Appearance-based 3D Face Tracker: An Evaluation Study *

Fadi Dornaika and Angel D. Sappa
Computer Vision Center, Edi ci O, Campus UAB
08193 Bellaterra, Barcelona, SPAIN
{*dornaika, sappa*}*@cvc.uab.es*

Abstract

The ability to detect and track human heads and faces in video sequences is useful in a great number of applications. In this paper, we present our recent 3D face tracker that combines Online Appearance Models with an image registration technique. This monocular tracker runs in real-time and is drift insensitive. We introduce a scheme that takes into account the orientation of local facial regions into the registration technique. Moreover, we introduce a general framework for evaluating the developed appearance-based tracker. Precision and usability of the tracker are assessed using stereo-based range facial data from which ground truth 3D motions are inferred. This evaluation quantifies the monocular tracker accuracy, and identifies its working range in 3D space.

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. 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. However, detecting and tracking faces in video sequences is a challenging task because faces are non-rigid and their images have a high degree of variability. A huge research effort has already been devoted to vision-based head and facial feature tracking in 2D and 3D (e.g., [3, 4, 7, 10, 11, 13]).

Tracking the 3D head pose from a monocular image sequence is a difficult problem. Proposed techniques may be roughly classified to those based on optical flow and those based on tracking some salient features. Recently, we developed an appearance-based 3D face tracker adopting the concepts of Online Appearance Models (OAMs) and image-based registration. This approach does not suffer from drifting and seems to be robust in the presence of large head motions and facial animations. In this paper, we summarize the developed approach and propose an accuracy evaluation based on dense depth data obtained from a stereo head. The main innovation of this paper is the introduction of an evaluation of the appearance-based tracker using dense range data.

The rest of the paper is organized as follows. Section 2 describes the deformable 3D face model that we use to create shape-free facial patches from input images. Section 3 describes the problem we are focusing on, and the online reconstruction of the facial appearance model. Section 4 describes the proposed approach, that is, the recovery of the 3D head pose and facial actions using a registration technique. Section 5 gives some experimental results using monocular video sequences. Section 6 introduces an accuracy evaluation using range face data and gives quantitative accuracy evaluation obtained with a real video sequence. Section 7 concludes the paper.

2. Modeling faces

2.1. A deformable 3D model

In our study, 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:

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{S}\,\tau_{\mathrm{s}} + \mathbf{A}\,\tau_{\mathrm{a}} \qquad (1)$$

where $\bar{\mathbf{g}}$ is the standard shape of the model, τ_{s} and τ_{a} are shape and animation control vectors, respectively, and the columns of $\mathbf{S}$ and $\mathbf{A}$ are the Shape and Animation Units. A Shape Unit provides a way to deform the 3D wireframe such as to adapt the eye width, the head width, the eye separation distance etc. Thus, the term $\mathbf{S}\,\tau_{\mathrm{s}}$ accounts for shape

*This work was supported in part by the Government of Spain under the CICYT project TRA2004-06702/AUT and The Ramón y Cajal Program.

variability (inter-person variability) while the term $\mathbf{A}\,\tau_\mathbf{a}$ accounts for the facial animation (intra-person variability). The shape and animation variabilities can be approximated well enough for practical purposes by this linear relation. Also, we assume that the two kinds of variability are independent. With this model, the ideal neutral face configuration is represented by $\tau_\mathbf{a} = \mathbf{0}$.

In this study, we use twelve modes for the facial Shape Units matrix $\mathbf{S}$ and six modes for the facial animation units 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. Therefore, the mapping between the 3D face model and the image is given by a 2×4 matrix, $\mathbf{M}$, encapsulating both the 3D head pose and the camera parameters.

Thus, a 3D vertex $\mathbf{P}_i = (X_i, Y_i, Z_i)^T \subset \mathbf{g}$ will be projected onto the image point $\mathbf{p}_i = (u_i, v_i)^T$ given by:

$$(u_i, v_i)^T = \mathbf{M}\,(X_i, Y_i, Z_i, 1)^T \tag{2}$$

For a given person, $\tau_\mathbf{s}$ is constant. Estimating $\tau_\mathbf{s}$ can be carried out using either feature-based or featureless approaches. 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 $\tau_\mathbf{a}$. This is given by the 12-dimensional vector $\mathbf{b}$:

$$\mathbf{b} = [\theta_x,\ \theta_y,\ \theta_z,\ t_x,\ t_y,\ t_z,\ \tau_\mathbf{a}{}^T\ ]^T \tag{3}$$

2.2. Shape-free facial patches

A face texture is represented as a shape-free texture (geometrically normalized image). The geometry of this image is obtained by projecting the standard shape $\overline{\mathbf{g}}$ 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:

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

where $\mathbf{x}$ denotes the shape-free texture patch 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. Regarding photometric transformations, a zero-mean unit-variance normalization is used to partially compensate for contrast variations. The complete image transformation is implemented as follows: (i) transfer the texture $\mathbf{y}$ using the piecewise affine transform associated with the vector $\mathbf{b}$, and (ii) perform the grey-level normalization of the obtained patch.

(a) **(b)**

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

3. Problem formulation and adaptive observation model

Given a 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 $\tau_\mathbf{a}$. In other words, we would like to estimate the vector $\mathbf{b}_t$ (equation 3) 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 warped texture patch (the shape-free patch) 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 patch, that is,

$$\hat{\mathbf{x}}_t = \mathbf{x}(\hat{\mathbf{b}}_t) = \mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_t) \tag{5}$$

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

The appearance model associated to the shape-free facial patch at time t, A_t, is time-varying on that it models the appearances present in all observations $\hat{\mathbf{x}}$ up to time $(t-1)$. The appearance model A_t obeys a Gaussian with a center μ and a variance σ. Notice that μ and σ are vectors composed of d components/pixels (d is the size of $\mathbf{x}$) that are assumed to be independent of each other. In summary, the observation likelihood at time t is written as

$$p(\mathbf{y}_t|\mathbf{b}_t) = p(\mathbf{x}_t|\mathbf{b}_t) = \prod_{i=1}^{d} \mathbf{N}(x_i; \mu_i, \sigma_i) \tag{6}$$

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

$$\mathbf{N}(x; \mu_i, \sigma_i) = (2\pi\sigma_i^2)^{-1/2} \exp\left[-\frac{1}{2}\left(\frac{x-\mu_i}{\sigma_i}\right)^2\right] \quad (7)$$

We assume that A_t summarizes the past observations under an exponential envelop, 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.

It can be shown that the appearance model parameters, *i.e.*, μ and σ can be updated using the following equations (see [8] for more details on Online Appearance Models):

$$\mu_{t+1} = (1-\alpha)\,\mu_t + \alpha\,\hat{\mathbf{x}}_t \quad (8)$$

$$\sigma_{t+1}^2 = (1-\alpha)\,\sigma_t^2 + \alpha\,(\hat{\mathbf{x}}_t - \mu_t)^2 \quad (9)$$

In the above equations, all μ's and σ^2's are vectorized and the operation is element-wise. 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. Figure 2 shows an envelop having α equal to 0.01 where the current frame is 500.

Note that μ is initialized with the first patch $\mathbf{x}$. However, equation (9) 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 (9) is used with α being set to $\frac{1}{t}$.

Here we used a single Gaussian to model the appearance of each pixel in the shape-free template. However, modeling the appearance with Gaussian mixtures can also be used on the expense of additional computational load (e.g., see [14, 9]).

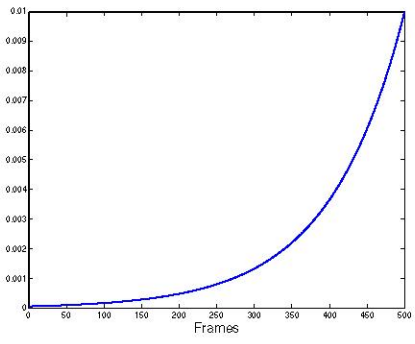

Figure 2: A sliding exponential envelop having an α equal to 0.01. The current frame/time is 500.

4 Tracking with a registration technique

Consider the state vector $\mathbf{b} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z, {\tau_\mathrm{a}}^T]^T$ encapsulating the 3D head pose and the facial animations. In this section, we will show how this state can be recovered for time t from the previous known state $\hat{\mathbf{b}}_{t-1}$.

The sought geometrical parameters $\mathbf{b}_t$ at time t are related to the previous parameters by the following equation ($\hat{\mathbf{b}}_{t-1}$ is known):

$$\mathbf{b}_t = \hat{\mathbf{b}}_{t-1} + \Delta\mathbf{b}_t \quad (10)$$

where $\Delta\mathbf{b}_t$ is the unknown shift in the geometric parameters. This shift is estimated using a region-based registration technique that does not need any image feature extraction. In other words, $\Delta\mathbf{b}_t$ is estimated such that the warped texture will be as close as possible to the facial appearance A_t. For this purpose, we minimize the *Mahalanobis* distance between the warped texture and the current appearance mean,

$$\min_{\mathbf{b}_t} e(\mathbf{b}_t) = \min_{\mathbf{b}_t} D(\mathbf{x}(\mathbf{b}_t), \mu_t) = \sum_{i=1}^{d}\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2 \quad (11)$$

The above criterion can be minimized using iterative first-order linear approximation which is equivalent to a gradient descent method. It is worthwhile noting that minimizing the above criterion is equivalent to maximizing the likelihood measure given by (6).

Gradient-descent registration: We assume that there exists $\mathbf{b}_t = \hat{\mathbf{b}}_{t-1} + \Delta\mathbf{b}_t$ such that the warped shape-free texture will be very close to the appearance mean, i.e,

$$\mathcal{W}(\mathbf{y}_t, \mathbf{b}_t) \simeq \mu_t$$

Approximating $\mathcal{W}(\mathbf{y}_t, \mathbf{b}_t)$ via a first-order Taylor series expansion around $\hat{\mathbf{b}}_{t-1}$ yields

$$\mathcal{W}(\mathbf{y}_t, \mathbf{b}_t) \simeq \mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_{t-1}) + \mathbf{G}_t(\mathbf{b}_t - \hat{\mathbf{b}}_{t-1})$$

where $\mathbf{G}_t$ is the gradient matrix. By combining the previous two equations we have:

$$\mu_t = \mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_{t-1}) + \mathbf{G}_t\,(\mathbf{b}_t - \hat{\mathbf{b}}_{t-1})$$

Therefore, the shift in the parameter space is given by:

$$\Delta\mathbf{b}_t = \mathbf{b}_t - \hat{\mathbf{b}}_{t-1} = -\mathbf{G}_t^{\dagger}\,(\mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_{t-1}) - \mu_t) \quad (12)$$

In practice, the solution $\mathbf{b}_t$ (or equivalently the shift $\Delta\mathbf{b}_t$) is estimated by running several iterations until the error cannot be improved. We proceed as follows.

Starting from $\mathbf{b} = \hat{\mathbf{b}}_{t-1}$, we compute the error vector $(\mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_{t-1}) - \mu_t)$ and the corresponding *Mahalanobis* distance $e(\mathbf{b})$ (given by (11)). We find a shift $\Delta\mathbf{b}$ by multiplying the error vector with the negative pseudo-inverse of the gradient matrix using (12). The vector $\Delta\mathbf{b}$ gives a displacement in the search space for which the error, e, can be minimized. We compute a new parameter vector and a new error:

$$\mathbf{b}' = \mathbf{b} + \rho\,\Delta\mathbf{b} \qquad (13)$$
$$e' = e(\mathbf{b}')$$

where ρ is a positive real.

If $e' < e$, we update $\mathbf{b}$ according to (13) and the process is iterated until convergence. If $e' \geq e$, we try smaller update steps in the same direction (i.e., a smaller ρ is used). Convergence is declared when the error cannot be improved anymore. In practice, we found that convergence is reached with less than ten iterations.

Computation of the gradient matrix. The gradient matrix is given by:

$$\mathbf{G} = \frac{\partial \mathcal{W}(\mathbf{y}_t, \mathbf{b}_t)}{\partial \mathbf{b}} = \frac{\partial \mathbf{x}_t}{\partial \mathbf{b}}$$

It is approximated by numerical differences. Once the solution $\hat{\mathbf{b}}_t$ becomes available for a given frame, it is possible to compute the gradient matrix from the associated input image. We use the following:

The j^{th} column of $\mathbf{G}$ ($j = 1, \ldots, \dim(\mathbf{b})$):

$$\mathbf{G}_j = \frac{\partial \mathcal{W}(\mathbf{y}_t, \mathbf{b}_t)}{\partial b_j}$$

can be estimated using differences

$$\mathbf{G}_j \simeq \frac{\mathcal{W}(\mathbf{y}_t, \mathbf{b}_t) - \mathcal{W}(\mathbf{y}_t, \mathbf{b}_t + \delta\,\mathbf{q}_j)}{\delta}$$

where δ is a suitable step size and $\mathbf{q}_j$ is a vector with all elements zero except the j^{th} element that equals one. To gain more accuracy, the j^{th} column of $\mathbf{G}$ is estimated using several steps around the current value b_j, and then averaging over all these, we get the final $\mathbf{G}_j$ as

$$\mathbf{G}_j = \frac{1}{K} \sum_{k=-K/2, k\neq 0}^{K/2} \frac{\mathcal{W}(\mathbf{y}_t, \mathbf{b}_t) - \mathcal{W}(\mathbf{y}_t, \mathbf{b}_t + k\,\delta_j\,\mathbf{q}_j)}{k\,\delta_j}$$

where δ_j is the smallest perturbation associated with the parameter b_j and K is the number of steps (in our experiments, K is set to 8). Note that other averaging windows can also be used, e.g. triangular or Gaussian windows.

Notice that the computation of the gradient matrix $\mathbf{G}_t$ at time t is carried out using the estimated geometric parameters $\hat{\mathbf{b}}_{t-1}$ and the associated input image $\mathbf{y}_{t-1}$ since the adaptation at time t has not been computed.

It is worthwhile noting that the gradient matrix is computed for each time step. The advantage is twofold. First, a varying gradient matrix is able to accommodate appearance changes. Second, it will be closer to the exact gradient matrix since it is computed for the current geometric configuration (3D head pose and facial animations) whereas a fixed gradient matrix can be a source of errors.

Improving the minimized criterion. When significant out-of-plane rotations of the face occur, local self occlusions and distortions may appear in the shape-free texture $\mathbf{x}$. In order to downweight their influence on the registration technique we incorporate the orientation of individual triangles of the 3D mesh in the minimized criterion such that the contribution of any triangle becomes less significant as it shies away from the frontal view. Recall that the orientation of any 3D triangle with respect to the camera can be recovered since the 3D rotation between the 3D head model and the camera frame is tracked. For a given triangle, m, the angle γ_m is given by the angle between the optical axis $\mathbf{k} = (0, 0, 1)^T$ and the normal to the triangle expressed in the camera frame

For any given frame, the minimized criterion (11) becomes:

$$\min_{\mathbf{b}_t} e(\mathbf{b}_t) = \sum_{i=1}^{d} w(\gamma_i) \left(\frac{x_i - \mu_i}{\sigma_i}\right)^2 \qquad (14)$$

where γ_i is the angle associated to the triangle containing the pixel i and $w(\gamma_i)$ is a monotonic decreasing function. For example, we use $w(\gamma_i) = \frac{a}{1+\gamma_i}$.

Figure 3 displays three real 3D face poses: a frontal view and two non-frontal views. The left column shows the orientation of all individual triangles of the 3D mesh. The right column shows the corresponding shape-free texture.

5. Tracking results

Figure 4 displays the tracking results associated with a 300-frame-long sequence. The sequence is of resolution 640×480 pixels. Only frames 38, 167, 247, and 283 are shown. The upper left corner of each image shows the current appearance (μ_t) and the current shape-free texture ($\hat{\mathbf{x}}_t$). The plots of this figure display the estimated values of the 3D head pose parameters (the three rotations and the three translations) as a function of the sequence frames. Since the used camera is calibrated the absolute translation is recovered.

On a 3.2 GHz PC, a non-optimized C code of the approach computes the 12 degrees of freedom (the six 3D head pose parameters and the six facial actions) in less than 50 ms if the patch resolution is 1310 pixels. About half that time is required to compute the 3D head pose parameters.

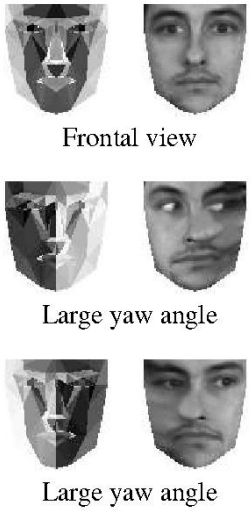

Figure 3: Three different 3D face orientations (from top to bottom): frontal view, a vertical rotation to the left, and a vertical rotation to the right. The first column depicts the angle of all 3D triangles with respect to the camera. Dark grey level corresponds to small angles (the 3D triangle is in fronto-parallel plane) while bright grey level corresponds to large angles. The right column depicts the corresponding shape-free texture.

Figure 5 displays the head and facial action tracking results associated with two video sequences. These sequences are of resolution 640×480 pixels.

6. Accuracy evaluation

The evaluation of the above appearance-based tracker has not been more formal than observing that it works quite well and that the features of the 3D model projects onto their corresponding 2D features in the image sequence. The problem with an objective evaluation is that the absolute truth is not known. This is particularly true for the 3D head pose/motion which is given by six degrees of freedom. However, it is less problematic for the facial feature motion since their estimated motion can be assessed by checking the alignment between the projected 3D model (feature points and line segments) and the actual location of the facial features. In our case, the facial features are given by the lips and the eyebrows so evaluating their motion is straightforward. We point out that their corresponding motions essentially belong to the frontal plane of the face. There are other techniques for measuring face motion, such as motion capture systems based on acoustic trackers [12] or magnetic sensors. However, such systems are expensive and encumbering, and may not succeed to capture small motion accurately. Since we are using a deformable 3D mesh we can adopt an inexpensive solution that employs synthetic

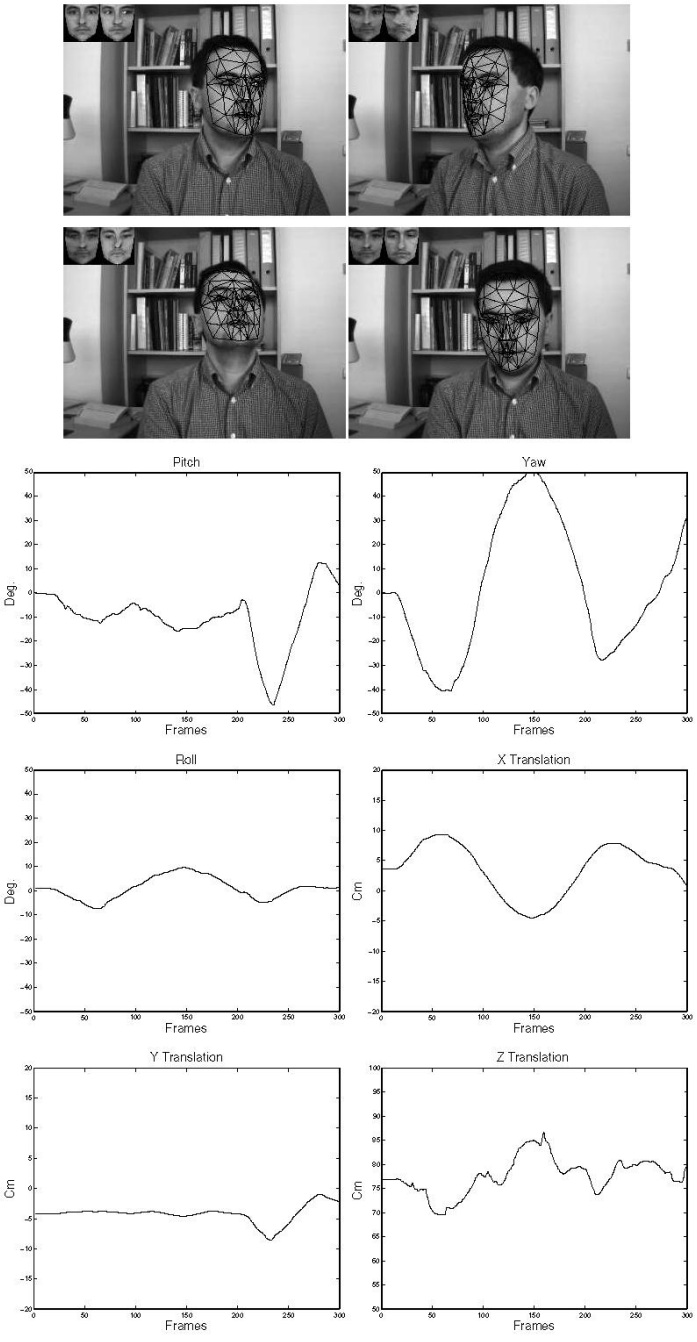

Figure 4: Tracking the 3D head pose with appearance-based tracker. The sequence length is 300 frames. Only frames 38, 167, 247, and 283 are shown. The upper left corner of each image shows the current appearance μ_t and the current texture. The six plots display the six degrees of freedom of the 3D head pose as a function of time.

Figure 5: Tracking the 3D head pose and the facial actions applied to two video sequences.

test sequences with known ground truth similarly to [5, 1]. In [1], a 3D face tracker based on a statistical facial texture was evaluated. A synthetic video sequence is created using a 3D mesh mapping a texture onto it, and then animating it according to some captured or semi-random motion. The tracker then tracks the face in the synthetic video sequence and the discrepancy between the used synthetic motion (ground-truth) and the estimated motion yields the accuracy of the tracker.

Although this scheme can give an idea on the tracker accuracy, it has several shortcomings. First, one can note the self-referential nature of the test, since the same 3D mesh is used in the synthesis phase and in the test phase. Second, synthetic videos may not look very life-like. Third, since the synthetic motion should be realistic to some extent, one has to use the output of another tracker, and if the same tracker is used the evaluation test becomes self-referential regarding the used 3D motions in the sense that the tracker is tested with motion parameters that are easy to estimate. Therefore, our idea is to use stereo-based 3D facial surfaces (from which an accurate rigid 3D head motion can be retrieved), and at the same time run our appearance-based on the associated monocular sequence. Then, the accuracy is evaluated by comparing the 3D head motions provided by the stereo data and the developed monocular 3D face tracker.

6.1. 3D facial data and 3D face motion

A commercial stereo vision camera system (Bumblebee from Point Grey [6]) was used. It consists of two Sony ICX084 color CCDs with 6mm focal length lenses. Bumblebee is a precalibrated system that does not require in-field calibration. The baseline of the stereo head is 12cm

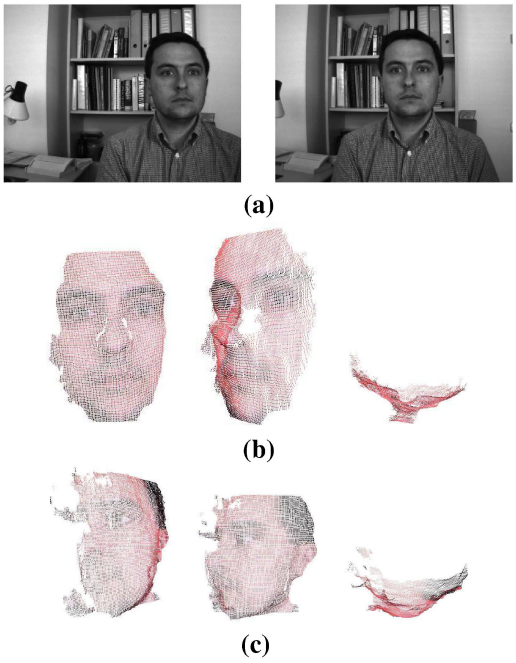

Figure 6: Dense 3D facial data provided by a stereo head. **(a)** a stereo pair. **(b)** the corresponding computed 3D facial data with mapped texture displayed from three different points of view. **(c)** 3D facial data associated to another stereo pair illustrating a non-frontal face.

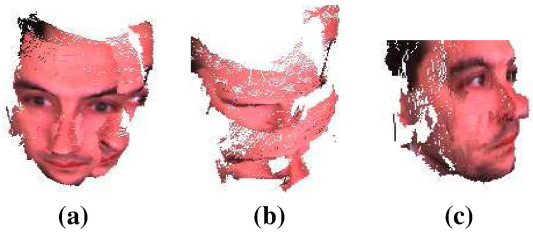

Figure 7: 3D registration of two facial clouds provided at frames 1 and 39, which are separated by a large yaw angle (about 40 degrees). **(a)** the range facial data associated to frames 1 and 39, expressed in the same coordinate system. **(b)** alignment results using the relative 3D face motion provided by our monocular tracker. **(c)** refinement of the registration using the Iterative Closet Point algorithm.

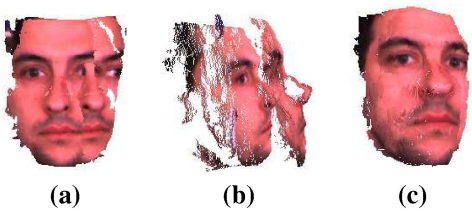

Figure 8: 3D registration of two facial clouds provided at frames 1 and 85.

and it is connected to the computer by a IEEE-1394 connector. Right and left color images were captured at a resolution of 640×480 pixels and a frame rate near to 30 fps. After capturing these right and left images, 3D data were computed by using the provided 3D reconstruction software. During the experiments the stereo head was placed at a distance of 80 centimeters from the person. Figure 6.**a** shows a stereo pair used in our evaluation. Figure 6.**b** shows the corresponding 3D facial data visualized from three different points of view. Figure 6.**c** depicts the 3D data associated with another stereo pair depicting a non-frontal face. In our case, the 3D face model (a cloud of 3D points) is manually selected in the first stereo frame which is captured in a frontal view (Figure 6.**b**). This 3D face model contains about 20500 3D points. More elaborated statistical techniques could be used for segmenting the 3D head in the range images (e.g., [10]). For subsequent frames, we adopt a moving head's bounding box to bound the head region in the range images (Figure 6.**b**). Note that for these frames accurate head segmentation is not required since we are using a global 3D registration technique (see below).

The proposed evaluation, as mentioned before, consists of computing the 3D face motions using two different methods: (i) the proposed appearance-based approach (Section 4) using the monocular sequence provided by the right camera, and (ii) the 3D face motions computed by 3D registration of 3D facial data in different frames. Recall that the 3D rigid displacement that align two facial clouds obtained at two different frames is equivalent to the performed 3D head motion between these two frames.

The 3D registration is computed by means of the well known Iterative Closest Point, ICP, algorithm. ICP, also referenced in the literature as a fine registration technique, assumes that the clouds to be registered are very close. ICP has been originally presented by [2]. In our evaluation, since we use the 3D facial data/cloud in the first reference frame as a face model, the 3D registration may fail in subsequent frames containing large rotations, thus our idea is to use the monocular tracker solution as a starting solution for the ICP algorithm (see Figures 7 and 8). Therefore, the ICP returns a 3D rigid displacement that directly quantifies the monocular tracker accuracy.

6.2. Tracker accuracy

The 3D head pose was tracked in the previous 300-frame long sequence using two approaches: (i) the monocular tracker, and (ii) the joint use of the stereo-based facial data and the ICP algorithm. The 3D head pose parameters provided by the monocular tracker gives the position and orientation of the 3D wireframe model with respect to the right camera frame (the one used by the monocular tracker). The 3D head motion is set to the 3D motion between the first frame and the current frame, which is easily recovered from the corresponding absolute 3D head poses. Figure 7 illustrates the estimated 3D head pose/motion associated with frame 39. Figure 7.(**a**) illustrates the range facial data associated to frames 1 and 39, expressed in the same coordinate system. One can easily see that the face has performed a vertical rotation of about 40 degrees. Figure 7.(**b**) displays the 3D alignment obtained using the relative 3D face motion provided by the monocular tracker. Figure 7.(**c**) shows the refinement of the registration using the ICP algorithm whose returned displacement gives the 3D error associated with the monocular tracker. One can notice that the monocular registration technique brings the two 3D clouds into a good alignment even though there is an offset in the in-depth translation. This is due to the monocular vision effect and to the use of a frontal texture model.

Figure 8 illustrates the estimated 3D head pose/motion associated with frame 85 using the monocular tracker and the ICP algorithm.

Figure 9 depicts the monocular tracker errors associated with the sequence depicted in Figure 4. These errors are computed by the ICP algorithm. As can be seen, the errors increase as the face shies away from the frontal view. However, the tracker never loose the track. As can be seen, due to the effect of monocular vision and very large out-of-plane rotations (more than 60 degrees) the estimated depth may suffer from 6cm error. However, within a useful working range about the frontal view, this error is about 3 cm which corresponds to one pixel error given our camera parameters and the actual depth of the face.

7. Conclusion

In this paper, we have described our appearance-based 3D face tracker. We have introduced a general evaluation framework that is based on stereo-based range facial data. We have found that the out-of-plane motions can be off the track whenever the absolute orientation of the face is so far from the frontal view, e.g, a vertical rotation of 60 degrees. However, even in these extreme cases, the appearance-based tracker is still usable and do not suffer from drifting due to these out-of-plane motion inaccuracies that can be explained not only by the monocular effect but also by the fact that the texture/appearance of the 3D wireframe is modelled in a frontal view. Adopting multi-view shape-free texture models that partition the 3D rotations of the face is expected to considerably decrease such inaccuracies.

Although the joint use of 3D facial data and the ICP algorithm can be used as a 3D head tracker, the significant computational overhead of ICP algorithm prohibits real-time performance. In the light of the evaluation, one is able to adjust the experimental set-up such that the monocular tracker provides optimal results.

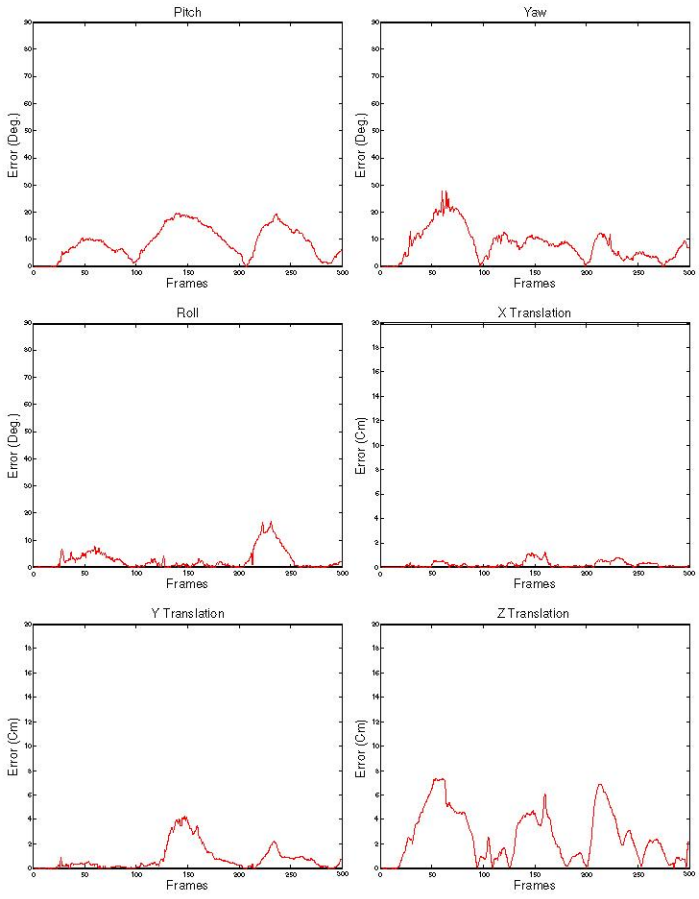

Figure 9: 3D head pose errors computed by the ICP algorithm associated with 300-frame long sequence (see Figure 4). For each degree of freedom, the absolute value of the error is plotted. For each frame, the ICP algorithm was initialized by the output of the monocular tracker, thus the refined 3D registration can be considered as the monocular tracker error.

References

[1] J. Ahlberg. *Model-based coding: Extraction, coding, and evaluation of face model parameters*. PhD thesis, No. 761, Linköping University, Sweden, September 2002.

[2] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[3] M.L. Cascia, S. Sclaroff, and V. Athitsos. 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(4):322–336, 2000.

[4] S.B. Gokturk, J.Y. Bouguet, and R. Grzeszczuk. A data-driven model for monocular face tracking. In *IEEE International Conference on Computer Vision*, 2001.

[5] M. Harville, A. Rahimi, T. Darell, G. Gordon, and J. Woodfill. 3D pose tracking with linear depth and brightness constraints. In *IEEE International Conference on Computer Vision*, 1999.

[6] http://www.ptgrey.com.

[7] T.S. Jebara and A. Pentland. Parameterized structure from motion for 3D adaptative feedback tracking of faces. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[8] A.D. Jepson, D.J. Fleet, and T.F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, 2003.

[9] D. Lee. Effective Gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832, 2005.

[10] S. Malassiotis and M. G. Strintzis. Robust real-time 3D head pose estimation from range data. *Pattern Recognition*, 38(2005):1153–1165, 8.

[11] F. Moreno, A. Tarrida, J. Andrade-Cetto, and A. Sanfeliu. 3D real-time tracking fusing color histograms and stereovision. In *IEEE International Conference on Pattern Recognition*, 2002.

[12] L.D. Mouse. Acoustic tracking system. http://www.vrdepot.com/vrteclg.htm.

[13] M.H. Yang, D.J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[14] S. Zhou, R. Chellappa, and B. Mogghaddam. Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, 13(11):1473–1490, 2004.