

---

# Head and Facial Animation Tracking using Appearance-Adaptive Models and Particle Filters

Franck Davoine<sup>1</sup> and Fadi Dornaika<sup>2</sup>

HEUDIASYC Mixed Research Unit,  
CNRS / Compiègne University of Technology, FRANCE  
Franck.Davoine@hds.utc.fr  
Fdornaika@yahoo.com

In this chapter, we address the problem of tracking a face and its facial features in real video sequences, considering two approaches. The first approach is based on a particle filter tracker capable of tracking the 3D head pose of a person. In this case, the distribution of observations is derived from an eigenspace decomposition. The second approach introduces an appearance-adaptive tracker capable of tracking both the 3D head pose and facial animations. It consists of an online adapted observation model of the face texture, together with adaptive dynamics in the sense that they are guided by a deterministic search in a state space. This approach extends the concept of online appearance models to the case of tracking 3D non-rigid face motion (3D head pose and facial animations). Experiments on real video sequences show the effectiveness of the developed methods. Accurate tracking was obtained even in the presence of perturbing factors such as significant head pose or local facial occlusions.

## 1 Introduction

This chapter addresses the problem of tracking in a single video the global motion of a face as well as the local motion of its inner features, due for instance to expressions or other facial behaviors. This task is required in many emerging applications, like surveillance, teleconferencing, emotional computer interfaces, motion capture for video synthesis, automated lipreading, driver drowsiness monitoring, etc. Face tracking poses challenging problems because of the variability of facial appearance within a video sequence, most notably due to changes in head pose, expressions, lighting or occlusions. Much research has thus been devoted to the problem of face tracking, as a specially difficult case of non-rigid object tracking.

In the object tracking literature, the following formulation of the tracking problem is conveniently used: at each time step  $t$ , the goal is to infer the unobserved state of the object, denoted  $\mathbf{b}_t \in \mathcal{B}$ , given all the observed data until time  $t$ , denoted  $\mathbf{y}_{1:t} \equiv \{\mathbf{y}_1, \dots, \mathbf{y}_t\}$ . When tracking a face in 3D, the unobserved state  $\mathbf{b}_t$  includes motion or pose parameters like the position, scale and orientation of the face; when facial features are also tracked, the unobserved state should contain parameters describing the face inner motion. The observed data  $\mathbf{y}_t$  consists of measurements derived from the current video frame, such as greylevel patches, edges, or color histograms. In order to evaluate a hypothesized state, the measurements are actually only considered in the image area corresponding to the hypothesized location. For instance, the most natural measurement consists of the pixel greylevel values themselves. Basically, a given state  $\mathbf{b}_t$  (motion parameters) is then evaluated by comparing the motion-compensated greylevel image patch  $\mathbf{x}(\mathbf{b}_t)$  with a greylevel template face patch  $\mathbf{g}_{model}$ .

The tracking task then essentially consists in searching the current state  $\hat{\mathbf{b}}_t \in \mathcal{B}$  that matches at best the measurements  $\mathbf{y}_t$  in the current image. The tracking history  $\hat{\mathbf{b}}_{1:(t-1)}$  is mainly used as a prior knowledge in order to search only a small subset of the state space  $\mathcal{B}$ . More specifically, for a hypothesized state  $\mathbf{b}_t$ , measurements  $\mathbf{x}(\mathbf{b}_t)$  are extracted from the image patch at the hypothesized location, and those measurements are matched against some model of the object. The various tracking methods can be categorized according to the considered class of measurements, joint model of state and measurements and induced matching criterion (error functional to minimize or probability to maximize), and inference technique (deterministic or stochastic). According to this classification, a brief and non-exhaustive survey of tracking approaches is given below.

In a non-probabilistic formulation of the tracking problem, the state  $\mathbf{b}_t$  is usually sought so as to minimize an error functional  $d[\mathbf{x}(\mathbf{b}_t); \mathbf{g}_{model}]$ , e.g. an euclidean or robust distance. Actually, in a tracking setting, the state is supposed to evolve little between consecutive time steps. The solution is thus searched as a small displacement from the previous frame estimation:  $\hat{\mathbf{b}}_t = \hat{\mathbf{b}}_{t-1} + \widehat{\Delta\mathbf{b}}_t$ . The optimal displacement is then typically obtained by a gradient-like descent method. The well-known Lucas-Kanade algorithm [11] is a particular case of such a formulation, and has been recently generalized in [2]. Instead of being specified by a single face greylevel template  $\mathbf{g}_{model}$ , the face model can span a subspace of greylevel patches, learnt by principal component analysis from a face training set. The error functional is then a distance from the image patch  $\mathbf{x}(\mathbf{b}_t)$  to the face subspace, usually taken to be the distance to the projection in face subspace. The subspace modeling allows to account for some variability of the global face appearance. The eigen-tracking method is based on such a principle [3]. Using also principal component analysis, the Active Appearance Models (AAMs) encode the variations of face appearance by learning the shape and texture variations [5].

They enable thus the tracking of both global motion and inner features. In the case of AAMs, the gradient jacobian matrix is pre-computed in order to save processing time. In practice, tracking using the deterministic AAM search appears to work well while the lighting conditions remain stable and only small occlusions are present. However, large occlusions often make the AAM search converge to incorrect positions and loose track of the face.

In probabilistic formulations, the hidden state and the observations are linked by a joint distribution; this statistical framework offers rich modeling possibilities. A Markovian dynamic model describes how the state evolves through time. An observation model specifies the likelihood of each hypothesised state, i.e. the probability that the considered state may generate the observed data. Such generative models represent the variability in the motion and appearance of the object to track. Note that even the non-probabilistic minimization of an error functional can be recast as the maximization of a likelihood:

$$p(\mathbf{x}_t|\mathbf{b}_t) \propto \exp -d[\mathbf{x}(\mathbf{b}_t); \mathbf{g}_{model}]$$

Based on such a generative model, Bayesian filtering methods recursively evaluate the posterior density of the target state at each time step conditionally to the history of observations until the current time.

Stochastic implementations of Bayesian filtering are generally based on sequential Monte Carlo estimation, also known as particle filtering [7]. When compared with the analytical solution provided by the well-known Kalman filter, particle filtering has two advantages: it is not restricted to the case of linear and Gaussian models, and it can maintain multiple hypotheses of the current state, a desirable property when the background is complex and contains distracting clutter.

For video tracking, the CONDENSATION algorithm was first proposed in conjunction with edge measurements produced by an edge detector [9]. Since then, this algorithm has attracted much interest, and other kinds of measurements have given valuable variants. For instance, tracking based on color histograms has gained recent interest due to the development of efficient search techniques, whether based on the deterministic mean-shift search paradigm [4] or the stochastic particle filtering framework [14]. In the case of particle filtering, motion is used as an additional information in order to remove ambiguities due to color used alone. However, since color histograms are global, they do not allow to track the motion of internal facial features as is the goal here. A grayscale patch is used as measurement vector by Zhou *et al.* [15]. In order to cope with the changing appearance of the face, the likelihood is taken to be a mixture of three appearance templates, and the parameters of the mixture are re-estimated during the tracking. They consider a modified version of the online appearance model (OAM) proposed by Jepson *et al.* [10]. Their model considers global inter-frame face motion and appearance templates in the likelihood, and global motion parameters in the state vector. In order to

track the local motion of facial features, De la Torre *et al.* [6] model the face appearance as a set of image patches taken at feature points.

This chapter has two main contributions. The first one consists in combining a modified version of the AAM as developed in [1] with the CONDENSATION stochastic search in order to augment its robustness to occlusions and strong out-of-plane face rotations. This approach uses a 3D wireframe model of a human face. It is developed to track the global 3D head pose of a person. The second contribution consists in combining a modified version of the OAM as proposed in [15, 10] with the 3D parameterized wireframe model deformed by Animation Units (AUs) that describe deformations that are possible to perform by a human face. This second contribution extends the concept of OAM to the case of tracking 3D non-rigid face motion (3D head pose and facial animations).

The chapter 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 condensation-based 3D head tracking approach, and presents some experimental results. Section 4 describes the head and facial animation tracking using an adaptive facial appearance model and an adaptive transition model, and presents some experimental results. And finally, section 5 concludes the chapter.

## 2 Modeling faces

### 2.1 A deformable 3D model

We introduce here 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  $\mathbf{P}_i, i = 1, \dots, n$  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:

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{S} \tau_s + \mathbf{A} \tau_a \quad (1)$$

where  $\bar{\mathbf{g}}$  is the standard shape of the model,  $\tau_s$  and  $\tau_a$  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  $\mathbf{S} \tau_s$  accounts for shape variability (inter-person variability) while the term  $\mathbf{A} \tau_a$  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.

In this study, we use twelve modes for the Shape Units matrix and six modes for the Animation Units matrix. Without loss of generality, we have chosen the six following AUs: jaw drop, 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).

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  $2 \times 4$  matrix,  $\mathbf{M}$ , encapsulating both the 3D head pose and the camera parameters.

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

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

For a given person,  $\tau_s$  is constant. Estimating  $\tau_s$  can be carried out using either feature-based or featureless approaches. Thus, the state of the 3D wire-frame 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 state vector  $\mathbf{b}$ :

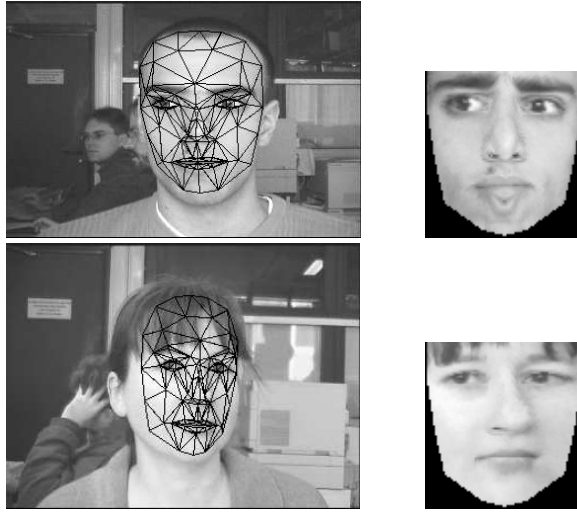
$$\mathbf{b} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z, \tau_{\mathbf{a}}^T]^T \quad (3)$$

## 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  $\bar{\mathbf{g}}$  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:

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

where  $\mathbf{x}$  denotes the shape-free texture patch and  $\mathbf{b}$  denotes the geometrical parameters. Two resolution levels have been considered for the shape-free textures, encoded by 1310 or 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 piece-wise affine transform associated with the vector  $\mathbf{b}$ , and (ii) perform the grey-level normalization of the obtained patch.



**Fig. 1.** Left column: two input images with correct adaptation. Right column: the corresponding shape-free facial images.

### 3 Condensation-based head pose tracking

Given a video sequence depicting a moving face, the tracking consists in estimating, for each frame, the head pose as well as the facial animations encoded by the control vector  $\tau_{\mathbf{a}}$ . In other words, one would like to estimate the vector  $\mathbf{b}_t$  (equation 3) at time  $t$ . In a tracking context, the model parameters associated with the current frame will be handed over to the next frame.

In this section, we are interested in tracking the global 3D head pose. Therefore, the state vector  $\mathbf{b}$  is given by  $\mathbf{b} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z]^T$ . In this particular case, the animation parameters  $\tau_{\mathbf{a}}$  are set to zero. We propose a CONDENSATION-based method to tracks the six degrees of freedom associated with the head motion where the face model is given the *Candide* model.

Particle filtering approximates the posterior state density  $p(\mathbf{b}_t | \mathbf{y}_{1:t})$  by a set of  $J$  random weighted samples (particles) at each time step. The CONDENSATION algorithm consists in propagating this sample set  $\{\mathbf{b}_t^{(j)}, w_t^{(j)}\}_{j=1}^J$  through time using a dynamic model and in weighting each sample proportionally to its likelihood function value [9] (the particles explore the state space following independent realizations from a state evolution model, and are re-distributed according to their consistency with the observations). Finally, the state estimate  $\hat{\mathbf{b}}_t$  at time  $t$  is set to the maximum *a posteriori*:

$$\hat{\mathbf{b}}_t = \arg \max_{\mathbf{b}_t} p(\mathbf{b}_t | \mathbf{y}_{1:t}) \approx \arg \max_{\mathbf{b}_t} w_t^{(j)} \quad (5)$$

In this work, we use the following simple state evolution model:

$$\mathbf{b}_t = \hat{\mathbf{b}}_{t-1} + \mathbf{U}_t \quad (6)$$

$\mathbf{U}_t$  is a random vector having a centered normal distribution,  $\mathbf{N}(\mathbf{0}, \Sigma)$ . The covariance matrix  $\Sigma$  is learned off-line from the state vector differences  $\mathbf{b}_t - \mathbf{b}_{t-1}$  associated with previously tracked video sequences.

Since image data  $\mathbf{y}$  are represented as shape-free texture patches  $\mathbf{x}$ , we can set the observation likelihood  $p(\mathbf{y}_t|\mathbf{b}_t)$  to  $p(\mathbf{x}_t|\mathbf{b}_t)$ . It quantifies the consistence of the texture  $\mathbf{x}(\mathbf{b}_t)$  with the statistical texture model represented by texture modes (eigenvectors). For this purpose, we use a likelihood measure such as the one proposed in [12]:

$$p(\mathbf{x}_t|\mathbf{b}_t) = c \exp\left(-\frac{1}{2} \sum_{i=1}^M \frac{\xi_i^2}{\lambda_i}\right) \exp\left(-\frac{e^2(\mathbf{x}_t)}{2\rho^*}\right) \quad (7)$$

where  $e^2(\mathbf{x}_t)$  is the “distance-from-feature-space”,  $\lambda_i$ s are the eigenvalues associated with the first  $M$  eigenvectors,  $\xi_i$ s are the first  $M$  principal components and  $\rho^*$  is the arithmetic average of the remaining eigenvalues.

1. Initialization  $t = 0$ : Generate  $J$  state samples  $\mathbf{a}_0^{(1)}, \dots, \mathbf{a}_0^{(J)}$  according to some prior density  $p(\mathbf{b}_0)$  and assign them identical weights,  $w_0^{(1)} = \dots = w_0^{(J)} = 1/J$
2. At time step  $t$ , we have  $J$  weighted particles  $(\mathbf{a}_{t-1}^{(j)}, w_{t-1}^{(j)})$  that approximate the posterior distribution of the state  $p(\mathbf{b}_{t-1}|\mathbf{x}_{1:(t-1)})$  at previous time step
  - a) Resample the particles proportionally to their weights, *i.e.* keep only particles with high weights and remove particles with small ones. Resampled particles have the same weights
  - b) Draw  $J$  particles according to the dynamic model  $p(\mathbf{b}_t|\mathbf{b}_{t-1} = \mathbf{a}_{t-1}^{(j)})$ . These particles approximate the predicted distribution  $p(\mathbf{b}_t|\mathbf{x}_{1:(t-1)})$
  - c) Compute the geometrically normalized texture  $\mathbf{x}(\mathbf{b}_t)$  according to (4)
  - d) Weight each new particle proportionally to its likelihood:

$$w_t^{(j)} = \frac{p(\mathbf{x}_t|\mathbf{b}_t = \mathbf{a}_{t-1}^{(j)})}{\sum_{m=1}^J p(\mathbf{x}_t|\mathbf{b}_t = \mathbf{a}_{t-1}^{(m)})}$$

The set of weighted particles approximates the posterior  $p(\mathbf{b}_t|\mathbf{x}_{1:t})$

- e) Give an estimate of the state  $\hat{\mathbf{b}}_t$  as the MAP:

$$\hat{\mathbf{b}}_t = \arg \max_{\mathbf{b}_t} p(\mathbf{b}_t|\mathbf{y}_{1:t}) \approx \arg \max_{\mathbf{a}_t^{(j)}} w_t^{(j)}$$

**Fig. 2.** CONDENSATION algorithm.

The sketch of the CONDENSATION algorithm is recalled in figure 2. For a good introduction to the algorithm, the reader is referred to the seminal paper of Isard and Blake [9].

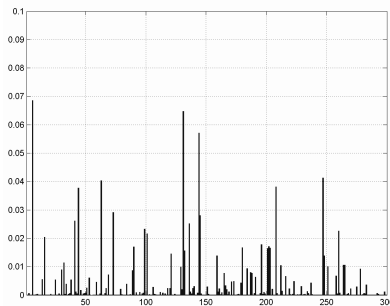
During a filtering iteration, due to the resampling step, samples with a high weight may be chosen several times while others with relatively low weights may not be chosen at all. Note that the initial distribution  $p(\mathbf{b}_0)$  can be either a Dirac or Gaussian distribution centered on a solution provided by a detector algorithm or manually specified.

### 3.1 Experiments

Figure 3 displays the tracking results associated with several frames of a long test sequence. It corresponds to the particle-filter-based tracking algorithm using a statistical texture model. The number of particles is set to 300. For each frame in this figure, only the MAP solution is displayed. The statistical facial texture is built with 330 training face images and the number of the principal eigenvectors is 20. The bottom plot displays the weights associated with the bottom-right image.



**Fig. 3.** Particle filter-based 3D head tracking with a statistical facial texture model.



**Fig. 4.** Drawing of the 300 particle weights associated to the bottom-right image of Figure 3.

## 4 Head and facial animation tracking using an adaptive appearance model

In this section, we consider now the 3D head pose as well as the facial animations, that is, the state vector  $\mathbf{b}$  is given by  $\mathbf{b} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z, \tau_{\mathbf{a}}^T]^T$ .

### 4.1 Motivations

The efficiency of the stochastic tracking algorithm presented in section 4 depends on different factors. However, the main factor which limits the efficiency of stochastic tracking algorithms is the lack of a suitable state evolution model. Indeed, there are two ways for handling the transition model. (i) the first is to learn state transition models directly from training video sequences. For example, in [13] the authors use Expectation-Maximization combined with the CONDENSATION algorithm to learn multiclass dynamics associated with a juggled ball. However, such models may not necessarily succeed when presented with testing videos featuring different types of motions. (ii) the second is to use a fixed model with fixed noise variance for simplicity, that is, the predicted state is simply the previous state (or a shifted version of it) to which a random noise with fixed variance is added (this methodology was adopted in section 4). If the variance is very small, it is hard to model rapid movements; if the variance is large, it is computationally inefficient since many more particles are needed to accommodate large noise variance.

In addition to the problems associated with the state transition model, the observation model has its own limitations. For example, if the observation model (observation likelihood) is built upon a statistical texture model, any significant change in the imaging conditions will make the corresponding learned observation model useless and one should build a new observation model based on a new statistical texture model.

For all these factors, we develop a new tracking framework capable of coping with the limitations mentioned above. Our approach is to make both observation and state transition models adaptive in the framework of a particle filter, with provisions for handling outliers embedded. The main features of the developed approach are:

- Adaptive observation model. We adopt an appearance-based approach, using the concept of online appearance model (OAM) developed in [10] and modified in [15], where the appearance is learned on-line from the tracked video sequence. However, in our case, we extend this paradigm to the case of tracking the 3D non-rigid face motion (3D head pose together with facial animations). Therefore, the observation model is adaptive as the appearance of the texture.
- Adaptive state transition model. Instead of using a fixed state transition model, we use an adaptive-velocity model, where the motion velocity is predicted using a registration technique between the incoming observation and the current appearance configuration. We also use an adaptive noise component whose magnitude is a function of the registration error. We vary the number of particles based on the noise component.
- Handling occlusion. Occlusion and large image variations are handled using robust statistics. We improve the robustness of the likelihood measurement and the adaptive velocity estimate by downweighting the outlier pixels.

## 4.2 Adaptive observation model

The appearance model at time  $t$ ,  $A_t$ , is a time-varying one that it models the appearances present in all observations  $\mathbf{x}$  up to time  $(t - 1)$ . For each frame, the observation is simply the warped texture patch associated with the computed 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 texture patch, that is,

$$\hat{\mathbf{x}}_t = \mathbf{x}(\hat{\mathbf{b}}_t) = \mathcal{W}(\mathbf{y}_t, \hat{\mathbf{b}}_t) \quad (8)$$

The appearance model  $A_t$  obeys a Gaussian with a center  $\mu$  and a variance  $\sigma$ . Notice that  $\mu$  and  $\sigma$  are vectors consisting of  $d$  pixels ( $d$  is the size of  $\mathbf{x}$ ) that are assumed to be independent of each other. In summary, the observation likelihood is written as

$$p(\mathbf{y}_t | \mathbf{b}_t) = p(\mathbf{x}_t | \mathbf{b}_t) = \prod_{i=1}^d \mathbf{N}(x_i; \mu_i, \sigma_i) \quad (9)$$

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

$$\mathbf{N}(x; \mu_i, \sigma_i) = (2\pi\sigma_i^2)^{-1/2} \exp \left[ -\frac{1}{2} \left( \frac{x - \mu_i}{\sigma_i} \right)^2 \right] \quad (10)$$

We assume that  $A_t$  summarizes the past observations under an exponential envelop with a forgetting factor  $\alpha$ . 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.*,  $\mu$  and  $\sigma$  can be updated using the following equations (see [10] for more details on OAMs):

$$\mu_{t+1} = \alpha \mu_t + (1 - \alpha) \hat{\mathbf{x}}_t \quad (11)$$

$$\sigma_{t+1}^2 = \alpha \sigma_t^2 + (1 - \alpha) (\hat{\mathbf{x}}_t - \mu_t)^2 \quad (12)$$

In the above equations, all  $\mu$ 's and  $\sigma^2$ '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.

Note that  $\mu$  is initialized with the first patch  $\mathbf{x}$ . However, equation (12) is not used until the number of frames reaches a certain value (*e.g.*, the first 40 frames). For these frames, the classical variance is used, that is, equation (12) is utilized with  $\alpha$  being set to  $1 - \frac{1}{t}$ .

### 4.3 Adaptive transition model

Instead of using a fixed function to predict the transition state from time  $(t - 1)$  to time  $t$ , we use the following adaptive transition model:

$$\mathbf{b}_t = \hat{\mathbf{b}}_{t-1} + \Delta \mathbf{b}_t + \mathbf{U}_t \quad (13)$$

where  $\Delta \mathbf{b}_t$  is the shift in the geometric parameters and  $\mathbf{U}_t$  is the random noise. Our basic idea allowing to recover the solution  $\mathbf{b}_t$  or equivalently the deterministic part of equation (13) is to use region-based registration techniques. In other words, the current input image  $\mathbf{y}_t$  is registered with the current appearance model  $A_t$ . For this purpose, we minimize an error measure between the warped texture and the current appearance mean,

$$\min_{\mathbf{b}_t} e(\mathbf{b}_t) = \min_{\mathbf{b}_t} d[\mathbf{x}(\mathbf{b}_t), \mu_t] = \sum_{i=1}^d \left( \frac{x_i - \mu_i}{\sigma_i} \right)^2 \quad (14)$$

Note the appearance parameters  $\mu_t$  and  $\sigma_t$  are known. The above criterion can be minimized using iterative first-order linear approximation.

#### *Gradient-descent registration*

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

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

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

$$\mathbf{x}(\mathbf{b}_t) \simeq \mathbf{x}(\hat{\mathbf{b}}_{t-1}) + \mathbf{G}_t \Delta \mathbf{b}_t$$

where  $\mathbf{G}_t$  is the gradient matrix, and  $\mathbf{G}_t^+$  its pseudo-inverse. By combining the above two equations we have:

$$\mu_t = \mathbf{x}(\hat{\mathbf{b}}_{t-1}) + \mathbf{G}_t \Delta \mathbf{b}_t$$

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

$$\Delta \mathbf{b}_t = \mathbf{b}_t - \hat{\mathbf{b}}_{t-1} = -\mathbf{G}_t^+ (\mathbf{x}(\hat{\mathbf{b}}_{t-1}) - \mu_t) \quad (15)$$

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  $(\mathbf{x}(\hat{\mathbf{b}}_{t-1}) - \mu_t)$  and the corresponding error measure  $e(\mathbf{b})$  (equation (14)). We find a shift  $\Delta \mathbf{b}$  by multiplying the error vector with the negative pseudo-inverse of the gradient matrix using (15).  $\Delta \mathbf{b}$  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:

$$\mathbf{b}' = \mathbf{b} + \theta \Delta \mathbf{b} \quad (16)$$

$$e' = e(\mathbf{b}') \quad (17)$$

where  $\theta$  is a positive real.

If  $e' < e$ ,  $\mathbf{b}$  is updated according to (16) and the process is iterated until convergence. If  $e' \geq e$ , smaller update steps are tested, using the same direction (*i.e.*, smaller  $\theta$  is used). Convergence is declared when the error cannot be improved anymore.

#### *Gradient matrix computation*

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, as explained in [5]. 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. The  $j^{th}$  column of  $\mathbf{G}$  ( $j = 1, \dots, \dim(\mathbf{b})$ ) is given by:

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

and is estimated using differences

$$\mathbf{G}_j \simeq \frac{\mathbf{x}(\mathbf{b}_t) - \mathbf{x}(\mathbf{b}_t + \delta \mathbf{q}_j)}{\delta}$$

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

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

where  $\delta_j$  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 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 for the 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 for some kinds of motions such as out-of-plane motions.

#### 4.4 Handling outliers and occlusions

We assume that occlusion and large image differences can be treated as outliers. Outlier pixels cannot be explained by the underlying process (the current appearance model  $A_t$ ) and their influences on the estimation process should be reduced. Robust statistics provide such mechanisms [8].

The mechanism will have impact on three items: (i) the likelihood measure, (ii) the gradient descent method, and (iii) the update of the online appearance model  $A_t$ .

Following the ideas developed in Zhou *et al.* [15], we use the Huber's cost function  $\rho$  defined as follows [8]:

$$\rho(x) = \begin{cases} \frac{1}{2} x^2 & \text{if } |x| \leq h \\ h|x| - \frac{1}{2} h^2 & \text{if } |x| > h \end{cases}$$

where  $x$  is the value of a pixel in the patch  $\mathbf{x}$ , normalized by the mean and the variance of the appearance at the same pixel, i.e.  $\mu_i$  and  $\sigma_i$ . This function is a hybrid between the L1 and least-squares function. It is continuous, with continuous first derivative. The cutoff threshold  $h$  controls the outlier rate. In our application, we take  $h = 3$  based on experimental experience.

### Likelihood measure

To make the likelihood measure robust, we replace the one-dimensional normal density  $\mathbf{N}(x; \mu_i, \sigma_i)$  by

$$\hat{\mathbf{N}}(x; \mu_i, \sigma_i) = \frac{1}{Z} \exp \left[ -\rho \left( \frac{x - \mu_i}{\sigma_i} \right) \right]$$

### Gradient method

To downweight the influence of the outlier pixels in the registration technique, we introduce a  $d \times d$  diagonal matrix  $L_t$  with its  $i^{\text{th}}$  diagonal element being  $L_t(i) = \eta(x_i)$  where  $x_i$  is the  $i^{\text{th}}$  element of the difference image  $(\mathbf{x}(\hat{\mathbf{b}}_{t-1}) - \mu_t)$  normalized by the corresponding variance  $\sigma_i$  and

$$\eta(x) = \frac{1}{x} \frac{d\rho(x)}{dx} = \begin{cases} 1 & \text{if } |x| \leq h \\ \frac{h}{|x|} & \text{if } |x| > h \end{cases}$$

$\eta(x)$  is used to attenuate the influence of the outliers. Therefore, the shift used in the gradient-descent registration becomes

$$\Delta \mathbf{b}_t = -\mathbf{G}_t^+ L_t (\mathbf{x}(\hat{\mathbf{b}}_{t-1}) - \mu_t) \quad (18)$$

### Appearance update

Once the solution  $\mathbf{b}_t$  is ready, the corresponding patch  $\mathbf{x}$  will be used to update the appearance. For non-outlier pixels the update equations are given by (11) and (12); for outlier pixels the corresponding means and variances are not updated. This mechanism is very useful for preventing occlusions from deteriorating the online appearance model.

## 4.5 The tracking algorithm

Tracking the 3D head pose and the facial animations is performed as follows. Starting from the solution,  $\hat{\mathbf{b}}_{t-1}$ , associated with the previous frame, we predict the state using (13) in which the deterministic part of the prediction, i.e.  $\hat{\mathbf{b}}_{t-1} + \Delta \mathbf{b}_t$ , is computed by the registration technique and the noise variance was set as a monotonically increasing function of the registration error obtained at convergence. Once a set of particles is obtained, the MAP of (9) is again chosen to be the solution of the current frame.

As can be seen, unlike the classical particle filtering, the propagation concerns the MAP solution only and not the whole particle set. It is worthwhile noting that although the solutions provided by the deterministic and stochastic parts of (13) have utilized the same observation model, there are some differences. The deterministic solution is obtained by a directed continuous search starting from the solution associated with the previous frame. The stochastic solution is obtained by diffusing the deterministic solution in order to obtain possible refinement.

#### 4.6 Experiments

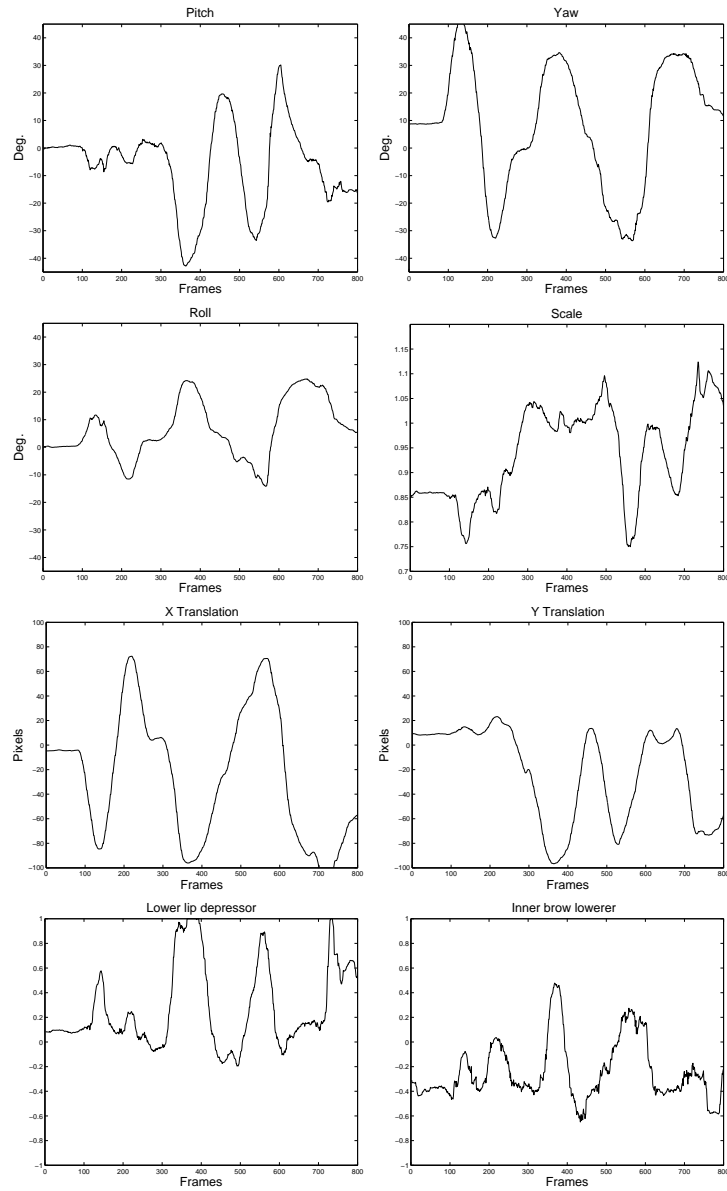
Figure 5 displays the head and facial animation tracking results associated with a 800 frame long (only four frames are shown). These results correspond to the online tracker based on appearance-adaptive model (described in section 4). The sequence features quite large head pose variations as well as large facial animations. The sequence is of resolution  $640 \times 480$  pixels. As can be seen with the very little prior information, the 3D motion of the face as well as the facial animations associated with the mouth and the eyebrows are accurately recovered. The upper left corner shows the current appearance  $\mu_t$  and the current shape-free texture  $\hat{x}_t$ .



**Fig. 5.** Our real-time framework for tracking the 3D head pose and the facial animations with an appearance-adaptive model. The sequence length is 800 frames.

Figure 6 displays the estimated values of the 3D head pose parameters (the three rotations and the three translations) as well as the lower lip depressor, and the inner brow lowerer as a function of the frames of the same sequence.

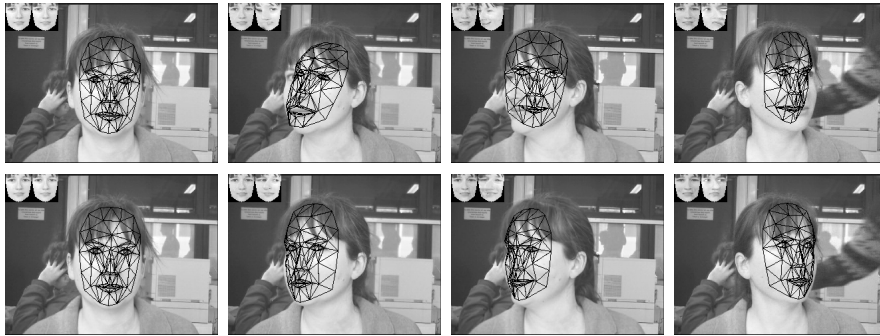
Figure 7 displays the face and facial animation results associated with a 602 frame long sequence. The first row displays the tracking results of the online tracker using the appearance-adaptive model with a fixed gradient matrix computed at the first video frame.



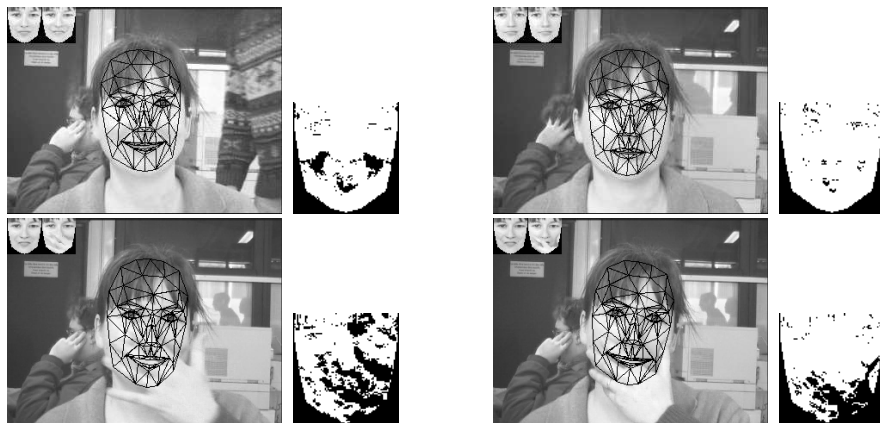
**Fig. 6.** The tracked parameters as a function of time associated with the 800 frame long sequence. The first six plots display the six degrees of freedom of the 3D head pose. The two bottom plots display the lower lip depressor and inner brow parameters, respectively.

The second row displays the tracking results when a time-varying gradient matrix has been used. We have noticed, for this sequence, that whenever

the face performs an out-of-plane motion the tracker with a time-varying gradient is more accurate than the one using a fixed gradient. Moreover, in other experiments, the tracker using a fixed gradient matrix has totally lost the track.



**Fig. 7.** From left to right: frames 94, 201, 376, and 569. The upper row displays the head and facial animation tracking using the appearance-adaptive framework with a fixed gradient. The lower row displays the tracking result when a time-varying gradient is used.



**Fig. 8.** Tracking another test sequence featuring two occlusions. Frames 36, 58, 218, and 265 are displayed. Frames 218 and 265 show respectively the start and the end of the second occlusion present in the video. The black and white patches show the corresponding shape-free map of the outlier pixels (shown in black).

Figure 8 displays the face and facial animation results associated with another 402 frame long sequence featuring two occlusions (only the second occlusion is displayed).



**Fig. 9.** Face and facial animation tracking results obtained with a 708 frame long sequence using the gradient-based registration method. At the end of the sequence the person is putting on his glasses. The two upper rows display tracking results obtained without any temporal downsampling. The two lower rows display tracking results obtained when every fourth image was used, *i.e.* the temporal downsampling is set to four.

The two occlusions are caused by putting the hand in front of the face. The frames 218 and 265 show the start and the end of the second occlusion, respectively. As can be seen on the the black and white patches of figure 8, pixels associated with the region of smiling and occlusions are considered as outliers.

On a 2 GHz PC, a non-optimized C code of the algorithm computes the adaptation parameters associated to one image in about 50 ms assuming that the patch resolution is 1310 pixels,  $K$  is eight. 80% of the CPU power is devoted to the gradient matrix computation.

In order to explore the behavior of the time-varying gradient based registration method in the presence of fast head movements and facial animations, we have conducted the following experiment. A video sequence was captured. This 708 frame long sequence features a bearded subject putting on his glasses. The sequence was tracked by the registration technique. Figure 9 displays the tracking results associated with four frames of the sequence. In order to simulate rapid movements, the sequence was subjected to a temporal downsampling factor of four. In other words, every fourth image of the original sequence was used. As can be seen, the tracking is still accurate and almost the same accurate tracking results were obtained in both cases.

## 5 Conclusion

In this chapter, we have proposed two tracking methods. The first method is fully stochastic and uses a particle filter with an observation likelihood based on statistical facial textures. This method has been utilized for 3D head tracking. The second method combines the merits of both stochastic and deterministic methods and is capable of tracking the head and facial animation. It employs an Online Appearance Model where both the observation and transition models are adaptive. The deterministic part exploits a directed continuous search aiming at minimizing the discrepancy between the upcoming observation and the current appearance model. Tracking long video sequences demonstrated the effectiveness of the developed methods. Accurate tracking was obtained even in the presence of perturbing factors such as illumination changes, significant head pose and facial expression variations as well as occlusions. Currently, we are investigating the recognition of facial expressions and gestures from the tracked parameters.

## References

1. J. Ahlberg. An active model for facial feature tracking. *EURASIP Journal on Applied Signal Processing*, 2002(6):566–571, June 2002.
2. S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *Int. Journal of Computer Vision*, 56(3):221–255, February 2004.

3. M. Black and A. Jepson. Eigen-tracking: Robust matching and tracking of articulated objects using a view-based representation. *Int. Journal of Computer Vision*, 36(2):101–130, 1998.
4. D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.
5. T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–684, 2001.
6. F. de la Torre, Y. Yacoob, and L. Davis. A probabilistic framework for rigid and non-rigid appearance based tracking and recognition. In *Proc. 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 491–498, 2000.
7. A. Doucet, J. F. G. De Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.
8. P.J. Huber. *Robust Statistics*. John Wiley and Sons, 1981.
9. M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *Int. Journal of Computer Vision*, 29(1):5–28, 1998.
10. 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.
11. B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 674–679, 1981.
12. B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710, 1997.
13. B. North, A. Blake, M. Isard, and J. Rittscher. Learning and classification of complex dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):1016–1034, 2000.
14. P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proc. Europ. Conf. Computer Vision*, pages 661–675, 2002.
15. S. Zhou, R. Chellappa, and B. Moghaddam. Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Trans. on Image Processing*, 13(11), November 2004.

---

## Index

- active appearance model, 2
- bayesian filtering, 3
- Candide model, 4
- condensation algorithm, 3, 8
- facial animation tracking, 9, 19
- facial animation units, 5
- facial shape units, 5
- facial state estimate, 6
- gradient-descent registration, 11
- head pose tracking, 6
- maximum a posteriori, 6
- numerical differences, 12
- observation likelihood, 7, 10
- occlusions, 10
- online appearance model, 3, 10, 11
- out-of-plane motions, 13
- particle filtering, 6, 10
- region-based registration, 11
- robust statistics, 13
- robustness, 10
- shape-free texture patch, 5
- state evolution model, 7
- texture modes, 7

