Robust Parameterized Component Analysis
Theory and Applications to 2D Facial Modeling

Fernando De la Torre[1] and Michael J. Black[2]

[1] Department of Communications and Signal Theory, La Salle School of Engineering, Universitat Ramon LLull, Barcelona 08022, Spain.
ftorre@salleURL.edu, http://www.salleURL.edu/~ftorre/

[2] Department of Computer Science, Brown University, Box 1910, Providence, RI 02912, USA.
black@cs.brown.edu http://www.cs.brown.edu/~black/

Abstract. Principal Component Analysis (PCA) has been successfully applied to construct linear models of shape, graylevel, and motion. In particular, PCA has been widely used to model the variation in the appearance of people's faces. We extend previous work on facial modeling for tracking faces in video sequences as they undergo significant changes due to facial expressions. Here we develop person-specific facial appearance models (PSFAM), which use modular PCA to model complex intra-person appearance changes. Such models require aligned visual training data; in previous work, this has involved a time consuming and error-prone hand alignment and cropping process. Instead, we introduce parameterized component analysis to learn a subspace that is invariant to affine (or higher order) geometric transformations. The automatic learning of a PSFAM given a training image sequence is posed as a continuous optimization problem and is solved with a mixture of stochastic and deterministic techniques achieving sub-pixel accuracy. We illustrate the use of the 2D PSFAM model with several applications including video-conferencing, realistic avatar animation and eye tracking.

1 Introduction

Many computer vision researchers have used Principal Component Analysis (PCA) to parameterize appearance, shape or motion [3,10,23,30]. However, one major drawback of this traditional technique is that it needs normalized samples in the training data. In the case of computer vision applications, the result is that the samples have to be aligned or geometrically normalized (we assume that other normalizations, e.g. photometric, have already been done). Previous methods for constructing appearance or shape models [10, 16,17,23,30,31] have cropped the region of interest by hand, or have used a hand-labeled pre-defined feature points to apply the translation, scaling and rotation that brought each image into alignment with a prototype. These manual approaches are likely to introduce errors into the model due to inaccuracies which arise from labeling the points by hand. In addition, manual cropping is a tedious, unpleasant, and time consuming task. This paper automates this process with a general framework for learning low dimensional linear subspaces while automatically solving for the alignment of the input data with sub-pixel accuracy.

To illustrate the idea, Figure 1 shows some frames from a training set for learning an eigen-eye (a subspace for the eye's variation). The images were captured by asking

A. Heyden et al. (Eds.): ECCV 2002, LNCS 2353, pp. 653–669, 2002.
© Springer-Verlag Berlin Heidelberg 2002

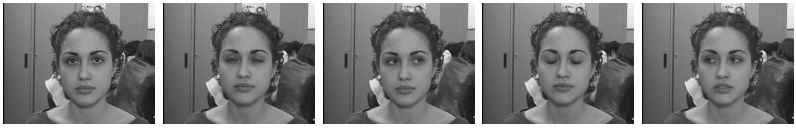


Fig. 1. Some frames of the original image sequence.

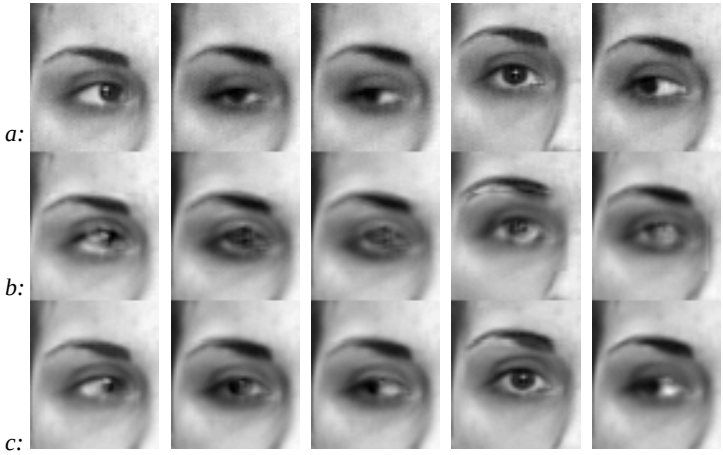


Fig. 2. a) Original data. b) Reconstruction of the right eye without any alignment. c) Reconstruction of the right eye with the proposed method.

the user to change the configuration of the eyes (open, close, look right, etc.) while holding the head still. However, it is not a reasonable to assume that the person is absolutely still during the training time, and in practical situations there are always small motions between frames. Observe that in this kind of sequence it is difficult to gather aligned data due to person's motion and the lack of labeled points for solving the correspondence problem between frames. The aim of the paper is illustrated in Figure 2, where Figure 2.a shows some original images used for training. A low dimensional linear model of the eye is constructed using PCA applied to the non-aligned images; that is, assuming that the person is not moving, the spatial domain of the eyes does not change over the sequence. Figure 2.b plots the reconstructed images computed with the bases derived from the non-aligned images. Figure 2.c shows the reconstructed images obtained using the parameterized component analysis method described here. This "eigen-registration" technique iteratively computes a linear subspace while aligning the training images w.r.t. this subspace. That is, the algorithm that we propose in this paper will simultaneously learn the local appearance basis, creating modular eigenspaces (ME) [26,30] while computing the motion to align the images w.r.t. the ME. The masks which define the spatial domain of the ME are defined by hand in the first frame (no appearance model is previously learned) and after that the method is fully automatic.

In this paper we focus on the application of face modeling. Most of the previous work on face tracking and modeling is focused on generic trackers, which are independent of the person's identity [5,9,10,23,24]. In particular, appearance based face trackers [10,26,

34] make use of PCA in order to construct a linear model of the face's subspace (variation across people) rather than the intra-person variations due to changes in expression. When working with person-specific models [15,17,20,23], PCA will model the complex intra-person appearance changes due mostly to variations of expression (eyes' blinking, wrinkles in the mouth area, appearance of the teeth, etc.) rather than modeling the appearance changes due to identity. Although PSFAM are just valid for one person, they remain useful in many vision related applications such as vision-based human computer interaction (VBHCI), speech driven animation (to animate faces from audio), facial animation in general, video-conferencing, face verification, etc, which usually involve a particular user. In related but different work, Edwards et al. [19,20] have proposed a method for approximately isolating the sources of image variation such as identity, pose, lighting, etc [19] by using linear discriminant analysis. Edwards et al. [20] use this factorized basis to update some characteristics to personalize a model. In this paper, we will apply Robust Parameterized Component Analysis to learn a PSFAM and will illustrate the method with applications involving facial modeling. Preliminary results of this paper were presented in [12].

2 Previous Work

This paper is related to previous work on subspace learning methods and PCA. It is beyond the scope of the paper to review all possible applications of PCA, therefore we just briefly describe the theory and point to related work for further information.

2.1 Subspace Learning

Let $\mathbf{D} = [\mathbf{d}_1\ \mathbf{d}_2 \ldots \mathbf{d}_N] = [\mathbf{d}^1\ \mathbf{d}^2 \ldots \mathbf{d}^d]^T$ be a matrix $\mathbf{D} \in \Re^{d \times N}$ [1], where each column $\mathbf{d}_i$ is a data sample (or image), N is the number of training images, and d is the number of pixels (variables) in each image. If the effective rank of $\mathbf{D}$ is much less than d, we can approximate the column space of $\mathbf{D}$ with $k << d$ principal components. Let the first k principal components of $\mathbf{D}$ be $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k] \in \Re^{d \times k}$. The columns of $\mathbf{B}$ span the subspace of maximum variation of the data $\mathbf{D}$.

Although a closed form solution for computing the principal components ($\mathbf{B}$) can be achieved by computing the k largest eigenvectors of the covariance matrix $\mathbf{D}\mathbf{D}^T$ [18], here it is useful to exploit work that formulates PCA as the minimization of an energy function [14,18]. Related formulations have been studied in various communities

[1] Bold capital letters denote a matrix $\mathbf{D}$, bold lower-case letters a column vector $\mathbf{d}$. $\mathbf{d}_j$ represents the j-th column of the matrix $\mathbf{D}$ and $\mathbf{d}^j$ is a column vector representing the j-th *row* of the matrix $\mathbf{D}$. d_{ij} denotes the scalar in row i and column j of the matrix $\mathbf{D}$ and the scalar i-th element of a column vector $\mathbf{d}_j$. All non-bold letters represent scalar variables. d_{ji} is the i-th scalar element of the vector $\mathbf{d}^j$. $diag$ is an operator that transforms a vector to a diagonal matrix, or a matrix into a column vector by taking each of its diagonal components. $tr(\mathbf{D})$ is the trace operator for a square matrix $\mathbf{D} \in \Re^{d \times d}$, that is, $\sum_{i=1}^{d} d_{ii}$. $||\mathbf{d}||_2^2$ denotes the L_2 norm of the vector $\mathbf{d}$, that is $\mathbf{d}^T\mathbf{d}$. $||\mathbf{d}||_{\mathbf{W}}^2$ denotes the weighted L_2 norm of the vector $\mathbf{d}$, that is $\mathbf{d}^T\mathbf{W}\mathbf{d}$, and $||\mathbf{D}||_F^2$ is the Frobenius norm of a matrix, $tr(\mathbf{D}^T\mathbf{D}) = tr(\mathbf{D}\mathbf{D}^T)$. $\mathbf{D}_1 \circ \mathbf{D}_2$ denotes the Hadamard (point wise) product between two matrices of equal dimension.

(see [14]): machine learning, statistics, neural networks and computer vision. In spirit, all these approaches essentially minimize the following energy function (although with different noise models, deterministic or Bayesian frameworks, or different metrics):

$$E_{pca}(\mathbf{B},\mathbf{C}) = ||\mathbf{D}-\mathbf{B}\mathbf{C}||_F^2 = \sum_{i=1}^{N} ||\mathbf{d}_i - \mathbf{B}\mathbf{c}_i||_2^2 = \sum_{t=1}^{N}\sum_{p=1}^{d}(d_{pt} - \sum_{j=1}^{k} b_{pj}c_{jt})^2 \quad (1)$$

where $\mathbf{C} = [\mathbf{c}_1\ \mathbf{c}_2\ \cdots \mathbf{c}_n]$ and each $\mathbf{c}_i$ is a vector of coefficients used to reconstruct the data vector $\mathbf{d}_i$. It is interesting to note that the three equivalent previous equations can give different insights into the subspace learning technique. The first matrix formulation clearly poses PCA as a simple factorization of $\mathbf{D}$ into $\mathbf{B}$ and $\mathbf{C}$. The problem of subspace learning translates to a bilinear estimation process of matrices $\mathbf{B}$ and $\mathbf{C}$. The second equivalence shows more explicitly how each data sample $\mathbf{d}_i$ is reconstructed with a coefficient $\mathbf{c}_i$ and a common basis $\mathbf{B}$. Finally the last equation expresses the subspace constraint at a pixel level. Many methods exist for minimizing (1) (Alternated Least Squares (ALS), Expectation-Maximization (EM), etc.), but in the case of PCA, share the same basic philosophy. These algorithms alternate between solving for the coefficients $\mathbf{C}$ with the bases $\mathbf{B}$ fixed and then solving for the bases $\mathbf{B}$ with $\mathbf{C}$ fixed. Typically, both updates are computed by solving a linear system of equations.

2.2 Adding Motion into the Subspace Formulation

Principal component analysis has been widely applied to the construction of facial models using linear subspaces [34]. During recognition or tracking it is common to automatically align the input images with the eigenspace using some optimization technique [4, 10,34]. In contrast, little work has addressed problems posed by facial misalignment at the learning stage. Mis-registration introduces significant non-linearities in the manifold of faces and can reduce the accuracy of tracking and recognition algorithms. While previous approaches have dealt with these issues as a separate, off-line registration processes (often manual), here it is integrated into the learning procedure.

Recently there has been an interest in the simultaneous computation (although the existing algorithms compute it iteratively) of appearance bases and motion. This problem is is a classical chicken-and-egg problem (like motion segmentation). Once the pixel correspondence between the images in the training data is solved, learning the appearance model is straightforward, and if the appearance is known, solving for the correspondence is easy. De la Torre et al. [15] proposed a method for face tracking which recovers affine parameters using subspace methods. This method dynamically updates the eigenspace by utilizing the most recent history. The updating algorithm estimates the parametric transformation, which aligns the actual image w.r.t. the eigenspace and recalculates a local eigenspace. Because the new images usually contain information not available in the eigenspace, the motion parameters are calculated in a robust manner. However, the method assumes that an initial eigenspace is learned from a training set aligned by hand. Schweitzer [33] has proposed a deterministic method which registers the images with respect to their eigenfeatures, applying it to the *flower garden* sequence for indexing purposes. However, the assumption of affine or quadratic motion models is only valid

when the scene is planar. The extension to the general case of arbitrary 3D scenes and camera motions remains unclear. As Schweitzer notices [33] the algorithm is likely to get stuck in local minima, since it comes from a linearization and uses gradient descent methods. On the other hand, Rao [32] has proposed a neural-network which can learn a translation-invariant code for natural images. Although he suggests updating the appearance basis, the experiments show only translation-invariant recognition, as proposed by Black and Jepson [4].

Frey and Jojic [21] took a different approach and they introduce an Expectation Maximization (EM) algorithm for factor analysis (similar to PCA) that is invariant to geometric transformations. While their work represents a significant and pioneering contribution, the proposed method can be problematic because it discretizes the space of spatial transformations and the computational cost grows exponentially with the number of possible transformations. Our work attempts to solve a similar problem but with a continuous optimization framework that is more appropriate for common parameterized transformations of the data (e.g. affine).

In a different direction, there has been intensive research on automatically or semi-automatically aligning facial shape models using extracted landmarks. See [11] for a good review of automatic 2D and 3D landmark placement. In contrast to previous automatic landmark methods, we use parameterized matching with a low dimensional model (e.g. affine) and generalize the matching by solving for both the subspace of the appearance variation and the alignment of the training data with the subspace.

In this paper, unlike previous methods we use stochastic and multi-resolution techniques to avoid local minima in the minimization process. Also, we extend previous approaches to multiple regions within a robust (to outliers) and continuous optimization framework. We apply the method to learn 2D modular PSFAMs and several potential applications of PSFAMs are proposed.

3 Generative Face Models: Motivation

Our eigen-registration algorithm will be introduced with examples from face modeling. In this section we describe one possible generative model for dynamic faces. Similar to the previous work of Black and Jepson [4], Black et *al.* [3] and Cootes et *al.* [10], the generative model that we propose for image formation takes into account motion and appearance, but in our case we also exploit predefined masks and learn the appearance bases. Figure 3 shows some frames of a training set for learning a 2D PSFAM. Given this training data as an input, the algorithm that we propose in this paper is able to factorize the training data into appearance and motion of some predefined regions.

3.1 Modular Eigenspaces

The eyes and the mouth, while weakly correlated, can perform independent graylevel changes (over a long sequence), so they should be represented in different eigenspaces in order to facilitate interpretation, to allow a more flexible model and to generate a more compact representation. Consider, for instance, a training set of people with both eyes open or both eyes closed. If we consider the face as a unique region, it would no

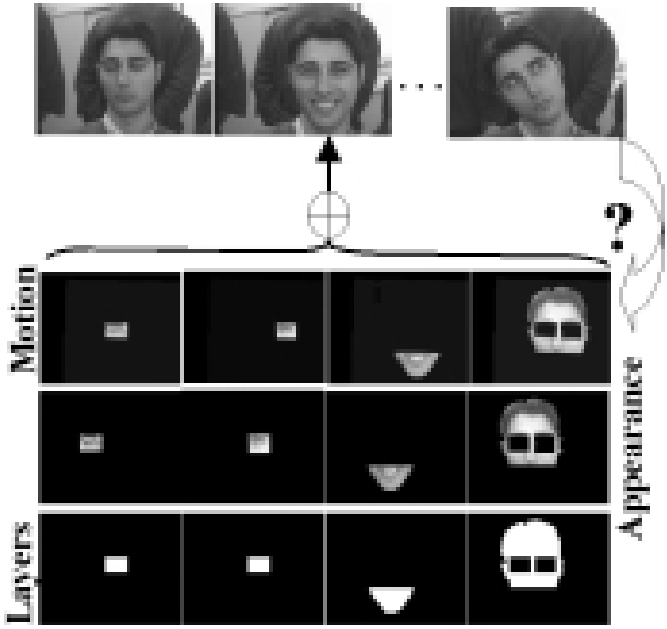


Fig. 3. Generative model for an image sequence. **Layers**: binary masks or layers specified by hand in the first frame. **Appearance**: linear appearance model recovered in the mask areas. **Motion**: transformation between the eigenspace and the images.

longer be possible to accurately reconstruct one person with the left eye closed and the right open. A similar example has been pointed out by Jebara et *al.* [26]. These facts suggested the idea of constructing modular eigenspaces (ME), that is, performing PCA on patches in the image, originally suggested by Pentland et *al.* [26,30] in the context of face recognition. Working with ME has a number of additional benefits: The first one is that ME will provide more accurate reconstruction of the regions of interest (e.g. eyes and mouth versus hair). Due to the reduction of the dimensionality of the space (the images of each local training set are smaller) the estimation of the covariance is more likely to be full rank and probabilistic appearance methods are better conditioned [30]. Also, the computational cost of computing the eigenspace is smaller.

In principle, the regions of support for independent appearance changes (the masks), could be computed as an eigenspace based segmentation problem. That is, constructing an algorithm which tries to divide the image into connected regions in order to minimize some criterion (e.g. MDL). However, in the case of the face, these regions are quite clear, and a rough approximation is sufficient. Therefore, we define the masks in the first image and they will remain the same for the entire training image sequence (simply performing rigid transformations). Fig. 3 shows one example of how to divide the face into non-overlapping regions (left eye, right eye, the mouth and the rest of the face).

Let $\mathbf{d}_t \in \Re^{d\times 1}$ be the region of d pixels belonging to the face, defined by hand in the first image. $\boldsymbol{\pi}^l = [\pi_1^l\ \pi_2^l\ ...\ \pi_d^l]^T \in \Re^{d\times 1}$ denotes the binary mask for the layer l and it has the same size as the face region (d pixels). Each of the mask's pixels takes a binary

value, $\pi_p^l \in \{0,1\}$ and there is no overlap between masks, that is, $\sum_{l=1}^{L} \pi_p^l = 1 \ \forall\, p$. $\boldsymbol{\pi}^l$ will contain d_l pixels with value 1, which define the spatial domain of the mask l (see Fig. 3) and $\sum_{l=1}^{L} d_l = d$.

Each of these masks will have an associated eigenspace. The graylevel of the patch, or layer l, will be reconstructed by a linear combination of an appearance basis $\tilde{\mathbf{B}}^l$:

$$\mathbf{d}_t = \begin{bmatrix} \mathbf{d}_t^1 \\ \vdots \\ \mathbf{d}_t^L \end{bmatrix} = \begin{bmatrix} \mathbf{B}^1\mathbf{c}_t^1 \\ \vdots \\ \mathbf{B}^L\mathbf{c}_t^L \end{bmatrix} = \sum_{i=1}^{L} (\boldsymbol{\pi}^l \circ \tilde{\mathbf{B}}^l \mathbf{c}_t^l) \tag{2}$$

where $\mathbf{d}_t^l \in \Re^{d_l \times 1}$ is the patch of the layer l and $\mathbf{c}_t^l$ are the appearance coefficients of the layer l at time t. $\mathbf{B}^l = [\mathbf{b}_1^l \ \mathbf{b}_2^l \ ... \ \mathbf{b}_{k_l}^l] \in \Re^{d_l \times k_l}$ are the k_l appearance bases for the l layer. $\tilde{\mathbf{B}}^l \in \Re^{d \times k_l}$ will be equal to $\mathbf{B}^l$ for all pixels where $\pi_p^l = 1$ (i.e. belonging to the lth mask) and otherwise can take an arbitrary value.

3.2 Motion

If the face to be tracked can be considered to be far away from the camera, it can be approximated by a plane [5]. The motion of planar surfaces, under orthographic or perspective projection, can be recovered with a parametric model of 6 or 8 parameters. The rigid motion of the face will be parameterized by an affine model:

$$\mathbf{f}_1(\mathbf{x}_p, \mathbf{a}_t^l) = \begin{bmatrix} a_{1t}^l \\ a_{4t}^l \end{bmatrix} + \begin{bmatrix} a_{2t}^l & a_{3t}^l \\ a_{5t}^l & a_{6t}^l \end{bmatrix} \begin{bmatrix} x_p - x_c^l \\ y_p - y_c^l \end{bmatrix} \tag{3}$$

where $\mathbf{a}_t^l = [a_{1t}^l \ a_{2t}^l \ \ldots \ a_{6t}^l]^T$ denotes the vector of motion parameters of the mask l at time t, $\mathbf{x}_p = [x_p \ \ y_p]^T$ are the Cartesian coordinates of the image at the pth pixel and $\mathbf{x}_c^l = [x_c^l \ \ y_c^l]^T$ is the center of the l layer. Throughout the paper, we will assume that the rigid motion of all the modular eigenspaces (w.r.t. the center of the face) is the same. That is, $\mathbf{a}_t^1 = \mathbf{a}_t^2 \ldots = \mathbf{a}_t^L$.

Once the appearance and motion models have been defined, the graylevel of each pixel of the image $\mathbf{d}_t$ is explained as a superposition of a layer-subspace plus a warping, see Fig. (3); that is, $\mathbf{d}_t = \sum_{l=1}^{L} \left(\boldsymbol{\pi}^l \circ \tilde{\mathbf{B}}^l \mathbf{c}_t^l\right)\left(\mathbf{f}_1(\mathbf{x}, \mathbf{a}_t^l)\right)$, where $\mathbf{x} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_d]^T$. The notation $\left(\boldsymbol{\pi}^l \circ \tilde{\mathbf{B}}^l \mathbf{c}_t^l\right)\left(\mathbf{f}_1(\mathbf{x}, \mathbf{a}_t^l)\right)$ means that the reconstructed image within the mask, $\left(\boldsymbol{\pi}^l \circ \tilde{\mathbf{B}}^l \mathbf{c}_t^l\right)$, is warped (or indexed) by the parameterized transformation $\mathbf{f}_1(\mathbf{x}, \mathbf{a}_t^l)$. Observe that this image model is essentially the same as previous appearance representations [4,10,16] but with the addition of modular eigenspaces.

4 Learning the Model Parameters

Once the model has been established, in order to automatically learn the PSFAM, it is necessary to learn the model parameters. In this section we describe the learning procedure; that is, given an observed image sequence ($\mathbf{D} \in \Re^{d \times N}$) (N is the number of images) and L masks in the first image ($\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \ \ldots \ , \boldsymbol{\pi}^L\}$), finding the parameters $\mathcal{B}$, $\mathcal{C}$, $\mathcal{A}$ and $\boldsymbol{\sigma}$,

which best reconstruct the sequence. Where $\mathcal{A} = \{\mathbf{A}^1,\ \mathbf{A}^2,\ \ldots,\mathbf{A}^L\}$ is the set of motion parameters of all the layers in all the image frames. $\mathbf{A}^i = [\mathbf{a}_1^i\ \mathbf{a}_2^i\ \cdots\ \mathbf{a}_N^i]$ is the matrix which contains the motion parameters for each image in the ith layer. Analogously, $\mathcal{C} = \{\mathbf{C}^1,\ \mathbf{C}^2,\ \ldots,\mathbf{C}^L\}$ where $\mathbf{C}^i = [\mathbf{c}_1^i\ \mathbf{c}_2^i\ \cdots\ \mathbf{c}_N^i]$ and $\mathcal{B} = \{\mathbf{B}^1,\ \mathbf{B}^2,\ \ldots,\mathbf{B}^L\}$.

At this point, learning the model parameters can be posed as a minimization problem. In this case the residual will be the difference between the image at time t and the reconstruction with the model. In order to take into account outlying data, we introduce a robust objective function, minimizing $E_{rereg}(\mathcal{B},\mathcal{C},\mathcal{A};\boldsymbol{\sigma})$:

$$E_{rereg} = \sum_{t=1}^{N}\sum_{p=1}^{d}\rho\left(d_{pt} - \sum_{l=1}^{L}\left(\pi_p^l\sum_{j=1}^{k}b_{pj}^l c_{jt}^l\right)\left(\mathbf{f}_1(\mathbf{x}_p,\mathbf{a}_t^l)\right),\sigma_p\right) \quad (4)$$

where b_{pj}^l is the pth pixel of the jth basis of $\mathbf{B}^l$ for the layer l. Observe that the pixel residual is *filtered* by the Geman-McClure robust error function [22] given by $\rho(x,\sigma_p) = \frac{x^2}{x^2+\sigma_p^2}$, in order to reduce the influence of outlying data. σ_p is a parameter that controls the convexity of the robust function and is used for the deterministic annealing in a Graduated Non-Convexity algorithm [4,7] (we do not minimize E_{rereg} over σ_p). Benefits of the robust formulation in the subspace related problems are explained elsewhere [14]. Observe that the previous Eq. (4) is similar to *Eigentracking* [4] but is applied in image patches (masks). It is also similar to AAM [10] or *Flexible Eigentracking* [16] without shape constraints. However, in contrast to these approaches [4,10,16], in E_{rereg} the appearance bases $\mathcal{B}$ are now treated as parameters to be estimated.

4.1 Stochastic State Initialization

The error function E_{rereg}, Eq. (4), is non-convex and, without a good starting point, gradient descent methods are likely to get trapped in local minima. When computing the motion parameters, as in the case of optical flow, a coarse-to-fine strategy [4,11] can help to avoid local minima. Although a coarse-to-fine strategy is helpful, this technique is insufficient in our case, since in real image sequences the size of the face can be small in comparison to the number of pixels in the background, and large motions can be performed (e.g. in the sequences that we tried, the face can move more than 20 pixels from frame to frame). In order to cope with such real conditions, we explore the use of stochastic methods such as Simulated Annealing (SA) [2], Genetic Algorithms (GA) [27,29] or CONDENSATION (particle filtering) [6,17] for motion estimation. Although the techniques are very similar computationally speaking, here we make use of GA [29] within a coarse-to-fine strategy.

Given the first image of the sequence, we manually initialize the layers or masks at the highest resolution level and assign the graylevel to the first bases for each layer $\mathcal{B} = \{\mathbf{b}_1^1,\ \ldots,\mathbf{b}_1^L\}$. Afterwards, we take the subset of the m frames closest in time (typically m=15), and use a GA for a first estimation of the motion parameters which minimize Eq. (4) (the least squares version). Given the genetic estimation of these parameters, we recompute the bases $\mathcal{B}$ which preserve 60% of the energy. This initialization procedure is repeated until all the frames in the image sequence are initialized. The procedure is summarized as:

- Manual initialization in the first frame.
  - Initialize the mask in the image $\mathbf{d}_1$.
  - Initialize the bases $\mathcal{B} = \{\mathbf{b}_1^1, ..., \mathbf{b}_1^L\}$ with the graylevel values of $\mathbf{d}_1$.
- Stochastic initialization of the motion and appearance parameters for $\mathbf{D}$.
  - for i=2 : m : N (Matlab notation)
  - Run the GA for computing the motion and appearance parameters in $\{\mathbf{d}_i, \ \ldots \ , \mathbf{d}_{i+m}\}$.
  - Add basis and recompute the modular eigenspace, $\mathcal{B}$. Keep the number of bases which preserve 60% of the energy.
  - end

The GA uses 300 individuals over 13 generations for each frame. The selection function that we use is the normalized geometric ranking, which defines the probability of one individual as $P_i = \frac{q}{1-(1-q)^P}(1-q)^{(r-1)}$ where q is the probability of selecting the best individual, r is the rank of the individual, and P the population size. See [29] for a more detailed explanation of the GA. At the beginning, q has a low value, and it is successively increased over generations acting as a temperature parameter in the deterministic annealing [4,7] for improving the local search. The crossover process is a convex combination between two samples, i.e. $\alpha * cromosome_1 + (1-\alpha) * cromosome_2$ where $1 \geq \alpha \geq 0$. The genetic operator is a simple Gaussian random perturbation, which also depends on the temperature parameter.

4.2 Robust Deterministic Learning

The previous section describes a method for computing an initial estimate of the parameters $\mathcal{B}$, $\mathcal{C}$, $\mathcal{A}$. In order to improve the solution and achieve sub-pixel accuracy, a normalized gradient descent algorithm for minimizing Eq. (4) has been employed in [12]. Alternatively (and conveniently) we can reformulate the minimization problem as one of iteratively reweighted least-squares (IRLS), which provides an approximate, iterative, solution to the robust M-estimation problem [28]. For a given $\boldsymbol{\sigma}$, a matrix $\mathbf{W} \in \Re^{d \times N}$, which contains the positive weights for each pixel and each image, is calculated for each iteration as a function of the previous residuals $e_{pi} = d_{pt} - (\pi_p^l \sum_{j=1}^k b_{pj}^l c_{jt}^l)\big(\mathbf{f}_1(\mathbf{x}_p, \mathbf{a}_t^l)\big)$. Each element, w_{pi} (p^{th} pixel of the ith image) of $\mathbf{W}$ will be equal to $w_{pi} = \psi(e_{pi}, \sigma_p)/e_{pi}$, where $\psi(e_{pi}, \sigma_p) = \frac{\partial \rho(e_{pi}, \sigma_p)}{\partial e_{pi}} = \frac{2 e_{pi} \sigma_p^2}{(e_{pi}^2 + \sigma_p^2)^2}$, [**?**]. Given an initial error, the weight matrix $\mathbf{W}$ is computed and the Eq. (4) becomes

$$E_{wereg}(\mathcal{B}, \mathcal{C}, \mathcal{A}; \boldsymbol{\sigma}) = \sum_{t=1}^{N} ||\mathbf{d}_t - \sum_{l=1}^{L} \big(\boldsymbol{\pi}^l \circ \tilde{\mathbf{B}}^l \mathbf{c}_t{}^l\big)\big(\mathbf{f}_1(\mathbf{x}, \mathbf{a}_t^l)\big)||_{\mathbf{W}_t}^2 \tag{5}$$

$$= \sum_{t=1}^{N} \sum_{l=1}^{L} ||\mathbf{d}_t^l\big(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^l)\big) - \mathbf{B}^l \mathbf{c}_t^l||_{\mathbf{W}_t^l}^2 \tag{6}$$

where $\mathbf{f}$ will warp the image w.r.t. the eigenspace, whereas $\mathbf{f}_1$ warps the basis to the image. Observe that $\mathbf{f}$ will be approximately the inverse of $\mathbf{f}_1$. Recall that $||\mathbf{d}||_{\mathbf{W}}^2 = \mathbf{d}^T \mathbf{W} \mathbf{d}$ is a weighted norm. $\mathbf{W}_t \in \Re^{d \times d}$ is a diagonal matrix, such that the diagonal elements are

the tth column of $\mathbf{W}$. $\mathbf{W}_t^l \in \Re^{d_l \times d_l}$ is diagonal matrix, where the diagonal is created by the elements of the t column of $\mathbf{W}$ which belong to the lth layer. Observe that if $\mathbf{W}$ is a matrix with all ones we have the least-squares solution.

Eq. (6) provides the formulation for robust *eigen-registration* or robust parameterized component analysis. Minimizing (6) with respect to the parameters gives a subspace that is invariant to the allowed geometric transformations and robust to outliers on a pixel level. Clearly, finding the minimum is a challenge and the process for doing so is described below.

Notice that, if the motion parameters are known, computing the basis ($\mathcal{B}$) and the coefficients ($\mathcal{C}$) translates into a weighted bilinear problem. In order to compute the updates of the bases and coefficients in closed form in the simplest way, we use the following observation:

$$E_{wereg}(\mathcal{B}, \mathcal{C}, \mathcal{A}; \boldsymbol{\sigma}) = \sum_{t=1}^{N} \sum_{l=1}^{L} ||(\mathbf{d}_w^l)_t - \mathbf{B}^l \mathbf{c}_t^l||^2_{\mathbf{W}_t^l} \tag{7}$$

$$= \sum_{p=1}^{d_l} \sum_{l=1}^{L} ||(\mathbf{d}_w^l)^p - (\mathbf{C}^l)^T (\mathbf{b}^l)^p||^2_{(\mathbf{W}^l)^p} \tag{8}$$

where $(\mathbf{d}_w^l)_t$ is the warped image $\mathbf{d}_t^l\big(\mathbf{f}(\mathbf{x}, \mathbf{a}_t)\big)$ and it is the tth column of the matrix $\mathbf{D}_w$ (just the d_l elements corresponding to the l layer). Recall that $(\mathbf{d}_w^l)^p$ is a column vector which corresponds to the pth row of the matrix $\mathbf{D}_w$ and that $(\mathbf{W}^l)^p$ is a diagonal matrix which contains the pth row of the matrix $\mathbf{W}$ of the layer l.

Minimizing Eq. (6) is a non-linear optimization problem w.r.t. the motion parameters. Following previous work on motion estimation [4,24], we linearize the variation of the function, using a 1st order Taylor series approximation. Without loss of generality, rather than linearizing the transformation which warps the eigenspace towards the image $\mathbf{f}_1(\mathbf{x}, \mathbf{a}_t)$, we linearize the transformation which aligns the incoming image w.r.t. the eigenspace $\mathbf{f}(\mathbf{x}, \mathbf{a}_t)$ (see Eq. 6). Expanding, $\mathbf{d}_t^l(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^{l0} + \boldsymbol{\Delta}\mathbf{a}_t^l))$ in the Taylor series about the initial estimation of the motion parameters $\mathbf{a}_t^{l0}$ (given by the GA):

$$\mathbf{d}_t^l(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^{l0} + \boldsymbol{\Delta}\mathbf{a}_t^l)) = \mathbf{d}_t^l(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^{l0})) + \mathbf{J}_t^l \boldsymbol{\Delta}\mathbf{a}_t^l + h.o.t. \tag{9}$$

where $\mathbf{J}_t^l$ is the Jacobian at time t of the lth layer and $h.o.t.$ denotes the higher order terms. Observe that after the linearization the function E_{wereg}, Eq. (6), is convex in each of the parameters. For instance, $\boldsymbol{\Delta}\mathbf{a}_t$ can be computed in closed form solving a linear system of Equations:

$$\begin{bmatrix} \big((\mathbf{J}_t^1)^T \mathbf{W}_t^1 \mathbf{J}_t^1\big) \\ \big((\mathbf{J}_t^2)^T \mathbf{W}_t^2 \mathbf{J}_t^2\big) \\ . \\ . \\ \big((\mathbf{J}_t^L)^T \mathbf{W}_t^L \mathbf{J}_t^L\big) \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta}\mathbf{a}_t \end{bmatrix} = \begin{bmatrix} (\mathbf{J}_t^1)^T \mathbf{W}_t^1 (\mathbf{d}_t(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^0)) - \mathbf{B}^1 \mathbf{c}_t^1) \\ (\mathbf{J}_t^2)^T \mathbf{W}_t^2 (\mathbf{d}_t(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^0)) - \mathbf{B}^2 \mathbf{c}_t^2) \\ . \\ . \\ (\mathbf{J}_t^1)^T \mathbf{W}_t^L (\mathbf{d}_t(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^0)) - \mathbf{B}^L \mathbf{c}_t^L) \end{bmatrix}$$

where recall that $\mathbf{W}_t^l$ is a matrix containing the weights for the layer l at time t. In this case, we have assumed that $\boldsymbol{\Delta}\mathbf{a}_t^l = \boldsymbol{\Delta}\mathbf{a}_t \ \forall\, l$ because the modular eigenspaces share the motion parameters, therefore we drop the superscript l in the motion parameters.

However, E_{wereg} is no longer convex as a joint function of these variables. In order to learn the parameters, we break the estimation problem into two sub-problems. We alternate between estimating $\mathcal{C}$ and $\mathcal{A}$ with a Gauss-Newton scheme [4] and learning the basis $\mathcal{B}$ and scale parameters $\boldsymbol{\sigma}$ until convergence, see [14] for more detailed information. Each of the updates for $\mathcal{C}, \mathcal{A}$ and $\mathcal{B}$ are done in closed form. This multi-linear fitting algorithm monotonically reduces the cost function, although it is not guaranteed to converge to the global minimum. We also use a coarse-to-fine strategy [4,11] to cope with large motions and to improve the efficiency of the algorithm. Towards that end, a Gaussian image pyramid is constructed. Each level of the pyramid is constructed by taking the image at the previous resolution level, convolving it with a Gaussian filter and sub-sampling. Details of the learning method are given below.

- For each image resolution level until convergence of $\mathcal{C}, \mathcal{A}$ and $\mathcal{B}$
  - Until convergence of $\mathcal{C}, \mathcal{A}$
    - Until convergence of $\mathcal{A}$, rewarp $\mathbf{D}$ to $\mathbf{D}_w$ and update the motion parameters for each layer by computing:
      $(\mathbf{a}_t^l) = (\mathbf{a}_t^l) + \boldsymbol{\Delta}\mathbf{a}_t \;\; \forall \; l = 1 \; \ldots \; L$
    - Update the appearance coefficients for each layer and each image
      $((\mathbf{B}^l)^T \mathbf{W}_t^l \mathbf{B}^l)\mathbf{c}_t^l = (\mathbf{B}^l)^T \mathbf{W}_t^l \mathbf{d}_t(\mathbf{f}(\mathbf{x}, \mathbf{a}_t^l)) \;\; \forall \, l = 1 \ldots L, \; \forall \; t = 1 \ldots N$
  - Update $\mathcal{B}$ preserving 85% of the energy, solving:
    $(\mathbf{C}^l(\mathbf{W}^l)^p(\mathbf{C}^l)^T)(\mathbf{b}^l)^p = \mathbf{C}^l(\mathbf{W}^l)^p(\mathbf{d}_w^l)^p \;\; \forall \, l = 1 \; \ldots \; L, \; \forall \, p = 1 \; \ldots \; d_l$
  - Recompute the error, weights $(\mathbf{W})$ and the scale statistics $\boldsymbol{\sigma}$ [14].
- Propagate the motion parameters to the next resolution level [4,11] (the translation parameters are multiplied by a factor 2). Once the motion parameters are propagated the bases are recomputed.

5 Experiments and Applications

5.1 Automatic Learning of Eigeneyes

Eyes are one of the key elements in Vision Based Human Computer Interaction (VBHCI). In this experiment, we automatically learn a person-specific "eigeneye" without any manual cropping, except in the first image. We assume that during the training process the person is not moving far away (around 5-8 pixels) from the first frame.

Recall that Fig. 1 illustrates the eigen-registration method and shows a few images from a training set. In the first frame, we manually select the mask for the eyes, face, and background (in Fig. 6 the regions are represented). In this case, because we are assuming a small motion, the GA has not been applied for initializing the algorithm, and we minimize Eq. (6) with the robust deterministic learning method proposed, with a coarse-to-fine strategy (2 levels) over the entire training set (around 300 frames). We have presupposed that the data had few outliers, so we give $\boldsymbol{\sigma}$ a high value.

In Fig. 2 the reconstruction of some right eye training images are shown. Fig. 2.a shows the original images, Fig. 2.b plots the reconstructed images with non-aligned basis (assuming that the person is not moving, each layer does not change over the sequence). Fig. 2.c represents the reconstructed images after minimizing Eq. (6). As

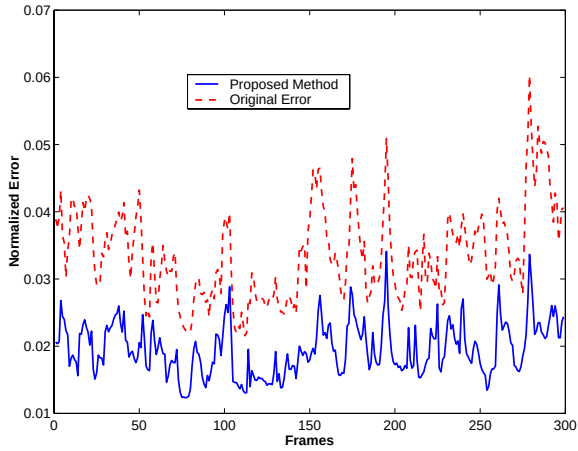


Fig. 4. Normalized reconstruction error versus number of frames.

can be observed, the reconstruction is visually better when we align and simultaneously learn the basis (with the same number of bases). We have chosen the number of bases which preserve 85% of the energy (seven bases in this case).

Fig. 4 shows the normalized reconstruction error of the face for the original training set $\mathbf{D}$ and the aligned training set $\mathbf{D}_w$ with the same number of bases. The normalized reconstruction error for the image i, will be $r_i = \frac{||\mathbf{d}_i - \mathbf{B}\mathbf{c}_i||}{||\mathbf{d}_i||}$. Observe that, in general, due to the warping process, $\mathbf{D}$ and $\mathbf{D}_w$ have different energy. As can be observed, the reconstruction error is lower in the case of iteratively warping the image w.r.t. the eigenspace and computing the eigenspace itself (continuous line) than when this procedure is not performed (dotted line).

Once the eigeneyes have been learned, tracking can be performed with deterministic techniques [4,10] or stochastic ones [17]. Fig. 5 shows some results of eye tracking, using an eigen-eye model learned with the previous method. The tracking system exploits a stochastic tracking framework, runs at 10 Hz, and is able to track the eye position and configuration of the user (for further details see [17]). Observe, that when the person performs changes in pose the tracker eventually loses track. However, due to its stochastic nature, the tracker can recover after a few frames. The method is being tested for driver fatigue detection purposes.

5.2 Automatic Face Learning

In this experiment, we explore the possibility of learning the entire face model, including modeling mouth changes. The modular face model is composed of 4 layers, see Fig. 6. Some frames of the sequence (240×320 pixels and 320 frames) are shown in Fig. (7.a). In this sequence, the person can suddenly move more than 20 pixels from frame to frame, along with large scale and rotation changes. In this case, we make use of the stochastic initialization with the GA for an initial estimation of the parameters.

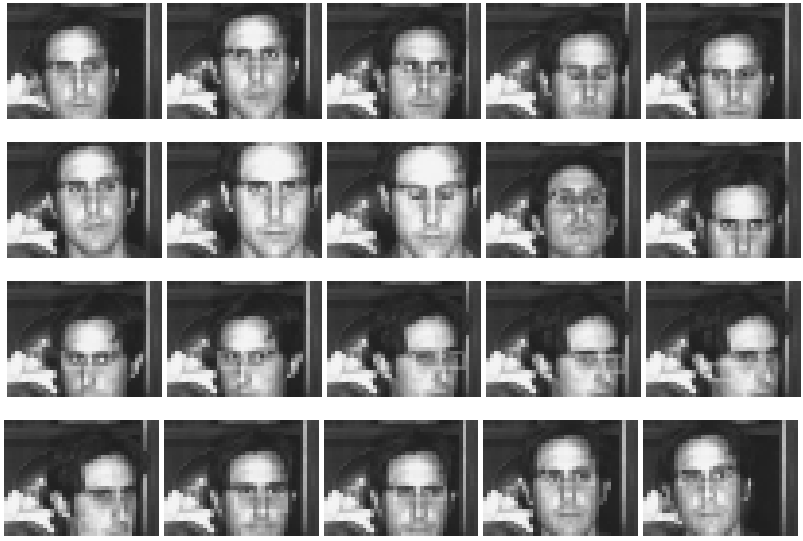
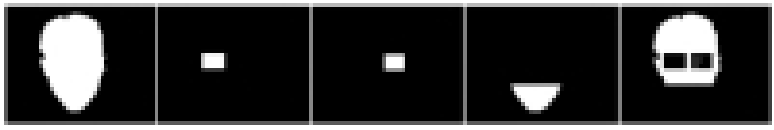


Fig. 5. Frontal eye-tracking using an eye model learned with parameterized component analysis.



a *b* *c* *d* *e*

Fig. 6. a) Mask for the whole face. b),c),d),e) Mask of the eyes, mouth and the rest of the face.

Fig. (7.b) shows the normalized face (w.r.t. the first frame) reconstructed with the learned bases after the convergence of the algorithm. Recall that we have just initialized the layers in the first image and no previous appearance model was given. The faces in Fig. 7.b display variations due simply to appearance and not to motion. In this case we preserve 85% of the energy in each modular eigenspace. Each face image (Fig. 7.b) can be reconstructed with 23 parameters and further research needs to be done in order to study the viability to apply it for video-conferencing.

5.3 Facial Animation

In this experiment we animate one face given another. In general it is hard to model and animate faces, even when they are cartoon characters. Usually complex models encoding the physical underlying musculature of the face are used (e.g. Candide model [8]). We use a PSFAM to parameterize the expression using modular PCA, and parameterized component analysis to learn the PSFAM.

Fig. 8 shows frames of a virtual female face animated by the appearance of the input male face. The first column shows the original input stream ($\hat{\mathbf{D}}$); the second one, ($\mathbf{D}$), is

Fig. 7. a) Original image sequence. b) Normalized face.

the result of animating the face with Asymmetric Coupled Component Analysis (ACCA) [13] plus the affine motion of the head. As we can observe this approach allows us to model the rich texture present on the face providing fairly realistic animations. See [13] for further information.

Fig. 8. a) Original face. b) Animated virtual face.

6 Discussion and Future Work

This paper has introduced robust parameterized component analysis to learn modular subspaces that are invariant to various geometric transformations. The robust and continuous formulation of the problem extends previous work and has proven effective for learning low dimensional models of human faces. We have illustrated the method with several applications of face modeling, tracking, and animation. In particular we have shown how the method can simultaneously construct an eigenspace while aligning misregistered training images. The resulting model better describes the training data and the method can be applied to arbitrary parameterized deformations. Due to the complexity of the objective function, a stochastic initialization of the algorithm has proven to be essential for avoiding local minima.

Observe that parameterized component analysis, always improves the quality of the appearance basis if some misalignment exists in the training set (due to manual cropping, motion of the person, etc). Although we have presented a method for learning person-specific models, the method may also be useful when improving the basis of a training set containing faces from different people. As described here, the method is appropriate for learning appearance models in an off-line process. The method could be extended to be useful for on-line learning by simply replacing the closed form solution with a gradient descent algorithm or any adaptive method. Based on the recent extension of EigenTracking [4] to deal with Support Vector Machines [1] it would also be interesting to consider extending our method to other statistical learning techniques like SVM or independent component analysis.

Modeling the face with modular eigenspaces coupled by the motion can result in the loss of correlations between the parts (e.g. when smiling some wrinkles appear in the eye region). Now we are working on modeling the face with symmetric coupled component analysis [13] and are experimenting with hierarchical component analysis in which one set of coefficients models the coupling between regions while each individual region has its own coefficients for local variation.

Finally, the work presented in this paper on automatic learning of 2D PSFAMs has the limitation of being applicable to some particular view of the face, in this case the frontal view. We are working on extending the PSFAM to model 3D changes within the same continuous optimization framework described here. Also, we are exploring improving the 2D model (e.g. adding a more complex geometric transformation that takes into account chin deformations).

Videos with the results for all the experiments performed in this paper can be downloaded from http://www.salleURL.edu/∼ftorre/.

Acknowledgements. The first author has been partially supported by a grant from the the Direcció General de Recerca of the Generalitat of Catalunya (#2001BEAI200220). This research was also supported by the DARPA HumanID project (ONR contract N000140110886) and a gift from the Xerox Foundation. We would like to thank Allan Jepson for discussions on robust PCA and eigen-registration.

References

1. S. Avidan. Support vector tracking. In *Conference on Computer Vision and Pattern Recognition*, 2001.
2. M. Betke and N. Makris. Fast object recognition in noisy images using simulated annealing. In *International Conference Computer Vision*, pages 523–530, 1994.
3. M. J. Black, D. J. Fleet, and Y. Yacoob. Robustly estimating changes in image appearance. *Computer Vision and Image Understanding*, 78(1):8–31, 2000.
4. M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of objects using view-based representation. *International Journal of Computer Vision*, 26(1):63–84, 1998.
5. M. J. Black and Y. Yacoob. Recognizing facial expressions in image sequences using local parameterized models of image motion. *International Journal of Computer Vision*, 25(1):23–48, 1997.
6. A. Blake and M. Isard. *Active Contours*. Springer Verlag, 1998.
7. A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press series, Massachusetts, 1987.
8. M. Brand. Voice puppetry. In *SIGGRAPH*, pages 21–28, 1999.
9. R. Cipolla and A. Pentland. *Computer vision for Human-Machine Interaction*. Cambridge university press, 1998.
10. T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *European Conference Computer Vision*, pages 484–498, 1998.
11. T. F. Cootes and C. J. Taylor. Statistical models of appearance for com-puter vision. In *World Wide Web Publication*, February 2001. (Available from http://www.isbe.man.ac.uk/bim/refs.html).
12. F. de la Torre. Automatic learning of appearance face models. In *Second International Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems*, pages 32–39, 2001.

13. F. de la Torre and M. J. Black. Dynamic coupled component analysis. In *Computer Vision and Pattern Recognition*, 2001.
14. F. de la Torre and M. J. Black. Robust principal component analysis for computer vision. In *International Conference on Computer Vision*, pages 362–369, 2001.
15. F. de la Torre, S. Gong, and S. McKenna. View alignment with dynamically updated affine tracking. In *Int. Conf. on Automatic Face and Gesture Recognition*, pages 510–515, 1998.
16. F. de la Torre, J. Vitrià, P. Radeva, and J. Melenchón. Eigenfiltering for flexible eigentracking. In *International Conference on Pattern Recognition*, pages 1118–1121, Barcelona, 2000.
17. F. de la Torre, Y. Yacoob, and L. Davis. A probabilisitc framework for rigid and non-rigid appearance based tracking and recognition. In *Int. Conf. on Automatic Face and Gesture Recognition*, pages 491–498, 2000.
18. K. I. Diamantaras.*Principal Component Neural Networks (Therory and Applications)*. John Wiley & Sons, 1996.
19. G. J. Edwards, A. Lanitis, C. Taylor, and T. F. Cootes. Statistical models of face images-improving specificity. *Image and Vision Computing*, 16:203–211, 1998.
20. G. J. Edwards, C. J. Taylor, and T.F. Cootes. Improving identitification performance by integrating evidence from sequences. In *Computer Vision and Pattern Recognition*, pages 486–491, 1999.
21. B. J. Frey and N. Jojic. Transformation-invariant clustering and dimensionality reduction. *Submitted to IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2000.
22. S. Geman and D. McClure. Statistical methods for tomographic image reconstruction. *Bulletin of the International Statistical Institute*, LII:4:5, 1987.
23. S. Gong, S. Mckenna, and A. Psarrou. *Dynamic Vision: From Images to Face Recognition.* Imperial College Press, 2000.
24. G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.
25. F. Hampel, E. Ronchetti, P. Rousseeuw, and W. Stahel. *Robust Statistics: The Approach Based on Influence Functions.* Wiley, New York., 1986.
26. T. Jebara, K. Russell, and A. Pentland. Mixtures of eigenfeatures for real-time structure from texture. In *International Conference on Computer Vision*, 1998.
27. A. Lanitis, A. Hill, T. F. Cootes, and C. J. Taylor. Locating facial feature using genetic algorithms. In *International Conference on Digital Signal Processing*, pages 520–525, 1995.
28. G. Li. Robust regression. In D. C. Hoaglin, F. Mosteller, and J. W. Tukey, editors, *Exploring Data, Tables, Trends and Shapes.* John Wiley & Sons, 1985.
29. M. Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, 1996.
30. B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *Pattern Analysis and Machine Intelligence*, 19(7):137–143, July 1997.
31. S. K. Nayar and T. Poggio. *Early Visual Learning.* Oxford University Press, 1996.
32. R. P. N. Rao. Development of localized oriented receptive fields by learning a translation-invariant code for natural images. *Network: Comput. Neural Systems*, 9:219–234, 1998.
33. H. Schweitzer. Optimal eigenfeature selection by optimal image registration. In *Conference on Computer Vision and Pattern Recognition*, pages 219–224, 1999.
34. M. Turk and A. Pentland. Eigenfaces for recognition.*Journal Cognitive Neuroscience*, 3(1):71–86, 1991.