Journal of Marine Science and Technology

Volume 22 | Issue 3 Article 2

ACTIVE APPEARANCE MODEL ALGORITHM WITH K-NEAREST NEIGHBOR CLASSIFIER FOR FACE POSE ESTIMATION

Bing-Fei Wu
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C,
bwu@cssp.cn.nctu.edu.tw

Chih-Chung Kao
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C

Cheng-Lung Jen
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C

Chia-Rong Chiang
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C

Po-Hung Lai
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C

Follow this and additional works at: https://jmstt.ntou.edu.tw/journal

 Part of the Controls and Control Theory Commons

Recommended Citation

Wu, Bing-Fei; Kao, Chih-Chung; Jen, Cheng-Lung; Chiang, Chia-Rong; and Lai, Po-Hung (2014) "ACTIVE APPEARANCE MODEL ALGORITHM WITH K-NEAREST NEIGHBOR CLASSIFIER FOR FACE POSE ESTIMATION," *Journal of Marine Science and Technology*: Vol. 22: Iss. 3, Article 2.
DOI: 10.6119/JMST-013-0716-1
Available at: https://jmstt.ntou.edu.tw/journal/vol22/iss3/2

This Research Article is brought to you for free and open access by Journal of Marine Science and Technology. It has been accepted for inclusion in Journal of Marine Science and Technology by an authorized editor of Journal of Marine Science and Technology.

ACTIVE APPEARANCE MODEL ALGORITHM WITH K-NEAREST NEIGHBOR CLASSIFIER FOR FACE POSE ESTIMATION

Acknowledgements

This work was supported by the National Science Council, Taiwan, under Grant NSC 102-2221-E-009-141.

This research article is available in Journal of Marine Science and Technology: https://jmstt.ntou.edu.tw/journal/vol22/iss3/2

Journal of Marine Science and Technology, Vol. 22, No. 3, pp. 285-294 (*2014*)
DOI: 10.6119/JMST-013-0716-1

ACTIVE APPEARANCE MODEL ALGORITHM WITH K-NEAREST NEIGHBOR CLASSIFIER FOR FACE POSE ESTIMATION

Bing-Fei Wu, Chih-Chung Kao, Cheng-Lung Jen,
Chia-Rong Chiang, and Po-Hung Lai

Key words: active appearance model, shape model, texture model, face pose estimation, k-nearest neighbor.

ABSTRACT

In this paper, a face pose estimation (FPE) algorithm using active appearance model (AAM) with a k-nearest neighbor (kNN) classifier is presented. AAM is a model-based image representation method used to describe non-rigid visual objects, with both shape and texture variations, using a mean vector and linear combinations of a set of variation modes. Since AAM is a deformable model, it has several variations. Owing to the variations, the model is adjusted to the input test face image using iterative searching and fitting. The error, which measures the difference between the model and a test image, is minimized with the proposed searching algorithm. The face pose is then estimated using the distances between the landmark points in the AAM model with a kNN classifier. Experimental results demonstrate that the proposed FPE algorithm can fit the face location with different face poses, with or without a hat, even wearing glasses, and identify the face pose accurately.

I. INTRODUCTION

In recent years, the human interface issue has become a very popular research field, particularly in robot controlling, computer gaming, computer vision, medical services, and people identification. Many enthusiastic researchers, therefore, have dedicated themselves to developing and investigating systems to detect and recognize human kinetics. Face recognition systems are receiving the most attention. Using biometric methods with low intrusiveness, they are delivering high accuracy. Face recognition systems include face detection, face identification, face pose estimation, and facial expression recognition. Early face recognition systems focused on feature-based methods, and face contour is a popular example. Because the contour of the human face resembles an ellipse, in the early years of facial research, this feature played an important role in identifying a face. Specific distinct features were marked as reliable landmarks for featured-based approaches. Early works on face recognition were mostly based on feature-based methods. Kanade [12] used landmark features as judging points to identify a face. This research made use of a simple template-based method of calculating Euclidean distances to recognize faces. Following this, some researchers employed more sophisticated feature extraction methods, including the Hough Transform [17], morphological operations [9], and Reisfeld's symmetry operator [20]. These methods, however, could not fit the shape of the input images perfectly. Statistical analysis is the task of calculating the correlation between input images and based-data. Principal component analysis (PCA) is a popular method used to analyze correlation information. Eigenvalues and eigenfeatures, which estimate the data distribution, are calculated by PCA. However, the results of the data distribution prediction by PCA were not acceptable. Other methods investigated include linear discriminate analysis(LDA), independent component analysis (ICA), and probabilistic eigenfaces. Other methods, such as machine learning and neural network, were able to distinguish important features from the training data. Feature information is divided into several classes. The best known binary classification methods are Adaboost and support vector machine (SVM). Existing methods of face recognition can be categorized into three groups: holistic, local, and hybrid.

1. Holistic Methods

Conforming to the definition of the word holistic, the input data for these methods make use of a whole face image. Early research used knowledge-based methods, such as Kirby and Sirovich [13], who applied features in eigenspaces to recognize faces using PCA. Because PCA cannot distinguish faces of different people, other methods, such as LDA [25], and

Paper submitted 08/20/12; revised 04/30/13; accepted 07/16/13. Author for correspondence: Bing-Fei Wu (e-mail: bwu@cssp.cn.nctu.edu.tw).
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.

Fisherface [2] were used with PCA to make the algorithm more robust. The key benefit of these methods is that they retain the entire shape, texture, and details of the human face. This information can be used to evaluate the human face. When the size of the judgment features differ from the size of the input images, however, holistic methods encounter problems, including pose angle and illumination.

2. Local Methods

Local methods extract essential feature points from limited regions of the face as the base for determining judgment features. A local template comparison was proposed [5] and Pentland *et al*. [19] extended PCA, extracting only the eyes, nose, mouth, and facial contour, instead of the whole face image. Other researchers did not use the facial features directly. The local facial features were mapped to other spaces. For example, Tan *et al*. [22] found a probability distribution for these local features. With this relationship, the facial features were mapped to a corresponding feature plane, called a self-organizing map (SOM) [14]. The plane with the mapping features was called the SOM-face. This could express different facial features. However, local methods are significantly affected by face pose. They are applied mainly, therefore, to frontal face views. Localized gradient orientation histograms are employed with support vector regressors (SVRs) for pose estimation [16].

3. Hybrid Methods

Hybrid methods combine both holistic and local methods [23]. However, the challenge is to integrate these two feature methods and use them effectively. There are still many obstacles to be conquered. First, illumination is an important issue. Testing images [1] are seriously affected by luminance. Second, the recognition of different face poses using kn1owledge-based methods is difficult. This has become a popular topic in recent studies.

Face pose estimation (FPE) is an interesting research topic in the field of human computer interface. It is easy for human beings to determine face poses. Unfortunately, it is a difficult technical challenge for computer vision. In recent years, research has addressed new methods to solve this challenge. A brief description of some of these techniques follows. Appearance template methods: A database of faces with different poses is created. An input image is then compared with this database for a likeness [3]. Detector Array: The data array structure is similar to the appearance template. However, the comparison process is quite different. With detector array, every pose is set up in the database using supervised learning to acquire the judgment statistics. These data are then used to recognize an input face pose. In early research, SVM was used to define and recognize three standard face poses [10]. Recently, methods utilizing neural networks [21] or Adaboost [11] to achieve FPE have been developed. Nonlinear regression is used to develop a nonlinear function for every pose and features can fit to the function. Neural networks, using the structure of supervised learning, use feedback values, iteratively, to modify the weighting of previous layers and base points. Until the results converge [18], Cootes *et al*. [6] utilized the relationship between facial features, including the eyes, nose and mouth. Input images are compared to distinct locations to find the most similar face. A flexible model is built utilizing a set of training data. Initially, essential landmarks are manually labeled and analyzed statistically to find the principal components. Active appearance model (AAM) [7] is a popular fitting model. Generally, the face in a non-frontal pose is harder to recognize. In this paper, the face is approximated by an AAM model. Using several iterations to determine the most fitting position, the face is recognized even though it may be in a different pose. After the AAM fitting, the face pose can be identified using k-nearest neighbor (kNN) classifier and the distances of the landmarks. In FPE, the fitting procedure exhibits good performance even if the person is wearing glasses or is in a different pose.

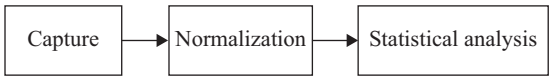


Fig. 1. Setup process for shape and texture models of AAM.

The remainder of this paper is organized as follows. In Section II, AAM is introduced, including shape, texture, and combined modeling. Section III presents the procedures for AAM fitting. Our FPE approach is discussed in Section IV. The experimental evaluations of our FPE methodology are discussed in Section V. Finally, the conclusion is presented in SectionVI.

II. AAM MODELING

Dealing with the large variability of the image data has been difficult. Recently, however, the deformable template model, which is a model-based method, has proven very successful in addressing this challenge. In our face pose estimation approach, AAM, which is a deformable model method, is an important procedure to model and fit the face image for pose estimation. The setup process of the shape and texture models is the same. Following the steps in Fig. 1,the shape and texture models are created. The shape model is built first. The texture model is then established based on the reference mean shape using the following procedure. In the capture step, a shape model uses a finite number of points to determine the contour of the deformable shape. At the same time, a texture model uses the piece-wise Affine warp bilinear interpolation to extract the texture features. In the normalization step, a shape model aligns the shape with translation, scaling, and rotation. Finally, in the statistical analysis step, the shape or texture model is analyzed by PCA to evaluate the eigenvalues and eigenvectors to describe the variations of the shape and texture model.

1. Shape Model

In order to extract the shape of faces, the landmark points of

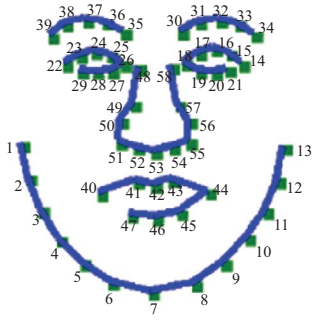


Fig. 2. Example landmarks for connectivity scheme.

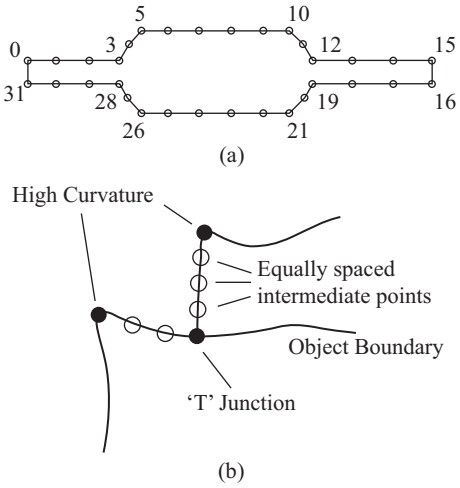


Fig. 3. Example landmarks for connectivity scheme [4].

the database images must first be labeled manually. Fig. 2 represents these points and contour.

To prevent unrelated points from distorting the analysis result, choosing the landmark points is an essential task. According to [4], three corresponding features should be sufficient for our purpose, for example, an intersection of two boundaries, such as points 0, 3, and 5 in Fig. 3(a). Faces, eye corners, mouth corners or other similar points could be selected. Prominent points of the image are shown in Fig. 3(b). Boundary line features label points along a boundary, such as points 1 to 2, and 6 to 9 in Fig. 3(a).

To obtain a statistical model correctly, analyzing all the shapes should be based on the same reference point. The classical data alignment solution is Procrustesanalysis (PA). The alignment of two shapes requires finding the best match of one shape to another by minimizing the Procrustes distance in (1), which indicates the distance between two shapes, with respect to scaling, rotation, and translation.

$$D(\mathbf{x}_1,\mathbf{x}_2)=\sqrt{\sum_{i=1}^{n}(x_{1i}-x_{2i})^2+(y_{1i}-y_{2i})^2}, \qquad (1)$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ indicate the landmark points of two labeled

Table 1. Procrustes Analysis.

Algorithm 1
1. Find the centroid of each shape: $(\bar{x},\bar{y})=(\frac{1}{n}\sum_{i=1}^{n}x_i,\frac{1}{n}\sum_{i=1}^{n}y_i)$
2. Align two shapes to their origin: $(\mathbf{x}_c,\mathbf{y}_c)=(\mathbf{x}-\bar{x},\mathbf{y}-\bar{y})$
3. Normalize each shape: $\hat{\mathbf{x}}_c=\frac{\mathbf{x}_c}{\|x_c\|}$
4. Set shape matrices: $\mathbf{X}=[\hat{x}\,\|\,\hat{y}]_{n\times 2}$
5. Evaluate $SVD(X_1X_2) = USV^T$
Find the optimal rotation matrix by SVD: $R = UV^T$

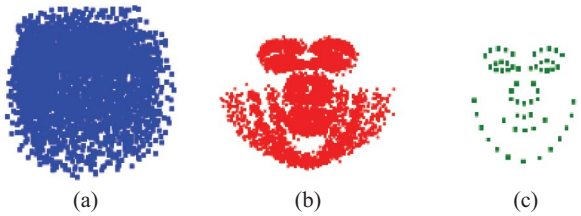


Fig. 4. Result of Generalized Procrustes Analysis. (a) Raw data, (b) The result of Generalized Procrustes Analysis, and (c) Mean shape.

shapes.x_{1i}, y_{1i}, x_{2i}, and y_{2i} represent the coordinates of the i^{th} point of shape $\mathbf{x}_1$ and $\mathbf{x}_2$. During PA, the centroid of every shape is calculated first. Then every shape is moved to its origin. This removes the mean of every shape. To prevent the scaling of a shape affecting the data alignment, every shape is normalized to the same size. The rotation problem is solved by singular value decomposition (SVD). The purpose of using SVD is to find the optimal rotation matrix R that leads to the minimum distance difference of the two shapes. After obtaining the rotation matrix R, the last step of PA is to modify the aligned shapes using this rotation matrix R. As a result, the minimum distance of the two shapes is determined. The details of Procrustes distance are listed in Table 1. However, PA only addresses the data alignment between two shapes. Generalized Procrustesanalysis (GPA), which specifically processes serial shape data alignment, is an extension of PA.

In GPA, the first shape is set as the initial estimate mean shape and all the other shapes are aligned to this estimate mean shape using PA. After the first data alignment pass, a new estimate mean shape is generated. This data alignment procedure runs repetitively. The procedure stops when the data alignment converges; that is, the new estimate mean shape is similar to the previous estimate mean shape. The initial data of all the shapes are shown in Fig. 4(a); the results of GPA are shown in Fig. 4(b); and the mean of all the aligned data is addressed in Fig. 4(c). Upon completing GPA, the serially new-aligned data is analyzed by PCA. In the PCA process, the covariance matrix $\mathbf{C}_s$ is calculated in (2),

$$\mathbf{C}_s=\frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i-\bar{\mathbf{x}})(\mathbf{x}_i-\bar{\mathbf{x}})^T, \qquad (2)$$

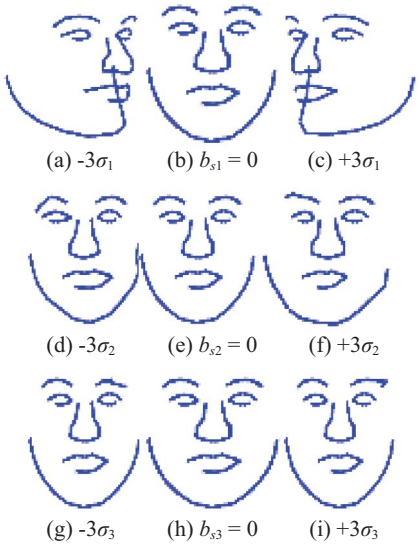


Fig. 5. The top three shape variation modes in the shape model.

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i, \tag{3}$$

where N is the total number of shapes in the training set, $\bar{\mathbf{x}}$ indicates the mean shape of the deformable model as shown in Fig. 4(c), and $\mathbf{x}_i$ represents each labeled shape. Eigenvalues and eigenvectors, which demonstrate possible data distribution coordinates and the variance of each coordinate, are obtained by PCA data analysis.

The statistical variation shape $\mathbf{x}$ can be modeled in Eq. (4).

$$\mathbf{x} = \bar{\mathbf{x}} + \Phi_s b_s, \tag{4}$$

where Φ_s and b_s represent the eigenvector and a parameter, which controls the shape variance of the covariance matrix. The calculated mean shape is $\bar{\mathbf{x}}$. Choosing different parameters for Φ_s and b_s leads to different variations of the shape model. Fig. 5 displays the different variations of the top three modes. In Fig. 5, each row implies one mode, and each mode represents a variation of each eigenvector. The mean shapes are presented in Fig. 5(b), (e), and (h). The variances of the first, second, and third eigenvectors are illustrated in Fig. 5(a) and (c), Fig. 5(d) and (f), and Fig. 5(g) and (i).

2. Texture Model

Texture feature extraction is an essential process in the development of a texture model. A texture model describes the intensity of the entire face and can display precise facial changes. Every texture face of m pixels can be represented as

$$\mathbf{g} = \left[g_1, g_2, \ldots, g_{m-1}, g_m\right]^T, \tag{5}$$

where g_i is the i^{th} pixel value of the texture vector $\mathbf{g}$. The first step of building a texture model is to determine the points of the mean shape. These indicate the important control points of the face. The relationship of each shape and the mean shape must be established. Delaunay triangulation (DT) is a method used to obtain the relationship of two shapes and textures. The purpose of DT is to connect a finite set of points in 2-D space into several triangulation networks. The vertexes of each network are the control points in the mean shape. These triangle networks do not intercept each other. The resulting mean shape after applying DT is illustrated in Fig. 6.

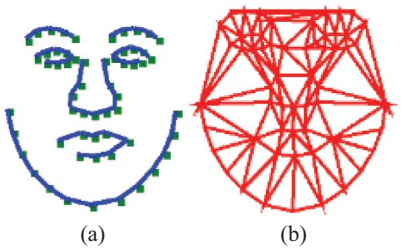


Fig. 6. Mean shape Delaunay Triangulation (DT). (a) Mean shape; (b) Mean shape after Delaunay Triangulation.

After the DT process, the aligned shapes are triangulated into a similar number of triangle networks. To find the corresponding vertex of two shapes, barycentric coordinates are used. Any point $\mathbf{x} = [x\ y]^T$ in a triangle can be expressed as:

$$\mathbf{x} = \alpha\mathbf{x}_1 + \beta\mathbf{x}_2 + \gamma\mathbf{x}_3, \tag{6}$$

where $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ are vertexes of the triangle, α, β, γ are barycentric coordinates of $\mathbf{x}$ in relation to $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$. The solution can be obtained using (7),

$$\begin{aligned}
\alpha &= 1-(\beta+\gamma)\\
\beta &= \frac{yx_3 - x_1y - x_3y_1 - y_3x + x_1y_3 + xy_1}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2}\\
\gamma &= \frac{xy_2 - xy_1 - x_1y_2 - x_2y + x_2y_1 + x_1y}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2}.
\end{aligned} \tag{7}$$

When finding point $\mathbf{x}$ in a triangle network with vertices $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, the solution of barycentric coordinates parameter should be between 0 and 1. To minimize the amount of data that cannot be mapped to a corresponding data coordinate, backward barycentric coordinate mapping is applied. After the backward mapping, some of the textures may still have tiny holes in the mapping texture. A bilinear interpolation technique is used to eliminate these holes.

When the texture is processed by bilinear interpolation and barycentric coordinates, reconstructed texture faces are obtained. Fig. 7 shows the original face, a textured face mapped into a mean shape, and the original shape with DT. Combining the texture of every triangle network produces an entire texture face. Every sample will generate a mapped texture face.

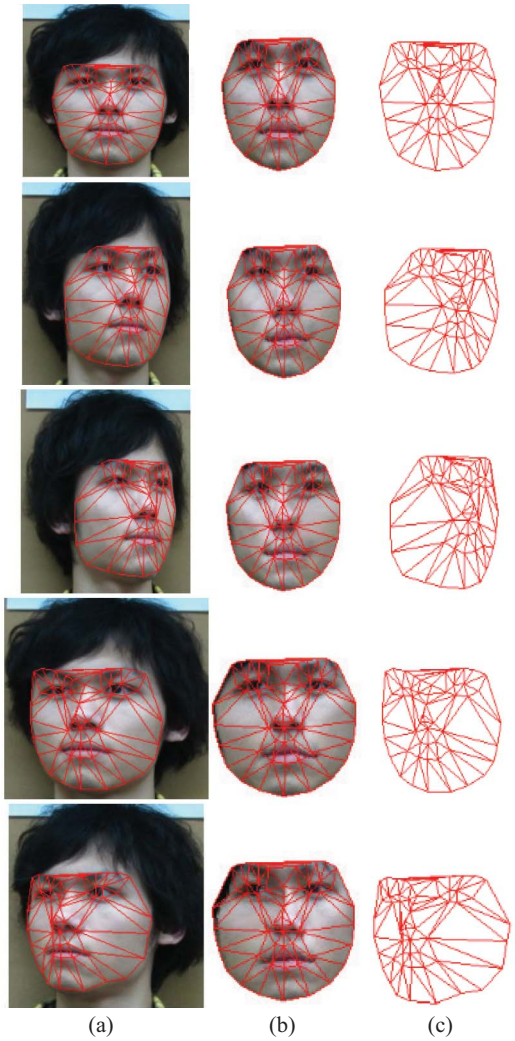


Fig. 7. The mapping faces. (a) original image, (b) mapping texture face into mean shape, and (c) original shape with DT.

With these mapped texture faces, the characteristic of the data can be analyzed. To find the variance of the texture, these mapped sample faces are analyzed using PCA. The PCA for the texture model is computed as:

$$\bar{\mathbf{g}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{g}_i, \tag{8}$$

$$\mathbf{C}_{g(m\times m)} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{g}_i - \bar{\mathbf{g}})(\mathbf{g}_i - \bar{\mathbf{g}})^T, \tag{9}$$

where $\bar{\mathbf{g}}$ and $\mathbf{C}_g$ represent the mean of the texture and the covariance matrix of the texture. $\mathbf{g}_i$ is the i^{th} texture image, and N is the sample number. However, the dimension of $\mathbf{C}_{g(m\times m)}$ will be too large to calculate. Using the Eckart-Young theorem [8], this dimension problem is overcome. The $N \times N$ covariance matrix is calculated in (10). The first N eigenvalues are the same as the eigenvalues of $\mathbf{C}_{g(m\times m)}$.

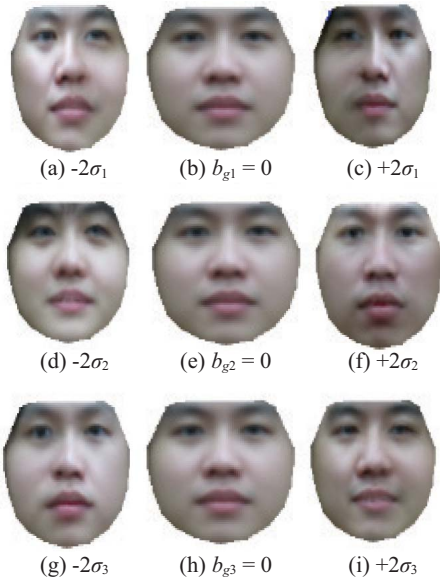


Fig. 8. The top three texture variation modes in the texture model.

$$\mathbf{C}_{g(N\times N)} = \frac{1}{N-1}\sum_{i=1}^{m}(\mathbf{g}_i - \bar{\mathbf{g}})(\mathbf{g}_i - \bar{\mathbf{g}})^T, \tag{10}$$

where $\mathbf{g}_i$ is the i^{th} pixel in the texture image, and m is the total pixel number in the mean texture.

After applying PCA to the texture image, each texture image $\mathbf{g}$ can be expressed as:

$$\mathbf{g} = \bar{\mathbf{g}} + \Phi_g b_g \tag{11}$$

where Φ_g and b_g indicates the matrix of the eigenvectors and the eigenvalues.

Fig. 8 presents the variance of the texture faces. The mean textures are shown in Fig. 8(b), (e), and (h). Fig. 8(a) and (c), Fig. 8 (d) and (f), and Fig. 8(g) and (i) are the variances of the first, second, and third eigenvectors, respectively. As seen in Fig. 8, the first mode represents the change of face direction in the texture image. The second mode indicates the change in the eye and eyebrow part. Finally, the third mode shows the change in the mouth part.

3. Combined Model

Shape and texture are described by the parameters $\mathbf{b}_s$ and $\mathbf{b}_g$. The combining parameter is described in (12):

$$\mathbf{b} = \begin{pmatrix} \mathbf{W}_s\mathbf{b}_s \\ \mathbf{b}_g \end{pmatrix} = \begin{pmatrix} \mathbf{W}_s\Phi_s^T(\mathbf{x}-\bar{\mathbf{x}}) \\ \Phi_g^T(\mathbf{g}-\bar{\mathbf{g}}) \end{pmatrix}, \tag{12}$$

where $\mathbf{b}_s$ and $\mathbf{b}_g$ indicate the distance units and intensity units, respectively. Because distance units and intensity units do not have a direct relationship, $\mathbf{W}_s$ is the combination weighting matrix of $\mathbf{b}_s$ and $\mathbf{b}_g$.

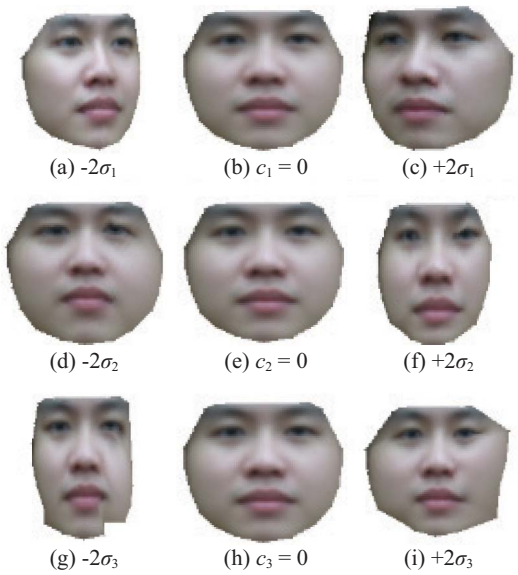


Fig. 9. The top three variation modes of the combined model.

$$\mathbf{W}_s = \mathrm{Diag}(r), \tag{13}$$

where r is the weight ratio calculated in (14),

$$r = \sum_{i=1}^{N} \lambda_{g_i} \Big/ \sum_{j=1}^{n} \lambda_{s_j}, \tag{14}$$

where λ_s and λ_g are the eigenvalues of the shape and texture, respectively.

To remove the correlation between the shape and texture model, the covariance of **b** is again analyzed by PCA. After PCA, the new combinations of eigenvalues and eigenvectors are obtained as:

$$\mathbf{b} = \Phi_c \mathbf{c} = \begin{pmatrix} \Phi_{cs} \\ \Phi_{cg} \end{pmatrix} \mathbf{c}, \tag{15}$$

where Φ_c and **c** are the eigenvectors and variance of the vector, respectively. Moreover, parameter **c** is the essential parameter that can control both the shape and texture of the combined model.

Due to the linear nature of the model, it is possible to express the shape**s**, and the texture ***g***, using the combined model simultaneously in (16). Several shape and texture faces are yielded.

$$\begin{aligned} \mathbf{x} &= \bar{\mathbf{x}} + \Phi_s \mathbf{W}_s^{-1} \Phi_{cs} \mathbf{c}, \\ \mathbf{g} &= \bar{\mathbf{g}} + \Phi_g \Phi_{cg} \mathbf{c}. \end{aligned} \tag{16}$$

To combine these two features, DT is used as a mapping tool that maps the texture features to the shape features. The result of the combined model is shown in Fig. 9. Fig. 9(b), (e), and (h) represent the mean texture of each mode. Fig. 9(a) and (c), (d), and (f), and (g) and (i) are the variance of the first, second, and third eigenvector, respectively.

III. AAM FITTING

1. Training Mode

The AAM search method calculates the texture difference between the combined model and an input image in (17). The appearance variation parameters are updated iteratively. Therefore, the correlation between the texture difference and the parameters must be established in the training mode.

$$\arg\min_{\mathbf{c}} \left| \mathbf{I}_{\text{image}} - \mathbf{I}_{\text{model}} \right| \tag{17}$$

where $\mathbf{I}_{\text{image}}$ and $\mathbf{I}_{\text{model}}$ are the input and model image, and **c** is the variance of the vector.

Because we are using more than one parameter, multilinear approximation is applied to estimate the regression matrix representing the relationship between the texture difference and the parameters. Using this concept, only a few different displacements are needed, and each of these displacements is part of the training data for evaluating the regression matrix.

$$\delta_{\mathbf{p}} = \mathbf{R} \delta_{\mathbf{g}} \tag{18}$$

where $\delta_{\mathbf{p}}$ is the difference of the appearance parameters (including translation parameter **t** and appearance parameter **c**), $\delta_{\mathbf{g}}$ is the texture difference, and **R** indicates the regression matrix of $\delta_{\mathbf{p}}$ and $\delta_{\mathbf{g}}$. The consequence of the regression has a significant effect on the prediction of the appearance parameter. The details of the training procedure are addressed in Table 2.

2. Searching Mode

The input image first generates a texture model. To achieve this goal, the error between the input image and the base texture model is calculated. This result is a judgment to evaluate the similarity between the base model and the input image. The current error (E_{curr}) is compared to the previous error (E_{prev}). If E_{curr} is bigger than E_{prev}, the fitting procedure modifies the step size and compares the step size with the threshold. Otherwise, FPE records the current data and updates the parameters that are used to predict the final position. The above step processes iteratively until the final position, which indicates the correct pose, is found. From the previous steps and the camera parameters, the proposed FPE can estimate the different poses. The flowchart for this process is addressed in Fig. 10. After performing the searching procedure, the input image is fitted by the model and the positions of the landmarks are output as the face pose estimation inputs.

Table 2. Generation of displacement for training procedure.

Algorithm 2

for each sample image
for each displacement
1. decide the dimension of parameter **p** (that implies the dimension of parameter **t**; parameter **c** should be decided)
2. update values of appearance parameter $\mathbf{c} = \delta\mathbf{c} + \mathbf{c}_0$
3. update values of $\mathbf{t} = \delta\mathbf{t} + \mathbf{t}_0$
4. using the new parameters in steps 2 and 3, establish a new shape model **x** and texture model $\mathbf{g}_m$
5. align shape $\mathbf{x}_{\text{image}}$ by using parameter **t**
6. obtain texture vector $\mathbf{g}_{\text{image}}$ under the shape $\mathbf{x}_{\text{image}}$
7. map texture vector $\mathbf{g}_{\text{image}}$ onto the normalize texture vector $\mathbf{g}_i$
8. once all the texture vectors are mapped onto the normalized texture vector, the texture difference can be obtained, $\delta_{\mathbf{g}} = \mathbf{g}_i - \mathbf{g}_m$
9. fill $\delta\mathbf{t}$ into the corresponding places of the matrix $\Delta_{\mathbf{p}}$
10. fill $\delta_{\mathbf{g}}$ into the corresponding places of matrix $\Delta_{\mathbf{g}}$

end for
end for

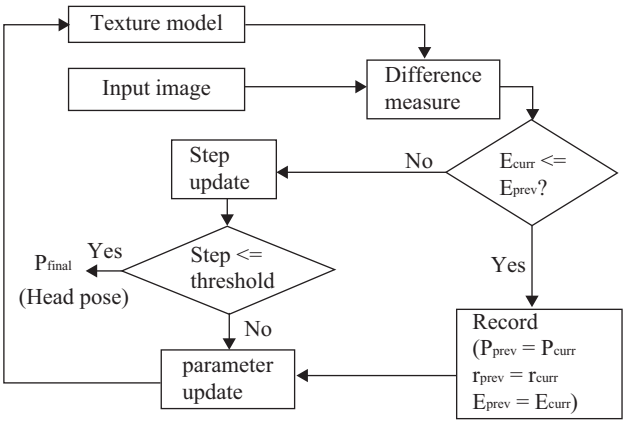


Fig. 10. The flowchart for AAM searching.

IV. FACE POSE ESTIMATION

In order to identify the face pose of the input image, kNN classifier is adopted to distinguish the face direction. kNN is an instance-based classifier that gives a high accuracy recognition ratio with less computing load. The half samples that include the front, left, and right pose, are set as training samples. Then, the FPE process is executed according to the landmark distances of the AAM fitted result. The flowchart of FPE is illustrated in Fig. 11. Once the kNN classifier is trained, the distances of the landmark points from the AAM fitted result are fed as input **x** to the kNN classifier. The pose is then identified by the kNN classifier. Although the shape model and texture model are adjusted simultaneously, only

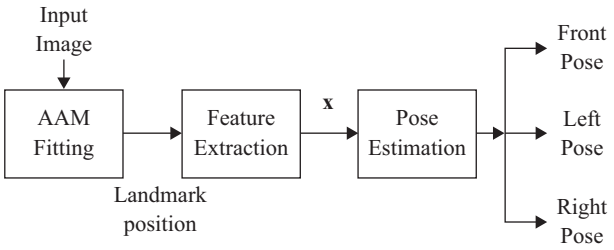


Fig. 11. Flowchart of face pose estimation.

the positions of the landmarks in the shape model are useful for FPE. Therefore, the input parameters **x** are extracted according to the distances of the landmark positions after the AAM fitting. Though there are 58 landmark points defined in Fig. 2, the landmarks of the eyebrow, eye, mouth, and nose parts exhibit a weak difference in the different poses. Accordingly, points in the chin and cheek are used for the feature extraction. Input **x** is a feature vector with 24 dimensions, and the features are defined as follows,

$$dx_i^j = x_i - x_j,\ dy_i^j = y_i - y_j, \tag{19}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \mathbf{x}_4 \end{bmatrix}^T, \tag{20}$$

$$\mathbf{x}_1 = \begin{bmatrix} dx_1^2 & dx_{13}^{12} & dx_2^3 & dx_{12}^{11} & dx_3^4 & dx_{11}^{10} \end{bmatrix}, \tag{21}$$

$$\mathbf{x}_2 = \begin{bmatrix} dx_4^5 & dx_{10}^9 & dx_5^6 & dx_9^8 & dx_6^7 & dx_8^7 \end{bmatrix}, \tag{22}$$

$$\mathbf{x}_3 = \begin{bmatrix} dy_1^2 & dy_{13}^{12} & dy_2^3 & dy_{12}^{11} & dy_3^4 & dy_{11}^{10} \end{bmatrix}, \tag{23}$$

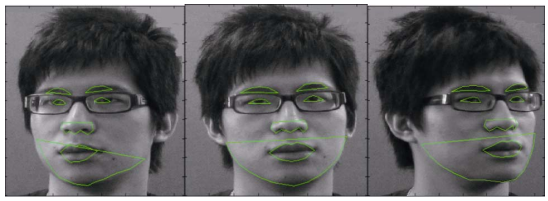
$$\mathbf{x}_4 = \begin{bmatrix} dy_4^5 & dy_{10}^9 & dy_5^6 & dy_9^8 & dy_6^7 & dx_8^7 \end{bmatrix} \tag{24}$$

where *dx* and *dy* represent the distances in the x and y direction between the i^{th} point and the j^{th} point, respectively.

The order of the feature vector is arranged according to the importance of the feature elements. Because the difference in the x-direction is more obvious than the difference in the y-direction, the *dx* elements are placed in front of the *dy* elements. Moreover, the movements of the landmarks in the cheek are clearer than the movements of the landmarks in the chin. Therefore, the *dx* elements of the cheek are placed in front of the *dx* of the chin.

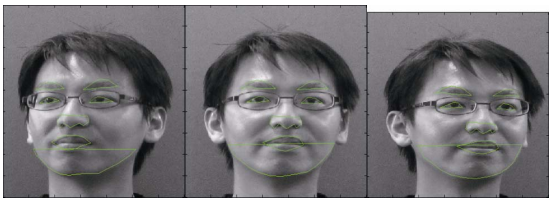
V. EXPERIMENTAL RESULTS

Experiments were carried out to evaluate the performance of the proposed FPE using AAM. The testing images included several different frontal and angle face poses. After the AAM fitting, the images that are fit successfully are divided into two sets, the training set and the testing set. In the quantitative

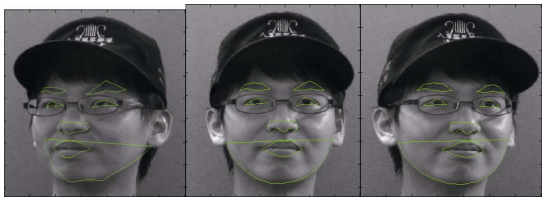


(a) Person 1 without hat.

(b) Person 1 with hat.

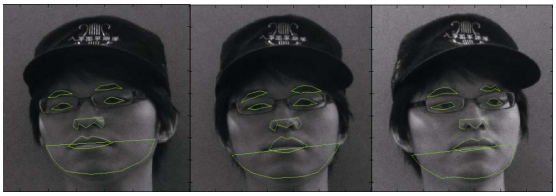


(c) Person 2 without hat.



(d) Person 2 with hat.

(e) Person 3 without hat.



(f) Person 3 with hat.

Fig. 12. Different face pose estimation. (a) (c) (e) Left, frontal and right face pose estimation without hat. (b) (d) (f) Left, frontal and right face pose estimation with hat.

evaluations of the proposed FPE approach, Recall and Precision in (25) and (26), are used to evaluate the performances.

$$\text{Recall} = T_p / (T_p + F_n), \tag{25}$$

$$\text{Precision} = T_p / (T_p + F_p), \tag{26}$$

Table 3. Comparison of face pose estimation with different *k* numbers in the training set.

k	definition	1	3	5	7	9
Left	T_p	19	18	19	18	18
	F_n	1	2	1	2	2
	F_p	0	1	2	0	0
	Recall	95	90	95	90	90
	Precision	100	94.74	90.48	100	100
Front	T_p	20	19	18	20	20
	F_n	0	1	2	0	0
	F_p	2	5	5	7	7
	Recall	100	95	90	100	100
	Precision	90	79.17	78.26	74.07	74.07
Right	T_p	19	17	16	15	15
	F_n	1	3	4	5	5
	F_p	0	0	0	0	0
	Recall	95	85	80	75	75
	Precision	100	100	100	100	100
Total	T_p	58	54	53	53	53
	F_n	2	6	7	7	7
	F_p	2	6	7	7	7
	Recall	96.67	90	88.33	88.33	88.33
	Precision	96.67	90	88.33	88.33	83.33

Table 4. Comparison of face pose estimation with different *k* numbers in the testing set.

k	definition	1	3	5	7	9
Left	T_p	8	11	12	12	10
	F_n	5	2	1	1	3
	F_p	2	3	3	4	4
	Recall	61.54	84.62	92.31	92.31	76.92
	Precision	80	78.57	80	75	71.43
Front	T_p	33	33	34	33	33
	F_n	6	6	5	6	6
	F_p	7	3	3	4	5
	Recall	84.62	84.62	87.18	84.62	84.62
	Precision	82.50	91.67	91.89	89.19	86.84
Right	T_p	8	9	8	7	8
	F_n	2	1	2	3	2
	F_p	4	3	2	2	2
	Recall	80	90	80	70	80
	Precision	66.67	75	80	77.78	80
Total	T_p	49	53	54	52	51
	F_n	13	9	8	10	11
	F_p	13	9	8	10	11
	Recall	79.03	85.48	87.10	83.87	82.25
	Precision	79.03	85.48	87.10	83.87	82.25

where T_p (true positives), F_p (false positives), and F_n (false negatives) represent the number of correctly identified face pose direction, falsely identified face pose from the other pose direction, and the number of missing face in the correct pose, respectively.

Fig. 12 shows the pose estimation results with hat, as in Fig. 12(a), (c), and (e), and without hat, as in Fig. 12(b), (d), and (f). As seen in Fig. 12, the AAM model can be adaptively adjusted to the input image with different poses. Moreover, the proposed FPE algorithm can fit the model to input images under different illuminations or wearing glasses. As seen in Fig. 12(c) and (f), even with different illumination, AAM can still fit the face accurately. After the AAM fitting, the positions of the landmarks are output and the feature vector is extracted. Using the feature vector and kNN classifier, the face pose, left, front and right-directions, can be identified correctly.

Tables 3 and 4 illustrate the classification results of FPE in the training and testing sets, with a different k number for kNN. As shown in Table 3, Precision is always 100 percent regardless of what k is in the classifier, because there is no other directional face recognized as the right pose face. However, Recalls in the right and left poses are much lower than the front pose, since the feature vectors in the front pose are more uniform than the other two direction poses. Recall in the front face set is high and the precision is low, because front pose faces are not as easy to recognize as the other direction faces. Other direction pose faces are easily classified into a front pose. Table 4 illustrates that Recall and Precision in the testing set are not as good as in the training set. However, Recall and Precision do achieve 80 percent with a k of 3, 5, 7, and 9 in the total testing set. Precision in the front pose face is better than that of the other two directional faces, and Recall is still greater than 90 percent. The performance of 1NN in both sets are changed severely, since using only one nearest neighbor is unreliable when the variance of the image is large.

In order to achieve a high Precision, the number of features for kNN was analyzed. The classification results in the training set and testing set with different feature numbers are shown in Figs. 13 and 14, respectively. The number of used first feature in the feature vector **x**, denoted as F. F is eight, twelve, and twenty-four representing the x-directional distances in the cheek, entire x-directional distances of the feature vector, and both x- and y-directional distances used. F is two, indicating that only the first two features are used. As seen in Figs. 13 and 14, Precision approaches 100 percent when F is 8 and 12 in 1NN. However, Precision in 1NN does not perform well, because considering only one nearest neighbor as a classification result is unreliable. The result of using only two features is worse than others in both the training and testing sets. In general, Precision in the testing set when F is 12 in 5NN is better than other results. If we do not consider 1NN in the training set, when F is 12 in 5NN, we get the best results in the training set. Precision when F is 12 in 5NN in the training set and testing set is over 95 and 90 percent, respectively. The y-directional features are not helpful, especially when k increases. This is reasonable in that the y-directional distances of the landmarks in the chin and cheek are not obviously different in the three-directional pose faces.

The experimental results of the Adaboost classifier [11] are listed in Table 5. Adaboost is comprised of many weak classifiers. As seen in Fig. 15, after the weak classifier number (W) is adjusted, the best classification result, which appears when W is 30 and F is 12, is still lower than 85 percent. In Table 5, in any directional face, Recall and Precision of the proposed FPE approach are better than the Adaboost classifier when F is 12.

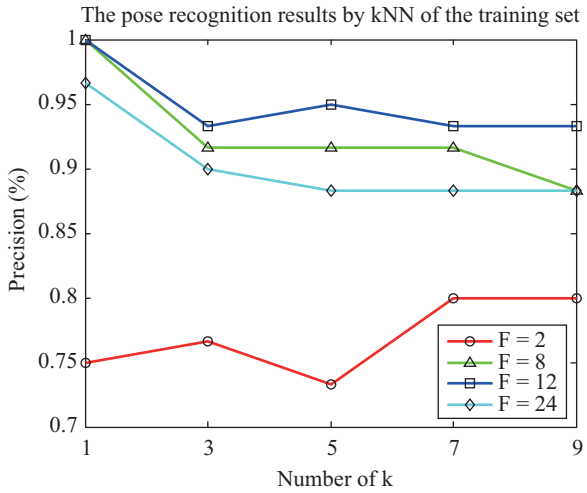


Fig. 13. Pose recognition results by *k*NN witha different feature number F in the training set.

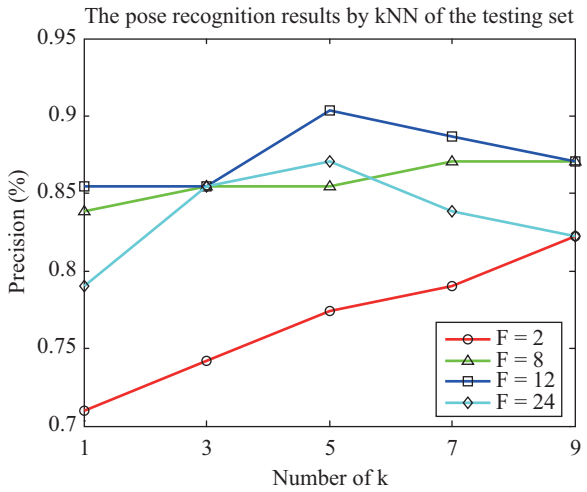


Fig. 14. Pose recognition results by *k*NN with a different feature number F in the testing set.

VI. CONCLUSION

A new FPE algorithm based on AAM and kNN is presented in this work. The shape and texture variation are modeled in AAM simultaneously. In FPE, the face is searched by iteratively fitting the model to the face image. By using AAM, the input images are fitted, including the different pose faces under different luminance and wearing glasses. Using AAM,

Table 5. Comparison of face pose estimation when F is 12 in the testing set. W is 30 for Adaboost, and k is 5 for kNN.

Definition	Left		Front		Right		Total	
	Ada.	kNN	Ada.	kNN	Ada.	kNN	Ada.	kNN
T_p	11	12	34	35	7	9	52	56
F_n	2	1	5	4	3	1	10	6
F_p	3	2	5	2	2	2	10	6
Recall	84.6	92.3	87.2	89.7	70	90	83.9	90.3
Precision	78.6	85.7	87.2	94.6	77.8	81.8	83.9	90.3

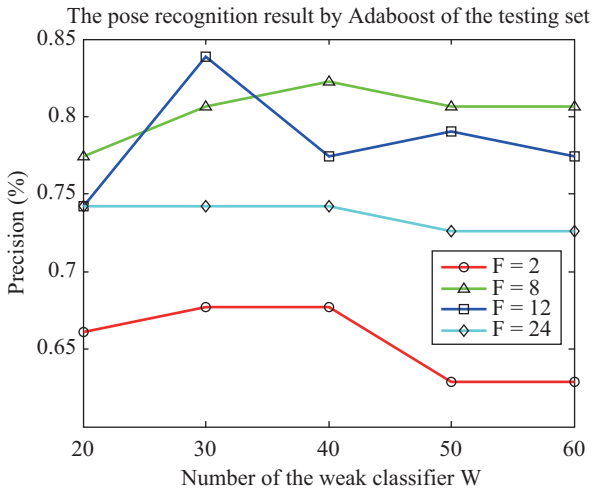


Fig. 15. The pose recognition resultsusing the Adaboost classifier with different feature number F in the testing set.

the controlling parameters that adjust the model to fit the input image are obtained, and the feature vector for FPE is extracted. The different face poses can be estimated by using the trained kNN classifier. According to the analysis of the value of k, the performance in 5NN is better than the other k numbers when twelve features, meaning only x-directional distances of the landmark points, are used. The results showed that the proposed FPE performs better than the Adaboost classifier.

ACKNOWLEDGMENTS

This work was supported by the National Science Council, Taiwan, under Grant NSC 102-2221-E-009-141.

REFERENCES

1. Adini, Y., Moses, Y., and Ullman, S., "The problem of compensating for changes in illumination direction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 721-732 (1997).
2. Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J., "Eigenfaces *vs.* fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 711-720 (1997).
3. Beymer, D., "Face recognition under varying pose," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 756-761 (1994).
4. Bookstein, F. L., *Morphometric Tools for Landmark Data*, Cambridge University Press (1991).
5. Brunelli, R. and Poggio, T., "Face recognition: Features versus templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 10, pp. 1042-1052 (1993).
6. Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J., "Active shape models - their training and application," *Computer Vision and Image Understanding*, Vol. 61, No. 1, pp. 38-59 (1995).
7. Cootes, T. F., Taylor, C. J., and Edwards, G. J., "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 6, pp. 681-685 (2001).
8. Eckart, C. and Young, G., "The approximation of one matrix by another of lower rank," *Psychometrika*, Vol. 1, No. 3, pp. 211-218 (1936).
9. Graf, H. P., Chen, T., Petajan, E., and Cosatto, E., "Locating faces and facial parts," *Proceeding of the International Conference on Automatic Face and Gesture Recognition*, pp. 41-46 (1995).
10. Huang, J., Shao, X., and Wechsler, H., "Face pose discrimination using support vector machines (SVM)," *Proceeding of the International Conference on Pattern Recognition*, Vol. 1, pp. 154-156 (1998).
11. Jones, M. and Viola, P., *Fast Multi-View Face Detection*, Technical Report 096, Mitsubishi Electric Research Laboratories (2003).
12. Kanade, T., *Picture Processing System by Computer Complex and Recognition of Human Faces*, Ph.D. Dissertation, Kyoto University, Japan (1973).
13. Kirby, M. and Sirovich, L., "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 1, pp. 103-108 (1990).
14. Kohonen, T., *Self-Organizing Map*, Springer, Berlin (1997).
15. Li, H. Q., Wang, S. Y., and Qi, F. H., "Automatic face recognition by support vector machines," *Proceeding of the Combinatorial Image Analysis*, Vol. 3322, pp. 716-725 (2004).
16. Murphy-Chutorian, E. and Trivedi, M. M., "Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 11, No. 2, pp. 300-311 (2010).
17. Nixon, M., "Eye spacing measurement for facial recognition," *SPIE Proceedings*, Vol. 0575, pp. 279-285 (1985).
18. Osadchy, M., Cun, Y. L., and Miller, M. L., "Synergistic face detection and pose estimation with energy-based model," *The Journal of Machine Research*, Vol. 8, pp. 1197-1215 (2007).
19. Pentland, A., Moghaddam, B., and Starner, T., "View-based and modular eigenspaces for face recognition," *Proceeding of the International Conference on Computer Vision and Pattern Recognition*, pp. 84-91 (1994).
20. Resfeld, D., *Generalized Symmetry Transforms: Attentional Mechanisms and Face Recognition*, Ph.D. Dissertation, Tel-Aviv University (1994).
21. Rowley, H., Baluja, S., and Kanade, T., "Rotation invariant neural network-based face detection," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 38-44 (1998).
22. Tan, X., Chen, S. C., and Zhou, Z.-H., "Recognizing partially occluded, expression variant faces from single training image per person with SOM and soft k-NN ensemble," *IEEE Transaction on Neural Networks*, Vol. 16, No. 4, pp. 875-886 (2005).
23. Valenti, R., Sebe, N., and Gevers, T., "Combining head pose and eye location information for gaze estimation," *IEEE Transactions on Image Processing*, Vol. 21, No. 2, pp. 802-815 (2012).
24. Viola, P. and Jones, M. J., "Robust real-time face detection," *International Journal of Computer Vision*, Vol. 52, No. 2, pp. 137-154 (2004).
25. Zhao, W., Chellappa, R., and Krishnaswamy, A., "Discriminate analysis of principal component for face recognition," *Proceeding of the International Conference on Computer Vision and Pattern Recognition*, pp. 336-341 (1998).