Biomedicine and Nursing

Websites:
http://www.nbmedicine.org
http://www.sciencepub.net/nurse

Emails:
editor@sciencepub.net
nbmeditor@gmail.com

MARSLAND PRESS
Multidisciplinary Academic Journal Publisher

Probabilistic PCA for texture modeling of Adaptive Active Appearance Models and its application for head pose estimation

Navid Mahmoudian Bidgoli [1], Abolghasem A. Raie [1], M. Naraghi [2]

[1.] Electrical Engineering Department, Amir Kabir University of Technology, Tehran, Iran
[2.] Mechanical Engineering Department, Amir Kabir University of Technology, Tehran, Iran
navid_mahmoudian@aut.ac.ir

Abstract: This paper suggests a 3D real-time monocular head pose tracker in which Active Appearance Models (AAM) are used to extract facial features. In order to improve texture model, two probabilistic approaches are proposed for principal component analysis in the presence of missing values and are evaluated. it is finally observed that applying Bayesian model results in improving model fitting. On the other hand, contrary to the common assumption in AAM, the gradient matrix must not suppose to be constant. In this investigation a method is suggested in which gradient matrix is adapted with new images during model fitting of video sequences as much as possible. In the next step, by means of suggested methods, operator's head pose will be estimated by POSIT algorithm and by implementing this system on a People Bot robot, enhancement of the interaction between human and robot is presented in order to control the orientation of robot camera.
[Navid Mahmoudian Bidgoli, Abolghasem A. Raie, M. Naraghi. **Probabilistic PCA for texture modeling of Adaptive Active Appearance Models and its application for head pose estimation.** *Biomedicine and Nursing* 2020;6(1): 59-73]. ISSN 2379-8211 (print); ISSN 2379-8203 (online). http://www.nbmedicine.org. 8. doi:10.7537/marsbnj060120.08.

Keywords: Head pose estimation, Human-Robot Interaction (HRI), Probabilistic PCA, missing values, Active appearance models (AAM)

1 Introduction

Most of the individuals' behavioral data could be obtained by investigating their head pose. Ability to distinguish and track individuals' head and facial characteristics are very useful and applied in video sequences, including human-robot interaction [1], Auto security system for drivers' awareness [2], face identification systems and smart environments [3].

Nowadays, considering surveillance systems requirements, it is necessary for operators to interact with robots and other intelligent devices easily. Since individual's head orients toward subjects which are interesting to them initially, head pose estimation acts as a filter by which the importance of each object is determined in the camera FOV (Field Of View).

So far many researches have been done in the field of head pose estimation including a survey on this subject [4]. Dornaika and Raducano used a head pose estimation system on AIBO robot, [1], through online appearance models [5]. One limitation of their method is that recommended system is applicable solely by specific persons.

In this paper 3D estimation of face orientation and position of individuals (6 DOF) and its application for facilitating human-robot interaction by means of a camera is investigated. In this manner, a 3D rigid face model is required in which Candide-3 face model [6] and POSIT algorithm [7] have been used. Since POSIT algorithm uses 3D coordinates of Candide-3 and its correspondent points in 2D images, in order to obtain 2D points of the face in that image, Active Appearance Models [8] are used.

AAM fitting is based on texture residual produced by the model. Therefore, it is expected that preparing a precise and perfect model of the face texture would be effective for model fitting improvement. So in this paper, two possible approaches are suggested for PCA, one with and the other without considering prior probability for parameters.

Unlike the current assumption in Active Appearance Models fitting (constant Jacobian matrix), an approach is presented in order to update matrix relative to the current frame of video sequences.

The research objective is facilitating robot surveillance and supervision system through operator's head pose estimation. We will show more precisely that how robot camera direction could be controlled through head pose estimation.

This paper is organized as follows: in section 2, Active Appearance Models are introduced. In the 3rd one probabilistic approaches for texture modeling are

described. In section 4, a method is suggested for updating Jacobian matrix and finally in section 5 head pose estimation system is described by means of POSIT algorithm and its implementation on PeopleBot robot.

2 Active Appearance Models (AAM)

Active appearance model [9] is a way to match statistical models of shape and texture to unseen images. In this model, introduced by Cootes et al (1998) [8], variation in shape and texture of face are captured from a series of representative training set and by combining them, Active Appearance Models are produced.

AAM algorithm is an efficient method for model parameter estimation to make the model synthesized image similar to the target image as much as possible. To achieve appropriate statistical model, IMM database has been used, which contains 240 annotated face images [10].

2.1 Obtaining statistical model

The representation used for a n-point shape is given by $\boldsymbol{s} = [x_1, \dots, x_{n-1}, x_n, y_1, \dots, y_{n-1}, y_n]^T$. The shapes of training images are aligned by means of Procrustes Analysis method. Finally, shape model will be obtained by applying PCA [11] to the aligned data:

$$\tilde{\boldsymbol{s}} \approx \bar{\boldsymbol{s}} + \boldsymbol{\Phi}_s \boldsymbol{b}_s \quad (1)$$

In which $\bar{\boldsymbol{s}}$ is the mean shape, the orthonormal columns of $\boldsymbol{\Phi}_s$ represents modes resulted from shape statistical model and vector $\boldsymbol{b}_s$ is shape parameters. Shape mode numbers are selected so that 95% of the variance in the training set will be explained. Since the precise amount of shape $\boldsymbol{s}$ is specified in training set, vectors $\boldsymbol{b}_s$ relevant to database shapes are shown by $\boldsymbol{b}_{s_{opt}}$ and calculated as followed:

$$\boldsymbol{b}_{s_{opt}} = \boldsymbol{\Phi}_{\mathrm{s}}^{\mathrm{T}}(\boldsymbol{s} - \bar{\boldsymbol{s}}) \quad (2)$$

Since the object shape is not enough to obtain complete model of appearance, it is also required to model the object texture (pixel intensities) in the image. The texture is obtained by texture mapping from the triangular 2-D mesh covering the face in the input image to the reference shape -mean shape image with given resolution- using a piecewise affine warp.

With m gray scale sampled pixel from the object image, texture is displayed as $\mathbf{g} = [\mathrm{g}_1, \mathrm{g}_2, \dots, \mathrm{g}_m]^{\mathrm{T}}$. In order to decrease light and contrast effects, zero-mean unit-variance normalization is used for texture vector (Figure 1). By applying PCA to normalized textures, texture model would be as:

$$\tilde{\boldsymbol{g}} \approx \bar{\boldsymbol{g}} + \boldsymbol{\Phi}_{\mathbf{g}} \boldsymbol{b}_g \quad (3)$$

In which $\tilde{\boldsymbol{g}}$ is an approximation of normalized texture vector $\boldsymbol{g}$, orthonormal columns of Matrix $\boldsymbol{\Phi}_{\mathbf{g}}$ represents modes resulted from texture statistical model and $\mathrm{t_g}$-dimensional vector $\boldsymbol{b}_g$ is texture parameters. Similar to shape model, number of texture modes are selected to explain 95% of the variation in training set. Vectors $\boldsymbol{b}_g$ relevant to database images are displayed by $\boldsymbol{b}_{g_{opt}}$ and is calculated as bellow:

$$\boldsymbol{b}_{g_{opt}} = \boldsymbol{\Phi}_{\mathbf{g}}^{\mathrm{T}}(\boldsymbol{g} - \bar{\boldsymbol{g}}) \quad (4)$$

In order to remove correlations between model parameters of the shape and texture, principal component analysis method for the 3rd time was applied to data of:

$$\boldsymbol{b}_{\boldsymbol{i}} = \begin{pmatrix} \boldsymbol{W}_s \boldsymbol{b}_{s_{opt_i}} \\ \boldsymbol{b}_{g_{opt_i}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{W}_s \boldsymbol{\Phi}_s^T (\boldsymbol{s_i} - \bar{\boldsymbol{s}}) \\ \boldsymbol{\Phi}_g^T (\boldsymbol{g_i} - \bar{\boldsymbol{g}}) \end{pmatrix} \quad (5)$$

In which i displays i[th] image of database and $\boldsymbol{W}_s$ is diagonal matrix of weights which is used to compensate unit difference between shape (pixel) and texture (intensity). Again we retain enough modes to explain 95% of total combined variance. The average of RMS changes in vector $\boldsymbol{g}$ per unit change in elements of $\boldsymbol{b}_{s_{opt}}$ gives the weight $\boldsymbol{W}_s$ for the relevant element.

Since averages of both $\boldsymbol{b}_{s_{opt}}$ and $\boldsymbol{b}_{g_{opt}}$ vectors are zero, vectors $\boldsymbol{b}$ -concatenated vector of these two vectors- as a resultant has an average equal to zero. Applying PCA for the 3rd time, we obtain the combined model $\boldsymbol{b} \approx \boldsymbol{\Phi}_c \boldsymbol{c}$. With regard to linear nature of the model, an AAM instance could be created including shape data ($\boldsymbol{s}$) and texture data ($\boldsymbol{g}$), according to the following equation by appearance parameters $\boldsymbol{c}$:

$$\begin{aligned} \tilde{\boldsymbol{s}} &= \bar{\boldsymbol{s}} + \boldsymbol{\Phi}_s \boldsymbol{W}_s^{-1} \boldsymbol{\Phi}_{cs} \boldsymbol{c} \\ \tilde{\boldsymbol{g}} &= \bar{\boldsymbol{g}} + \boldsymbol{\Phi}_g \boldsymbol{\Phi}_{cg} \boldsymbol{c} \end{aligned} \quad (6)$$

In which $\boldsymbol{\Phi}_{\mathrm{c}_{(\mathrm{t_s}+\mathrm{t_g})*\mathrm{t_c}}} = \begin{pmatrix} \boldsymbol{\Phi}_{\mathrm{cs}\ \mathrm{t_s}*\mathrm{t_c}} \\ \boldsymbol{\Phi}_{\mathrm{cg}\ \mathrm{t_g}*\mathrm{t_c}} \end{pmatrix}$ and $t_c \leq t_s + t_g$.

2.2 Model training

If we combine appearance parameter, $\boldsymbol{c}$, with 2D pose parameters ($\boldsymbol{t}$) in $\mathrm{t_p}$-dimensional vector $\boldsymbol{p} = [\boldsymbol{c}^T | \boldsymbol{t}^T]^T$, we can make synthesized image of face in any part of image plane. Active Appearance Model fitting is followed by updating vector $\boldsymbol{p}$ to minimize error between model texture and target image texture (texture residual) according to the following relation:

$$\arg\min_{c,t} \boldsymbol{r}(\boldsymbol{p})^2 = \arg\min_{c,t} \left| \boldsymbol{g}_{\boldsymbol{image}} - \boldsymbol{g}_{model} \right| \quad (7)$$

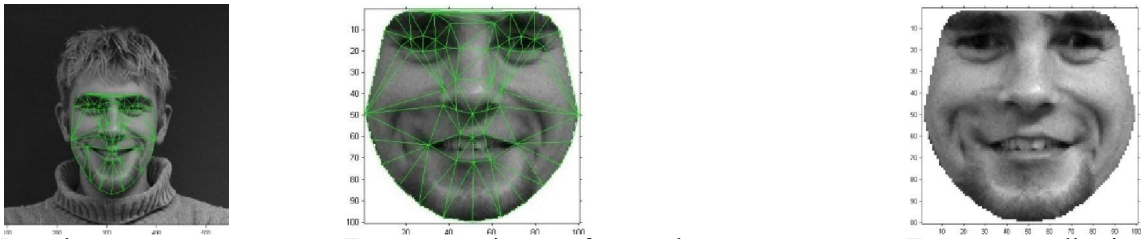
Consider an estimation of parameters as $\boldsymbol{p} = \boldsymbol{p}^* + \delta\boldsymbol{p}$ in which $\delta\boldsymbol{p}$ is parameter displacement from true solution, $\boldsymbol{p}^*$. If we present texture residual vector resulted from $\boldsymbol{p}$ by $\boldsymbol{r}(\boldsymbol{p})$, according to Eq (7), objective of Active Appearance Models is parameter modification by $\delta\boldsymbol{p}$ to minimize $|\boldsymbol{r}(\boldsymbol{p}+\delta\boldsymbol{p})|^2$. It is assumed that in Active Appearance Models $\delta\boldsymbol{p}$ could be predicted linearly from residual vector by $\delta\boldsymbol{p} = \boldsymbol{R}\,\delta\boldsymbol{g}$. By use of first order Taylor expansion for texture residual around $\boldsymbol{p}$ Eq (8) would be obtained:

$$\boldsymbol{r}(\boldsymbol{p}+\delta\boldsymbol{p}) = \boldsymbol{r}(\boldsymbol{p}) + \boldsymbol{J}\,\delta\boldsymbol{p} \quad (8)$$

Where $\mathbf{J}_{m\times \mathrm{t_p}} = \frac{\partial \boldsymbol{r}(\boldsymbol{p})}{\partial \boldsymbol{p}}$ is the Jacobian matrix. By deviating Eq (8), setting it to zero and comparing it with $\delta\boldsymbol{p} = \boldsymbol{R}\,\delta\boldsymbol{g}$, Regression matrix, $\boldsymbol{R}$, will be calculated as followed:

$$\boldsymbol{R} = (\boldsymbol{J}^T\boldsymbol{J})^{-1}\boldsymbol{J}^T = \boldsymbol{J}^\dagger \quad (9)$$

In which $\boldsymbol{J}^\dagger$ is Jacobian matrix pseudo-inverse. For calculating Jacobian matrix from training set, we displace model parameters from $\boldsymbol{p}_{opt}$ by $\delta\boldsymbol{p}_i$ systematically. Vectors $\delta\boldsymbol{p}_i$ and the corresponding residual $\boldsymbol{r}_i = \boldsymbol{r}(\boldsymbol{p}_{opt} + \delta\boldsymbol{p}_i)$, would be stored in the columns of $\boldsymbol{\Delta p}$ and $\boldsymbol{\Delta r}$ respectively. Jacobian matrix then will be calculated as follows [12]:



Input image Texture mapping to reference shape Texture normalization

Figure 1. Example of image warping and texture normalization

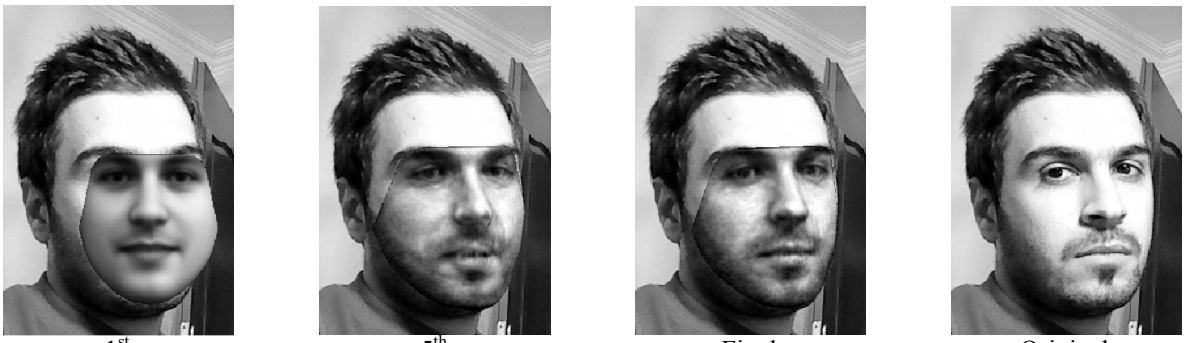
$$\frac{\partial \boldsymbol{r}(\boldsymbol{p})}{\partial \boldsymbol{p}} = \arg\min_{\boldsymbol{J}} ||\boldsymbol{\Delta r} - \boldsymbol{J}\boldsymbol{\Delta p}||_F^2 \rightarrow \boldsymbol{J} = \boldsymbol{\Delta r}\,\boldsymbol{\Delta p}^T(\boldsymbol{\Delta p}\,\boldsymbol{\Delta p}^T)^{-1} = \boldsymbol{\Delta r}\,\boldsymbol{\Delta p}^\dagger \quad (10)$$

2.3 Model fitting

Fitting process of AAM algorithm (Algorithm 1) begins with initial estimate for model parameters ($\boldsymbol{p_0}$), then by following a series of iterative steps the model would be fitted to the target image. While there is no information about vector $\boldsymbol{p_0}$ quantity (such as first frame in video sequences), we can estimate it by $\boldsymbol{p_0} = [\mathbf{0} \mid \boldsymbol{t}_0]^{\mathrm{T}}$, in which Adaboost face detection [13] is used for locating face in the image ($\boldsymbol{t}_0$). Figure 2 shows a successful fitting example of Active Appearance Models.

Algorithm 1. AAM fitting

1: initialize: set $\boldsymbol{p} = \boldsymbol{p_0}$, iteration =1
2: **while** iteration < Maxiteration and error $E = |\boldsymbol{r}|^2$ is improving:
3: use $\boldsymbol{p}$ to compute the residual,$\boldsymbol{r}(\boldsymbol{p})$, and corresponding error $E = |\boldsymbol{r}|^2$.
4: predict the displacement $\delta\boldsymbol{p} = \boldsymbol{R}\,\boldsymbol{r}$
5: set $\alpha = 1$
6: update the model parameters $\boldsymbol{p}' = \boldsymbol{p} - \alpha\,\delta\boldsymbol{p}$
7: calculate the new residual vector $\boldsymbol{r}'(\boldsymbol{p}')$, and corresponding error $E' = |\boldsymbol{r}'|$
8: **if** $E' < E$ then
9: set $\boldsymbol{p} = \boldsymbol{p}'$, $E = E'$, $\boldsymbol{r} = \boldsymbol{r}'$
10: **else**
11: Try $\alpha = 1.5, \alpha = 0.5, \alpha = 0.25, \alpha = 0.125$
12: **end if**
13: iteration=iteration + 1
14: **end while**



1st 5th Final Original

Figure 2. successful fitting of Active Appearance Models applied for unseen images

Table 1. Mean and standard deviation (SD) of point to point error regarding existence or nonexistence of mouth area pixels in image warping. m is number of sampled pixels and size ($\boldsymbol{p}$) is number of elements in vector $\boldsymbol{p}$

	m	size ($\boldsymbol{p}$)	$\delta t_x = 0$ $\delta t_y = 0$		$\delta t_x = +8$ $\delta t_y = 0$		$\delta t_x = -8$ $\delta t_y = 0$		$\delta t_x = 0$ $\delta t_y = +8$		$\delta t_x = 0$ $\delta t_y = -8$		**Overall average**	
			mean	SD	mean	SD	mean	SD	mean	SD	mean	SD	mean	SD
warping all parts of face	1917	83	5.944	2.828	6.020	2.604	6.111	3.102	6.304	3.416	5.861	2.639	**6.048**	**2.918**
Eliminate mouth in texture warping	1835	80	5.821	2.700	5.734	2.542	6.013	2.974	6.020	2.933	5.697	2.288	**5.857**	**2.687**

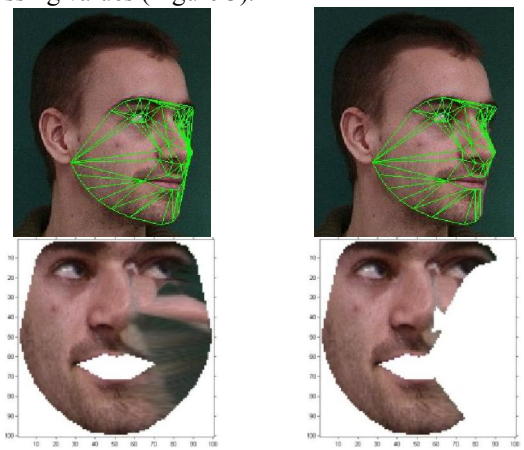
In order to investigate mapping effects on various parts of the face to the reference shape, two cases regarding existence or nonexistence of mouth area pixels in image warping were compared with each other. In both cases same training images were used to obtain Regression matrix $\boldsymbol{R}$ and AAM fitting was executed 5 times for each of the cases. Each time, we started with different initial point by displacing the model systematically as ± 5% width and length of the mean shape image (±8 pixels) in the horizontal and vertical direction respectively. Remaining model parameters (appearance parameters $\boldsymbol{c}$) were set to zero in initialization. After each convergence we recorded the mean and standard deviation (SD) of point to point distance error -the distance between each model point and their corresponding manually determined ground truth location in pixel- in order to obtain a metric for AAM fitting.

Results are shown in Table 1, and indicate that eliminating mouth area from texture mapping not only decreases number of sampled pixels (which leads to increase image warping speed), but also enhances system accuracy. Reason for this would be mouth interior area pixels insert misleading information particularly during changing facial expressions (such as revealing teeth while laughing, mouth opening during yawning) which leads to model fitting failure.

3 Probabilistic PCA for texture modeling

In order to make a richer model of texture, better capability of dealing with the large variation in texture and obtaining the texture model even in the presence of missing values in the training data Probabilistic PCA with missing data is proposed in this paper.

Since the convex nature of human face shape, the texture warping should be a straight full procedure but across several 3D pose variations, which occlusion occurs, background pixels appear in the texture. Using proposed method we can remove these pixels from texture and gain the texture model. For this purpose, if any triangle of the face transfers background pixels to the texture, all pixels of that triangle are considered as missing values (Figure 3).



Training Texture in Classic PCA Training Texture in Probabilistic PCA

Figure 3. Example of background pixels omission. Upper row illustrates the input image and lower row shows the relevant training texture.

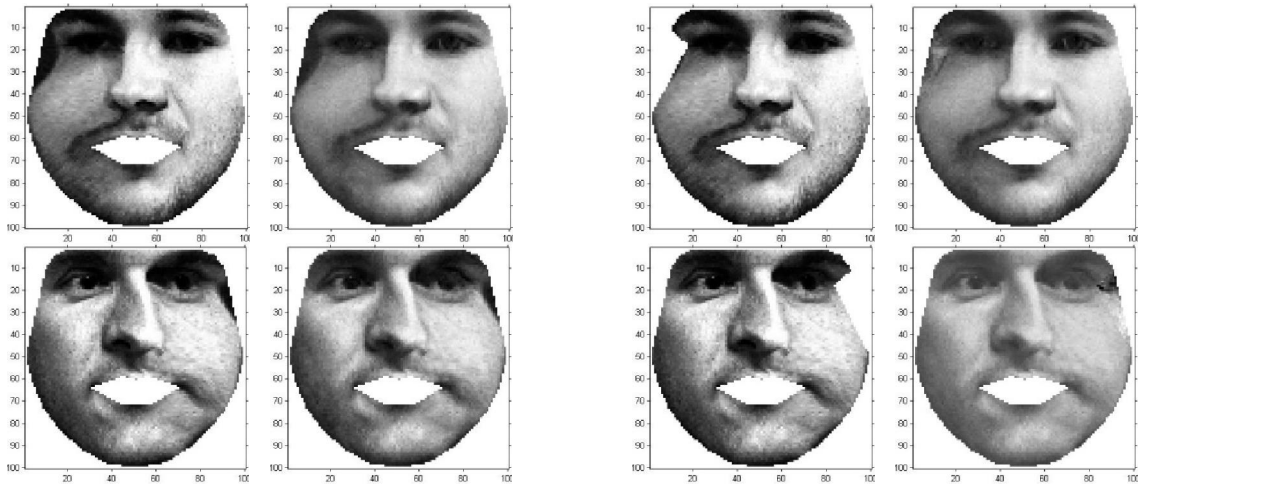
In this paper two probabilistic approaches are suggested; one without considering any prior probabilities for parameters (PPCA) and the other with considering prior probability for them (Variational Bayesian PCA) which provides a good foundation for handling of missing values.

3.1 PPCA without any prior probabilities for model parameters

Probabilistic PCA is suggested by Bishop & Tipping [14] through which it is possible to find out an EM algorithm for Principal Component Analysis and apply the same for missing values. PPCA is in fact an example of Linear-Gaussian model because all conditional & marginal distributions are in Gaussian form. Such a probabilistic model is formulated by introducing latent variables, $\boldsymbol{b}_g$, which corresponds to principal-component subspace. After that a Gaussian prior distribution $p(\boldsymbol{b}_g)$ over the latent variables, together with a Gaussian conditional distribution $p(\boldsymbol{g}|\boldsymbol{b}_g)$, for the observed variables $\boldsymbol{g}$ conditioned on the value of the latent variables, is defined as follows [15]:

$$p(\boldsymbol{b}_g) = \mathcal{N}(\boldsymbol{b}_g|\boldsymbol{0}, \boldsymbol{I}) \quad (11$$

$$p(\boldsymbol{g}|\boldsymbol{b}_g, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g, \sigma^2) = \mathcal{N}(\boldsymbol{g}|\boldsymbol{\Phi}_g \boldsymbol{b}_g + \boldsymbol{\mu}_g, \sigma^2\mathbf{I}) \quad (12$$



Input texture | Reconstructed texture from classic PCA | Input texture | Reconstructed texture from probabilistic PCA

Classic PCA | Probabilistic PCA

Figure 4. A comparison between reconstructed texture by PPCA and classic PCA

Where $\boldsymbol{g}$ is a linear function of t_g-dimensional vector $\boldsymbol{b}_g$, m*t_g matrix $\boldsymbol{\Phi}_g$ and m-dimensional vector $\boldsymbol{\mu}_g$. Matrix $\boldsymbol{I}$ shows the unitary matrix. In order to calculate the amount of t_g, we assume that all data are observed and none of which is missing. Then we may calculate this amount in a way to explain 95% of observed data variance. The other parameter of this model is the Scalar σ^2 explaining the variance of conditional distribution. Note that there is no loss of generality in assuming a zero mean, unit covariance Gaussian for the latent distribution in Eq (11) because a more general Gaussian distribution would finally resulted in an equivalent probabilistic model [15]. We may consider this probabilistic model from a generative viewpoint in which a noise term is considered as follows:

$$\boldsymbol{g} = \boldsymbol{\Phi}_g \boldsymbol{b}_g + \boldsymbol{\mu}_g + \boldsymbol{\epsilon} \quad (13$$

In which $p(\boldsymbol{\epsilon}) = \mathcal{N}(\boldsymbol{\epsilon}|0, \sigma^2\boldsymbol{I})$. If we display the set of texture vectors $\{\boldsymbol{g}_1, \boldsymbol{g}_2, \dots, \boldsymbol{g}_N\}$ for N images of database with $\boldsymbol{G}$ and the set of corresponding latent variables of $\{\boldsymbol{b}_g^{(1)}, \boldsymbol{b}_g^{(2)}, \dots, \boldsymbol{b}_g^{(N)}\}$ with $\boldsymbol{B}_g$, the marginal distribution can be written as bellow:

$$p(\boldsymbol{G}|\boldsymbol{\mu}_g, \boldsymbol{\Phi}_g, \sigma^2) = \int p(\boldsymbol{B}_g)\, p(\boldsymbol{G}|\boldsymbol{B}_g, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g, \sigma^2)\, d\boldsymbol{B}_g \quad (1$$

$$= \prod_{n=1}^{N} \int p(\boldsymbol{b}_g^{(n)}) \prod_{i|in \in O}^{m} \mathcal{N}\left(g_{in}|\tilde{\boldsymbol{\phi}}_{(i)_g}^T \boldsymbol{b}_g^{(n)} + \mu_g^{(i)}, \sigma^2\boldsymbol{I}\right) d\boldsymbol{b}_g^{(n)}$$

Where indice n represents n^{th} image of training set. g_{in} and $\mu_g^{(i)}$ are the ith element of vectors $\boldsymbol{g}_n$ and $\boldsymbol{\mu}_g$ respectively. $\widetilde{\boldsymbol{\phi}}_{(i)_g}$ is the column vector including ith row of matrix $\boldsymbol{\Phi}_g$. Symbol $i|in \in O$ shows set of indices i in which g_{in} is observed. By using EM algorithm one can find the maximum likelihood estimates of model parameters. At E step of the algorithm it is necessary to obtain posterior distribution of latent variables, $p(\boldsymbol{B}_g|\boldsymbol{G}, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g.\sigma^2)$, (where $\boldsymbol{G}$ is the set of observed data). Since all distributions are based upon linear Gaussian form, posterior distribution of latent variables $\boldsymbol{b}_g^{(n)}$ has also Gaussian distribution as $\prod_{n=1}^{N} \mathcal{N}(\boldsymbol{b}_g^{(n)}|\boldsymbol{m}_{\boldsymbol{b}_g}^{(n)}, \Sigma_{\boldsymbol{b}_g}^{(n)})$. In the E step we evaluate:

$$\boldsymbol{\Sigma}_{\boldsymbol{b}_g}^{(n)} = \left(\boldsymbol{I} + \sum_{i|in\in O}^{m} \frac{1}{\sigma^2} \widetilde{\boldsymbol{\phi}}_{(i)_g} \widetilde{\boldsymbol{\phi}}_{(i)_g}^{T} \right)^{-1} \tag{15}$$

$$E_{\boldsymbol{b}_g^{(n)}}\left[\boldsymbol{b}_g^{(n)}\right] = \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)} = \boldsymbol{\Sigma}_{\boldsymbol{b}_g}^{(n)} \frac{1}{\sigma^2} \sum_{i|in\in O}^{m} (g_{in} - \mu_g^{(i)}) \widetilde{\boldsymbol{\phi}}_{(i)_g} \tag{16}$$

At M step we compute the equations:

$$\widetilde{\boldsymbol{\phi}}_{(i)_g} = \left(\sum_{n|in\in O}^{N} \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)} \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)^T} + \boldsymbol{\Sigma}_{\boldsymbol{b}_g}^{(n)} \right)^{-1} \times \sum_{n|in\in O}^{N} \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)} (g_{in} - \mu_g^{(i)}) \tag{17}$$

$$\mu_g^{(i)} = \frac{1}{N_{n|in\in O}} \sum_{n|in\,\in O}^{N} (g_{in} - \widetilde{\boldsymbol{\phi}}_{(i)_g}^{T} \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)}) \tag{18}$$

$$\sigma^2 = \frac{1}{N^O} \sum_{n=1}^{N} \sum_{i|in\in O}^{m} \left\{ \left(g_{in} - \widetilde{\boldsymbol{\phi}}_{(i)_g}^{T} \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)} - \mu_g^{(i)} \right)^2 + \widetilde{\boldsymbol{\phi}}_{(i)_g}^{T} \boldsymbol{\Sigma}_{\boldsymbol{b}_g}^{(n)} \widetilde{\boldsymbol{\phi}}_{(i)_g} \right\} \tag{19}$$

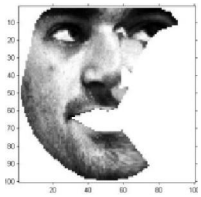
In which $N_{n|in\in O}$ is number of indices n for which g_{in} is observed and N^O is the total number of the observed g_{in}. Continue E & M steps until model parameters do not change significantly within iterations. If necessary we may transfer model parameters especially columns of matrix $\boldsymbol{\Phi}_g$ to be orthonormal. By using the probabilistic PCA all database texture vectors $\boldsymbol{g}$ can be reconstructed even in the presence of missing data in the texture by taking the expectation of Eq (13) with respect to the posterior distribution of latent variables and noise ϵ as follows:

$$\widetilde{\boldsymbol{g_n}} = E_{\boldsymbol{b}_g,\boldsymbol{\epsilon}}\left[\boldsymbol{\Phi}_g \boldsymbol{b}_g^{(n)} + \boldsymbol{\mu_g} + \boldsymbol{\epsilon}\right] = \boldsymbol{\Phi}_g \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)} + \boldsymbol{\mu_g} \tag{20}$$

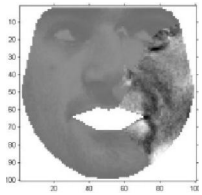
In which Eq (16), (17) and (18) are used in order to compute $\boldsymbol{m}_{\boldsymbol{b}_g}^{(n)}, \boldsymbol{\Phi}_g, \boldsymbol{\mu_g}$. When some of the texture values are lost in PPCA, it is possible for the model to reconstruct them as much as possible. Figure 4 illustrates a comparison between reconstructed textures by PPCA with classic PCA.

However PPCA can overfit; this will arise when a great volume of information is lost for a special data. In this situation, not only the model cannot estimate any missing data, but also it may change the observed data significantly at the time of reconstruction (Figure 5).

Original image



Input texture



Reconstructed texture from probabilistic PCA

Figure 5. An example of overfitting while lots of data is missing

3.1.1 Results

By replacing $\boldsymbol{m}_{\boldsymbol{b}_g}^{(n)}$ instead of $\boldsymbol{b}_{g_{opt}}$ mentioned in Eq (4) and applying classic PCA to the combined vectors $\boldsymbol{b}$ -Eq (5)- Active Appearance Models are created. However, in Eq (6) instead of $\overline{\boldsymbol{g}}$ & $\boldsymbol{\Phi}_g$, $\boldsymbol{\mu}_g$ and $\boldsymbol{\Phi}_g$ are used respectively as mentioned in Eq (20). In order to evaluate model fitting, we performed relevant experiments same as Table 1 for the proposed probabilistic model. Results are shown in Table 2. Comparing Table 2 with Table 1 indicates that using PPCA for texture modeling may cause better fitting. Nevertheless, it seems that solving the overfitting problem may cause better results. Therefore we may consider Bayesian Probabilistic model in which prior probabilities are considered for the model parameters.

3.2 . Variational Bayesian PCA

Variational PCA in case of fully observed data first introduced by Bishop [16]. Here we discuss the problem when some of the data values are missing. In a variational Bayesian PCA (variational PCA) model parameters $\boldsymbol{\Phi}_g$ and $\boldsymbol{\mu}_g$ also have prior probabilities and accompanied by $\boldsymbol{B}_g$ are considered as latent variables.

$$p(\boldsymbol{\mu}_g|\sigma_\mu^2) = \mathcal{N}(\boldsymbol{\mu}_g|0,\sigma_\mu^2\boldsymbol{I}) \tag{21}$$

$$p\left(\boldsymbol{\Phi}_g\middle|\sigma_{\boldsymbol{\Phi}_{gk}}^2\right) = \prod_{k=1}^{t_g}\mathcal{N}(\boldsymbol{\Phi}_{g:k}|0,\sigma_{\boldsymbol{\Phi}_{gk}}^2\boldsymbol{I}) \tag{22}$$

Where $\boldsymbol{\Phi}_{g:k}$ represents k[th] column of matrix $\boldsymbol{\Phi}_g$. According to the above-mentioned equation the model considers same prior probability for all elements positioned in the same column of matrix $\boldsymbol{\Phi}_g$. In order to compute t_g again we may assume that all data is observed and none of which is missing. Then it will be calculated in a way to explain %95 of observed data variance.

Considering prior probability for model parameters causes to cope with the overfitting problem. Since working with true posterior distribution of latent variables and taking the required expectation terms with respect to this posterior is intractable (this may occur due to high dimension of latent variables or their complex form), approximation schemes could be applied for solving this problem. Here we turn to a family of approximation techniques called variational inference [15]. If we show all latent variables of model by $\boldsymbol{Z} = \{\boldsymbol{B}_g, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g\}$ and observed variables with $\boldsymbol{G}$, the objective of Bayesian method is finding an estimation for posterior distribution $p(\boldsymbol{Z}|\boldsymbol{G})$ and marginal distribution $p(\boldsymbol{G})$. The relevant approximation of Bayesian method is to restrict the functional form of posterior distribution $q(\boldsymbol{Z})$ by assuming that it factorizes over the components variables $\{\boldsymbol{Z}_i\}$ so that:

$$q(\boldsymbol{Z}) = \prod_{i=1}^{M} q_i(\boldsymbol{Z}_i) = q(\boldsymbol{B}_g)q(\boldsymbol{\mu}_g)q(\boldsymbol{\Phi}_g) \tag{23}$$

Note that no further assumptions are applied regarding distribution $q(\boldsymbol{Z})$ and no restriction for the functional forms of distributions $q_i(\boldsymbol{Z}_i)$ is considered. It is possible to decompose the log marginal probability for each arbitrary distribution $q(\boldsymbol{Z})$ as follows [15]:

$$\ln p(\boldsymbol{G}) = \mathcal{L}(\boldsymbol{q}) + KL(\boldsymbol{q}(\boldsymbol{Z})||\boldsymbol{p}(\boldsymbol{Z}|\boldsymbol{G})) \tag{24}$$

Where:

$$\mathcal{L}(\boldsymbol{q}) = \int q(\boldsymbol{Z}) \ln\left\{\frac{p(\boldsymbol{G},\boldsymbol{Z})}{q(\boldsymbol{Z})}\right\} d\boldsymbol{Z} \tag{25}$$

$$KL(\boldsymbol{q}||\boldsymbol{p}) = -\int q(\boldsymbol{Z}) \ln\left\{\frac{p(\boldsymbol{Z}|\boldsymbol{G})}{q(\boldsymbol{Z})}\right\} d\boldsymbol{Z} \tag{26}$$

The term KL (q||p) is named as Kullback-Leibler divergence. Since Kullback-Leibler divergence satisfies $KL(\boldsymbol{q}||\boldsymbol{p}) \geq 0$, therefore it follows from Eq (24) that $\mathcal{L}(\boldsymbol{q}) \leq \ln p(\boldsymbol{G})$. In other words, $\mathcal{L}(\boldsymbol{q})$ provides a lower bound on log marginal distribution. As a result the optimization problem will be presented by maximizing the lower bound with respect to the distribution $q(\boldsymbol{Z})$, which is equivalent to minimizing the KL divergence.

Table 2. Mean and standard deviation of point to point distance error for AAM model fitting when texture model is obtained through PPCA. size ($\boldsymbol{p}$) represents number of elements in vector $\boldsymbol{p}$.

	Size ($\boldsymbol{p}$)	$\delta t_x = 0$ $\delta t_y = 0$		$\delta t_x = +8$ $\delta t_y = 0$		$\delta t_x = -8$ $\delta t_y = 0$		$\delta t_x = 0$ $\delta t_y = +8$		$\delta t_x = 0$ $\delta t_y = -8$		**Overall average**	
		mean	SD	mean	SD	mean	SD	mean	SD	mean	SD	mean	std
Probabilistic PCA without any prior probabilities for model parameters	79	5.749	2.666	5.701	2.500	5.858	2.871	5.770	2.901	5.614	2.333	**5.738**	**2.654**

Thus the general expression for the optimal solution $q_j^*(\boldsymbol{Z}_j)$ is given by [15]:

$$\ln q_j^*(\boldsymbol{Z}_j) = E_{i\neq j}[\ln p(\boldsymbol{G},\boldsymbol{Z})] + const \tag{27}$$

In which, $E_{i\neq j}[.]$ denotes an expectation with respect to all distributions $q_j(\boldsymbol{Z}_j)$ for $i \neq j$. Following is the joint distribution of latent and observed variables:

$$\begin{aligned} P\left(\boldsymbol{G},\boldsymbol{Z}\middle|\sigma^2,\sigma_\mu^2,\sigma_{\boldsymbol{\Phi}_{gk}}^2\right) &= p\left(\boldsymbol{G}\middle|\boldsymbol{B}_g,\boldsymbol{\mu}_g,\boldsymbol{\Phi}_g,\sigma^2,\sigma_\mu^2,\sigma_{\boldsymbol{\Phi}_{gk}}^2\right)p(\boldsymbol{B}_g) \\ &\times p(\boldsymbol{\mu}_g|\sigma_\mu^2)p\left(\boldsymbol{\Phi}_g\middle|\sigma_{\boldsymbol{\Phi}_{gk}}^2\right) \\ &= p(\boldsymbol{\mu}_g|\sigma_\mu^2)p\left(\boldsymbol{\Phi}_g\middle|\sigma_{\boldsymbol{\Phi}_{gk}}^2\right) \\ &\times \prod_{n=1}^{N} p(\boldsymbol{b}_g^{(n)}) \prod_{i|in\in O}^{m} \mathcal{N}(g_{in}|\tilde{\boldsymbol{\phi}}_{(i)_g}^T \boldsymbol{b}_g^{(n)} + \mu_g^{(i)}, \sigma^2\boldsymbol{I}) \end{aligned} \tag{28}$$

Where g_{in} and $\mu_g^{(i)}$ are the i[th] element of random vectors $\boldsymbol{g}_n$ and $\boldsymbol{\mu}_g$ respectively. $\tilde{\boldsymbol{\phi}}_{(i)_g}$ is the random column vector including i[th] row of random matrix $\boldsymbol{\Phi}_g$. O is the set of indeces i,n corresponding to observed values g_{in}. According to Eq (27) & (28), together with the explicit forms for the various p (.) distributions, we obtain the following results for the posterior distributions of $q_j^*(\boldsymbol{Z}_j)$:

$$\begin{aligned} q^*(\boldsymbol{B}_g) &= \prod_{n=1}^{N} q^*(\boldsymbol{b}_g^{(n)}) \\ &= \prod_{n=1}^{N} \mathcal{N}\left(\boldsymbol{b}_g^{(n)}|\boldsymbol{m}_{b_g}^{(n)},\boldsymbol{\Sigma}_{b_g}^{(n)}\right) \end{aligned} \tag{29}$$

$$q^*(\boldsymbol{\Phi}_g) = \prod_{i=1}^{m} q^*\left(\tilde{\boldsymbol{\phi}}_{(i)_g}\right) = \prod_{i=1}^{m} \mathcal{N}\left(\tilde{\boldsymbol{\phi}}_{(i)_g}|\boldsymbol{m}_{\boldsymbol{\Phi}_g}^{(i)},\boldsymbol{\Sigma}_{\boldsymbol{\Phi}_g}^{(i)}\right) \tag{30}$$

$$q^*(\boldsymbol{\mu}_g) = \mathcal{N}\left(\boldsymbol{\mu}_g\middle|\boldsymbol{m}_{\boldsymbol{\mu}_g},\boldsymbol{\Sigma}_{\boldsymbol{\mu}_g}\right) = \prod_{i=1}^{m} \mathcal{N}\left(\mu_g^{(i)}|m_{\mu_g}^{(i)},v_{\mu_g}^{(i)}\right) \tag{31}$$

All aforementioned parameters of distributions are obtained via following equations:

$$\boldsymbol{m}_{b_g}^{(n)} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}_{b_g}^{(n)} \sum_{i|in\in O}^{m} \boldsymbol{m}_{\boldsymbol{\Phi}_g}^{(i)}(g_{in} - m_{\mu_g}^{(i)}) \tag{32}$$

$$\boldsymbol{\Sigma}_{b_g}^{(n)} = \left(\boldsymbol{I} + \frac{1}{\sigma^2}\sum_{i|in\in O}^{m}\left[\boldsymbol{m}_{\boldsymbol{\Phi}_g}^{(i)}\,\boldsymbol{m}_{\boldsymbol{\Phi}_g}^{(i)\,T} + \boldsymbol{\Sigma}_{\boldsymbol{\Phi}_g}^{(i)}\right]\right)^{-1} \tag{33}$$

$$\boldsymbol{m}_{\boldsymbol{\Phi}_g}^{(i)} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}_{\boldsymbol{\Phi}_g}^{(i)} \sum_{n|in\in O}^{N}\left[\boldsymbol{m}_{b_g}^{(n)}\left(g_{in} - m_{\mu_g}^{(i)}\right)\right] \tag{34}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\Phi}_g}^{(i)} = \tilde{\boldsymbol{\phi}}_{(i)_g}^T\left(diag\left((\sigma_{\boldsymbol{\Phi}_{gik}}^2)^{-1}\right) + \frac{1}{\sigma^2}\sum_{n|in\in O}^{N}[\boldsymbol{m}_{b_g}^{(n)}\,\boldsymbol{m}_{b_g}^{(n)\,T} + \boldsymbol{\Sigma}_{b_g}^{(n)}]\right)^{-1} \tag{35}$$

$$m_{\mu_g}^{(i)} = \frac{v_{\mu_g}^{(i)}}{\sigma^2} \sum_{n|in\in O}^{N} \left(g_{in} - \boldsymbol{m}_{\Phi_g}^{(i)\,T} \boldsymbol{m}_{b_g}^{(n)} \right) \quad (36)$$

$$v_{\mu_g}^{(i)} = \frac{\sigma^2 \sigma_\mu^2}{\sigma^2 + \sigma_\mu^2 \, N_{n|in\in O}} \quad (37)$$

Where $diag\left(\left(\sigma_{\Phi_{gik}}^2\right)^{-1}\right)$ is a diagonal t_g*t_g matrix whose k^{th} diagonal element is $1/\sigma_{\Phi_{gk}}^2$. In order to find deterministic parameters of $\boldsymbol{\theta} = \sigma^2, \sigma_\mu^2, \sigma_{\Phi_{gk}}^2$ we may use EM algorithm. As mentioned before, latent variables in variational Bayesian PCA are: $\boldsymbol{Z} = \{\boldsymbol{B}_g, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g\}$. Therefore at E step of EM algorithm by using posterior distribution $q(\boldsymbol{Z}|\boldsymbol{G}, \boldsymbol{\theta^{old}})$ and Eq (32) to (37), $\boldsymbol{m}_{b_g}^{(n)}$، $\boldsymbol{\Sigma}_{b_g}^{(n)}$، $\boldsymbol{m}_{\Phi_g}^{(i)}$، $\boldsymbol{\Sigma}_{\Phi_g}^{(i)}$، $m_{\mu_g}^{(i)}$ و $v_{\mu_g}^{(i)}$ will be computed. Then we may update all parameters at M step by following equations:

$$\sigma^2 = \frac{1}{N^O} \sum_{n=1}^{N} \sum_{i|in\in O}^{m} \left\{ \left(g_{in} - \boldsymbol{m}_{\Phi_g}^{(i)\,T} \boldsymbol{m}_{b_g}^{(n)} - m_{\mu_g}^{(i)} \right)^2 + \boldsymbol{m}_{\Phi_g}^{(i)\,T} \boldsymbol{\Sigma}_{b_g}^{(n)} \boldsymbol{m}_{\Phi_g}^{(i)} + \boldsymbol{m}_{b_g}^{(n)T} \boldsymbol{\Sigma}_{\Phi_g}^{(i)} \boldsymbol{m}_{b_g}^{(n)} + trace\left(\boldsymbol{\Sigma}_{b_g}^{(n)} \boldsymbol{\Sigma}_{\Phi_g}^{(i)} \right) + v_{\mu_g}^{(i)} \right\} \quad (38)$$

$$\sigma_\mu^2 = \frac{1}{m} \sum_{i=1}^{m} \left(m_{\mu_g}^{(i)\,2} + v_{\mu_g}^{(i)} \right) \quad (39)$$

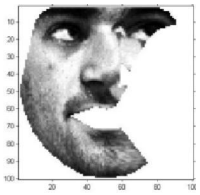
$$\sigma_{\Phi_{gk}}^2 = \frac{1}{m} \sum_{i=1}^{m} \left(m_{\Phi_g}^{(i,k)} + var(\phi_{ik}) \right) \quad (40)$$

In which $var(\phi_{ik})$ is the k^{th} element on the diagonal $\boldsymbol{\Sigma}_{\Phi_g}^{(i)}$. We should continue Eq (32) to (40) till there are no significant changes in model parameters. We may transfer model parameters especially columns of matrix $E_{\Phi_g}[\boldsymbol{\Phi}_g]$ in a way to be orthonormal, if needed. In order to reconstruct texture vectors of training set according to the variational PCA, we may take the expectation of Eq (13) with respect to the posterior distribution of latent variables $\boldsymbol{Z} = \{\boldsymbol{B}_g, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g\}$ and noise $\boldsymbol{\epsilon}$ as follows:

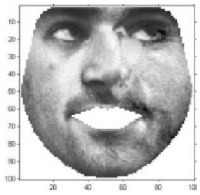
$$\widetilde{\boldsymbol{g_n}} = E_{\boldsymbol{B}_g, \boldsymbol{\mu}_g, \boldsymbol{\Phi}_g, \boldsymbol{\epsilon}}\left[\boldsymbol{\Phi}_g \boldsymbol{b}_g^{(n)} + \boldsymbol{\mu}_g + \boldsymbol{\epsilon} \right] = \widetilde{\boldsymbol{\Phi}_g} \boldsymbol{m}_{\boldsymbol{b}_g}^{(n)} + \boldsymbol{m}_{\boldsymbol{\mu}_g} \quad (41)$$

Where $\widetilde{\boldsymbol{\Phi}_g}$ is m*t_g matrix whose rows are $\boldsymbol{m}_{\boldsymbol{\Phi}_g}^{(i)\,T}$. Introducing prior probability for $\boldsymbol{\mu}_g, \boldsymbol{\Phi}_g$ acts as a regulator in order to overcome the overfitting problem. For instance, Figure 6 shows the reconstructed texture of Figure 5 by the use of Variational PCA.

Original image



Input texture



Reconstructed texture from Bayesian PCA

Figure 6. Reconstruction of texture in variational Bayesian PCA

3.2.1 Results

Upon inserting of $\boldsymbol{m}_{\boldsymbol{b}_g}^{(n)}$ instead of $\boldsymbol{b}_{g_{opt}}$ in Eq (4) and then applying third classic PCA to the combined vectors $\boldsymbol{b}$ -Eq (5)- we may create Active Appearance Models. The results of AAM fitting in which variational PCA has been used for texture modeling are shown in Table 3. Upon comparing Table 3 with Tables 1 & 2, it is obvious that using variational PCA for texture modeling, not only increases fitting accuracy of the model, but also reduces the number of vector p elements which may cause an increase in the speed of computing Eq (6).

4 Updating the Jacobian Matrix

Active appearance model fitting is based on finding model parameters which minimize texture residual between model and target image. Jacobian matrix shows that how texture residual varies when

model parameters change. Since determining Jacobian matrix requires numerous computations in classic AAM, this matrix would be calculated once during training and then it is assumed to be fixed around the optimum value of model parameters in test (fitting) period.

In this paper a method is presented to update Jacobian matrix while tracking video sequence. Such models are considered as adaptive Active Appearance Models. Various methods have been presented so far [16], [17]. The proposed method is similar to Cootes [17], but final results and also the method for obtaining equations are different.

In order to update Jacobian matrix during model fitting, partitioned matrix property is used. For this purpose after any successful model fitting we displace parameters from the final model parameters, $\boldsymbol{p}$, by $\delta\boldsymbol{p}'_j$ and denote the corresponding texture residual as $\boldsymbol{r}'_j$. $\Delta\boldsymbol{p}_j$, $\Delta\boldsymbol{r}_j$ are defined as following partitioned matrix:

$$\begin{aligned}\Delta\boldsymbol{p}_j &= \left[\Delta\boldsymbol{p}_{j-1} \quad | \quad \delta\boldsymbol{p}'_j\right]_{t_p * j} \\ \Delta\boldsymbol{r}_j &= \left[\Delta\boldsymbol{r}_{j-1} \quad | \quad \boldsymbol{r}'_j\right]_{m * j}\end{aligned} \tag{42}$$

In which $\Delta\boldsymbol{p}_j$, $\Delta\boldsymbol{r}_j$ are the previous step matrices ($\Delta\boldsymbol{p}_{j-1}$, $\Delta\boldsymbol{r}_{j-1}$) which $\delta\boldsymbol{p}'_j$ and $\boldsymbol{r}'_j$ columns have been added to the end of each matrix respectively. We can rewrite the Eq (10) as follows:

$$J_j = \Delta\boldsymbol{r}_j \, \Delta\boldsymbol{p}_j^{\dagger} = \Delta\boldsymbol{r}_j \, \Delta\boldsymbol{p}_j^T \, \boldsymbol{A}_j^{-1} \tag{43}$$

where $\boldsymbol{A}_j = \Delta\boldsymbol{p}_j \, \Delta\boldsymbol{p}_j^T$ is a $t_p * t_p$ matrix. By applying multiplication property in partitioned matrices and use of Sherman-Morrison formula, $\boldsymbol{A}_j^{-1}$ is calculated as:

$$\begin{aligned}\boldsymbol{A}_j^{-1} &= \left(\boldsymbol{A}_{j-1} + \delta\boldsymbol{p}'_j \delta\boldsymbol{p}'^T_j\right)^{-1} \\ &= \boldsymbol{A}_{j-1}^{-1}\left(\boldsymbol{I}_{t_p * t_p} - \frac{\delta\boldsymbol{p}'_j \delta\boldsymbol{p}'^T_j \, \boldsymbol{A}_{j-1}^{-1}}{1 + \delta\boldsymbol{p}'^T_j \boldsymbol{A}_{j-1}^{-1} \delta\boldsymbol{p}'_j}\right)\end{aligned} \tag{44}$$

Considering $\Delta\boldsymbol{p}_{j-1}^{\dagger} = \Delta\boldsymbol{p}_{j-1}^T \, \boldsymbol{A}_{j-1}^{-1}$ and again using partitioned matrix properties, $\Delta\boldsymbol{p}_j^{\dagger}$ is calculated as:

$$\Delta\boldsymbol{p}_j^{\dagger} = \begin{bmatrix} \Delta\boldsymbol{p}_{j-1}^{\dagger}\left(\boldsymbol{I}_{t_p * t_p} - \frac{\delta\boldsymbol{p}'_j \delta\boldsymbol{p}'^T_j \, \boldsymbol{A}_{j-1}^{-1}}{1 + \delta\boldsymbol{p}'^T_j \boldsymbol{A}_{j-1}^{-1} \delta\boldsymbol{p}'_j}\right) \\ \hline \frac{\delta\boldsymbol{p}'^T_j \, \boldsymbol{A}_{j-1}^{-1}}{1 + \delta\boldsymbol{p}'^T_j \boldsymbol{A}_{j-1}^{-1} \delta\boldsymbol{p}'_j} \end{bmatrix} \tag{45}$$

Finally Jacobian matrix is calculated for j[th] updating as bellow:

$$\begin{aligned}J_j &= \left[\Delta\boldsymbol{r}_{j-1} \quad | \quad \boldsymbol{r}'_j\right] \Delta\boldsymbol{p}_j^{\dagger} \\ &= \boldsymbol{J}_{j-1}\left(\boldsymbol{I}_{t_p * t_p} - \frac{\delta\boldsymbol{p}'_j \delta\boldsymbol{p}'^T_j \, \boldsymbol{A}_{j-1}^{-1}}{1 + \delta\boldsymbol{p}'^T_j \boldsymbol{A}_{j-1}^{-1} \delta\boldsymbol{p}'_j}\right) + \frac{\boldsymbol{r}'_j \delta\boldsymbol{p}'^T_j \, \boldsymbol{A}_{j-1}^{-1}}{1 + \delta\boldsymbol{p}'^T_j \boldsymbol{A}_{j-1}^{-1} \delta\boldsymbol{p}'_j}\end{aligned} \tag{4}$$

Since $\boldsymbol{A}_j$ is symmetric, the inverse of it is symmetric, too. $\boldsymbol{d}_j$ vector is defined as:

$$\boldsymbol{d}_j = \boldsymbol{A}_j^{-1} \, \delta\boldsymbol{p}'_j \rightarrow \boldsymbol{d}_j^T = \delta\boldsymbol{p}'^T_j \, \boldsymbol{A}_j^{-1} \tag{47}$$

Table 3. Mean and standard deviation of point to point distance error for AAM model fitting when texture model is obtained through the variational Bayesian PCA. size ($\boldsymbol{p}$) represents number of elements in vector $\boldsymbol{p}$.

	Size ($\boldsymbol{p}$)	$\delta t_x = 0$ $\delta t_y = 0$		$\delta t_x = +8$ $\delta t_y = 0$		$\delta t_x = -8$ $\delta t_y = 0$		$\delta t_x = 0$ $\delta t_y = +8$		$\delta t_x = 0$ $\delta t_y = -8$		**Overall average**	
		mean	SD	mean	SD	mean	SD	mean	SD	mean	SD	mean	std
variational PCA	51	5.709	2.815	5.602	2.372	5.782	2.862	6.018	2.745	5.469	2.080	**5.716**	**2.575**

To prevent numerical instabilities coefficient $\alpha = 1/(\beta + |\delta \boldsymbol{p}'_j|^2)$ is used, in which β is small [17]. Since obtaining inverse of matrix $\boldsymbol{A}_j$ is a time-consuming process from calculation point of view, its inverse could be calculated once in training period, and then it can be updated by using Sherman-Morrison formula. Using Eq (47) and applying regulating coefficient, α, following equations will be obtained:

$$\boldsymbol{A}_j^{-1} = \boldsymbol{A}_{j-1}^{-1} + \frac{\alpha \boldsymbol{d}_j \boldsymbol{d}_j^T}{1 + \alpha \boldsymbol{d}_j^T \delta \boldsymbol{p}'_j} \quad (48)$$

$$\boldsymbol{J}_j = \boldsymbol{J}_{j-1}\left(\boldsymbol{I}_{t_p * t_p} - \frac{\alpha\, \delta \boldsymbol{p}'_j \boldsymbol{d}_j^T}{1 + \alpha \boldsymbol{d}_j^T \delta \boldsymbol{p}'_j}\right) + \frac{\alpha\, \boldsymbol{r}'_j \boldsymbol{d}_j^T}{1 + \alpha \boldsymbol{d}_j^T \delta \boldsymbol{p}'_j} \quad (49)$$

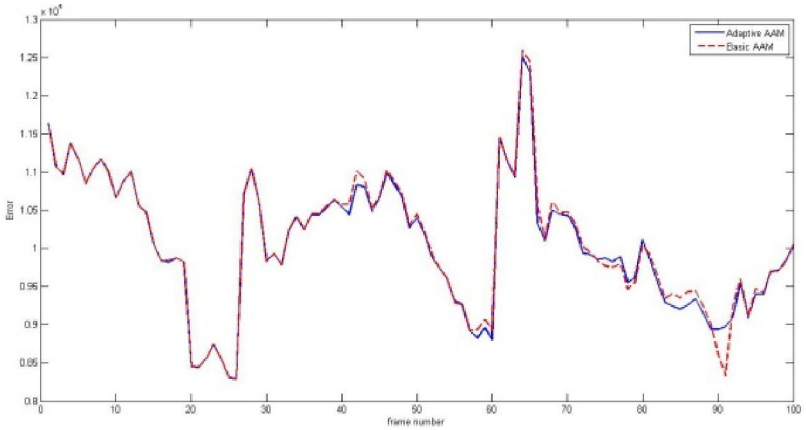
The steps of updating Regression matrix are described in Algorithm 2.

Algorithm 2. Algorithm of updating regression matrix

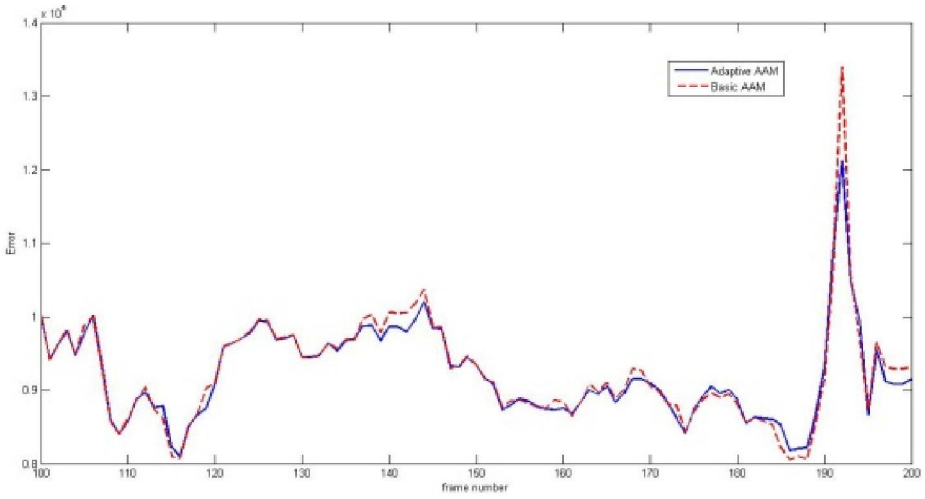
1:	set $\boldsymbol{J}_j = \boldsymbol{J}_0, \boldsymbol{A}_j^{-1} = (\Delta \boldsymbol{p}_0\, \Delta \boldsymbol{p}_0^T)^{-1}$ for j=0, *where Jacobian matrix, $\boldsymbol{J}_0$, and $\Delta \boldsymbol{p}_0$ are computed from training set*
2:	Use Algorithm 1 in order to fit AAM to target image. Obtain final model parameters, $\boldsymbol{p}_{final}$.
3:	**repeat**
4:	displace $\boldsymbol{p}_{final}$ by $\delta \boldsymbol{p}'_j$.
5:	compute $\boldsymbol{J}_j$ by using Eq (47), (48), (49)
6:	j=j+1
7:	**until** desired number of displacement occurs (we used 6 displacements vector $\delta \boldsymbol{p}'_j$ in our experiment).
8:	compute Matrix $\boldsymbol{R}_j = \boldsymbol{J}_j^{\dagger}$
9:	go to step 2 in order to fit AAM with new Regression matrix R to the next frame.

The most significant difference of suggested method from Cootes' is that in their method a minimized criterion function has been used for calculating Jacobian matrix updating equations instead of partitioned matrix. He also initialized $\boldsymbol{A}_0$ with unitary matrix while suggested method in this paper used $\boldsymbol{A}_0 = \Delta \boldsymbol{p}_0\, \Delta \boldsymbol{p}_0$ for initialization which $\Delta \boldsymbol{p}_0$ is obtained from the training set. Moreover, they update the Jacobian during fitting procedure of AAM, yet we update it after each successful fitting of AAM.

The less the texture residual error ($E = |\boldsymbol{r}|^2 = |\boldsymbol{g}_{image} - \boldsymbol{g}_{model}|^2$) is after converging fitting to target image, the more the similarity is between the model produced by AAM and the target image. Figure 7 shows final texture residual error in a video sequence with 200 frames for classic (basic) and adaptive AAM.



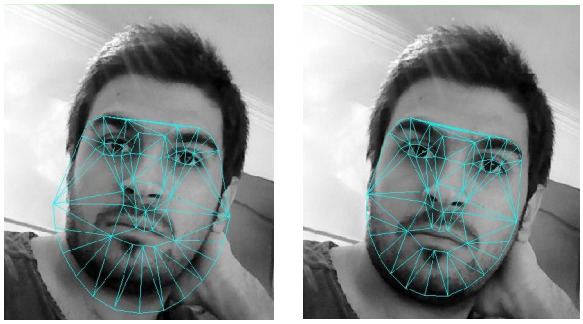
Error of frames 1 to 100



Error of frames 100 to 200

Figure 7. Texture residual error. The solid blue curve and the dotted red curves correspond to the Adaptive and Classic AAM respectively.

As it is observed in Figure 7, in adaptive form the amount of error has been decreased compared with the classic form particularly for frames 100 to 200. Figure 8 shows an example of model fitting for frame 192 from this video sequence.



Classic AAM Adaptive AAM

Figure 8. Frame 192 of the sequence depicted in Figure 7

5 Application of head pose estimation for Human-Robot Interaction (HRI)

Nowadays there is a close relationship between robots and human-beings, thus for better interaction with robots, instead of using traditional items (joystick, GUI and etc.), more user-friendly interfaces between these two are needed through which natural ways of human communication (Head pose, Hand and body gesture and even lip reading) are used. Head pose has a great role in HRI systems because it could be used as a rich source of information. For instance, being able to estimate 3-D head pose in real time, we can get a clue about the user's intentions [1].

This part of the research is about finding the 6 DOF pose of the face using proposed modified adaptive Active Appearance Model. In order to obtain AAM model we used variational PCA for texture modeling as mentioned in section 3.2.

POSIT algorithm [7] is used in this research for 3D estimation of head pose. For this purpose, we need a 3D rigid model of the face which is obtained from Candide-3 3D model [6]. In fact, Candide-3 is a parameterized face mask with 113 3D vertices. Since the face shape of IMM database consists of 58 points, in order to match these points with candide-3 points we used some kind of interpolation. Figure 9 shows some examples of head pose estimation by using POSIT algorithm and Candide-3 model.

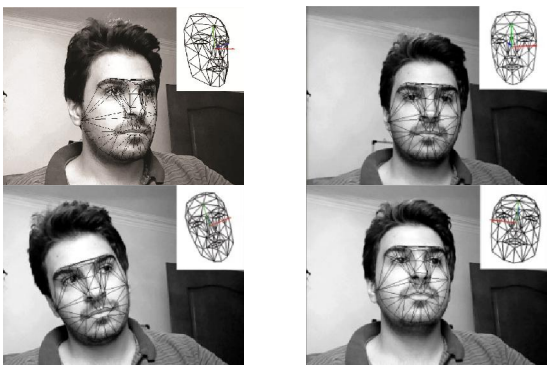


Figure 9. Estimation of head pose by using POSIT method and corresponding OpenGL model

Our application is intended to show how to move robot camera by changing of head orientation. We used Active Media's People Bot robot in our experiments. The System input includes video sequences which capture operator's face from a fixed camera. Then the major 2D points of face are obtained by the proposed adaptive Active Appearance Model.

The operator's 3D head pose is estimated in real time by using POSIT algorithm and Candide-3 model. Finally pitch & roll angles are sent to the robot camera using a wireless network. Robot camera orientation is updated online according to the desired direction imposed by the user's head pose (Figure 10). In order to accelerate the speed of data transfer, robot camera video signal is forwarded from robot (server) to the operator's computer (client) by an additional wireless audio/video transmitter. Following figure illustrates this system on a schematic basis.

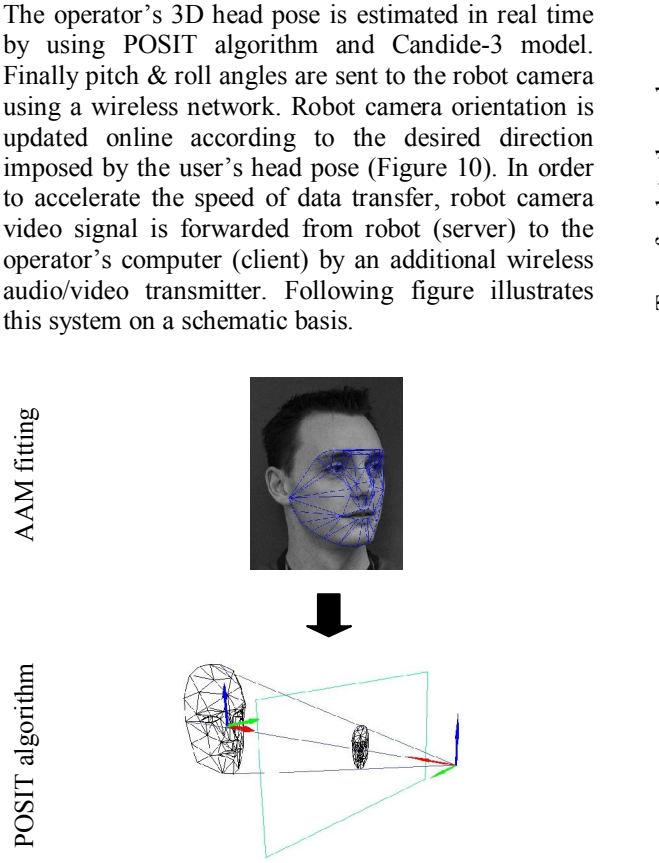
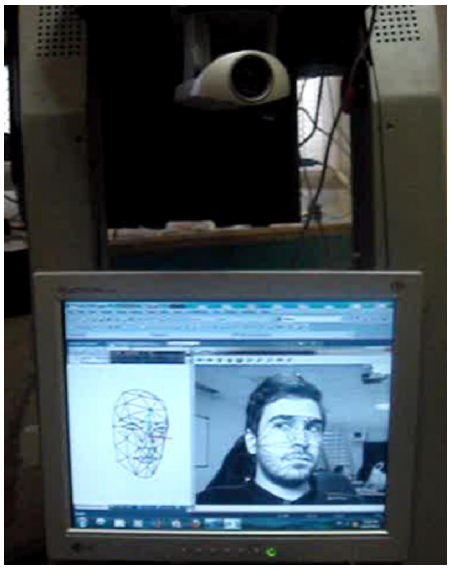


Figure 10. Dataflow to control orientation of robot camera by head pose estimation



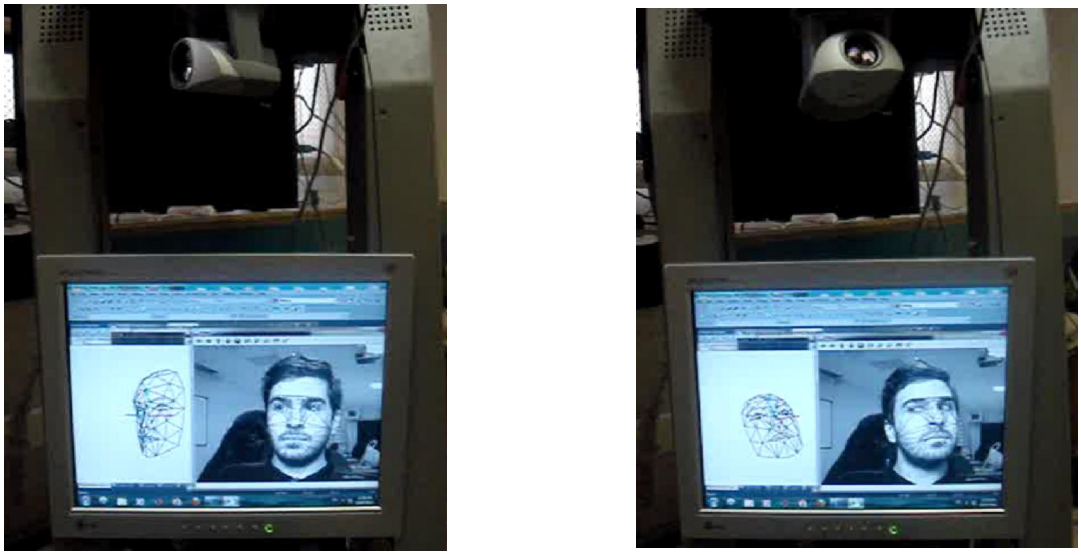


Figure 11. Examples of operator head pose Imitating. The 3 DOF rotational pose is represented by OpenGL model

Figure 11 illustrates the results of imitating operator's head pose by robot camera. The proposed HRI system is applicable in Telepresence. Meaning that operator and robot could be far from each other. So the operator is able to inspect robot environment without any presence in the place and just by changing his head direction. It is specifically important for rescue robots sent to areas affected by earthquakes or fires. For obstacle avoidance all range finder devices of the robot (IR, Sonar, Laser, bump sensors) have been used.

6 Conclusion

In this paper three major factors are described. First, we present two probabilistic methods for texture modeling in Active Appearance Models with further comparisons. As it was obvious, variational PCA method was better than classic AAM in accuracy and speed of calculations.

Then an adaptive model was presented for updating gradient matrix in AAM. Finally we used the mentioned method as an application for head pose estimation.

One of the advantages of this method is that the operator does not need to be in place for further inspection which may cause a reduction in any damages at high-risks areas to do the same by a robot. Since one of the major weak points of statistical models is their sensitivity to imaging conditions, therefore one of our future works is using Active Appearance Models in presence of cluttered background.

Acknowledgement

Authors are grateful to the Mobile Robot Research Laboratory, Amir Kabir University of Technology for support to carry out this work.

Corresponding Author:
Navid Mahmoudian Bidgoli
Department of Electrical Engineering
AmirKabir University of Technology, Tehran, Iran
E-mail: navid_mahmoudian@aut.ac.ir

References

1. Dornaika, Fadi and Raducanu, Bogdan. Three-Dimensional Face Pose Detecion and Tracking Using Monocular Videos: Tool and Application. IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS--PART B: CYBERNETICS, AUGUST 2009, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS--PART B: CYBERNETICS, VOL.39, NO. 4, Vol. 39, pp. 935-944.
2. Murphy-Chutorian, Erik and Manubhai Trivedi, Mohan. Head Pose Estimation and Augmented Reality Tracking: An Integrated System and Evaluation for Monitoring Driver Awareness. IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, June 2010, Vol. 11, pp. 300-311.
3. Ba, Sileye O, and Odobez, Jean-Marc. Multiperson Visual Focus of Attention from Head Pose and Meeting Contextual Cues. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, January 2011, Vol. 33, pp. 101-116.

4. Murphy-Chutorian, Erik and Manubhai Trivedi, Mohan. Head Pose Estimation in Computer Vision: A Survey. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 2008.
5. Dornaika, Fadi and Davoine, Frank. On Appearance Based Face and Facial Action Tracking. IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, SEPTEMBER 2006, Vol. 16.
6. Ahlberg, Jörgen. CANDIDE-3 - AN UPDATED PARAMETERISED FACE. Linköping, SWEDEN: Image Coding Group, Dept. of Electrical Engineering, Linköping University, 2001.
7. DeMenthon, Daniel F. and Davis, Larry S. Model-based object pose in 25 lines of code. International Journal of Computer Vision, 1995.
8. Cootes, T.F., Edwards, G.J. and Taylor, C.J. Active Appearance Models. Springer, 1998. European Conf. on computer vision. Vol. 2, pp. 484-498.
9. Cootes, Timothy F., Edwards, Gareth J. and Taylor, Christopher J. —. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, June 2001, Vol. 23, pp. 681-685.
10. Nordstrom, Michael M., et al. The IMM Face Database- An Annotated Dataset of 240 Face Images. Lyngby: Informatics and Mathematical Modelling, Technical University of Denmark, 2003.
11. Fukunaga, Keinosuke. Introduction to Statistical Pattern Recognition. Academic Press, 1990.
12. Tresadern, Phil, et al. Face Alignment Models. [book auth.] Stan Z. Li and Anil K. Jain. Handbook of Face Recognition (2nd edition). London: Springer, 2011.
13. Viola, Paul and Jones, Michael. Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE conf. Computer vision and Pattern Recognition, 2001.
14. Tipping, Michael E. and Bishop, Christopher M. Probabilistic Principal Component Analysis. 61, J. of the Royal statistical society, 1999, Vol. Series B (Statistical Methodology).
15. Bishop, Christopher M. Pattern Recognition and Machine Learning. Springer, 2006.
16. Bishop, Christopher M. Variational Principal components. In Proceedings Ninth International Conference on Aritificial Neural Networks, ICANN'99, 1999. Vol. 1, pp. 509-514.
17. Batur, Aziz Umit and Hayes, Monson H. Adaptive Active Appearance Models. IEEE TRANSACTIONS ON IMAGE PROCESSING, 2005, Vol. 14.
18. Cootes, T.F. and Taylor, C.j. An Algorithm for Tuning an Active Appearance Appearance Model to New Data. British Machine Vision Conference, 2006. Vol. 3, pp. 919-928.

3/25/2020