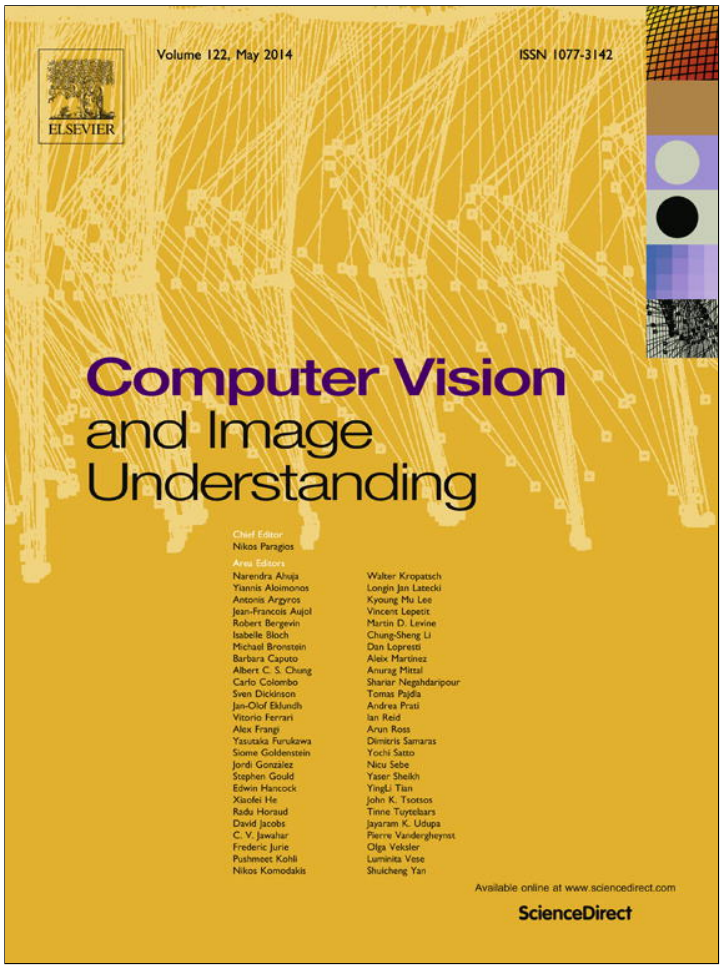


Computer Vision and Image Understanding 122 (2014) 182–201

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

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

Recursive head reconstruction from multi-view video sequences ☆

Catherine Herold [a,b,c,*], Vincent Despiegel [a,b], Stéphane Gentric [a,b], Séverine Dubuisson [d,1], Isabelle Bloch [a,c]

[a] Identity & Security Alliance, The Morpho and Télécom ParisTech Research Center, France
[b] Morpho, Safran Group, 11 boulevard Galliéni, Issy-les-Moulineaux, France
[c] Institut Mines-Télécom, Télécom ParisTech, CNRS LTCI, Paris, France
[d] Sorbonne Universités, UPMC Univ Paris 06, CNRS, UMR 7222, ISIR, F-75005, Paris, France

ARTICLE INFO

Article history:
Received 20 June 2013
Accepted 21 January 2014
Available online 30 January 2014

Keywords:
3DMM
Particle filter
Shape estimation
Facial biometry

ABSTRACT

Face reconstruction from images has been a core topic for the last decades, and is now involved in many applications such as identity verification or human–computer interaction. The 3D Morphable Model introduced by Blanz and Vetter has been widely used to this end, because its specific 3D modeling offers robustness to pose variation and adaptability to the specificities of each face.

To overcome the limitations of methods using a single image, and since video has become more and more affordable, we propose a new method which exploits video sequences to consolidate the 3D head shape estimation using successive frames. Based on particle filtering, our algorithm updates the model estimation at each instant and it is robust to noisy observations. A comparison with the Levenberg–Marquardt global optimization approach on various sets of data shows visual improvements both on pose and shape estimation. Biometric performances confirm this trend with a mean reduction of 10% in terms of False Rejection Rate.

1. Introduction

The recent rise of biometric techniques stimulates their use to automate the process of people recognition in a wide variety of systems, from computer locking devices to people authentication in airports. For each application, a compromise has to be found between the recognition rate of the biometric system on the one hand, and its easiness of use, cost and computation time on the other hand. The different types of biometric identifiers used for human recognition (fingerprints, iris, face, veins, etc.) have different requirements in terms of acquisition and do not lead to the same recognition accuracy.

Among all of them, facial biometry offers the advantage of being easily acquired without any contact with sensors, but suffers from specific issues of acquisition conditions (illumination, pose, facial expression). This is especially the case in video surveillance or in recognition systems designed to avoid behavior constraints in order to simplify the process from the user point of view. As such systems are not intrusive for users and due to the easiness of face acquisition, specific work has focused on face reconstruction and comparison methods. Moreover, facial biometry is sometimes the only biometric identifier available. To solve the different problems outlined above, the field of face recognition has been an active research area for many years, first on still images [30,9,5], then on video [23]. This extension is particularly interesting since video-based systems become more and more affordable, and have the advantage of increasing available observations. When people move about in uncontrolled scenarios, the information from a face observed under different poses in the sequence can be merged, and is then compared to a reference picture. Among existing face recognition algorithms, a number of methods are based on the comparison of frontal views (the reference view is generally the frontal picture on ID documents). A frontal view therefore has to be generated from the acquisitions. This can be performed via a 3D reconstruction of the face using the acquired images, from which synthesized views at any pose can be derived. Given the specificity of the face reconstruction problem (as opposed to object reconstruction without prior knowledge), model-based methods are privileged as they limit the risk of aberrant reconstruction, achieving a compromise between the information coming from the observations and the prior knowledge on the class of faces.

Most existing algorithms designed to estimate parameters of such 3D models are based on a single image input and highly depend on the quality of the observations [5,27]. Nevertheless, in order to obtain more accurate results, it is interesting to use several images to consolidate the reconstruction. In [2], the authors proposed to fuse images based on stereovision. The use of video

☆ This paper has been recommended for acceptance by Kevin W. Bowyer, Ph.D.

* Corresponding author at: Morpho, Safran Group, 11 boulevard Galliéni, Issy-les-Moulineaux, France.
E-mail address: catherine.herold@morpho.com (C. Herold).

[1] Part of this work has been done as S. Dubuisson was at LIP6 laboratory, Université Pierre et Marie Curie, 4 place Jussieu, Paris, France. C. Herold is also associated with this laboratory.

http://dx.doi.org/10.1016/j.cviu.2014.01.006
1077-3142/

sequences has not been widely exploited, except for structure from motion methods, where images are considered as an ensemble to estimate the model parameters [10]. In [32], the authors extend a single image based method to video sequences by fusing the estimations obtained at each instant independently, without verifying the model coherence. However, temporal constraints between states estimated at successive instants are not integrated in the process, which would improve results.

To propose a real-time working system, we have to exploit the incoming video frames on the fly. To this end, we propose a new method based on the update of a 3D head model by using a particle filter framework, which extends the work in [13], and has, to our knowledge, never been proposed. An important feature of the proposed approach is that previous observations are implicitly taken into account to estimate the model at the current instant. The key of our algorithm is to integrate the unknown shape coefficients in the particle state and to consider them as static parameters, unlike the pose which varies over time. Besides an adaptation to real data, we propose here an improved algorithm for face estimation, robust to noisy or aberrant detections thanks to multiple hypotheses handling, contrary to common gradient methods which optimize a unique solution associated with a given set of observations.

In Section 2, we first present the chosen head model, before giving an overview of methods which estimate the associated parameters, both for single and multiple input images. In Section 3, we detail how to adapt a particle filter method to handle static parameters for facial shape estimation in video sequences, and propose some alternatives to improve this static parameter estimation. Section 4 presents how the observations are exploited in the particle filter and used to generate the frontal view. Section 5 details a method which is compared to our particle filter-based method in Section 6. This alternative method is based on a Levenberg–Marquardt optimization to estimate the pose and the shape. Experiments are done on both synthetic and real data. They are first analyzed on visual illustrations, to demonstrate the improvements at the image level. Then, since our final goal is to improve facial recognition performances by improving the head reconstruction using video sequences, an evaluation based on biometric performances is also proposed, before concluding with the perspectives of our method.

2. State of art: 3D face reconstruction

The method we propose for face reconstruction from video sequences relies on a head model which is described in this section. We will then present the existing methods to estimate its parameters.

2.1. 3D head reconstruction

As underlined previously, many facial biometric systems must be able to work with unconstrained user behavior, which implies handling non-frontal poses in the input images.

Since most recognition algorithms are based on the comparison between frontal views, a frontal view has to be generated from the acquisitions. This can be performed via a 3D reconstruction of the face from which images at any pose can be derived.

There are many ways to reconstruct a 3D object from a set of views. We can distinguish purely geometric approaches, which can be applied to any object, from model-based methods, which use some prior information on the object to reconstruct, and are therefore specific to a class of objects. Among generic methods based on one or a set of views, the best known algorithms are based on shape from shading [35], structure from motion [35] or stereovision [6,18,4]. For the latter, an important constraint is to perform point matching between images which therefore need to be acquired under quite similar points of view. Reconstruction algorithms can also exploit other devices like 3D-scans, depth sensors [36] or structured light projectors [34]. Here, we limit our approach to image-based methods.

There exist several solutions to reconstruct a 3D object from a set of views, and we chose model-based methods to exploit prior knowledge on the object, here the face. The contribution of the prior is twofold: first, it can be used to initialize a solution corresponding to a valid shape (for instance, a mean model), then, it prevents the algorithm from delivering a solution which does not belong to the face space.

2.2. 3D head model

We use a 3D deformable shape model constructed in a similar way as the *3D Morphable Model* (*3DMM*) introduced in [5]. This model has been chosen for several reasons, the first being its 3D modeling (by opposition to 2D), necessary when faces under any pose are considered (2D models learned on frontal views indeed cannot be used with non-frontal face inputs). Moreover, as the final aim is to establish a comparison score between the frontal view of the estimated face and its corresponding ID picture, it is necessary to adapt the model so that it fits the observed identity as well as possible. The *3DMM* allows for this adjustment, as it describes the deformations of a mean face on two levels:

- The shape space, characterized by a mean shape $\overline{S}$ and a set of deformations $\{s_i, i = 1, \ldots, M\}$ computed by principal component analysis over a database of aligned head scans. Each instance of this model can then be written as:

$$S = \kappa\left(\overline{S} + \sum_{i=1}^{M} \alpha_i s_i\right) \tag{1}$$

  where $\{\alpha_i, i = 1, \ldots, M\}$ are the weighting parameters which characterize the similarity with the mean shape and κ is a scaling factor. The mean shape $\overline{S}$ is defined by a set of n_v 3D vertices, and each vector s_i corresponds to deformations associated with this set of points. A mesh is then defined from these vertices by adding facets to describe the entire head surface.
- The texture, that associates a color with each vertex of the mesh, is stored in a texture map (Fig. 1) independently of the shape parameters.
  Like the shape space, the texture space can be described by a set of texture eigenvectors $\{t_i, i = 1, \ldots, M'\}$ and a mean texture map $\overline{T}$. Any instance of the model is then a linear combination of these vectors, so that:

$$T = \overline{T} + \sum_{i=1}^{M'} \beta_i t_i \tag{2}$$

  where M' is the number of texture eigenvectors, and $\{\beta_i, i = 1, \ldots, M'\}$ are the weighting parameters for the texture similar to $\{\alpha_i\}$ in Eq. (1) for the shape.

A probability is associated with each shape parameter value, as follows:

$$p(\alpha_i) \sim e^{-\frac{1}{2}\frac{\alpha_i^2}{\sigma_i^2}} \tag{3}$$

with σ_i the ith eigenvalue of the shape covariance matrix. A similar probability can be written for the texture parameters.

In this article, the parameter estimation focuses on the geometrical part of the model, and the texture is then estimated in a second step. Some instances of the morphable shape model are given

Fig. 1. Texture map examples (left and middle images). In this representation, each vertex of the shape model has a given 2D position (parametrized between 0 and 1) in the texture map. There is a mapping between the full 3D model and the texture map registered in the 2D image (right image), which explains the deformations observed in the texture images.

in Fig. 2 illustrating its variations depending on the parameter values.

2.3. Reconstruction based on a single image

A first algorithm to estimate the shape and texture parameters of the *3DMM* was proposed in the seminal paper [5]. This algorithm is based on the optimization of a similarity score between the input image and a rendered view synthesized given the estimated pose, shape and texture. In case of perfect fitting, the input image and the rendered one should be exactly the same. The optimization is performed using stochastic gradient descent, in order to speed up the process and to avoid local minima.

In [26], the authors introduced a faster method, based on *Analysis-by-Synthesis* as previously mentioned. The difference image (between the input image and the synthesized one) is expressed as a function of different derivative terms with respect to the unknown variables to estimate (pose, texture and shape parameters, and illumination). Because this equation is linear when some of the parameters are fixed, the proposed method first optimizes iteratively the rigid transformation, then the shape, the illumination, and finally the texture parameters. Thus, dimensions are reduced for each of these optimization steps, and most of them are linear (when estimating one set of parameters, the others are fixed). Similar results are obtained in [5], but with a computation time divided by five.

In [27], the authors add features to the cost function in order to expand the convexity domain around the optimum, and thus limit the problem of local minima. Moreover, the estimation accuracy is improved as the use of multiple features leads to a better fit between the model and the observations.

Nevertheless, even if the model fits the observations on a single image well, the 3D fitting is not guaranteed. Actually, due to the projection from the 3D world onto the image plane and the occlusions of some parts of a face in an image, some information is missing and therefore the estimation might be erroneous. This is why new algorithms based on multiple image fitting have been proposed. We present them briefly in the next subsection.

2.4. Multiview and temporal head reconstruction

In [2], the fitting algorithm proposed in [27] was adapted to use a set of images acquired simultaneously. Besides the estimation of the pose and the model parameters, the authors also estimate the

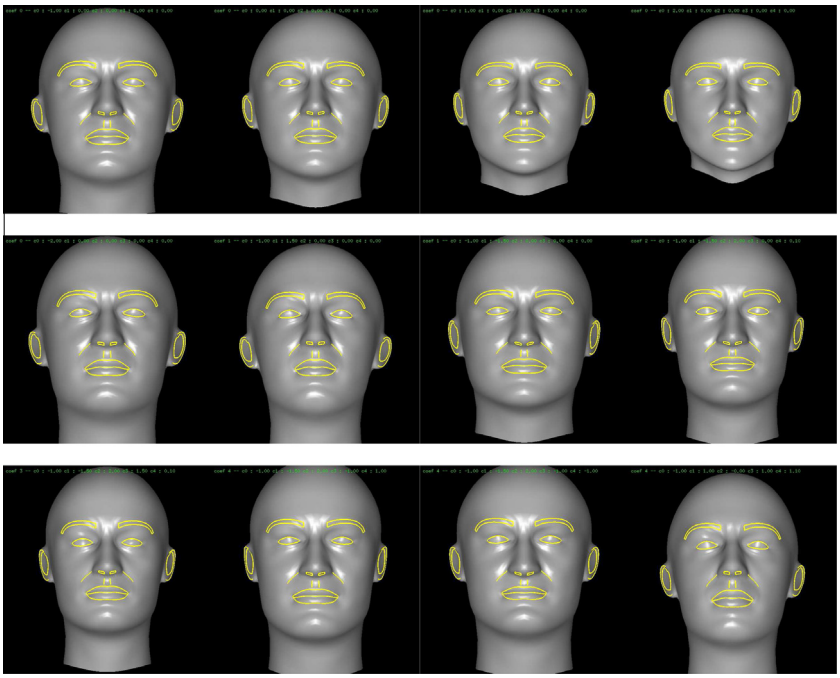


Fig. 2. Some instances of the deformable shape model (all faces are synthesized at the same pose and scale, and with identical lighting conditions). On the first line, only the first coefficient α_1 is non-zero and varies from -1.0 to 2.0 (from left to right). On the other lines, the five first coefficients (α_1 to α_5 in Eq. (1)) have been sampled between -2.0 and 2.0. We can notice that the global shape changes for each instance, and more specifically the mouth shape, the ear orientation or the width of the face.

camera calibration parameters. This method improves the results of algorithms relying only on a single image, but as for previous methods, noisy observations can lead to an inaccurate estimation. When facing non-frontal poses, the extraction of features such as points of interest (eye corners, ears, etc.) might be difficult. Moreover, when using gradient based optimization methods, the final estimation highly depends on the initialization: if the starting point is too far from the real value, the solution can be stuck in a local minimum.

To fit a head model to a face seen in a video sequence, some methods simply apply an additional selection step over all available images to extract the best one according to some criteria. One of the fitting methods introduced previously can then be applied on the single chosen image [7]. In [32], two experiments were proposed; the first one consists in estimating the parameters using each frame independently before making a fusion by a linear combination of these estimations. The second one uses all input images together to optimize parameters, leading to a single estimation based on the whole sequence.

The advantage of using stereovision or video sequences is that it guarantees a better 3D estimation as the model is fitted on observations under various poses. Moreover, there is more robustness to outliers or noisy detections, because one point being badly detected on one image may be correctly detected on other ones. If the feature points used for the reconstruction are well detected in most of the images, the estimation will lead to a fitting towards the good detections. Moreover, using stereovision or video sequences allows consolidating the estimation between views under various poses. Nevertheless, no specific method has been proposed to sequentially update the model using a video sequence. Indeed, stereovision is often based on images acquired simultaneously by a set of cameras, and the video-based method proposed in [32] is applied offline. We present here a new video-based approach, using sequences acquired from a set of calibrated cameras. Unlike the previous methods, we propose updating the parameter estimation online, with each new incoming observation. Thus, at each instant, we obtain an estimation built on all previous views, which can be exploited before the end of the acquisition.

3. Static shape parameter estimation by particle filtering.

In this paper, our goal is to estimate the parameters of the shape model introduced in Section 2.2. The methods which have been presented previously iteratively update an initial estimate, and the output is a unique instance of the morphable model. Unlike these types of algorithms, we propose here representing the previous estimation as a density, which characterizes the probability of realization over the whole shape space. This allows us to cope with the inherent nature of noisy data and to maintain multiple hypotheses during the estimation process, that are reinforced or eliminated with new frames.

We rely on the Gaussian assumption made in [5] to define the prior distribution of the model shape parameters. This initial density is then updated each time new information is available. Given a new frame (or a set of frames when multiple views are available), our goal is to update the previous distribution characterizing the prior constraint and the past observations by taking the current ones into account. This can be done using a Bayesian approach, which allows a compromise between the parameter validity and the correlation with the observations.

Several declinations of the Bayesian theory for sequential updating can be cited here, such as the Kalman filter [16], the extended Kalman filter [28], the Unscented Kalman [15] and the particle filter [8]. Due to the non-linearity of the involved functions (perspective projections, projection of a 3D object leading to partial occlusions) and the multi-modal distributions we handle, we choose to work with particle filters. As developed in Section 4.1, the particle filter offers a structure that maintains multiple hypotheses over time, which is useful when feature extraction is difficult and leads to outliers.

3.1. Particle filter

In this work, we use a particle filter to jointly estimate the pose and the facial shape parameters over a video sequence. In the experiments, the camera extrinsic and intrinsic parameters are known, and not subject to the estimation. We first describe how to estimate dynamical states (position, pose, etc.) and then discuss how to integrate and estimate static parameters in the filter.

The particle filter algorithm [3,8] aims at filtering a hidden state x by representing it as a density updated at each instant t given the observations y_t. At each time t, the hidden state x_t can be derived from the previous state x_{t-1} given an evolution equation:

$$x_t = g(x_{t-1}) + \mu_t \tag{4}$$

with g a (possibly non-linear) function characterizing the system dynamics, and μ_t the associated noise. The observations at time t are derived from the corresponding state x_t according to the following equation:

$$y_t = h(x_t) + \eta_t \tag{5}$$

with h the (possibly non-linear) observation function and η_t the noise associated with the observations. The set of observations from instant 1 to t is denoted by $y_{1:t}$.

The goal is to estimate at each time t the filtering density $p(x_t|y_{1:t})$ by a set of N elements $x_t^{(i)}$ called particles representing possible states at time t. Each particle $x_t^{(i)}$ is associated with a normalized weight $w_t^{(i)}$ which characterizes its likelihood. The density is approximated using the Monte-Carlo method given the set of N particles $\mathcal{P} = \left\{(x_t^{(i)}, w_t^{(i)}),\ i = 1, \ldots, N\right\}$:

$$\tilde{p}(x_t|y_{1:t}) = \sum_{i=1}^{N} w_t^{(i)} \delta_{x_t^{(i)}}(x_t)$$

where $\delta_{x_t^{(i)}}(x_t) = 1$ if $x_t = x_t^{(i)}$, 0 elsewhere.

Given the initial particle distribution $\{(x_0^{(i)}, w_0^{(i)} = 1/N),\ i = 1, \ldots, N\}$, the following steps will be repeated to estimate the *a posteriori* density at each instant:

- **Prediction**: each particle $x_{t-1}^{(i)}$ moves from time $t-1$ to time t given the probability $p(x_t|x_{t-1})$ associated with the system dynamics (Eq. (4)).
- **Correction**: each weight $w_t^{(i)}$ is then updated according to the particle likelihood given the current observations $p(y_t|x_t^{(i)})$:

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(y_t|x_t^{(i)}). \tag{6}$$

  This likelihood score defines how probable the state represented by each particle is, taking current observations and model knowledge into account.
- Finally, a **resampling** is performed if the particle weights are too scattered. This is characterized by the Effective Sample Size (ESS) approximated by $\frac{1}{\sum_{i=1}^{N}\left(w_t^{(i)}\right)^2}$.

3.2. Static parameter estimation by particle filter

Let us now consider the case of filtering with unknown static parameters. This is especially useful when some parameters of the object to track are unknown, and when their values impact the evolution and observation functions. For instance, if we

consider the size of the object as a parameter, it is clear that it will influence the observation function, as, the bigger it is, the larger its projection in the image for a given pose. To simplify the notations, we consider the concatenation of all unknown parameters in a vector θ (here, the shape parameters α_i and the scale factor κ). As previously mentioned, the aim is to estimate sequentially the dynamic state x_t (the pose) given the observations ($y_{1:t}$). However, we now take into account the unknown parameters in the evolution and observation equations. The parameter θ is a characteristic of the object and is supposed to be constant over time, but as it is unknown, it has to be estimated. We propose updating this estimation along the sequence, and we note θ_t the current estimation at time t. With these notations, the previous system can be rewritten as follows:

$$x_t = g(x_{t-1}, \theta_{t-1}, \mu_t) = g_{\theta_{t-1}}(x_{t-1}, \mu_t) \tag{7a}$$

$$y_t = h(x_t, \theta_t, \eta_t) = h_{\theta_t}(x_t, \eta_t) \tag{7b}$$

The functions g and h now depend on θ, meaning that the hidden state evolution depends on the parameter values, as do the observations given a state. In addition to the state estimation, a second task is to determine the static parameter vector θ.

In [17], the authors reviewed existing methods to deal with unknown parameters in particle filter algorithms. We do not hereby consider offline methods, which handle a set of observations and optimize the unknown static parameters and the poses globally. Instead, we focus on online methods which update the pose and parameter estimation recursively given the incoming observations. Thus, we can have the best estimation available at each instant, since it is computed from the previous estimations updated with the current observations. If the last computed estimation is good enough (for instance if it fits well with the observations), there is no need to continue the process. Otherwise, the last estimation is used for further improvement using new frames.

Let $p(\theta)$ be the prior distribution of the unknown static parameters. Our algorithm aims at estimating iteratively the vector θ_t^*, corresponding to the shape estimation at time t, and the associated pose x_t. The probability of θ_t at time t given the observations can be obtained by integrating over all possible values of the hidden state $x_{1:t}$:

$$p(\theta_t|y_{1:t}) = \int_{\mathcal{X}} p(x_{1:t}, \theta_t|y_{1:t}) dx_{1:t} \tag{8}$$

where $\mathcal{X}$ is the hidden dynamic state space. To estimate the value of θ iteratively, this static parameter can be integrated into the hidden state, thus increasing the size of the particle state [29,22]. The joint density $p(\theta_t, x_{1:t}|y_{1:t})$ can then be evaluated using Monte-Carlo methods.

Recall that $x \in \mathcal{X}$ is the dynamic state of dimension n_x, and $\theta \in \Theta$ the vector of static parameters of dimension n_θ. The complete vector to estimate is then $\{x, \theta\} \in \mathcal{X} \times \Theta$, with dimension $n_x + n_\theta$. Each particle will then be represented by a dynamic state part $x_t^{(i)}$ and a static state part $\theta_t^{(i)}$.

The integration of static parameters into the hidden state requires the application of an artificial move on the particle static part in order to explore the associated space. By definition, since the parameters are static, the dynamic function which determines their evolution in time is the identity function. In this case, the parameters do not change, and the only values tested over the sequence are the ones sampled at the particle initialization step, as they are not modified afterward. After some resampling steps, a few or even only one values will still be represented, and this impoverishment leads to a wrong estimation of the parameter values. A workaround to this issue is to apply an artificial move on the static parameters from frame to frame. Different types of moves which are listed below have already been proposed. We also propose two variants that improve the estimation process.

Gaussian noise. A first method consists in considering the static parameters as dynamic ones, as in [22]. Therefore, at each time t, a Gaussian noise is added to the perturbation, which can be considered as an artificial evolution:

$$\theta_{t+1} = \theta_t + \varepsilon_{t+1} \tag{9a}$$

$$\varepsilon_{t+1} \sim N(0, W) \tag{9b}$$

with W a covariance matrix characterizing the noise to add. Thanks to this step, diversification is introduced for the static part of the particle filter, and other states, than the ones initially sampled, can be evaluated.

Adaptive Gaussian noise depending on particle weight. With the previous move based on Gaussian noise addition, there is a loss of precision after the alteration, since good particles can be moved far away from their initial good position. To limit this effect, we propose making the covariance matrix W dependent on each particle weight $w_t^{(i)}$. If the weight is high (meaning that the particle is in agreement with the observations), we use a covariance matrix with small values to make a local exploration of the space. Conversely, if the weight is low, a noise with a higher covariance will be applied to the static parameters in order to move the particle in other subspaces.

Resample-move algorithm. In [11], each move applied on the static state is generated by a MCMC step (Monte Carlo Markov Chain). With this method, the new static states which are only generated are accepted under some likelihood conditions. This idea, called *Resample-Move algorithm*, has been introduced in [12]. The method works as follows: at each time t, a MCMC move is applied to each particle. This move is generated by a kernel $K_t(x'_{1:t}, \theta'|x_{1:t}, \theta)$ having $p(x_{1:t}, \theta|y_{1:t})$ as invariant distribution. The move can be limited to the static state θ, and can be obtained by the Metropolis–Hastings algorithm, computed in two steps:

- sample a new candidate for the static parameter: $\theta' \sim p(\theta'|\theta)$,
- sample $v \sim \mathcal{U}_{[0,1]}$, a random sample from the uniform distribution between 0 and 1. If

$$v \leqslant \min\left(1, \frac{p(y_{1:t}|x_{1:t}, \theta')}{p(y_{1:t}|x_{1:t}, \theta)}\right) \tag{10}$$

the move from θ to θ' is accepted. Otherwise, θ is kept.

By directly applying this formula, the computational cost is increased at each timestep, as more frames have to be considered for the likelihood computation. Indeed, as the static parameters are modified between frames, all likelihood values with respect to previous observations need to be recomputed. To limit this effect, we introduce a period ΔT, which defines the number of frames to be taken into account for the MCMC validation. The move is then accepted if:

$$v \leqslant \min\left(1, \frac{p(y_{t-\Delta T:t}|x_{t-\Delta T:t}, \theta')}{p(y_{t-\Delta T:t}|x_{t-\Delta T:t}, \theta)}\right) \tag{11}$$

with $\Delta T = 0$ if only the current observations are used.

For each particle, the sampling step

$$\theta_t'^{(i)} \sim p\left(\theta_t^{(i)}, x_{1:t}^{(i)}\right) \tag{12}$$

allows for a local diversification of the static parameters only. Like systematic Gaussian noise addition, we propose to evaluate another sampling step, based on the prior distribution of the static parameters, and independently of the particle current shape parameters $\theta_t^{(i)}$. Thus, new subspaces can be explored and moves are allowed

anywhere, as long as the acceptance condition in Eq. (10) is satisfied. With this method, it becomes possible to get a particle out of a local maximum. Nevertheless, if the space dimension is too high with very few areas with high likelihood, the probability to accept a move will be very low.

3.3. Application to head shape parameter estimation

Particle filters that include unknown static parameters have been detailed from a theoretical point of view this past decade but very few applications have been proposed in the field of image processing. One use of such methods was nevertheless given in [22], where the authors evaluated dimensions of simple geometrical objects in video sequences, by considering these values as unknown static parameters. The relation between the unknown dimensions and the observations remains simple, and there is no correlation between the different static values.

In this paper, we want to estimate the unknown parameters of a more complex model, namely the shape coefficients α_i and the scale κ (Eq. (1)) of the shape model presented in Section 2.2. Let us underline that the observations are highly dependent on the shape parameters, and this relation will be exploited to estimate them using the available observations. Pose and shape parameters must be estimated jointly, otherwise a pose error will be compensated by a parameter error and conversely. As illustrated in Fig. 2, due to the construction of our shape model, each parameter α_i in Eq. (1) modifies the whole face because each associated deformation eigenvector impacts all vertices. The process leading from the shape parameters to the observed image is as follows:

1. model deformation given $(\alpha_1, \ldots, \alpha_M)$ and the scale κ,
2. rotation and translation in the 3D world reference,
3. projection onto the image plane.

The dependence on θ needs therefore to be considered when using the observation function, due to the shape parameter impact on the 3D position of each point. The global system to consider is the following:

$$x_t = g(x_{t-1}, \mu_t) \tag{13a}$$
$$y_t = h(x_t, \theta, \eta_t) = h_\theta(x_t, \eta_t) \tag{13b}$$

For our study, instead of applying the usual prediction process of the particle filter (Eq. (13a)), we favor the use of feature points detected in the current frame to initialize the pose. Indeed, to handle low framerates, a very high number of particles would be necessary to cover the space of all possible poses, and only a few of them would be relevant. To avoid this step, we compute the particle poses directly given the current feature point detections. To this end, an initial pose x_t^0 is estimated by fitting the mean model using the method presented in [31]. This algorithm computes the translation, the rotation and the scale which minimize the least mean square error between two sets of 3D feature points. This is done by computing the Singular Value Decomposition (SVD) of the covariance matrix of these two sets of points, which is then used to determine the pose x_t^0. A pose $x_t^{(i)}$ is then sampled around x_t^0 for each particle, using a Gaussian noise (Eq. (14)).

Algorithm 1 presents the particle filter with two exclusive possibilities of move applied to the static parameters. In the case of Gaussian sampling, a move is automatically applied to each unknown parameter. When using MCMC, the move is applied conditionally to the gain in terms of likelihood between the previous and the new sampled states (Eq. (10)).

Algorithm 1. Static shape parameter estimation with a particle filter

Sample the shape parameters θ from a prior Gaussian distribution to initialize the set of particles $\left\{\theta_0^{(i)}, w_0^{(i)} = 1/N),\ i = 1, \ldots, N\right\}$
Define the move to apply: $Move = Gaussian\_Sampling$ or $MCMC$
for $t = 1 \rightarrow N_{frames}$ **do**
  Input: 2D feature point positions (possibly noisy).
  Mean shape model fitting to estimate the initial pose x_t^0 using the method by [31].
  for $i = 1 \rightarrow N$ **do**
    – Sample around the estimated pose:

$$x_t^{(i)} = x_t^0 + n_x, \quad \text{with } n_x \sim N(0, \Sigma). \tag{14}$$

    if $(Move = Gaussian\_Sampling)$ **then**
      Sample around the previous shape parameters: $\theta_t^{(i)} = \theta_{t-1}^{(i)} + n_\theta$, with $n_\theta \sim N\left(0, \frac{1}{w^{(i)}}\right)$.
    end if
    – Update the weight with the likelihood $p\left(y_t | x_t^{(i)}, \theta_t^{(i)}\right) : w_t^{(i)} \propto w_{t-1}^{(i)} p\left(y_t | x_t^{(i)}, \theta_t^{(i)}\right)$, as in Eq. (6), but taking the parameter values into account.
  end for
  Resampling
  if $(Move = MCMC)$ **then**
    for $i = 1 \rightarrow N$ **do**
      Apply a MCMC move
    end for
  end if
end for

4. 3D face reconstruction in videos

In our application, each particle state is decomposed into a dynamic part (the pose x_t) and a static part (the scale κ and shape parameters α_i, such that $\theta_t = \{\kappa, \alpha_1, \ldots, \alpha_M\}$) and must be updated and evaluated with the incoming observations. In this part, we detail how we use the images for these steps, and introduce a new way to handle noisy observations based on the particle filter structure. Then, we present the texture extraction process once the shape evaluation is done.

4.1. Multi-hypotheses feature point detections

At each time t, the particle weights are updated by computing the likelihood of their states with the current images. This can be done with commonly used criteria, such as edge or silhouette scores [22], which compare the edges in the input images and the ones of the projected object, or feature point projection, to verify their proximity to the detections [25]. These feature points are also used to initialize the pose as shown in Algorithm 1, by fitting the mean model on these points. As we have to handle non-frontal views, noisy or outlier detections can happen, which impact this initial pose estimation. If this pose is wrong, all particles will be badly initialized (Eq. (14)), and further shape estimation and pose improvement will then not be possible. In this part, we focus on the first pose estimation performed at each instant, and propose to exploit the multi-hypotheses structure of the particle filter to improve this step.

Different types of detectors can be learned to detect specific feature points of the face, using approaches such as AdaBoost, RealBoost and SVM-learning or Bayesian networks, which are detailed in [33]. Unfortunately, when faces vary depending on pose and acquisition conditions in the videos, it is a challenge to perfectly detect all facial feature points of a face. Depending on the video properties, some points can be missing, while others can be badly detected, as shown in Fig. 3. It is therefore important to take this uncertainty into account in order to handle outliers obtained with the feature point detectors.

In Algorithm 1, the pose for each particle is computed by adding some noise to the pose estimated from the set of detected feature points. Instead of using the same noisy pose version for all particles, we propose to exploit the property of multiple hypotheses representation of the particle filter by assigning different poses to particles, which are computed from different sets of feature points. Thus, we capitalize on the multi-hypothesis aspect of the particle filter to manage the outliers obtained with the feature point detectors, since some of the computed poses will be close to the correct one even if bad detections have been found (as all points are not necessarily used for this pose computation). More precisely, given a new frame, the pose of each particle is then computed as follows:

1. Select a set of valid points common to all particles:
   (a) For each feature point detected in all views of a given timestamp, compute its 3D coordinates and consider it as valid if the geometric reconstruction error considering the camera calibration parameters is below a given threshold σ_c. The outcome of this first step is a set of 3D feature points of the face.
   (b) Compute the pose parameters by fitting the mean 3D model on these reconstructed 3D points. This is done following the method in [31], combined with a RANSAC procedure to eliminate wrong feature points. The aim of this step is to find the 3D pose minimizing the distance between the 3D points selected at step (a) and the corresponding points in the 3D model. This distance is called geometric reconstruction error.
2. For each particle:
   (a) For each feature point not used previously, sample it given a probability p^D related to its detection confidence, and reconstruct the 3D point with the selected detections. Here again, the 3D point is kept only if the geometric reconstruction error is below a threshold σ_c.
   (b) Compute the pose parameters by fitting the deformed model (given the particle shape parameters) to the valid 3D points which have been reconstructed. No RANSAC algorithm is used at this step, as we want to see whether a pose is in accordance with the selected feature points.

The distribution p^D can be learned for each detector D given its outputs over an annotated database. To estimate it, we construct a histogram for each detector over the interval of its responses, characterizing the rate: (Number of good detections/Number of detections) for each bin. These histograms are then approximated by a density function in the form of a sigmoid function, characterizing the detector performances. Fig. 4 shows this rate given the detector output (horizontal axis) and the estimated sigmoid which has been fitted on these data. We can note that the curves differ from one detector to another, due to their different discriminative power (left eye and right eyebrow corner). A detection o^D associated with a confidence c is then kept if $p^D(c) > u, u \sim U[0,1]$, where u is sampled for each particle. Fig. 5 shows the points selected with this sampling method for a set of 10 particles.

The advantage of this method is twofold. First, unlike the pose sampling proposed in Eq. (14) of Algorithm 1, no noise is added to an initial pose estimated with the mean model. Here, each particle pose is optimized following the method in [31] using its own shape parameters and the observations.

Secondly, instead of using the same set of feature points for all particles based on a binary decision with respect to its confidence and a fixed threshold, this method adds diversity to the sets of points. Good points having an average confidence may still be selected for some particles, and conversely, noisy detections or outliers with good confidence may also be rejected. For instance, in Fig. 5, we can see that the confidence of the right ear detection is not high enough to pass above the previously fixed threshold. Nevertheless, it is sampled for some particles, and is then used to estimate the pose. Other criteria, like edge comparison, will then

Fig. 3. Left ear detections. In green: detection confidence above a threshold σ_{ear} – in red: detection confidence below σ_{ear}. The value of σ_{ear} has been chosen such that the probability to have a good detection at this threshold is $p = 0.5$. As the detector is not perfect, bad detections can be associated with high detection scores, and conversely. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

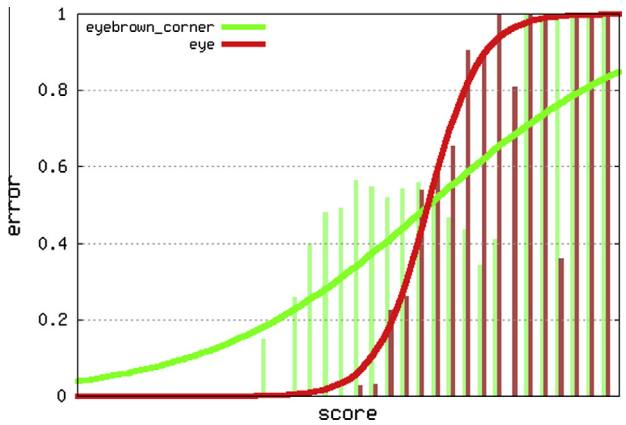


Fig. 4. Comparison of sigmoids fitted to the histograms representing the rate $\frac{\#\text{Good detections}}{\#\text{Total detections}}$ for two detectors. As semantic definitions of different feature points are not the same, their discriminating power is different, which explains the difference in terms of detection quality.

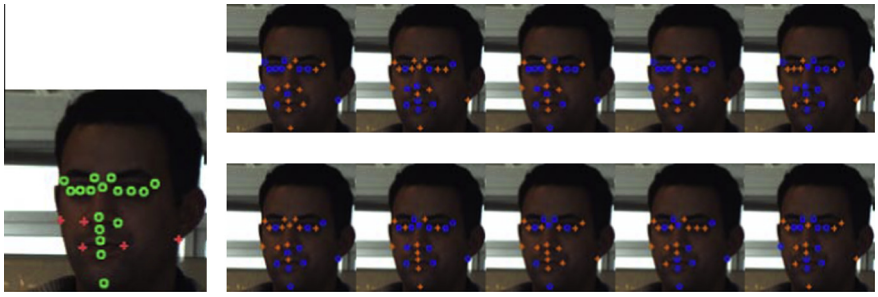


Fig. 5. Left image: initial detections. The red crosses (resp. the green circles) are detections with a confidence below 0.5 (resp. above 0.5). Right images: point sampling for ten particles. The orange crosses (resp. blue circles) are points which are rejected (resp. kept) for the current particle. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

differentiate the good particles from the bad ones, and assign higher weights to particles that selected the best set of feature points. Algorithm 2 summarizes the global workflow of this algorithm, taking the feature point sampling into account, unlike Algorithm 1. Only the systematic noise addition has been kept for the static parameter moves in this version.

Algorithm 2. Static shape parameter estimation with a particle filter and feature points management

Sample the shape parameters θ from a prior Gaussian distribution to initialize the set of particles $\left\{\left(\theta_0^{(i)}, w_0^{(i)} = 1/N\right),\ i = 1,\ldots,N\right\}$
for $t = 1 \rightarrow N_{frames}$ **do**
  Input: noisy 2D feature point positions.
  for $i = 1 \rightarrow N$ **do**
    – Sample around the previous shape parameters: $\theta_t^{(i)} = \theta_{t-1}^{(i)} + n_\theta$, with $n_\theta \sim N(0, \Sigma_\theta)$.
    – Sample a subset of feature points given their confidence and fit the pose of the current shape model using the method by [31].
    – Update the weight with the likelihood $p\left(y_t|x_t^{(i)}, \theta_t^{(i)}\right)$: $w_t^{(i)} \propto w_{t-1}^{(i)} p\left(y_t|x_t^{(i)}, \theta_t^{(i)}\right)$.
  end for
  Resampling
end for

4.2. Texture extraction and frontal view generation

For each frame and each camera, a texture map is extracted from the corresponding image to get color information on the face. For each pixel, a score is associated with this color, in order to define its quality. As the texture of the face is best viewed in a frontal view with respect to the camera optical axis, the quality is defined using a criterion expressing how frontal each facet of the model mesh is. Thus, for each pixel (x,y) in the texture map which is visible in the input image, its quality $qual(x,y)$ is computed as:

$$qual(x,y) = \left|\vec{n}_{f(x,y)} \cdot \vec{z}\right| \tag{15}$$

where $\vec{n}_{f(x,y)}$ is the normal to the face represented at pixel (x,y) and $\vec{z}$ the direction vector of the camera optical axis.

To obtain the most complete texture map, an intermediate step of texture map fusion has to be performed. We use a linear combination of the texture maps TM_i extracted from different views v $(v = 1,\ldots,V)$ at time t and weighted by the criterion presented above. Each pixel $TM(x,y)$ of the resulting texture map is computed from the corresponding pixels in the extracted texture maps TM_v as follows:

$$TM(x,y) = \sum_{v=1}^{V} qual_v(x,y) TM_v(x,y) \tag{16}$$

Finally, a frontal view is generated using the shape deformations and the texture map that have been computed from the video sequence. This global workflow is illustrated in Fig. 6.

5. Alternative algorithm: global optimization by Levenberg–Marquardt

To evaluate the proposed particle filter, we compare it to an optimization method based on the Levenberg–Marquardt (LM) algorithm [24]. This method attempts to iteratively minimize an error defined with criteria similar to those used in the particle filter, by mixing gradient descent and Gauss–Newton algorithms. Unlike the particle filter method, this method is global, meaning that it estimates jointly the poses for all frames and the shape parameters (the same for the whole sequence). We use the *levmar* library available online to this end [20].

Let $y_{1:T} = (y_1,\ldots,y_T)$ be the set of observations available in the video sequence, such as feature point positions, gradients, silhouettes, and $u = (x_1,\ldots,x_T,\theta)$ the vector containing the unknown values, which are the poses and shape parameters. We apply a global optimization using all observations together. The feature points $\tilde{y}$ corresponding to the estimated poses and deformations $\tilde{u}$ can be projected in order to compare them to the feature point detections, corresponding to the observations y. The idea of the algorithm is then to minimize the error $\|y - \tilde{y}\|^2$, considering the following energy:

$$E_{points} = \sum_{t=1}^{T} \frac{1}{D(t)} \left(\sum_{p=1}^{D(t)} \left\| \underbrace{x_{proj}(p,t,X_t,\theta)}_{\tilde{y}} - \underbrace{x_{det}(p,t)}_{y} \right\|^2 \right) \tag{17}$$

where t is the frame index, $D(t)$ is the number of detected feature points at time t, p the index of these feature points, $x_{det}(p,t)$ their 2D positions and $x_{proj}(p,t,X_t,\theta)$ the projection of the corresponding points from the model in the images given the current estimation of pose X_t and shape θ. Let x_{2d} be the 2D homogeneous coordinates of the projection; the value $x_{proj}(p,t,X_t,\theta)$ is computed as follows:

$$\begin{cases} x_{2d} = A\left(R_t \kappa\left(\bar{S}(p) + \sum_{i=1}^{M} \alpha_i s_i(p)\right) + T_t\right) \\ x_{proj}(p,t,X_t,\theta) = \left(\frac{x_{2d}[0]}{x_{2d}[2]}, \frac{x_{2d}[1]}{x_{2d}[2]}\right) \quad \text{(2D-coordinates after normalization)} \end{cases} \tag{18}$$

with $T_t = (x_t, y_t, z_t)$ the translation and R_t the rotation matrix derived from the pose X_t at time t, κ and $\{\alpha_i, i = 1,\ldots,M\}$,

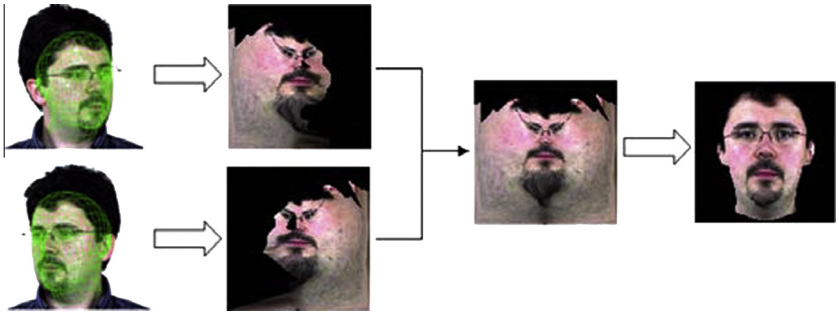


Fig. 6. Pose and shape estimation, texture map extraction, fusion and frontal view generation.

respectively the scale and the deformation values, and A the 3×3 projection matrix given the intrinsic camera parameters, which are supposed to be known in our experiments.

In addition to the retroprojection criterion, other criteria are used, as the ones introduced in [25]. In our case, the error then depends on the feature point projection term, an internal edge term, a silhouette term and a validity term for the shape parameters. The associated energy is a linear combination of those terms, with weights η_c, η_s and η_{model} which need to be empirically determined. Except for the shape validity term (E_{model}), all terms are computed for every frame. The function to minimize is then:

$$E = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{D(t)}\sum_{p=1}^{D(t)}\|x_{proj}(p,t,X_t,\theta) - x_{det}(p,t)\|^2\right) + \frac{1}{T}\sum_{t=1}^{T}\left(\eta_c d_c(t,X_t,\theta)^2 + \eta_s d_s(t,X_t,\theta)^2\right) + \eta_{model}E_{model} \quad (19)$$

with d_c and d_s two distances characterizing the error between the model and the observations for the internal edges and the silhouette criteria respectively, and the last term corresponding to the realism of the face (like the associated probability proposed in Eq. (3) or a regularity criterion for the deformed mesh).

The Levenberg–Marquardt algorithm needs an initial value u_0 for the unknown poses and shape parameters. The deformation parameters are set to zero (mean model), the scale κ_0 to a mean value calculated over a database and the poses are estimated using the method in [31] combined with RANSAC. The Levenberg–Marquardt algorithm uses the Jacobian of E, with respect to the unknown variables, to optimize the output state u. If we only use the feature points and the prior criteria, an analytical expression can be computed (Appendix A). If other criteria are included, a closed form solution is no longer available, and the Jacobian must be evaluated by finite differences, which increases computing time.

6. Evaluation

In this section, we will start by validating the proposed algorithm on a database of synthetic sequences for which the ground truth is available in terms of pose and shape parameters. After that, we will present the results of our method on real databases, both on visual aspects and biometric performances, which is the final purpose of the 3D face reconstruction in our case. Comparative results with the LM approach are presented in this second part, and show the interest of our approach.

6.1. Convergence results on synthetic data

6.1.1. Data generation and evaluation

To evaluate the different versions of the particle filter presented in Section 3.2 (systematic noise addition possibly parametrized by the weight, and MCMC with local or global sampling), we first give some convergence results on synthetic data (examples are given in Fig. 7), for which the ground truth values for the pose and the shape parameters are known.

The test faces are generated with two shape parameters sampled from the normal distribution, the pose during the sequence is similar to the one in real sequences, and the images are obtained with the same calibration parameters as the four cameras acquisition system which will be detailed in Section 6.2.

We tested the different particle filter methods on noisy data ($\sigma = 2$ pixels for the feature point inputs), to simulate detector answers on real data. To illustrate the impact of the chosen standard deviation, we compare it to the distribution of the feature point positions given the morphable model and represent it in Fig. 8. The distances between the eyes are 80 pixels in this figure, which can be compared to the corresponding distance in the input images, which ranges between 70 and 100 pixels. The standard deviations for each feature point are represented in Table 1, and are also given relatively to a distance of 80 pixels between the eyes. This feature point information leads to an approximate pose initialization and to a score function disturbed by the addition of Gaussian noise.

6.1.2. Parameter convergence

With known dynamic state. In a first experiment, we check the convergence of the particle static states towards the correct shape parameters when the head pose is known. In this case, the only unknown variables are the two static shape parameters, and we use therefore only 100 particles. The best particle at each instant (red cross), the filter mean (dashed light blue line) and variance (dotted dark blue line) are plotted for one sequence in Fig. 9. They can be

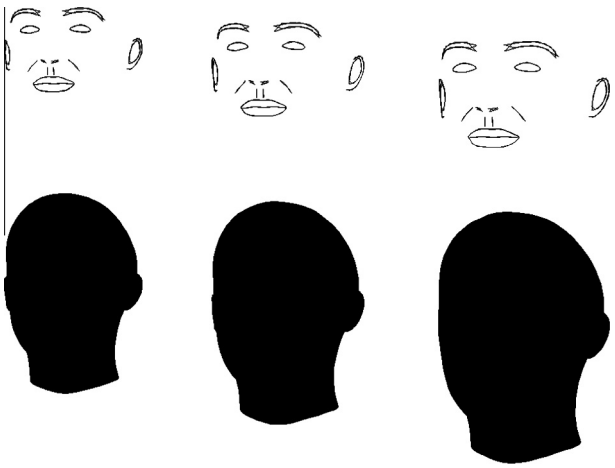
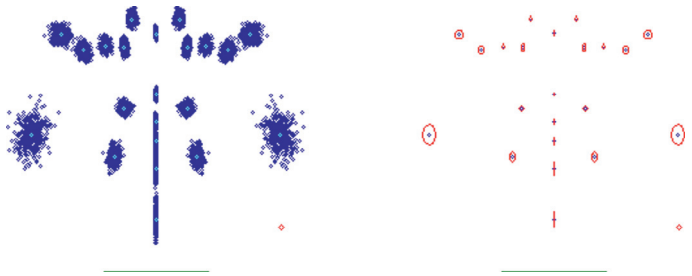


Fig. 7. Observations (internal edges and silhouettes) for a given camera at three instants of the sequence.



(a) Feature points corresponding to 500 samples (b) Mean and standard deviation

Fig. 8. Feature point distribution given the shape model seen frontally. In the right corner at the bottom, a 2 pixel radius circle is represented, corresponding to the standard deviation of the Gaussian noise applied to the positions. The distances between eyes on these two images are around 80 pixels, which is represented by the green line on the bottom. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
X- and *Y*-standard deviations (in pixels) for the feature points represented in Fig. 8. The corresponding distance between the eyes is 80 pixels.

Feature point	σ_X	σ_y
Left corner – left eyebrow	3.3	3.5
Right corner – left eyebrow	1.5	2.8
Left corner – left eye	2.2	3.1
Center of the left eye	1.8	2.8
Right corner – left eye	1.3	2.8
Top of the nose	$0.8 \cdot 10^{-3}$	2.0
Bottom of the nose	$0.5 \cdot 10^{-3}$	2.5
Tip of the nose	$0.6 \cdot 10^{-3}$	1.9
Left bottom of the ear	5.9	9.0
Left mouth corner	2.2	4.2
Bottom of the mouth	$0.5 \cdot 10^{-3}$	4.9
Chin	$0.6 \cdot 10^{-3}$	6.8

compared to the ground truth values plotted with a solid green line (GT). We observe that the mean of the filter converges towards the real parameters, and that the variance decreases at the beginning of the sequence before stabilizing. In this example, even if few particles are sampled around the true first parameter at the initialization step, the whole set of particles moves towards this value over the sequence.

With unknown dynamic state. To simulate real data issues, in which the pose is not known, we now integrate the hidden dynamic state x_t in the estimation process. Besides the two unknown shape parameters already estimated before, there are now six more time-varying unknown variables, corresponding to the 3D position of the head and the 3 rotation angles. This explains why we use $N = 2500$ particles in this experiment, still conducted on synthetic data, instead of $N = 100$ in the previous one. The pose $x_t^{(i)}$ of each particle is therefore generated as explained in Algorithm 1.

Robustness to the pose error. We initially evaluate the algorithm robustness to an initial pose error. To this end, we launch the algorithm using various input poses as initial pose estimation x_t^0 in Eq. (14) for each time t: first the true pose, before adding various yaw angle errors (2, 4, 6, 8 and 10 degrees) to it. Particle poses are then sampled around this modified pose input. Fig. 10

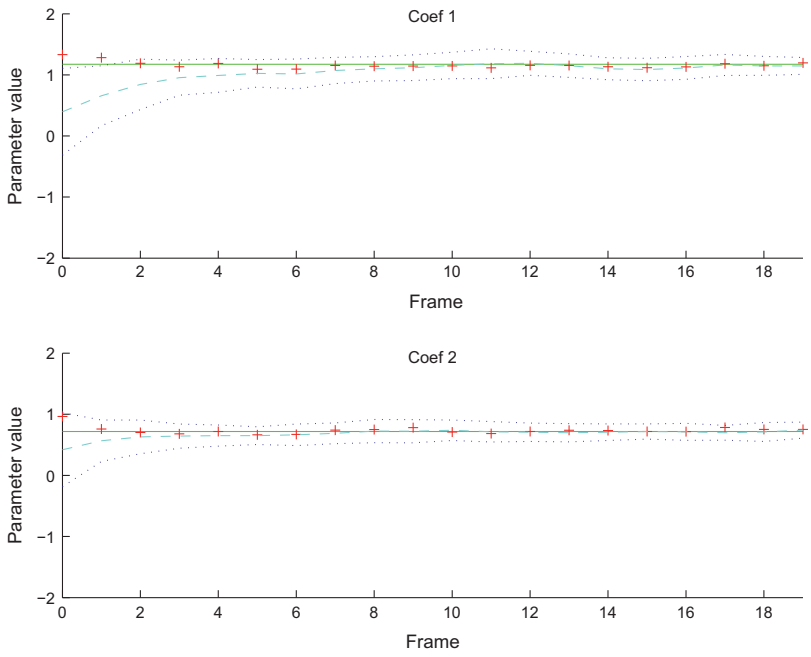


Fig. 9. Evolution of the filter static shape parameters when the pose is known. The curves *Coef1* and *Coef2* represent the particle distribution for α_1 and α_2 (Eq. (1)), the two unknown shape parameters to be estimated in this experiment. The best particle values are represented with red crosses, the filter mean with a light blue dashed line, and the filter variance with dark blue dotted lines. Ground truth values correspond to the green lines. $N = 100$ particles are used, and the artificial moves correspond to Gaussian noises with fixed covariance. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

illustrates the results, by representing for each error threshold ϵ (X-axis), the number of sequences (Y-axis) for which the error on the first parameter is below this threshold value. It shows that for an error on the initial yaw estimation that is less than 6 degrees, the convergence results are comparable. For higher errors, too few particles are sampled around the true pose which makes the convergence more difficult. An example of a sequence for which the initial pose is poorly estimated is shown in Fig. 11: given the dynamic noise n_x, the number of particles and the initial pose error, no particle is sampled close enough to the good values to have a high likelihood. The move and resampling steps cannot be guided to the appropriate subspaces, and the convergence can, therefore, not be achieved. A higher dynamic noise n_x in Eq. (14), associated with more particles, can be considered if larger pose errors are expected. However, increasing the noise would also lead to less accurate results in case of correct initial pose, and a compromise therefore needs to be found.

Gaussian noise on the static parameters. Fig. 12 shows the filter evolution for the same sequence as in Fig. 9, but with unknown pose parameters. The artificial dynamics are Gaussian noises with fixed covariance for all particles. We can see that the deviation of the particle distribution is larger than in the previous case. As the pose is estimated simultaneously with the parameters, the space dimension is higher, and it is difficult to differentiate particles having correct poses but wrong shape parameters from the ones having the inverse configuration. The shape parameter filtering therefore needs more time to eliminate the wrong states and to converge.

Adaptive noise. Fig. 13 shows some convergence results given input data without noise. When adding a noise of 2 pixels to the feature point positions, we get the results presented in Fig. 14. Despite this observation alteration, filter means for the static parameters are close to the true values.

MCMC moves. This method uses a validation step before modifying the static parameters sampled for a particle. We evaluate two types of sampling: local sampling around the current value, and global sampling given the Gaussian prior. The move is only applied on the static shape parameters, thus optimizing the shape at a fixed pose. This step requires a new likelihood computation, that should theoretically be done on the whole set of observations $y_1, \ldots, y_t$. In this case, the validity of the previously computed poses $x_1^{(i)}, \ldots, x_{t-1}^{(i)}$ is not guaranteed. This is why we use $\Delta_T = 0$, meaning that only the current view is used to compute the move acceptation. Fig. 15 shows the filter evolution for the two types of sampling methods, which lead to similar results.

Methods comparison. Let θ^1_{GT} be the true value of the first shape parameter and θ^1_{eval} the mean value over the particle states. To evaluate the different methods, we measure the error $\epsilon = |\theta^1_{eval} - \theta^1_{GT}|$ for our 39 synthetic sequences on the last frame of the sequence. Fig. 16 shows that all methods provide globally similar results. Curves 3–5 present results when a noise is added on the static parameters at each instant. Using an adaptive noise (curve 4) instead of a fixed noise (curve 5) results in more accuracy thanks to a better state space exploration. We can also notice that the prior sampling of the shape parameters (normal distribution with parameters $(m = 0, \sigma = 1.0)$ for curve 3, $(m = 0, \sigma = 1.7)$ for curve 4) influences slightly the curves: if the normal distribution has a larger deviation, it becomes easier to reach large values of the parameters, as more particles will then be sampled around the true value. Conversely, for narrow initial sampling, the particles are concentrated in a smaller area, which leads to more accurate results when the parameters are close to zero. This explains why curve 3 is above the others when the error is small. The highest error is around 0.6 for $\sigma = 1.7$ (curve 4), against 0.85 using standard normal distribution (curve 5). These values can be compared to the interval covered by all θ^1_{GT} values of our database, $[-2.97; 2.10]$, sampled from the standard normal distribution. Using the weight adaptive noise method and a large deviation for the initial parameter sampling, 87% of runs provide an error of less than 0.34 (6.7% of the interval width).

Although MCMC moves involve a validation step using the Metropolis–Hastings algorithm, the two evaluated methods (curves 1 and 2) do not outperform the previous ones, based on a systematic noise addition. Automatic noise methods may therefore be preferred since the other methods do not provide significant accuracy improvements despite their higher computational cost.

Failure cases. For some sequences, the true values are never reached during the filtering process. The explanation is twofold. First, it can be due to the model prior used to initialize the static parameter particles. The more true parameters are different from zero ($|\theta^i_{GT}| \geqslant 1$), the smaller the probability to sample particles around these true values, and the static parameter moves made afterward do not always compensate for the initialization (Fig. 11a). Secondly, the 3D pose can be poorly estimated, for instance with very noisy detections. As all particle poses are sampled around it, no poses will be close to the true one with a bad initialization. In this case, the shape optimization will not succeed, because a good pose approximation is required to estimate the parameters.

These two issues are sometimes related. When the observed face is very different from the mean shape (meaning that $|\theta^i_{GT}| \gg 0$ for some i), the pose estimated by fitting the mean model is not accurate, as the feature point positions are not the same on the current face and on the mean one (Fig. 11b). In this case, there are few particles in the appropriate pose and shape subspaces. This is why we proposed to initialize the pose with each particle shape model in Section 4.1, at the cost of N pose fittings. The following results on real data were obtained with this improvement.

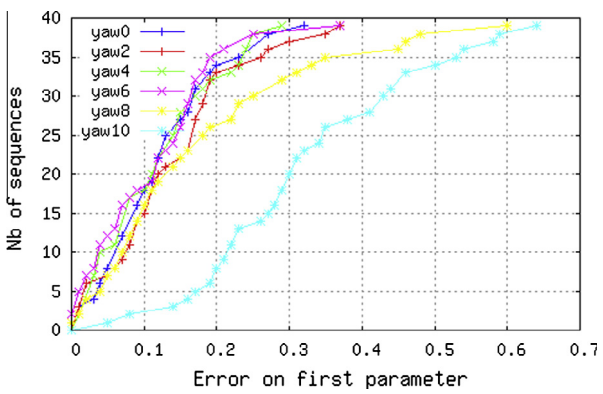


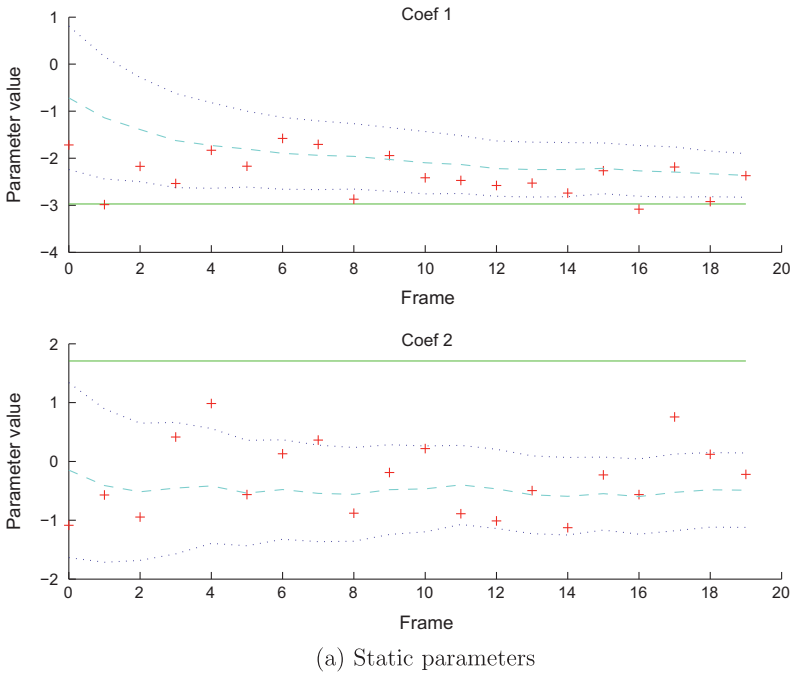
Fig. 10. Robustness of the first shape parameter estimation depending on the initial yaw angle error, varying from 0 (yaw0) to 10 degrees (yaw10). The Y-axis indicates the cumulative number of sequences for which the error on the first parameter is below a threshold (read on the X-axis).

6.2. Evaluation on real data

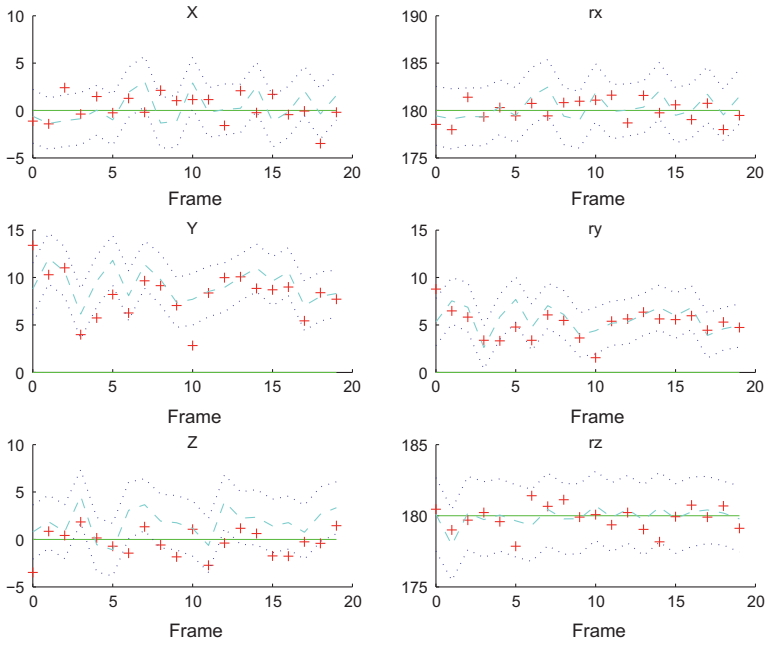
We extend now the evaluation of our algorithm on several real datasets acquired in our laboratory. The different scenarios are presented in the next section, before analyzing the visual and biometric results we obtained. The aim of this part is to evaluate the effectiveness of our method on real data, with noisy or outlier detections, and heads to be estimated which cannot be perfectly described by our head model.

6.2.1. Acquisition setup

The datasets used for our evaluation were acquired in our laboratory and correspond to a standard use of face recognition gates. They differ in terms of acquisition conditions (indoor or outdoor



(a) Static parameters



(b) Pose

Fig. 11. Example of non-convergence of the filter towards the ground truth values with unknown pose and noisy observations. $N = 2500$ particles are used, and artificial moves are Gaussian noise with adaptive covariance.

sequences, use of specific lightening system), of number of cameras (3 or 4), and of user behavior (stop during the walk through the gate). The different types of databases are listed in Table 2. As illustrated by the image samples in Fig. 17, the head poses seen in the camera coordinate system vary generally between half-profile and frontal pose, in addition to some pitch angle. There are no extreme poses (for instance, no profile views), as we consider the case of cooperative behavior of users who want to be recognized. Indeed, the system used to acquire these videos is an authentication system conceived to obtain good views of faces, but limiting the constraints for users during the acquisition.

No ground truth is available in terms of shape parameters and poses for these real databases, and only 2D-acquisitions are given to evaluate the quality of our results. For this reason, we propose two types of evaluation; the first one is based on visual control, to illustrate some results with the different methods implemented, and the second one is performed on biometric results.

6.2.2. Visual analysis

A first analysis has been conducted on visual outputs of our algorithm, such as the mesh projections on input images, the extracted texture maps and resulting frontal views. Fig. 18 shows

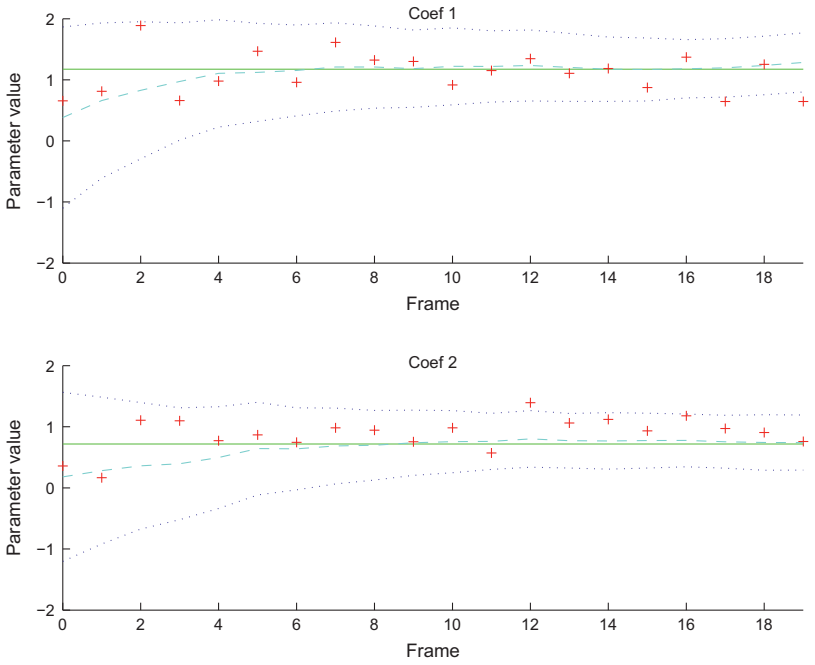


Fig. 12. Evolution of the filter static shape parameters when the pose is unknown, using $N = 2500$ particles. Artificial moves are Gaussian noises with fixed covariance.

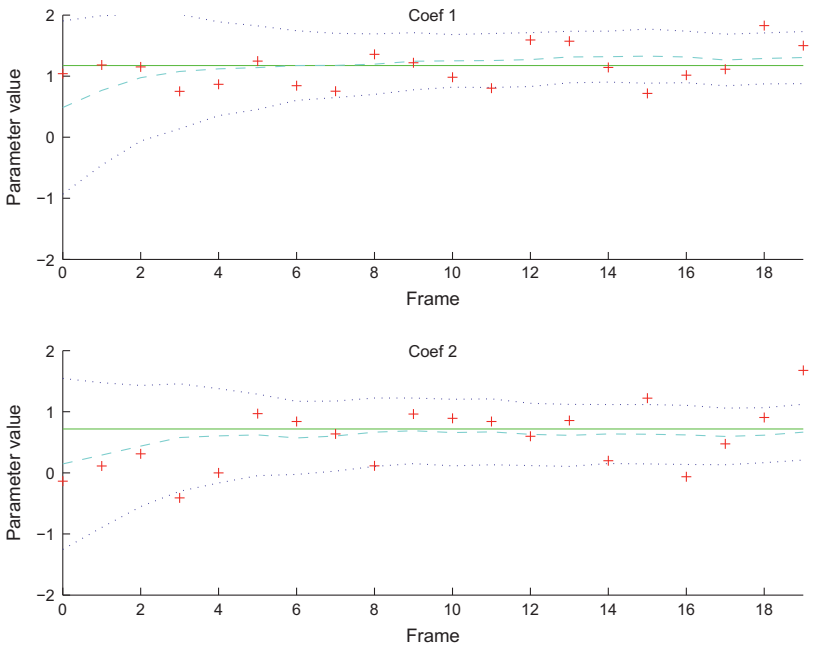


Fig. 13. Evolution of the filter static shape parameters, when the pose is unknown. No noise is added to the input feature points. $N = 2500$ particles are used, and artificial moves are Gaussian noises with adaptive covariance.

the mesh projection given the estimated pose and shape parameters at the end of the sequence. Additionally, it illustrates how the internal edges and the feature points fit to the corresponding features on the observations.

The evolution of the fitting quality between the beginning and the end of a sequence is presented in Fig. 19. The improvement is especially visible on the silhouette criterion, because the mean model used at the beginning is not adapted to the observed face.

Comparison with the Levenberg–Marquardt method. We first present some visual results to compare the output of the Levenberg–Marquardt method versus our particle filter. Fig. 20 shows the mesh projections for two sequences, using the LM optimization algorithm on the one hand and the particle filter method on the other hand. We can notice that the shape estimated by the Levenberg–Marquardt method is highly distorted on the first two images, because feature points have been badly detected and miss-detections are not handled. The multiple hypotheses evaluated with the particle filter method allows us to find a better pose, and thus leads to an improved set of shape parameters. Moreover, with our online method, even if one frame has bad inputs, it will not affect the whole estimation, because improvements can be obtained with further observations. We can also note that no

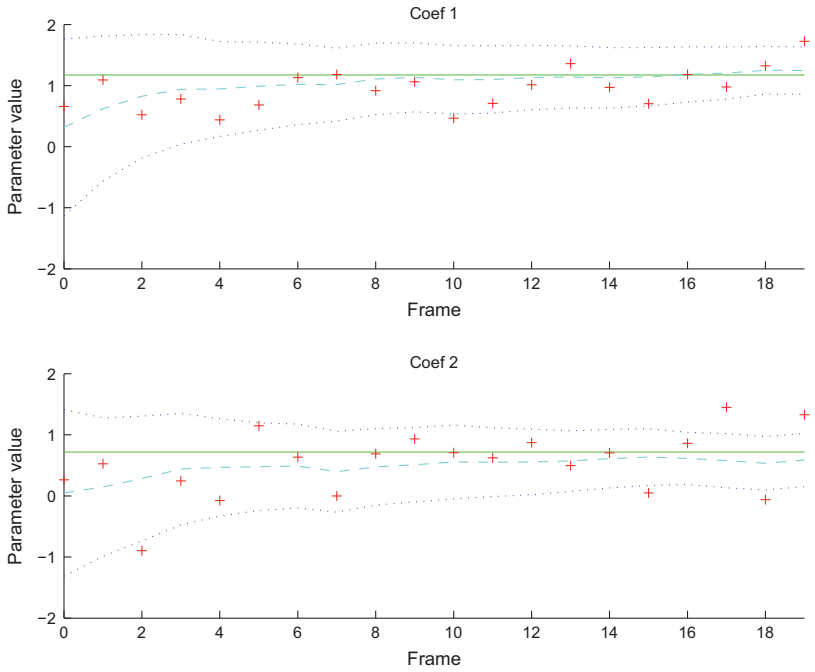


Fig. 14. Evolution of the filter static shape parameters when the pose is unknown. A Gaussian noise ($\sigma = 2$ pixels) is added to the input feature points. $N = 2500$ particles are used, and artificial moves are Gaussian noises with adaptive covariance.

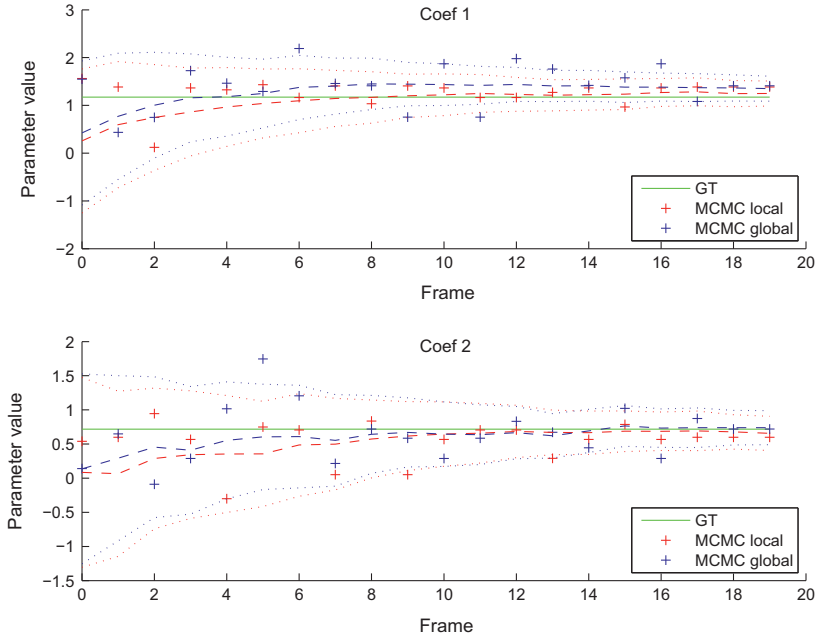


Fig. 15. Evolution of the filter static shape parameters when the pose is unknown. $N = 2500$ particles and MCMC moves are employed.

temporal consistency is verified when using the Levenberg–Marquardt method, while in our case, we verify that the pose estimation is consistent with the previous one for each particle. Thus, we avoid punctual aberrant poses in the trajectory. On the third image in the first line, we can see that the head pose is disturbed on the left by a bad ear detection: the head pose veered off to the left to get closer to this detection. With our method, some of the particles do not use this detection to fit the head, which leads to a better fitting regarding the different criteria. These particles will be duplicated during the resampling process, and lead to the displayed pose and shape estimation.

6.2.3. Biometric evaluation

In order to show the impact of the proposed algorithm on face recognition systems, we also analyze its performance relatively to a biometric evaluation. There is certainly a bias using this type of validation, as we cannot perfectly measure the impact of the shape estimation on the whole face comparison algorithm. Still, better pose and shape estimations improve the frontal views used as input for the comparison step. Moreover, due to the purposes of face recognition systems, we believe that this measure is relevant.

Face comparison. To compare images of faces for recognition purposes, discriminative information has to be extracted from each

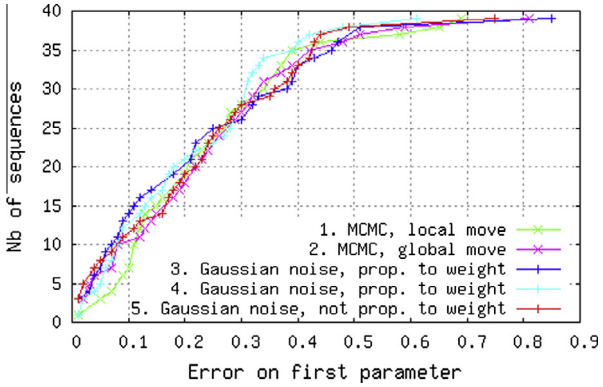


Fig. 16. Error on the first shape parameter estimation depending on the move used to diversify the static shape parameter and on the initial parameter sampling. The *Y*-axis indicates the number of sequences for which the error on the first parameter is below the threshold specified on the *X*-axis. The prior on θ is Gaussian with parameters $(m = 0, \sigma = 1.7)$ for curves 1, 2, 4, 5 and $(m = 0, \sigma = 1.0)$ for curve 3.

Table 2
Datasets used for the evaluation. For each dataset, several acquisitions were performed for each person (ID), with different walking speeds.

Base	Views	Scene	#ID	#Seq/ID	Total seq.
O1	3	Outdoor	36	1–5	122
I1	4	Indoor	61	3–6	273
I2	3	Indoor	30	1–5	183

of them. Among the existing methods used for face comparison, Gabor filters [19] or LBP [1] are the most commonly used descriptors. Once computed, these characteristics are associated and used to compute the distance between two facial feature vectors. A survey of face recognition algorithms using single images or videos is proposed in [14]. The extraction of facial specificities can be done on all images of a tracklet, or applied on a subset of images only. The output of this step is concatenated in a facial template. In our case, all templates are computed from a single image, which can be one of the input images directly, or the frontal view generated once the shape has been estimated with either our particle filter or the Levenberg–Marquardt algorithm. To establish biometric performances, we compare them to templates generated from ID picture, where the face is seen frontally with controlled illumination and neutral expression. The distance between these templates is finally related to a score which characterizes how similar the two faces are.

Temporal estimation improvement. The first experiment consists in studying the evolution of the face comparison score between the ID picture of the person walking through the gate, and the reconstructed frontal view at each instant. As the estimation is updated with each new incoming observation, its accuracy should be updated at each instant (at least if the acquisition conditions are constant). We evaluate the gain of our algorithm by analyzing the comparison score between the frontal view generated at each time t and the reference picture (the frontal face stored in the ID document).

Fig. 21 shows the evolution of this score for two sequences (time is on the *X*-axis). The red crosses correspond to frontal views generated from single images at time t, and the blue circles to

Fig. 17. Line 1: 4-cameras indoor; line 2: 3-cameras indoor; line 3: 3-cameras outdoor acquisitions.

Fig. 18. Projection of the estimated mesh and internal edges on the acquired image at the end of a sequence.

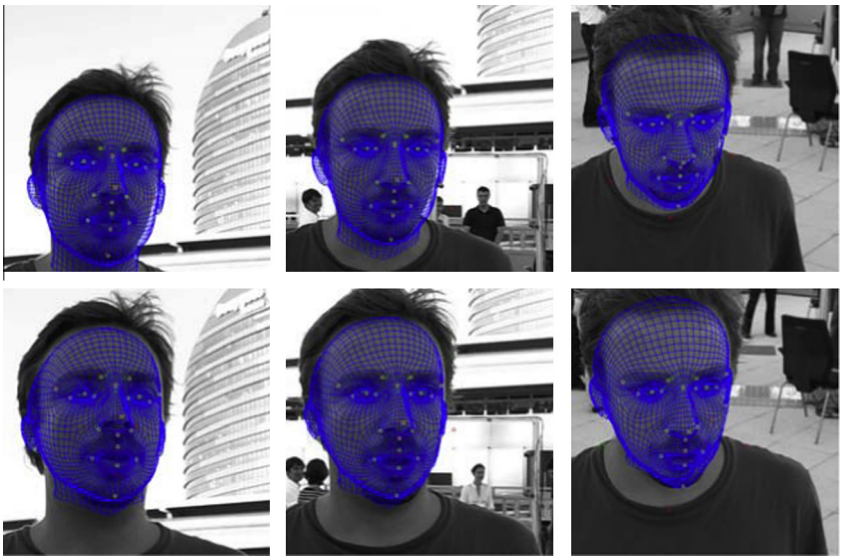


Fig. 19. Estimated mesh projection at the beginning (first line) and the end (second line) of the sequence. We can see that the shape differs more and more from the mean model over time to better fit to the observations.

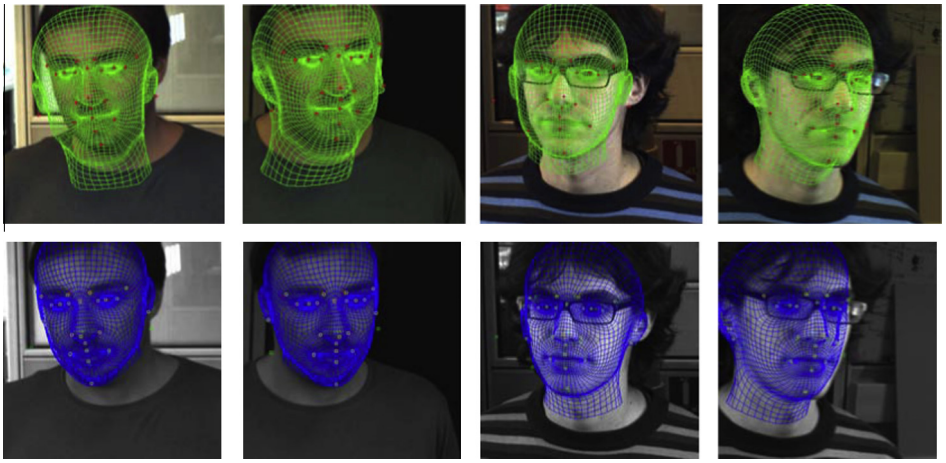


Fig. 20. Comparison between the pose and shape estimation computed by Levenberg–Marquardt (in green) and the one computed by particle filtering (in blue). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

views obtained after merging the texture maps in the available views at the same time, following the method described in Section 4.2. We can notice that the comparison scores are usually higher with the merged texture (especially for the I1 database), as the head is not necessarily seen frontally, and the texture map extracted in a single image is not complete. Besides, the comparison scores increase throughout the sequence on the whole. The score decrease at the end of the I1 sequence can be explained by two factors: first, in some configurations, the pose becomes less frontal when the face gets closer to the cameras. Less feature points are visible and the detection quality is lower, which leads to badly estimated poses. Secondly, the face can be over-exposed at the end when specific lights are used, which leads to less contrast or unwanted shadows over the face. Since face comparison algorithms are sensitive to the gradient positions in the face, such artifacts can modify the scores to a similar extend as modifications induced by errors on shape and pose. We therefore recommend using the estimation obtained a few images before the end with the 4-camera configuration. The comparison scores can differ from one identity to another, as they depend on the facial changes between the reference picture and the acquisitions. Indeed, if a person changed a lot between two acquisitions (ageing, make-up, etc.), the similarity score will be lower than for two images taken in a short time. This is related to the face comparison algorithms that are not necessarily robust to all these factors. This explains why the values obtained in the two curves of Fig. 21 are not the same (in addition to the difference between the two acquisition systems), because different identities have been considered.

The comparison with the LM method is shown in Fig. 22, in which we can see that the face comparison score is higher with the particle filter, thanks to a better pose and shape estimation.

Recognition rate evaluation. Finally we analyze our reconstruction algorithm in terms of biometric performances over a full database, using DET curves (False Reject Rate versus False Acceptance Rate). With such measures, one can easily determine how many persons have been rejected for a given rate of false acceptance. The smallest the FRR, the better are the results. Indeed, a perfect system ideally rejects all false pairs (no False Acceptance) and accepts all true pairs (which corresponds to 0 False Rejection).

Fig. 23 shows the FRR performances for different values of FAR, using observations at different instants to generate the frontal view. To compute these performances, each generated frontal view for a sequence has been compared to a reference basis containing 1086 items. As frontal views obtained after texture fusion between the views of the considered instant have better scores than the ones computed from each image separately, we only use the merged frontal view. This figure illustrates the performance evolution when new frames are used to update the model. We can notice some improvement at the beginning of the sequence, because new images help to increase the parameter accuracy which results in a better shape fitting. Nevertheless, performances deteriorate slightly when adding the last images, because of the non-frontal pose and the illumination conditions. These are reflected in the quality of the texture used for the frontal view generation which is only extracted from the images corresponding to the last instant used in the estimation. This observation can be linked to the one on temporal improvement analysis, when comparison scores decrease at the end of the sequence. An additional criterion taking lighting

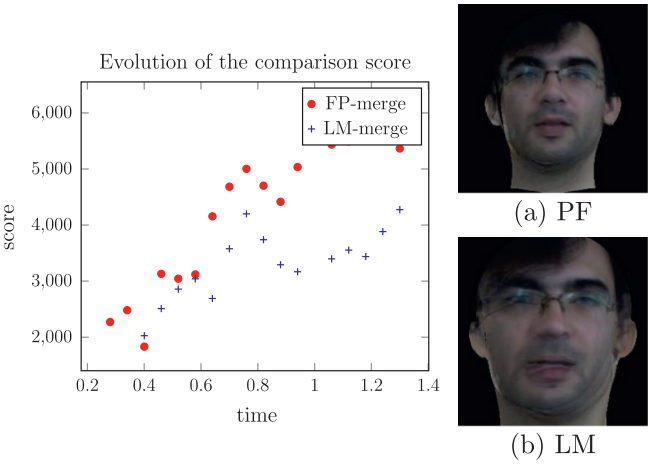


Fig. 22. Comparison score evolution over the sequence (left) and merged frontal views generated at the end with the LM (blue) and the PF (red) methods (right). The sample sequence belongs to the I2 database. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

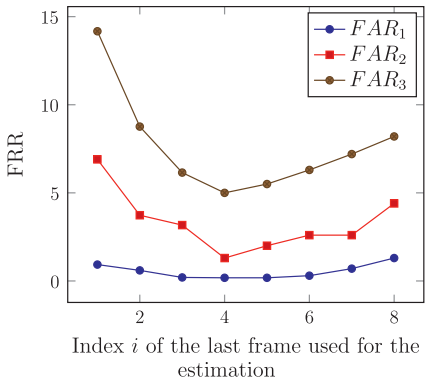


Fig. 23. FRR performances when integrating information over time, for three values of FAR. The value on the *x*-axis corresponds to the number of frames used for the estimation: the first value is the result using only the first frame, the following are the outputs using the video from time 1 to *i*, with our particle filter based method.

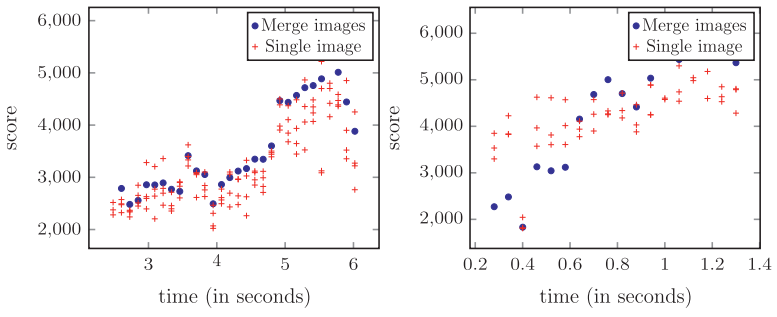


Fig. 21. Evolution of the comparison score between the synthesized frontal view and the ID picture during two sequences. Red crosses (respectively blue circles) correspond to views synthesized from texture maps extracted from each view (resp. from the merged texture map) at time *t* – Databases I1 (left) and I2 (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
Relative FRR reduction for PF with respect to LM: $(FRR_{PF} - FRR_{LM})/FRR_{LM}$.

	I1 database		I2 database		O1 database	
FAR	10^{-2} (%)	10^{-3} (%)	10^{-2} (%)	10^{-3} (%)	10^{-2} (%)	10^{-3} (%)
t_5	−14	−15	−21	−17	−8	−6
t_{10}	−19	−15	−10	−14	−7	−4
t_{15}	−6	−7	0	−11	−8	−3

conditions into account could therefore be interesting for choosing the appropriate images for the texture extraction.

Comparisons between the FRR-FAR performances for the LM and the PF methods are given in Table 3. We report results at $FAR = 10^{-2}$ and 10^{-3}, according to the constraints presented in [21], in terms of number of impostors tests. The estimations are computed at three instants, and with three different methods: the LM with the single frame t_T as input, and the PF and the LM with frames t_1 to t_T. Since results are similar with the two LM methods, we only indicate the gain of our method given the ones obtained with the LM over the whole sequence. Improvements are seen for all databases, which confirms the visual results presented in Section 6.2.2. Compared with the FRR obtained with the LM method, the mean reduction of false rejections reaches 10%.

To illustrate the improvement achieved by the step of frontalization, we add the following experiment on the database I1. Instead of using different images and generating a frontal view for the coding step, we select a single image using the following criterion:

$$Qual(image) = faceSize.\min(leftEye.conf, rightEye.conf) \tag{20}$$

The best image (BI) selected with this criterion corresponds to faces with high resolution (*faceSize*) and good detection scores (*leftEye.conf* and *rightEye.conf* are left and right eye confidence values). As our detectors were learned on frontal views, the detection confidences will be higher for such faces. We compare this method with the previous particle filter applied on a single instant corresponding to the one of the best image selected, and to the particle filter applied on the video sequence until the time associated to the best image. The improvement achieved by the use of synthetic frontal views for the coding and comparison steps is given by the relative FRR reduction of the particle filter algorithms with respect to the FRR provided by the method applied on the 2D best image only (without any frontalization) $(FRR_{PF} - FRR_{BI})/FRR_{BI}$:

- −41% on a single timestamp, −45% on the sequence, at $FAR = 10^{-2}$;
- −24% on a single timestamp, −27% on the sequence, at $FAR = 10^{-3}$.

These results show that applying our particle filter algorithm to estimate the pose and shape, and using the associated frontal views has a high impact on the biometric performances. Indeed, we cannot guarantee that faces are seen frontally in the best image, thus reducing the face comparison accuracy. The use of the particle filter on several timestamps provides even better results.

7. Conclusion and future work

We have presented a novel approach to estimation of the 3D pose and shape of a head in a video sequence. Considering the shape parameters as part of the hidden state in the particle filter algorithm, our method allows us to update the parameter distribution at each instant. Moreover, using the multi-hypothesis structure of the set of particles, we handle outliers in the set of feature points by varying the initial pose for each particle. In this way, there is also less chance of getting a solution trapped in a local maximum. Both visual and biometric results showed the interest of our particle filter-based method. In the future, we will adapt our algorithm to single camera and/or uncalibrated configurations in order to allow genericity and extend its use to new applications.

Acknowledgments

This work has been supported in part by the National Agency for Research and Technology (ANRT). We are grateful to all people which took part in our acquisitions in the Morpho laboratory and agreed to appear in this article.

Appendix A. Levenberg–Marquardt derivation

We present here the derivative computation for one point detected at time $\tilde{t}$ for the feature point retroprojection criterion only. Derivation with respect to the prior term is computed in the similar way, while finite differences are used for the other criteria, as the derivative expressions are not available.

The derivatives with respect to the scale parameter are given by:

$$\begin{cases} \frac{\partial x_{2d}}{\partial \kappa} = AR_t\left(\bar{S}(p) + \sum_{i=1}^{M}\alpha_i s_i(p)\right) \\ \frac{\partial x_{proj}(p,t,X_t,\theta)}{\partial \kappa}[k] = \frac{\frac{\partial x_{2d}[k]}{\partial \kappa}x_{2d}[2] - \frac{\partial x_{2d}[2]}{\partial \kappa}x_{2d}[k]}{x_{2d}[2]^2} \quad \text{for } k = 0,1 \end{cases} \tag{A.1}$$

and the ones with respect to the shape parameter α_i are:

$$\begin{cases} \frac{\partial x_{2d}}{\partial \alpha_i} = \kappa A(R_t s_i(p)) & \text{for } i = 1 \cdots M \\ \frac{\partial x_{proj}(p,t,X_t,\theta)}{\partial \alpha_i}[k] = \frac{\frac{\partial x_{2d}[k]}{\partial \alpha_i}x_{2d}[2] - \frac{\partial x_{2d}[2]}{\partial \alpha_i}x_{2d}[k]}{x_{2d}[2]^2} & \text{for } k = 0,1 \end{cases} \tag{A.2}$$

when the model constraint is related to the parameter probability (Eq. (3)). All derivatives $\frac{\partial E}{\partial x_t}, \frac{\partial E}{\partial y_t}, \frac{\partial E}{\partial z_t}, \frac{\partial E}{\partial r x_t}, \frac{\partial E}{\partial r y_t}$ and $\frac{\partial E}{\partial r z_t}$ are zero if $\tilde{t} \neq t$, as the projection at time t only depends on the pose at this time. If $\tilde{t} = t$ (we recall that $\tilde{t}$ is the time at which we consider the point), the derivatives have to be computed for the translation values:

$$\begin{cases} \frac{\partial x_{2d}}{\delta x_t} = A\begin{pmatrix}1\\0\\0\end{pmatrix} \\ \frac{\partial x_{proj}(p,t,X_t,\theta)}{\partial x_t}[k] = \frac{\frac{\partial x_{2d}[k]}{\partial x_t}x_{2d}[2] - \frac{\partial x_{2d}[2]}{\partial x_t}x_{2d}[k]}{x_{2d}[2]^2} \quad \text{for } k = 0,1 \end{cases} \tag{A.3}$$

respectively $(0,1,0)$ and $(0,0,1)$ for the derivatives with respect to y_t and z_t.

Using the derivated matrix $R_\psi = \frac{\partial R_t}{\partial \psi}$, we get the following derivatives with respect to the angles:

$$\begin{cases} \frac{\partial x_{2d}}{\partial \psi} = \kappa A\left(R_\psi\left(\bar{S}(p) + \sum_{i=1}^{M}\alpha_i s_i(p)\right)\right) \\ \frac{\partial x_{proj}(p,t,X_t,\theta)}{\partial \psi}[k] = \frac{\frac{\partial x_{2d}[k]}{\partial \psi}x_{2d}[2] - \frac{\partial x_{2d}[2]}{\partial \psi}x_{2d}[k]}{x_{2d}[2]^2} \quad \text{for } k = 0,1 \end{cases} \tag{A.4}$$

and similar equations are obtained for the derivative with respect to θ and ϕ respectively, using $R_\theta = \frac{\partial R_t}{\partial \theta}$ and $R_\phi = \frac{\partial R_t}{\partial \phi}$.

The lines in the jacobian J_E corresponding to the 2D detection of a point at time t has the following structure (x_p is used in place of $x_{proj}(p,t,X_t,\theta)$ for the sake of clarity):

$$\begin{pmatrix} \overbrace{0_{\{6(t-1)\times 1\}}}^{\text{derivatives are null w.r.t. other time poses}} & \underbrace{\begin{matrix} \frac{\partial x_p}{\partial x_t}[0] \cdots \frac{\partial x_p}{\partial z_t}[0] \\ \frac{\partial x_p}{\partial x_t}[1] \cdots \frac{\partial x_p}{\partial z_t}[1] \end{matrix}}_{\text{derivatives w.r.t. translation at } t} & \underbrace{\begin{matrix} \frac{\partial x_p}{\partial \phi_t}[0] \cdots \frac{\partial x_p}{\partial \psi_t}[0] \\ \frac{\partial x_p}{\partial \phi_t}[1] \cdots \frac{\partial x_p}{\partial \psi_t}[1] \end{matrix}}_{\text{derivatives w.r.t. rotation at } t} & \cdots & \begin{matrix} \frac{\partial x_p}{\partial \kappa}[0] \\ \frac{\partial x_p}{\partial \kappa}[1] \end{matrix} & \overbrace{\begin{matrix} \frac{\partial x_p}{\partial \alpha_I}[0] \cdots \frac{\partial x_p}{\partial \alpha_M}[0] \\ \frac{\partial x_p}{\partial \alpha_I}[1] \cdots \frac{\partial x_p}{\partial \alpha_M}[1] \end{matrix}}^{\text{derivatives w.r.t. the shape parameters}} \end{pmatrix}$$

References

[1] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: application to face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12) (2006) 2037–2041.

[2] B. Amberg, A. Blake, A. Fitzgibbon, S. Romdhani, T. Vetter, Reconstructing high quality face-surfaces using model-based stereo, in: IEEE International Conference on Computer Vision, October 2007, pp. 1–8.

[3] M.S. Arulampalam, S. Maskell, N. Gordon, A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, IEEE Trans. Signal Process. 50 (2002) 174–188.

[4] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, M. Gross, High-quality single-shot capture of facial geometry, ACM Trans. Graph. (SIGGRAPH) 29 (4) (2010) 40:1–40:9.

[5] V. Blanz, T. Vetter, A morphable model for the synthesis of 3D faces, in: SIGGRAPH, ACM Press/Addison-Wesley Publishing Co., 1999, pp. 187–194.

[6] D. Bradley, W. Heidrich, T. Popa, A. Sheffer, High resolution passive facial performance capture, ACM Trans. Graph. (SIGGRAPH) 29 (4) (2010) 41:1–41:10.

[7] P. Breuer, K.I. Kim, W. Kienzle, B. Schölkopf, V. Blanz, Automatic 3D face reconstruction from single images or video, in: IEEE International Conference on Automatic Face and Gesture Recognition, 2008, pp. 1–8.

[8] A. Doucet, S. Godsill, C. Andrieu, On sequential Monte Carlo sampling methods for Bayesian filtering, Stat. Comput. 10 (2000) 197–208.

[9] G.J. Edwards, C.J. Taylor, T.F. Cootes, Interpreting face images using active appearance models, in: IEEE International Conference on Automatic Face and Gesture Recognition, 1998, pp. 300–305.

[10] N. Faggian, A.P. Paplinski, J. Sherrah, 3D morphable model fitting from multiple views, in: IEEE International Conference on Automatic Face and Gesture Recognition, 2008, pp. 1–6.

[11] P. Fearnhead, MCMC, sufficient statistics and particle filters, J. Comput. Graph. Stat. 11 (4) (2002) 848–862.

[12] W.R. Gilks, C. Berzuini, Following a moving target – Monte Carlo inference for dynamic Bayesian models, J. Roy. Stat. Soc.: Ser. B (Stat. Methodol.) 63 (1) (2001) 127–146.

[13] C. Herold, V. Despiegel, S. Gentric, S. Dubuisson, I. Bloch, Head shape estimation using a particle filter including unknown static parameters, in: International Conference on Computer Vision Theory and Applications, SciTePress, 2012, pp. 284–293.

[14] R. Jafri, H.R. Arabnia, A survey of face recognition techniques, J. Inform. Process. Syst. 5 (2) (2009) 41–68.

[15] S. Julier, J. Uhlmann, A new extension of the Kalman filter to nonlinear systems, in: International Symposium on Aerospace/Defense Sensing, Simulation and Controls, SPIE, 1997, pp. 182–193.

[16] R.E. Kalman, A new approach to linear filtering and prediction problems, Trans. ASME: J. Basic Eng. 82 (Series D) (1960) 35–45.

[17] N. Kantas, A. Doucet, S.S. Singh, J.M. Maciejowski, An overview of sequential Monte Carlo methods for parameter estimation in general state-space models, in: 15th IFAC Symposium on System Identification, International Federation of Automatic Control, 2009.

[18] Y. Lin, G.G. Medioni, J. Choi, Accurate 3D face reconstruction from weakly calibrated wide baseline images with profile contours, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 1490–1497.

[19] C. Liu, H. Wechsler, Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition, IEEE Trans. Image Process. 11 (4) (2002) 467–476.

[20] M. Lourakis, levmar: Levenberg-Marquardt Nonlinear Least Squares Algorithms in C/C++, 2004. <http://www.ics.forth.gr/lourakis/levmar/>.

[21] A.J. Mansfield, J.L. Wayman, Best Practices in Testing and Reporting Performance of Biometric Devices, Tech. Rep., National Physical Laboratory, 2002.

[22] P. Minvielle, A. Doucet, A. Marrs, S. Maskell, A Bayesian approach to joint tracking and identification of geometric shapes in video sequences, Image Vis. Comput. 28 (2010) 111–123.

[23] U. Park, H. Chen, A.K. Jain, 3D model-assisted face recognition in video, in: Canadian Conference on Computer and Robot Vision, IEEE Computer Society, 2005, pp. 322–329.

[24] A. Ranganathan, The Levenberg–Marquardt Algorithm, Tech. Rep., 2004.

[25] S. Romdhani, Face Image Analysis using a Multiple Features Fitting Strategy, Ph.D. Thesis, Universität Basel, January 2005.

[26] S. Romdhani, V. Blanz, T. Vetter, Face identification by fitting a 3D morphable model using linear shape and texture error functions, in: European Conference on Computer Vision, Springer, 2002, pp. 3–19.

[27] S. Romdhani, T. Vetter, Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior, in: Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, 2005, pp. 986–993.

[28] H.W. Sorenson (Ed.), Kalman Filtering: Theory and Application, IEEE Press, 1985.

[29] G. Storvik, Particle filters for state-space models with the presence of unknown static parameters, IEEE Trans. Signal Process. 50 (2) (2002) 281–289.

[30] M.A. Turk, A.P. Pentland, Face recognition using eigenfaces, in: IEEE Conference on Computer Vision and Pattern Recognition, 1991, pp. 586–591.

[31] S. Umeyama, Least-squares estimation of transformation parameters between two point patterns, IEEE Trans. Pattern Anal. Mach. Intell. 13 (4) (1991) 376–380.

[32] R.T.A. Van Rootseler, L.J. Spreeuwers, R.N.J. Veldhuis, Application of 3D morphable models to faces in video images, in: Symposium on Information Theory in the Benelux, Werkgemeenschap voor Informatie- en Communicatietheorie, May 2011, pp. 34–41.

[33] C. Zhang, Z. Zhang, A Survey of Recent Advances in Face Detection, Tech. Rep., Microsoft Research, 2010.

[34] L. Zhang, N. Snavely, B. Urless, S.M. Seitz, Spacetime faces: high resolution capture for modeling and animation, ACM Trans. Graph. (SIGGRAPH) 23 (3) (2004) 548–558.

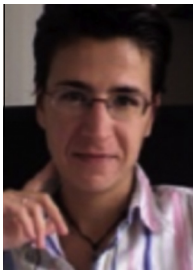
[35] R. Zhang, P.-S. Tsai, J.E. Cryer, M. Shah, Shape from shading: a survey, IEEE Trans. Pattern Anal. Mach. Intell. 21 (8) (1999) 690–706.

[36] M. Zollhöfer, M. Martinek, G. Greiner, M. Stamminger, J. Süßmuth, Automatic reconstruction of personalized avatars from 3D face scans, Comput. Anim. Virtual Worlds 22 (2011) 195–202.

Catherine Herold graduated from the Ecole des Mines de Nancy, Nancy, France, in 2009, she received an MSc in computer science and image processing from the Paris VI University in 2010. She is currently a Ph.D. student in Telecom Paris- Tech and LIP6, Paris, in collaboration with Morpho, Safran. Her research interests include computer vision, especially in the areas of face tracking and reconstruction, and particle filter methods for dynamic and static parameter estimation.

Vincent Despiegel received the Agrégation de Mathématiques degree in 2004, is a former student of the école Normale Supérieure de Lyon, Lyon, France, and received a Ph.D. degree in 2007 from the Université de Grenoble, Grenoble, France, on the study of Hyperelliptic curves and on how substitution boxes could be built for cryptographic applications. Since 2007, he has been a research and development staff member at Morpho, France, within the Biometric research team. From 2007 to 2011, he worked mainly on fingerprint algorithms improvement. In particular, he was involved in the European FP7 integrated project TURBINE (TrUsted Revocable Biometric IdeNtitiEs, 2008–2011) and worked on template protection and fingerprint templates binarization. Since 2011, he is the manager of a research team dedicated to face detection and tracking. His research interests include cryptography, image processing and pattern recognition dedicated to biometry.

Stéphane Gentric is Research Unit Manager at Morpho (www.morpho.com). He received his Ph.D. in 1999, on Pattern Recognition at UPMC. From 1999 to 2002, he worked mainly on fingerprint algorithms. From 2002, he focused on Face Recognition, then Iris Recognition. He is now team leader for both biometries, driving all algorithmic aspects, from Acquisition Device to Large Scale Matching System. He was involved in most of Morphos projects in biometrics of the past 10 years, such as Smartgate Australian border crossing System as well as NIST benchmarks, or the UIDAI project. His current research interests stay pattern recognition for improvement of biometric systems.



Séverine Dubuisson (M06) was born in 1975. She received the Ph.D. degree in system control from the Compi'egne University of Technology, Compi'egne, France, in 2001. Since 2002, she has been an Associate Professor with the Laboratory of Computer Sciences, University Pierre and Marie Curie (Paris 6), Paris, France. Her research interests include computer vision, probabilistic models for video sequence analysis, and tracking.

Isabelle Bloch is graduated from the Ecole des Mines de Paris, Paris, France, in 1986, she received the Master's degree from the University Paris 12, Paris, in 1987, the Ph.D. degree from the Ecole Nationale Supérieure des Télécommunications (Telecom ParisTech), Paris, in 1990, and the Habilitation degree from the University Paris 5, Paris, in 1995. She is currently a Professor with the Signal and Image Processing Department, Telecom ParisTech, in charge of the Image Processing and Understanding Group. Her research interests include 3D image and object processing, computer vision, 3D and fuzzy mathematical morphology, information fusion, fuzzy set theory, structural, graph-based, and knowledge-based object recognition, spatial reasoning, and medical imaging.