalent to a Gauss-Newton method where the initial solution is given by the previous known state $\hat{\mathbf{b}}_{t-1}$. It is worthwhile noting that the minimization is equivalent to maximizing the likelihood measure given by (5). In the above optimization, the gradient matrix $\frac{\partial \mathcal{W}(\mathbf{y}_t, \mathbf{b}_t)}{\partial \mathbf{b}_t} = \frac{\partial \mathbf{x}_t}{\partial \mathbf{b}_t}$ is computed for each frame and is approximated by numerical differences similarly to the work of Cootes [11].

On a 3.2 GHz PC, a non-optimized C code of the approach computes the 3D head pose and the six facial actions in 50 ms. About half that time is required if one is only interested in computing the 3D head pose parameters.

Accuracy evaluation. In [12], we have evaluated the accuracy of the above proposed monocular tracker. To this end, we have used ground truth data that were recovered by the Iterative Closest Point algorithm [6] and dense 3D facial data. Figure 2 depicts the monocular tracker errors associated with a 300-frame long sequence which contains rotational and translational out-of-plane head motions. The nominal absolute depth of the head was about 65 cm, and the focal length of the camera was 824 pixels. As can be seen, the out-of-plane motion errors can be large for some frames for which there is a room for improvement. Moreover, this evaluation has confirmed the general trend of appearance-based trackers, that is, the out-of-plane motion parameters (pitch angle, yaw angle, and depth) are more affected by errors than the other parameters. We point out that the facial feature motions obtained by the above appearance-based tracker can be accurately recovered. Indeed, these features (the lips and the eyebrows) have specific textures, so their independent motion can be accurately recovered by the appearance-based tracker.

One expects that the monocular tracker accuracy can be improved if an additional cue is used. In our case, the additional cue will be the 3D data associated with the mesh vertices provided by stereo reconstruction. Although the use of stereo data may seem as an excess requirement, recall that cheap and compact stereo systems are now widely available (e.g., [www.ptgrey.com]). We point out that stereo data are used to refine the static model $\mathbf{g}_s$ in the sense that the facial mesh can be more person-specific.

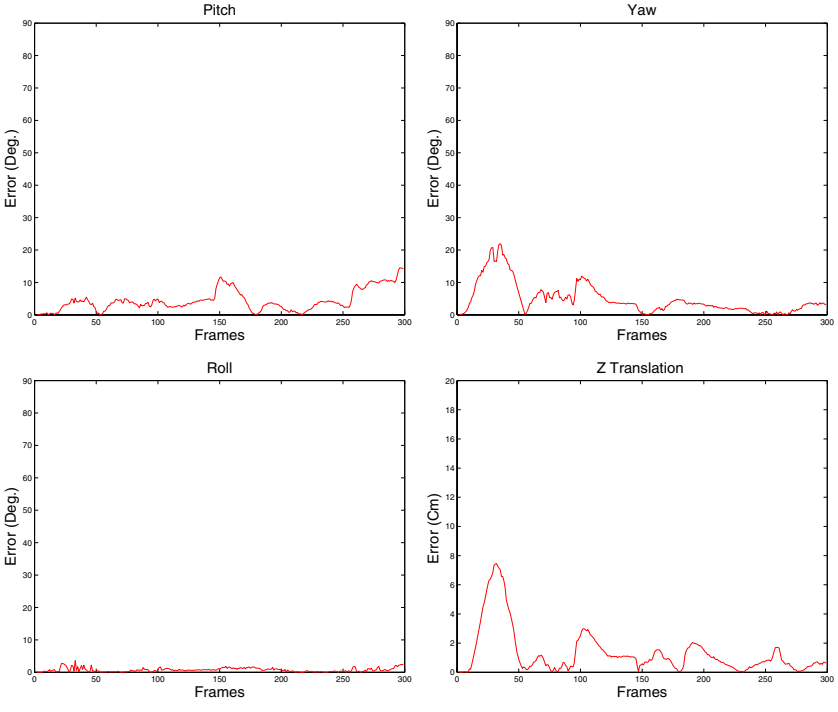


Fig. 2. 3D face motion errors computed by the ICP algorithm associated with a 300-frame long sequence

5 Tracking by Aligning Texture Maps and Stereo-Based 3D Models

In this section, we propose a novel tracking scheme that aims at computing a fast and accurate 3D face motion. To this end, we exploit the tracking results provided by the appearance-based tracker (Section 4) and the availability of a stereo system for reconstructing the mesh vertices. The whole algorithm is outlined in Figure 3. Note that the facial actions are already computed using the technique described in Section 4.

Our approach to 3D face tracking is simple and can be stated as follows: *If the 3D coordinates of the 3D mesh vertices at two different time instants are given in the same coordinate system, then the rigid transform corresponding to the 3D face motion can easily be recovered using a robust 3D point-to-point registration algorithm.* Without

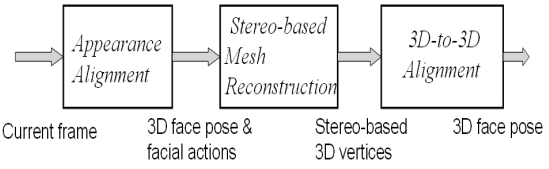


Fig. 3. The main steps of the developed robust 3D face tracker

loss of generality, the 3D face motion will be expressed with respect to the head coordinate system associated with the first video frame[1]. In order to invoke the registration algorithm one has to compute the 3D coordinates of the vertices associated with the two different video frames: the initial video frame and the current one (see Figure 4). This is carried out using stereo-based data associated with the first frame and the current frame. Recall that the first 3D head pose can be inferred using a classical model-based pose estimation algorithm. The proposed algorithm can be summarized as follows.

1. Invoke the appearance-tracker to recover the current 3D head pose and facial actions (Section 4).
2. Based on the estimated 3D head pose and facial actions, deform the 3D mesh and project it onto the current frame.
3. Reconstruct the obtained image points (stereo reconstruction of the mesh vertices).
4. Deform the initial mesh (first frame) according to the current estimated facial actions.
5. Eliminate the vertices that are not consistent with the 3D symmetry test. Recall that the Euclidean distance between two symmetrical vertices is invariant due to the model symmetry.
6. Invoke a robust registration technique that provides the rigid displacement corresponding to the actual 3D face motion between the initial frame and the current frame. This step is detailed in Figure 5.

As can be seen, the recovered 3D face motion has relied on both the appearance model and the stereo-based 3D data. Steps 4 and 5 have been introduced in order to reduce the number of outlier vertices. Since in step 4, the initial mesh is deformed according to the current facial expression, a rigid registration technique can be efficiently applied. Recall that for a profile view the vertices associated with the hidden part of the face may have erroneous depth due to occlusion. Thus, step 5 eliminates the vertices with erroneous depth since they do not satisfy the symmetry constraint. Reducing the number of outliers is very useful for obtaining a very fast robust registration in the sense that a very few random samples are needed. Note that although the appearance-based tracker may provide slightly inaccurate out-of-plane parameters, the corresponding projected mesh onto the current image is still useful for getting the current stereo-based 3D coordinates of the mesh vertices (steps 2 and 3).

[1] Upgrading this relative 3D face motion to a 3D head pose that is expressed in the camera coordinate system is carried out using the 3D head pose associated with the first video frame that can be inferred using a classical 3D pose estimation algorithm.

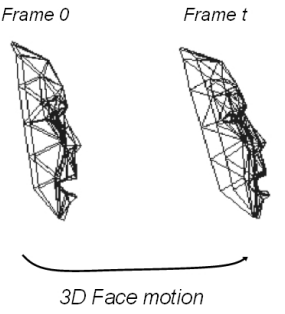


Fig. 4. The relative 3D face motion is recovered using a robust 3D-to-3D registration

We stress the fact that the proposed approach is not similar to a classical stereo-based 3D tracker where feature points are tracked across the image sequence. In our method, there is no feature matching and tracking across the image sequence. Instead, the whole face appearance is tracked in order to accurately locate the facial features (the projection of the mesh vertices) from which the 3D coordinates are inferred.

Robust 3D registration methods have been proposed in recent literature (e.g., see [13, 14]). In our work, we use a RANSAC-like technique that computes an adaptive threshold for inlier/outlier detection.

Inlier detection. The question now is: Given a subsample k and its associated solution $\mathbf{D}_k$, How do we decide whether or not an arbitrary vertex is an inlier? In techniques dealing with 2D geometrical features (points and lines) [15], this is achieved using the distance in the image plane between the actual location of the feature and its mapped location. If this distance is below a given threshold then this feature is considered as an inlier; otherwise, it is considered as an outlier. Here we can do the same by manually defining a distance in 3D space. However, this fixed selected threshold cannot accommodate all cases and all noises. Therefore, we use an adaptive threshold distance that is computed from the residual errors associated with all subsamples. Our idea is to compute a robust estimation of standard deviation of the residual errors. In the exploration step, for each subsample k, the median of residuals was computed. If we denote by $\overline{M}$ the least median among all K medians, then a robust estimation of the standard deviation of the residuals is given by [16]:

$$\hat{\sigma} = 1.4826 \left[1 + \frac{5}{N-3}\right] \sqrt{\overline{M}} \tag{10}$$

where N is the number of vertices. Once $\hat{\sigma}$ is known, any vertex j can be considered as an inlier if its residual error satisfies $|r_j| < 3\,\hat{\sigma}$.

Computational cost. On a 3.2 GHz PC, a non-optimized C code of the robust 3D-to-3D registration takes about 10ms assuming that the number of random samples K is set to 8 and the total number of the 3D mesh vertices, N, is 113. This computational time includes both the stereo reconstruction and the robust technique outlined in Figure 5. Thus, by appending the robust 3D-to-3D registration to the appearance-based tracker (described before) a video frame can be processed in about 60 ms.

Random sampling: Repeat the following three steps K times

1. Draw a random subsample of 3 different pairs of vertices. We have three pairs of 3D points $\{\mathbf{M}_i \leftrightarrow \mathbf{S}_i\}, \quad i = 1, 2, 3$. $\mathbf{M}_i$ denotes the 3D coordinates of vertex i associated with the first frame, and $\mathbf{S}_i$ denotes the 3D coordinates of the same vertex with the current frame t. $\mathbf{M}_i$ and $\mathbf{S}_i$ are expressed in the same coordinate system.
2. For this subsample, indexed by k ($k = 1, \ldots, K$), compute the 3D rigid displacement $\mathbf{D}_k = [\mathbf{R}_k|\mathbf{T}_k]$, where $\mathbf{R}_k$ is a 3D rotation and $\mathbf{T}_k$ a 3D translation, that brings these three pairs into alignment. $\mathbf{R}_k$ and $\mathbf{T}_k$ are computed by minimizing the residual error $\sum_{i=1}^{3} |\mathbf{S}_i - \mathbf{R}_k \mathbf{M}_i - \mathbf{T}_k|^2$. This is carried out using the quaternion method [17].
3. For this solution $\mathbf{D}_k$, compute the median M_k of the squared residual errors with respect to the whole set of N vertices. Note that we have N residuals corresponding to all vertices $\{\mathbf{M}_j \leftrightarrow \mathbf{S}_j\}, \quad j = 1, \ldots, N$. The squared residual associated with an arbitrary vertex $\mathbf{M}_j$ is $|\mathbf{S}_j - \mathbf{R}_k \mathbf{M}_j - \mathbf{T}_k|^2$.

Solution:

1. For each solution $\mathbf{D}_k = [\mathbf{R}_k|\mathbf{T}_k], k = 1, \ldots, K$, compute the number of inliers among the entire set of vertices (see text). Let n_k be this number.
2. Choose the solution that has the largest number of inlier vertices.
3. Refine the corresponding solution using all its inlier pairs.

Fig. 5. Recovering the relative 3D face motion using online stereo and robust statistics

6 Experimental Results

We use the stereo system Bumblebee from Point Grey `[www.ptgrey.com]`. It consists of two Sony ICX084 color CCDs with 6mm focal length lenses. The monocular sequence is used by the appearance-tracker (Section 4), while the stereo sequence is used by the 3D-to-3D registration technique (Section 5). Figure 6 (Top) displays the face and facial action tracking results associated with a 300-frame-long sequence (only three frames are shown). The tracking results were obtained using the proposed framework described in Sections 4 and 5. The upper left corner of each image shows the current appearance ($\boldsymbol{\mu}_t$) and the current shape-free texture ($\hat{\mathbf{x}}_t$). In this sequence, the nominal absolute depth of the head was about 65 cm.

As can be seen, the tracking results indicate good alignment between the mesh model and the images. However, it is very difficult to evaluate the accuracy of the out-of-plane motions by only inspecting the projection of the 3D wireframe onto these 2D images. Therefore, we have run three different methods using the same video sequence. The first method is given by the appearance-based tracker (Section 4). The second method is given the proposed method (Sections 4 and 5). Note that the number of random samples used by the proposed method is set to 8. The third method is given by the ICP registration between dense facial surfaces where the facial surface model is set to the one obtained with the first stereo pair [12]. Since the ICP registration results are accurate we can use them as ground-truth data for the relative 3D face motions.

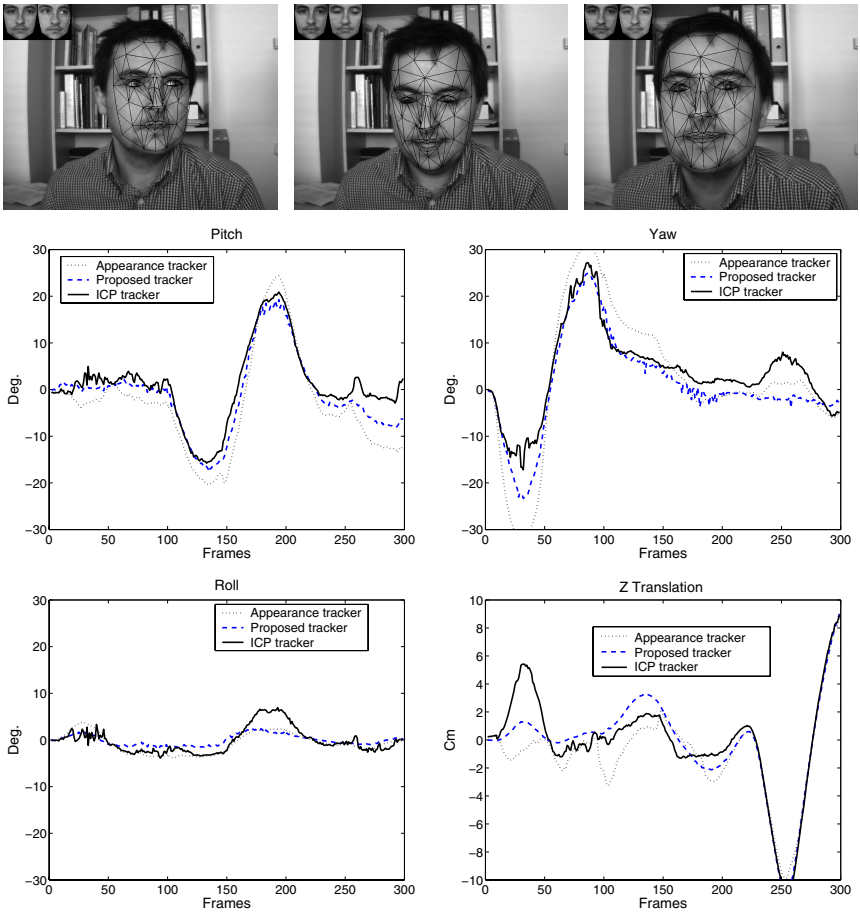


Fig. 6. Top: Tracking the face and facial feature in a 300-frame long sequence using the proposed tracker. Only frames 22, 179, and 255 are shown. **Bottom:** The relative 3D face motion obtained with the three trackers. The dotted curves correspond to the appearance-based tracker, the dashed ones to the proposed framework, and the solid ones to the ICP algorithm.

Figure 6 (Bottom) displays the computed relative 3D face motions obtained with the three methods. This figure displays the three angles and the in-depth translation. The dotted curves correspond to the appearance-based tracker, the dashed ones to the proposed framework, and the solid ones to the ICP algorithm.

From these curves, we can see that the proposed framework has outperformed the appearance-based tracker since the curves become close to those computed by the ICP algorithm - the ground-truth data. In this case, the first facial surface used by the ICP algorithm contained about 20000 3D points. Figure 7 displays the computed relative 3D face motions (pitch and yaw angles) obtained with another video sequence.

Since the ICP algorithm works with rigid surfaces the faces depicted in the sequences of Figures 6 and 7 were somehow neutral. Figure 8 displays the computed relative 3D

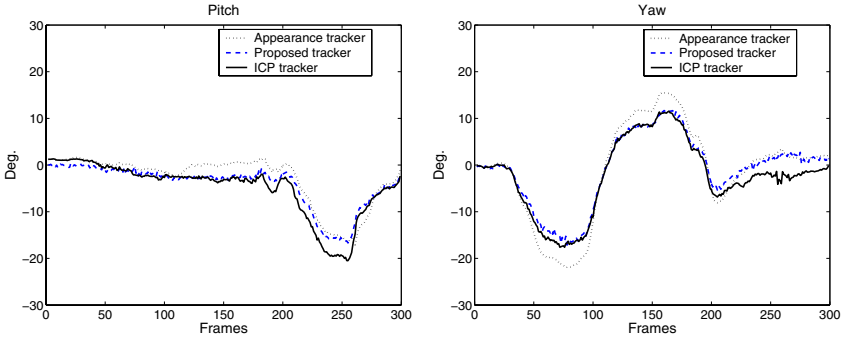


Fig. 7. The relative 3D face motion associated with another video sequence

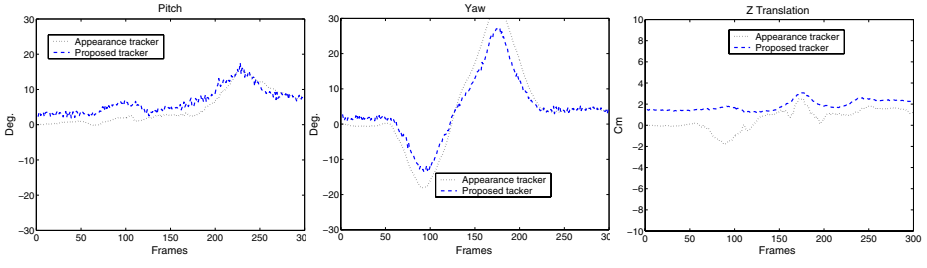


Fig. 8. The relative 3D face motion associated with a video sequence depicting simultaneous head motions and facial expressions

face motions (the three out-of-plane parameters) obtained with another video sequence depicting simultaneous head motions and facial expressions. For this sequence we have not used the ICP algorithm since the facial surface undergoes a rigid and large non-rigid motion. As can be seen, the motion parameters have been improved by using online stereo data.

7 Conclusion

In this paper, we have proposed a robust 3D face tracker that combines the advantages of both appearance-based trackers and 3D data-based trackers while keeping the CPU time very close to that required by real-time trackers. Experiments on real video sequences indicate that the estimates of the out-of-plane motions of the head can be considerably improved by combining a robust 3D-to-3D registration with the appearance model. Although the joint use of 3D facial data and the ICP algorithm as a 3D head tracker could be attractive, the significant computational cost of the ICP algorithm prohibits real-time performance.

References

1. Moreno, F., Tarrida, A., Andrade-Cetto, J., Sanfeliu, A.: 3D real-time tracking fusing color histograms and stereovision. In: IEEE International Conference on Pattern Recognition. (2002)
2. Cascia, M., Sclaroff, S., Athitsos, V.: Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 322–336
3. Ahlberg, J.: An active model for facial feature tracking. EURASIP Journal on Applied Signal Processing **2002** (2002) 566–571
4. Matthews, I., Baker, S.: Active appearance models revisited. International Journal of Computer Vision **60** (2004) 135–164
5. Malassiotis, S., Strintzis, M.G.: Robust real-time 3D head pose estimation from range data. Pattern Recognition **38** (2005) 1153–1165
6. Besl, P., McKay, N.: A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14** (1992) 239–256
7. Ahlberg, J.: CANDIDE-3 - an updated parametrized face. Technical Report LiTH-ISY-R-2326, Department of Electrical Engineering, Linköping University, Sweden (2001)
8. Xiao, J., Baker, S., Matthews, I., Kanade, T.: Real-time combined 2D+3D active appearance models. In: IEEE Int. Conference on Computer Vision and Pattern Recognition. (2004)
9. Jepson, A., Fleet, D., El-Maraghi, T.: Robust online appearance models for visual tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2003) 1296–1311
10. Dornaika, F., Davoine, F.: On appearance based face and facial action tracking. IEEE Transactions on Circuits and Systems for Video Technology (In press)
11. Cootes, T., Edwards, G., Taylor, C.: Active appearance models. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 681–684
12. Dornaika, F., Sappa, A.: Appearance-based tracker: An evaluation study. In: IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. (2005)
13. Chetverikov, D., Stepanov, D., Kresk, P.: Robust Euclidean alignment of 3D point sets: the trimmed iterative closet point algorithm. Image and Vision Computing **23** (2005) 299–309
14. Fitzgibbon, A.: Robust registration of 2D and 3D point sets. Image and Vision Computing **21** (2003) 1145–1153
15. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communication ACM **24** (1981) 381–395
16. Rousseeuw, P., Leroy, A.: Robust Regression and Outlier Detection. John Wiley & Sons, New York (1987)
17. Horn, B.: Closed-form solution of absolute orientation using unit quaternions. J. Opt. Soc. Amer. A. **4** (1987) 629–642