Head Pose Estimation via Manifold Learning

Chao Wang, Yuanhao Guo and Xubo Song

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/65903

Abstract

For the last decades, manifold learning has shown its advantage of efficient non-linear dimensionality reduction in data analysis. Based on the assumption that informative and discriminative representation of the data lies on a low-dimensional smooth manifold which implicitly embedded in the original high-dimensional space, manifold learning aims to learn the low-dimensional representation following some geometrical protocols, such as preserving piecewise local structure of the original data. Manifold learning also plays an important role in the applications of computer vision, i.e., face image analysis. According to the observations that many face-related research is benefitted by the head pose estimation, and the continuous variation of head pose can be modelled and interpreted as a low-dimensional smooth manifold, we will focus on the head pose estimation via manifold learning in this chapter. Generally, head pose is hard to directly explore from the high-dimensional space interpreted as face images, which is, however, can be efficiently represented in low-dimensional manifold. Therefore, in this chapter, classical manifold learning algorithms are introduced and the corresponding application on head pose estimation are elaborated. Several extensions of manifold learning algorithms which are developed especially for head pose estimation are also discussed and compared.

Keywords: manifold learning, head pose estimation, nonlinear feature reduction, supervised manifold learning, local linearity, global geometry

1. Introduction

Manifold learning becomes well known due to its property to learn the representative geometry in low-dimensional embedding, with which data analysis and visualization are significantly benefitted. From the observation of some nonlinear data, a low-dimensional smooth manifold (differentiable manifold) is embedded in the original high-dimensional space, which is implicit if we only consider the metrics of the original space. Manifold learning algorithm purpose is to learn such embedding according to some protocols, e.g., local linearity and global structure preserving. For the remainder of this chapter, the term manifold is used to refer as the smooth manifolds (differentiable manifolds) for convenience. As a complex high-dimensional data, face image analysis is a difficult topic in the field of computer vision due to the complicated facial appearance variations, among which the head pose challenges many face-related applications. Accurate head pose estimation is advantageous to face alignment and recognition, because frontal- or near-frontal faces are easier to handle compared with other poses. It has been found that facial appearance is lying on a manifold embedded form in the original high-dimensional space represented as face images. Correspondingly, the head pose can also be represented as a low-dimensional embedding, which is more representative and discriminative to model the variation. Therefore, the head pose estimation can be implemented by the manifold learning.

In principle, head pose refers to the view of the face to the imaging system, i.e., the camera center. 3D head transformation involves 6 degrees of freedom (DOF), which can be interpreted as the 3D translations $(t_x, t_y, t_z)^T$ and rotations $(\alpha, \theta, \gamma)^T$ from the head to the camera center. Among the six variables, the 3D rotations that are formally represented by pitch, roll, and yaw are taken as the head pose [1]. A schematic demonstration taken from reference [1] is shown in **Figure 1**. This definition reduces the head pose to 3 DOF which are sufficient to model most of the in- and out-plane rotation of the head. It can be found that the pitch and yaw generate more self-occlusions and roll can be easily corrected by the position of eyes. So, in this chapter, the 2 DOF including yaw and pitch are considered. Usually, the head poses of yaw and pitch lead to the problem of self-occlusion, which subsequently results in the loss of informative features, e.g., facial texture and shapes. By comparing, frontal- or near-frontal faces are relatively easier to deal with. Examples of various head poses [2] are given in **Figure 2**. The task of the head pose estimation is actually to determine the yaw and pitch of an unknown face image or search the frontal face in a database. Once the head pose is obtained, further applications such as face alignment and recognition will be benefitted in efficiency and accuracy. Therefore, modern development of face-related research prefers to estimate the head pose and correct the head orientation by positioning the face to near frontal or warping the faces in various poses to a frontal face template [3].

Basically, head pose estimation methods broadly fall into several categories. Template-based methods treat the head pose estimation as a verification (or classification) problem. The testing face is projected to the data set labeled with known poses, the one from which the most significant similarity measured by various metrics is retrieved for the testing pose [4, 5]. Furthermore, pose detectors can be learned to simultaneously localize the face and recognize the pose [6]. Regression-based methods estimate a linear or nonlinear function with the original faces or extracted facial features as input variables and discrete or continuous poses as output [2]. Deformable models learn flexible facial modes [7–9]. By manipulating a set of parameters which specifies the pose, specific face example can be generated, which will be used to match the testing face. With the development of manifold learning [10–13], more promising results of head pose estimation are achieved. The essence of such methods is based on the assumption that the discriminative modes for head pose lie on low-dimensional manifolds embedded in high-dimensional space, i.e., the original color space or other low level feature space [14]. The low-dimensional representation of the head pose images can be learned by unsupervised or supervised manifold learning.

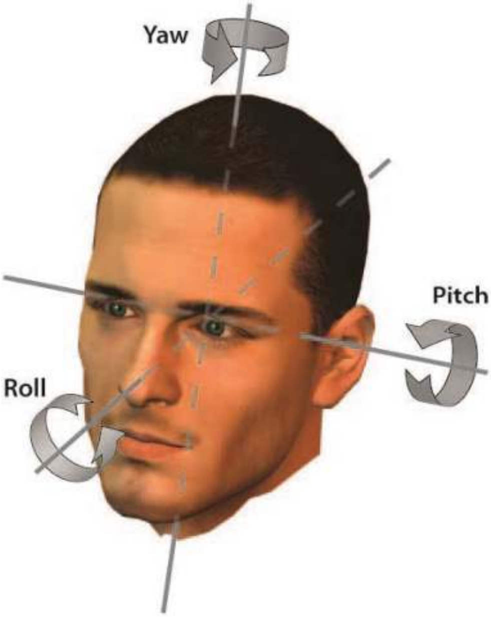

Figure 1. The 3 DOF of head pose proposed in reference [1]. The roll does not introduce any self-occlusion of the face, which can be easily corrected. Compared with the pitch, the yaw produces more serious self-occlusion problem.

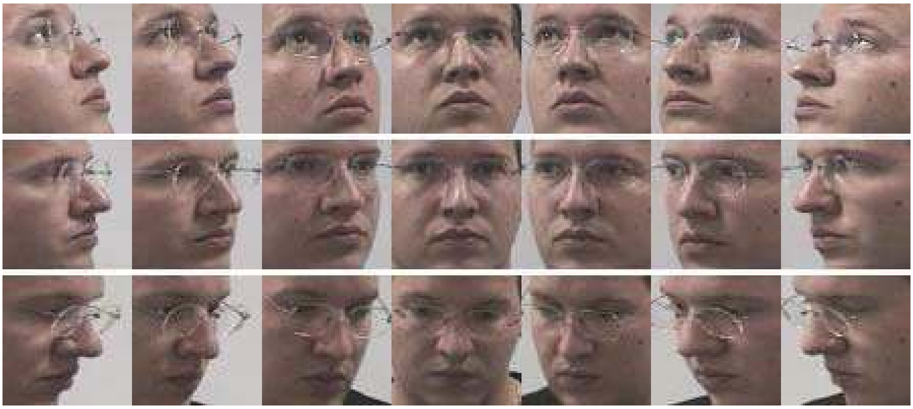

Figure 2. Examples of various head poses. The images are cropped, centered, and resized to 64 × 64 pixels from the originals. One individual is selected and shown in different yaw and pitch. From left to right represents the variation in yaw: −90, −60, −30, 0, 30, 60, and 90°. From top to bottom represents the variation in pitch: 30, 0, and −30°. One can find that the effects of self-occlusion occur with an increasing yaw and pitch. The frontal faces (center image) show a full overview of the face.

In contrast, the template-based methods have the problem of serious dependence on training data. If similar poses to the query pose do not exist in the training set, the estimated result would be biased. The regression-based methods often require to use complicated regression models, for example, a high-order polynomial. However, complicated nonlinear function would cause the problem of overfitting, which will result in poor generalization of the model. The deformable models require the localization of dense facial features, such as landmarks of facial components, which are seriously influenced by the head pose. The manifold learning-based methods are somehow limited by some problems, such as identity and noise sensitivity; however, simple efforts can be made to efficiently improve the performance [15]. More importantly, the manifold learning-based methods show promising performance of generalization. And the head pose can be easily modeled and better visualized with low-dimensional features. More details will be given in following sections.

According to the previous analysis, the main focus of this chapter will be on the manifold learning based on head pose estimation. The main notations used in this chapter are listed and interrupted in Section 2. In Section 3, classical manifold learning algorithms will be elaborated. In Section 4, adaptions and extensions of manifold learning algorithms, which are more suitable for head pose estimation, are discussed. Section 4 summaries the work, and some available resources of manifold learning are given.

2. Notations

$\boldsymbol{x_i} = (x_{i1}, x_{i2}, \ldots, x_{iD})^{\boldsymbol{T}}$ represents the ith data point in the original D-dimensional space. $\boldsymbol{x}$ without subscript is used to represent an arbitrary data point.

$\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M)$ represents the M-data points collection.

$N(i)$ represents the set of K-nearest neighbor of the ith data point.

$\boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots, y_{id})^{\boldsymbol{T}}$ represents the d-dimensional representation of the ith data point after dimensionality reduction. Similarly, $\boldsymbol{y}$ without subscript is used to represent an arbitrary data point.

$\mathbf{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_M)$ denotes the data collection in the low-dimensional space.

$\mathbf{C} = \mathbf{X}\mathbf{X}^{\boldsymbol{T}}$ is the covariance matrix for the centered data, where $\frac{1}{M}\sum_{i=1}^{M}\boldsymbol{x}_i = \mathbf{0}$. Usually, this can be enabled by mean subtraction: $\boldsymbol{x} = \boldsymbol{x} – \boldsymbol{\mu}$, and $\boldsymbol{\mu} = \frac{1}{M}\sum_{i=1}^{M}\boldsymbol{x}_i$.

$\mathbf{W} = \{w_{ij}\}$ is the weight matrix to model the graph for pairwise face images, which will be specified by different metrics, e.g. $w_{ij} = 1$ if the jth data point is one of the K nearest neighbors of the ith data point, and $w_{ij} = 0$ otherwise.

$\mathbf{D} = \{d_{ij}\}$ is the distance matrix measuring the pairwise distances among the data points, which can be Euclidean or other distance metrics.

$\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_D)$ is a diagonal matrix whose elements are the D eigenvalues (ranked decreasingly $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_D \geq 0$ decomposed from an $D \times D$ matrix.

$\mathbf{V} = (v_1, v_2, \ldots, v_d, \ldots, v_{D-d+1}, \ldots, v_D)$ is the projection matrix, which is consisted of the eigenvectors corresponding to the ranked eigenvalues. $(v_1, v_2, \ldots, v_d)$ are the top d eigenvectors, and $(v_{D-d+1}, \ldots, v_D)$ are the bottom d eigenvectors.

I denotes the identity matrix.

3. Characteristics of manifold learning algorithms

Given a set of data points, for example, face images, it is difficult to directly estimate or extract the most significant modes from such high-dimensional representation of the data. If the distribution of data in the original feature space can be linearly structured, the classical principal component analysis (PCA) will be able to estimate the discriminative modes and then reduce the feature dimensions. An example of such a type of data is shown in **Figure 3**. However, if the data distribution of the original data is nonlinear, for example, the famous "swiss roll" shown in **Figure 4(a)**, which is a smooth, continuous but nonlinear surface embedded in the 3D space, the structure interpreted as Euclidean distance is less preferable to represent the distribution of the data. Taking the two circled points sampled from the manifold shown in **Figure 4(b)**, for instance, their Euclidean distance is close, while this is not guaranteed if the 3D structure is considered. The embedded structure can be explored with the help of nonlinear dimensionality reduction, such as manifold learning algorithms. The learned low-dimensional representation can approximately model the real distance of the sampled data points as shown in **Figure 4(c)**.

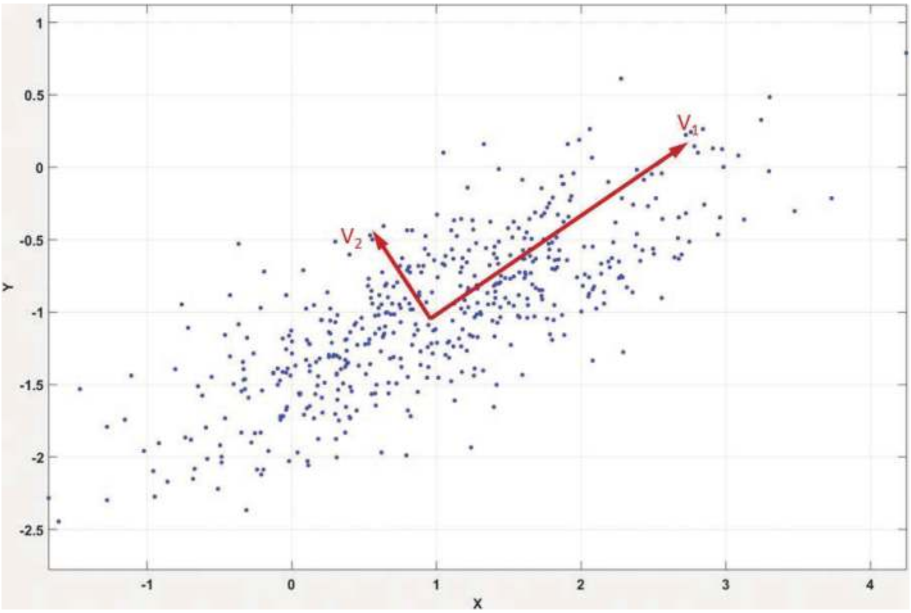

Figure 3. A data set sampled from a multivariate Gaussian distribution. The most significant modes indicated by the red orthogonal axis can be learned by PCA, which preserve the largest variations in the original data.

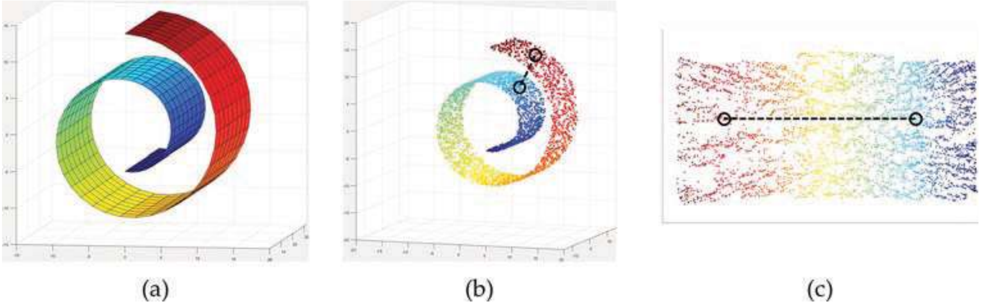

Figure 4. An example of the data set including a potential "swiss roll" structure. The figures are produced based on the code from [16]. (a) The original 3D surface. (b) The data points sampled from (a). The Euclidean distance indicated by dash line between the circled points cannot represent the distance lying on the potential structure. (c) The distance measured in the learned low-dimensional can more accurately model the data.

In this section, in order to reveal the essence of manifold learning, the PCA is initially detailed. Other classical manifold learning algorithms will be elaborated in the following.

3.1. Principal component analysis (PCA)

PCA is one of the most popular unsupervised linear dimensionality reduction algorithms. The intrinsic feature of PCA is to estimate a linear space whose basis will be able to preserve the maximum variations in the original data. Mathematically, the low-dimensional data can be obtained by a linear transformation from the original data as denoted in Eq. (1).

$$\boldsymbol{y} = \mathbf{V}^T \boldsymbol{x} \tag{1}$$

where $\boldsymbol{x}$ is the centered data point. The entry of the projection matrix $\mathbf{V}$ is the column vector that represents the principal components in the projection space. Let us take one of the principal components for instance. The objective is to preserve the maximum variations in the transformed data.

$$\max_{\|v\|=\mathbf{1}} Var(\boldsymbol{v}^T\mathbf{X}) = \max_{v} \frac{1}{M-1}\|\boldsymbol{v}^T\mathbf{X}\|^2 = \max_{v}\frac{1}{M-1}(\boldsymbol{v}^T\mathbf{X})(\mathbf{X}^T\boldsymbol{v}) = \max_{v}(\boldsymbol{v}^T\mathbf{C}\boldsymbol{v}) \tag{2}$$

$\frac{1}{M-1}$ makes Eq. (2) an unbiased estimation, which can be replaced by M if it is sufficiently large. Eq. (2) is a form of Rayleigh's quotient, which can be maximized by eigenvector decomposition of covariance matrix $\mathbf{C}$. The d eigenvectors corresponding to the top d positive eigenvalues are taken to construct the low-dimensional space $\boldsymbol{V}_{D\times d}$ to which the original data are projected. Actually, Eq. (2) can be converted to

$$\boldsymbol{x} = \mathbf{V}\boldsymbol{y} \tag{3}$$

which means that the original data can be linearly represented as a combination of the principal components.

Taking the head pose images of one identity shown in **Figure 2**, for example, the PCA is applied on the vectorized images. **Figure 5** visualizes the low-dimensional representation of the face images in the first 3D dimensions. One can find obvious transitions for pitch and yaw along a 3D shape of valley. The three principal components are visualized in **Figure 6**, from which one face image is decomposed into a weighted accumulation of variations in the mean face. The first and third eigenfaces (principal component) clearly show the variation in yaw. Therefore, PCA can model the head poses as some of the discriminative principal components.

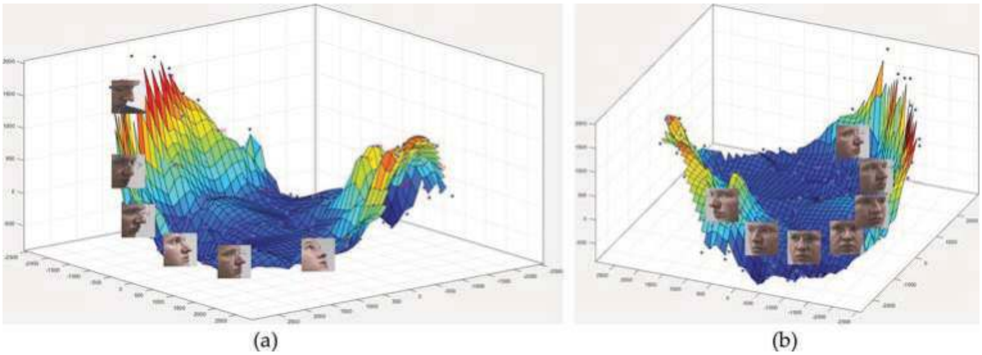

Figure 5. Visualization of the low-dimensional features obtained by PCA. A surface of valley can be found. Blue dots show the face images sampled from the surface. Some face images are selected and shown. (a) The variation in pitch is shown with the yaw of −90° in a specific view of the surface. (b) The variation in yaw is shown with the pitch of 0° in another view of the surface.

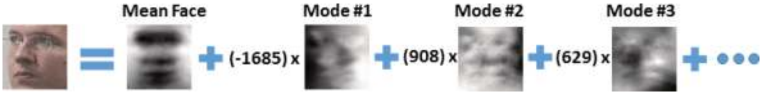

Figure 6. Representation of one face image by the mean face and first three eigenfaces obtained from PCA.

3.2. Locally linear embedding (LLE)

From the observation of the data shown in **Figure 4**, the smooth manifold is globally nonlinear but can be seen as linear from a local neighborhood. On the basis of this observation, the LLE attempts to represent each of the data by a weighted linear combination of a number of neighbors [11]. The weight matrix **W** can be obtained by the following objective function.

$$\min_{\|w_i\|=1} \sum_{i=1}^{M} \left\| \boldsymbol{x}_i - \sum_{j\in N(i)} w_{ij}\boldsymbol{x}_j \right\|^2 \tag{4}$$

where w_i denotes the ith row vector of matrix **W**. Eq. (4) shows that the LLE aims to minimize the total reconstruction error for the data from the corresponding nearest neighbors. Specifically, **W** is a sparse matrix which assigns optimal weights for neighboring data points and zeros for nonneighboring data points. As a result, both the global nonlinearity and local linearity are included in one identical form. A close form of the weights matrix $\mathbf{W} = \{w_{ij}\}$ can be efficiently computed.

$$w_{ij} = \frac{\sum_{n=1}^{K} c_{jn}^{-1}}{\sum_{m=1}^{K} \sum_{n=1}^{K} c_{mn}^{-1}} \tag{5}$$

where K is the number of nearest neighbors tuned for specific problem; $\mathbf{C} = \{c_{mn}\}$ is the neighborhood correlation matrix that is specified for each data: $c_{mn} = \xi_m^T \xi_n$ where ξ_m and ξ_n are the mth and nth neighbors of the ith data point.

Moreover, the weight matrix **W** is locally invariant to linear transformation, i.e., translation, rotation, and scaling. Therefore, it is reasonable to propose that low-dimensional representation of the data can also preserve the local geometry as featured in the original space with the same weight matrix **W**. The next step is then to estimate such low-dimensional (d-dimension) representation **Y** by the following equation.

$$\min_{\mathbf{Y}} \sum_{i=1}^{M} \left\| \boldsymbol{y}_i - \sum_{j \in N(i)} w_{ij} \boldsymbol{y}_j \right\|^2 \tag{6}$$

Eq. (6) can be rewritten as

$$\min_{\mathbf{Y}} \mathbf{Y} \mathbf{M} \mathbf{Y}^{\boldsymbol{T}} \tag{7}$$

where $\mathbf{M} = (\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^T$. Two constraints are made to center the data and avoid degenerate solutions: $\sum_{i=1}^{M} \boldsymbol{y}_i = \mathbf{0}$ and $\frac{1}{M} \sum_{i=1}^{M} \boldsymbol{y}_i \boldsymbol{y}_i^{\boldsymbol{T}} = \mathbf{I}_{d \times d}$. Then **Y** can be obtained, of which the columns correspond to the bottom d eigenvectors of **M**.

The same data set used in the last experiment is processed with LLE and shown with the first three dimensions in **Figure 7**. The variation of the head pose in yaw with different pitch is obviously shown. The transition from a pose to another pose tends to be continuous and easy to locate. The learned manifold is smoother and more discriminative than PCA.

3.3. Isomap

Isomap [10] is an abbreviation of isometric feature mapping [17], which is an extension of the classical algorithm of multidimensional scaling (MDS) [18]. From the previous section, one can learn that the LLE represents the nonlinearity of the original data by preserving the local geometrical linearity. In contrast, the algorithm of Isomap proposed a global solution by constructing a graph for all pairwise data. This idea ensures the global optimum.

Specifically, Isomap firstly constructs a graph that can be represented as $G = (V, E)$ with V as the vertices (the data points) and E as the edges (the adjacent connections). The adjacent vertices will be connected with edges according to particular metrics. Taking the ith and jth data points for instance, an edge will be added between them if $\|x_i - x_j\| \le \epsilon$ (ϵ-neighbor) or x_i is the K-nearest neighbor of x_j and vice versa (K-nearest neighbor). The graph is then assigned with distances among all pairwise vertices according to the edges. The distance between the vertices associated with edges will be $d_{ij} = \|x_i - x_j\|$ (Euclidean distance). For the other pairwise vertices, the geodesic distances are considered, which can be simply computed as the shortest path (Floyd's algorithm). This weighted graph is capable of modeling the isometric distances, which can preserve the global geometry in the learned manifold.

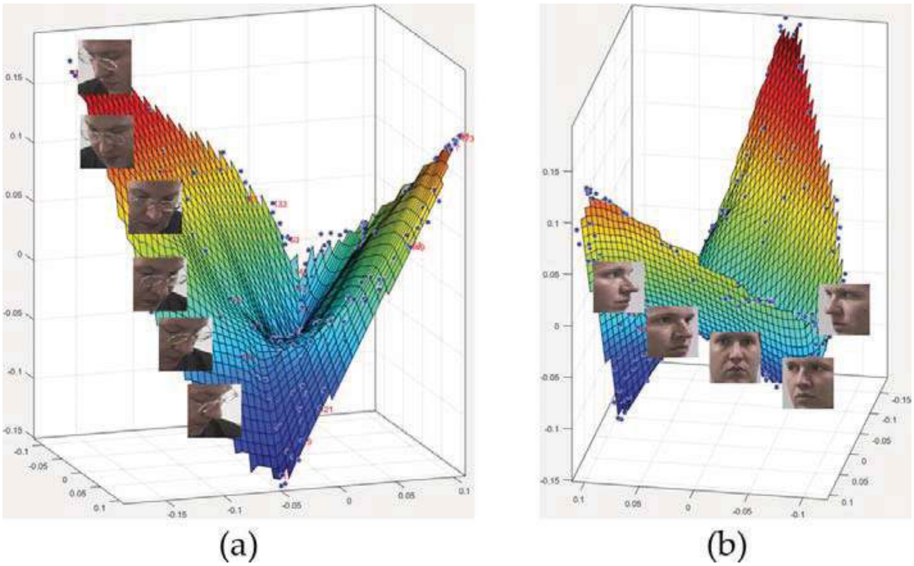

Figure 7. Visualization of the low-dimensional features obtained by LLE. A surface of "wings" is observed. (a) The variation in yaw with the pitch of -30° is found along the edge of one "wing" of the 3D surface. (b) Another variation in yaw with pitch of 0° is found along the ridge of the 3D surface.

From the distance matrix $\mathbf{D} = \{d_{ij}\}$, an objective function is defined as

$$\min_{\mathbf{Y}} \left\| \tau(\mathbf{D_X}) - \tau(\mathbf{D_Y}) \right\|^2 \tag{8}$$

where $\tau(\mathbf{D_X})$ and $\tau(\mathbf{D_Y})$ denote the distances conversion to inner products for the original data and the low-dimensional data, respectively. The operator τ is defined as $\tau(\mathbf{D}) = -\mathbf{HSH}/2$; $\mathbf{S} = \mathbf{D}^2$ is the squared distance matrix; $\mathbf{H} = \mathbf{I} - \frac{1}{M}\mathbf{1}\mathbf{1}^T$ is the centering matrix. This design of the objective function aims to preserve the global structure represented as a graph associated with geodesic distance. The low-dimensional embedding should be featured with similar global geometry with the original data. The optimization of the objective function can be solved by MDS. The d-dimensional representation $\mathbf{Y}$ is obtained by decomposing the matrix of $\tau(\mathbf{D_X})$ and preserving the top d eigenvectors.

Figure 8 shows the results of the low-dimensional head poses obtained from Isomap. An interesting shape of “bowl” of the embedding surface obtained for the head pose images.

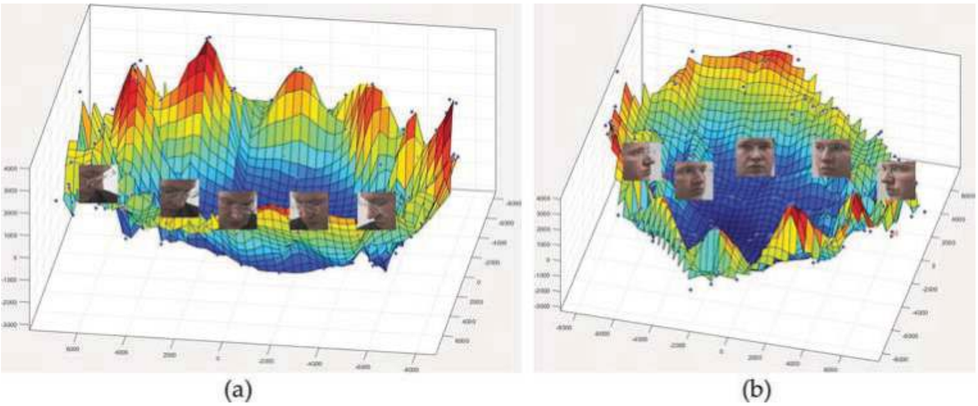

Figure 8. Visualization of the low-dimensional features obtained by Isomap. (a) The variation in yaw with the pitch of -30° is found along the edge of the shape. (b) Another variation in yaw with pitch of 0° is found along the geodesic path in the middle of the shape. The interesting thing is the frontal face locates approximated at the center.

3.4. Laplacian eigenmaps (LE)

Compared to Isomap, the idea of graph representation of the data is also taken by the algorithm of LE. However, the difference is the later attempts to construct a weighted graph (other than distance graph) for the data, which is then represented as a Laplacian [12].

The first step of LE is to construct an adjacent graph whose vertices are the data points and edges are the adjacent connections for neighbors. A pair of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are ϵ-neighbors and will be connected with edge if $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq \varepsilon$. The other criterion to connect or disconnect the pair of points is to find if they are K-nearest neighbors for each other. The second step is to choose appropriate weights for the graph. There are two options: the heat kernel defines the weight as $w_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$ if the two points of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are connected and zero otherwise. The other option is straightforwardly setting w_{ij} =1 for connected edges and zero otherwise.

The third step is to minimize an objective function

$$\min_{y:y\mathbf{A}y^T=1} \sum_{i,j} (\boldsymbol{y}_i - \boldsymbol{y}_j)^2 w_{ij} \tag{9}$$

where the diagonal matrix of $\mathbf{A} = diag\{a_{ii}\}$ is computed by column sums of $\mathbf{W}$: $a_{ii} = \sum_j w_{ji}$. From the definition of the objective function, the goal of LE is to preserve the weights for the mapped data from the original data. If the pair of data is close or apart from each other in the original space, they should be also kept close or apart in the embedding. The weight matrix strongly punishes the “connection” for apart data points. Next, the objective function can be derived to

$$\mathbf{L}y = \lambda \mathbf{A}y \tag{10}$$

where $\mathbf{L} = \mathbf{A} - \mathbf{W}$ is the Laplacian matrix for the weighted graph. Such form is a generalized eigenvector decomposition. And the matrix $\mathbf{Y}$ whose columns are the bottom d eigenvectors decomposed from Eq. (10) is the d-dimensional representation of the data. To be compared, the LE is less sensitive to outlier and noise due to its property of local preservation. The weights for nonedges are set to be zeros, which diminish the problem of short circuiting.

As shown in **Figure 9**, the embedding surface with the shape of parabolais generated by LE, which is similar to the results obtained by LLE. But the latter produces smoother and more symmetric shape of the surface. The variation in yaw from left to right is shown symmetrically, and the frontal face approximately locates on the vertex of bottom.

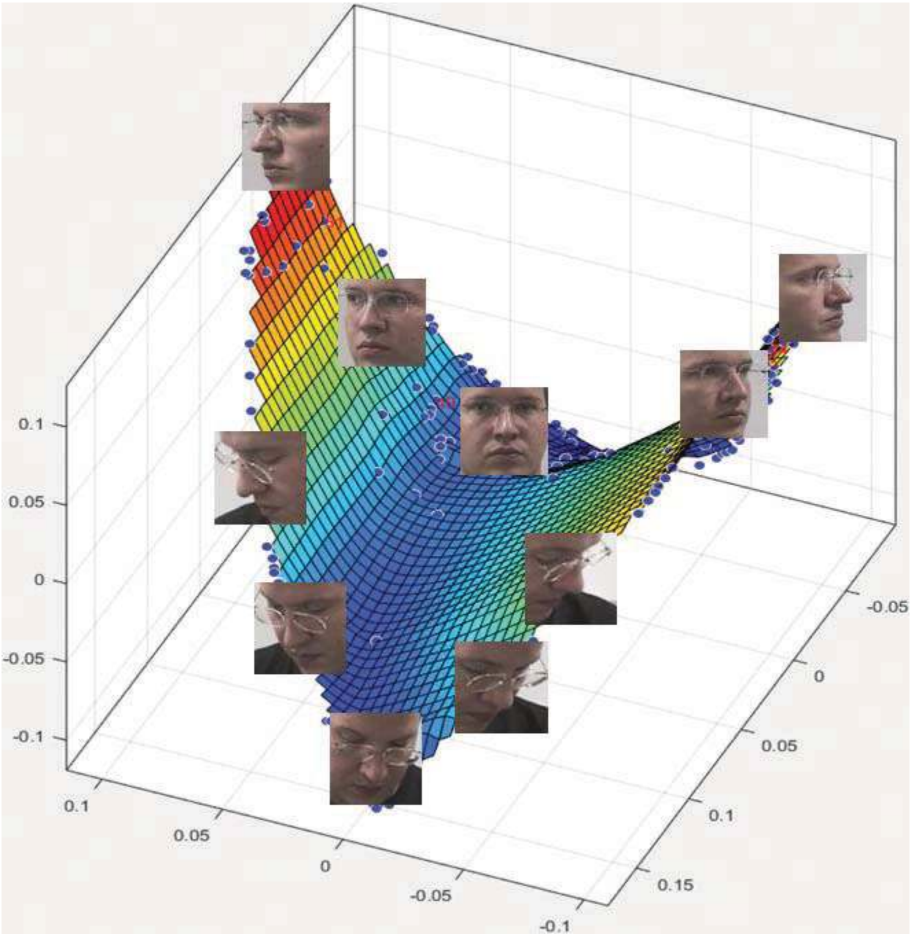

Figure 9. Visualization of the low-dimensional features obtained by LE.

3.5. Laplacian preserving projections (LPP)

The previously introduced algorithms do not clarify how an unseen data is projected to the low-dimensional space. To solve this problem, LPP reformulates the LE by representing the dimensionality reduction as a linear projection from the original to the low-dimensional data. The first two steps of LPP are exactly the same as LE, which construct the adjacent graph and compute the weights for each connection. The most significant difference is the LPP representing the dimensionality reduction from the original to the low-dimensional space as a projection $\boldsymbol{y} = \mathbf{V}^T\boldsymbol{x}$. The problem is converted to the one which aims to find a projection space instead of directly compute the low-dimensional features. The generalized eigenvector decomposition defined in Eq. (10) is then reformulated as follows:

$$\mathbf{XLX}^T\boldsymbol{v} = \lambda\mathbf{XAX}^T\boldsymbol{v} \tag{11}$$

The bottom d eigenvectors decomposed from Eq. (11) construct the projection matrix $\mathbf{V}_{D\times d} = \{\boldsymbol{v}_i\}$. Any data from the original space can be dimensionally reduced through $\boldsymbol{y} = \mathbf{V}^T\boldsymbol{x}$.

More improved nonlinear manifold learning algorithms are developed [13, 19], but in this section, the main idea of how to derive the low-dimensional representation of the head poses is the core. Details of the advanced versions of the manifold learning algorithms can be explored in the original references.

4. Head pose estimation via manifold learning

The manifold learning methods can successfully model the head pose variations in both yaw and pitch as discussed in the previous sections. However, there are still several difficulties to state. The introduction of noise, for example, identity, and illumination variations will affect the performance of those methods on the head pose estimation. Another point is that they do not infer how the low-dimensional representation of an unseen head pose image is obtained (except LPP) and how the pose is estimated. In this section, more sophisticated methods are introduced to solve these problems based on the original or extended manifold learning algorithms.

4.1. PCA-based head pose estimation

In Ref. [20], the PCA has been turned to be robust to invariance of identity. Another important conclusion is that the angle of 10° is found to be the lower bound to be discriminative. For the data set constructed following this finding, the PCA would produce promising results for head pose estimation.

A kernel machine-based method is proposed using the kernel PCA (KPCA) and kernel support vector classifier (KSVC) [21]. The KPCA is an extension of the classical PCA. Let $\phi(\boldsymbol{x}) : R^D \rightarrow R^{D'}$ be a kernel that maps the original dimensional into a higher dimensional, which makes the nonlinearly separable data linearly separable in the higher dimensional space. Correspondingly, the covariance matrix $\mathbf{C}$ is replaced by $\overline{\mathbf{C}} = \Phi\Phi^T$, of which the bold Φ is the kernel represented data points set. The projection matrix $\overline{\mathbf{V}}$ can be similarly obtained through eigenvector decomposition of matrix $\overline{\mathbf{C}}$. After the feature dimensionality reduction, a multiclass KSVC is trained which can estimate the view of head. Given a testing image $\boldsymbol{x}_{ts}$, it is first mapped by the kernel and then the low-dimensional features can be obtained by the projection matrix learned from KPCA $\boldsymbol{y}_{ts} = \overline{\mathbf{V}}^T \phi(\boldsymbol{x}_{ts})$ which will be fed into the KSVC to predict the head pose estimation. This method is proved to be outperformed its linear counterpart, i.e., PCA + SVC.

4.2. View representation by Isomap

The derivation of how the Isomap reduces the dimensionality of the original data to a low-dimension has been introduced in the previous section. Now the problem is how to connect the head pose to the features. A pose parameter map $\mathbf{F}$ is proposed in Ref. [22] to build such connections.

$$\boldsymbol{\Theta} = \mathbf{FY} \tag{12}$$

where $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_M)$ denotes the angles of the head poses from the training data and $\mathbf{Y}$ is the low-dimensional representation of the data obtained from Isomap. Actually, the matrix of $\mathbf{F}$ can be seen as a set of linear transformations that map the features to corresponding pose angles. During training time, the head poses $\boldsymbol{\Theta}$ are given as annotations, and the low-dimensional features $\mathbf{Y}$ can be learned by manifold learning, then, $\mathbf{F}$ can be obtained using the singular value decomposition (SVD) of $\mathbf{Y}^T$.

$$\mathbf{F}^T = \mathbf{P_Y W_Y^{-1} U_Y^T} \boldsymbol{\Theta}^T \tag{13}$$

where $\mathbf{P_Y}$, $\mathbf{W_Y}$, and $\mathbf{U_Y}$ are the SVD of $\mathbf{Y}^T$.

Given a testing image $\boldsymbol{x}_{ts}$, the goal now is to obtain the low-dimensional feature according to the embedding $\mathbf{Y}$. The first step is to construct a geodesic distance vector for the testing image to all the training images $\boldsymbol{d}_{ts,M} = (d^2_{ts,1}, d^2_{ts,2}, \ldots, d^2_{ts,M},)^T$. Then, $\boldsymbol{d}_{ts} = diag(\mathbf{Y}^T\mathbf{Y}) - \boldsymbol{d}_{ts,M}$. Next, the low-dimensional representation of the testing image is obtained by

$$\boldsymbol{y}_{ts} = \frac{1}{2}\left((\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{Y}^T\right)^T \boldsymbol{d}_{ts} \tag{14}$$

Finally, the estimated pose of the testing image is computed from

$$\boldsymbol{\theta}_{ts} = \mathbf{F}\boldsymbol{y}_{ts} \tag{15}$$

The insight of this method focuses on the conversion from testing data to the subspace learned by nonlinear manifold learning. The algorithms of LLE and LE can also be generalized by the proposed idea.

4.3. The biased manifold embedding (BME)

The head pose estimation is subjected to the identity variation. The ideal case is to eliminate such negative effects, which means the face images with close pose angles should maintain nearer and the ones with quite different poses should stay farther in the low-dimensional manifold, even the poses are from the same identity. Based on this statement, the BME is proposed to modify the distance matrix according to the pose angles, which can be extended with almost all the classical algorithms [23].

The modified distance between a pair of data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is given by:

$$\tilde{d_{ij}} = \begin{cases} \dfrac{\beta * p(i,j)}{\max_{m,n}\left(p(m,n)\right) - p(i,j)} * d_{ij}, p(i,j) \neq 0 \\ 0, \qquad\qquad\qquad\qquad\qquad\quad p(i,j) = 0 \end{cases} \tag{16}$$

where $p(i,j) = |p_i - p_j|$ is simply defined as the absolute difference of the angles of two poses. From the modified distance matrix, one can find that the distance between images with close poses is biased to be proportionally small. The images with the same poses are defined to be zero-distance.

In fact, the BME can be seen as a naÏve version of the supervised manifold learning. The head pose information is used as the supervision to enhance the construction of the graph. For the head pose estimation stage, the generalized regression neural network (GRNN) [24] is applied to learn the nonlinear mapping for the unseen data points, and linear multivariate regression is applied to estimate the head pose angle. This idea can be easily extended to the classical algorithms, e.g., Isomap, LLE, and LE, among which the biased LE achieves the lowest error rate on the data set of FacePix [25].

4.4. Head pose estimation as frontal view search

The two remarkable head poses, i.e., yaw and pitch, cause the problem self-occlusion. Compared with pitch, the yaw makes the problem more serious. An extended manifold learning (EML) method is proposed to specify the head pose estimation only considering the variation in the yaw [15]. This work resorts to the frontal view search instead of directly estimating the head pose, which is more efficient and robust. The idea is based on the observation that the frontal face locates nearly at the vertex in the symmetrical shape of the embedding. However, if the pose distribution of the data is asymmetric, the location of the frontal face in the manifold will shift from the vertex. Therefore, the first trial of the EML method is data enhancement. All the images are horizontally flipped and both the original and flipped images are used for manifold learning. In order to make the method more robust to variations in environment, for example, illumination, the localized edge orientation histogram (LEOH) is presented to represent the original color mappings as more representative features. The idea is inspired by the classical HoG feature [14]. The first step of LEOH is to apply a Canny edge detector on the original images. Then, the whole image is divided into $M \times N$ cells. The gradient orientation is quantized into N_B bins. Next, histograms of the gradient orientation of the cells locating in a block consisted of $P \times Q$ cells are accumulated and normalized. Finally, the LEOH feature is obtained by the block features concatenation. The proposed ideas can be easily incorporated in various manifold learning methods that improve the performance of the frontal view searching.

4.5. Head pose estimation by supervised manifold learning

A taxonomy of methods, which structures the general framework of manifold learning into several stages, is proposed to incorporate the head pose angles in one or some of the stages to enable the supervised manifold learning [26]. A straightforward solution could be the adaption of the distance and weight matrix according to the angle difference between pairwise face images. The head pose estimation problem is then interrupted as a regression problem, which was usually solved as a classification problem. As a result, continuous head poses can be generalized by the model.

The general framework of manifold learning can be represented as follows: Stage 1, neighborhood searching; Stage 2, graph weighting; Stage 3, low-dimensional manifold computation; and Stage 4, projection from unseen data to the manifold and pose estimation.

In Stage 1, the distance matrix of $\mathbf{D} = \{d_{ij}\}$ can be adapted as follows:

$$\tilde{d_{ij}} = f(|\theta_i - \theta_j|) \cdot d_{ij} \tag{17}$$

where θ_i and θ_j are the angles of two poses, which keep the same denotation as previous sections. The f is some reciprocal increasing positive function, for example, $f(u) = \alpha \cdot u/(\beta - u)$. The introduction of f encourages the distance decreasing of the nearer poses and increasing of farther poses. The farther the poses are, the more penalties the distance will gain.

In Stage 2, the weight matrix of $\mathbf{W} = \{w_{ij}\}$ can be adapted by similar idea of supervision information incorporation.

$$\tilde{w_{ij}} = w_{ij} \cdot g(|\theta_i - \theta_j|) \tag{18}$$

where g is defined as some positive decreasing function, which is similar to the f applied in Stage 1.

In Stage 3, let us take the LLE for an instance. The original objective function of LLE shown in Eq. (6) can be adapted as follows:

$$\min_{\mathbf{Y}} \sum_{i=1}^{M} \left\| \boldsymbol{y}_i - \sum_{j \in N(i)} w_{ij} \boldsymbol{y}_j \right\|^2 + \lambda \frac{1}{2} \sum_{i,j} (\boldsymbol{y}_i - \boldsymbol{y}_j)^2 \Lambda_{ij} \tag{19}$$

where the $\mathbf{\Lambda} = \{\Lambda_{i,j}\}$ measures the similarity between the angles of pairwise poses. A possible form of Λ is the heat kernel.

$$\Lambda_{ij} = \begin{cases} e^{-\frac{\left\|\theta_i - \theta_j\right\|^2}{2\sigma^2}}, & \text{if the } i^{th} \text{ and } j^{th} \text{data points are neighbours} \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

The adaption of the objective function can preserve the local linearity of the original data and enhance the similarity for neighborhoods, which are facilitated with similar poses. This is implemented by the second term of Eq. (19) that introduces the supervision information. Following the derivation from Eq. (6) to Eq. (7), Eq. (19) can be simplified as:

$$\min_{\mathbf{Y}} \mathbf{Y}\mathbf{M}\mathbf{Y}^{\boldsymbol{T}} + \lambda\mathbf{Y}\tilde{\mathbf{L}}\mathbf{Y}^{\boldsymbol{T}} = \min_{\mathbf{Y}} \mathbf{Y}(\mathbf{M} + \lambda\tilde{\mathbf{L}})\mathbf{Y}^{\boldsymbol{T}} \tag{21}$$

where $\tilde{\mathbf{L}}$ is the Laplacian matrix of $\boldsymbol{\Lambda}$. For the low-dimensional embedding, eigenvectors decomposition of $\mathbf{M}$ + $\lambda\mathbf{L}$ can be performed. By the supervision information incorporation, the method is much capable of imposing discriminative projection to the learned embedding.

In Stage 4, the GRNN algorithm is applied to produce the mapping from unseen data to the low-dimensional embedding. During testing time, the support vector regression (SVR) with RBF kernel and smoothing cubic splines are taken.

A novel method of supervised manifold learning for head pose estimation [27, 28] is proposed based on the framework from the former method. Similarly, angles of poses are incorporated in all three stages of the general manifold learning structure.

In Stage 1, an improved version of *f* is proposed as:

$$\tilde{d_{ij}} = f(|\theta_i - \theta_j|)^p \cdot d_{ij}(p > 0) \tag{22}$$

where f is defined as a rectified reciprocal form $f(|\theta_i - \theta_j|) = \alpha \frac{|\theta_i - \theta_j|}{\max_{m,n}\{|\theta_m - \theta_n|\} - |\theta_i - \theta_j| + \varepsilon}$. α is a positive constant and ε is an arbitrary small positive constant that avoids the denominator of *f* being zero. This adaption for the distance matrix further enhances the effects of the supervision information during the procedure of neighbors search.

In Stage 2, taking LLE (NPE [29]), for an example, the local distance matrix shown in Eq. (4) is modified as

$$\tilde{c_{mn}} = g_{mn} \cdot c_{mn} \tag{23}$$

where $g_{mn} = \frac{|\theta_i - \theta_m||\theta_i - \theta_n|}{(\max_{m,n}\{|\theta_m - \theta_n|\} - |\theta_m - \theta_n| + \varepsilon)^2}$. θ_i is the angle of the reference face image $\boldsymbol{x}_i$.This operation enhances the supervision during the computation of local correlated matrix.

In Stage 3, a supervised neighborhood-based fisher discriminant analysis (SNFDA) is proposed. The basic idea is to make the neighboring data points as close as possible and the nonneighboring data points as far as possible in the low-dimensional embedding. The SNFDA can be seen as a postprocessing procedure in this stage. Based on the low-dimensional represented data **Y** obtained from the original LLE or the modified LLE in Stages 1 and 2, the within- and between-neighborhood scatter matrices are defined as:

$$\mathbf{S}_w = \frac{K}{2}\sum_{i,j=1}^{M} A_{ij}^{w}(\boldsymbol{y}_i - \boldsymbol{y}_j)(\boldsymbol{y}_i - \boldsymbol{y}_j)^T \tag{24}$$

$$\mathbf{S}_B = \frac{K}{2}\sum_{i,j=1}^{M} A_{ij}^{B}(\boldsymbol{y}_i - \boldsymbol{y}_j)(\boldsymbol{y}_i - \boldsymbol{y}_j)^T \tag{25}$$

where

$$A_{ij}^{w} = \begin{cases} \dfrac{A_{ij}}{K}, & \boldsymbol{y}_j \in N(i) \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

$$A_{ij}^{w} = \begin{cases} A_{ij}\left(\dfrac{1}{M} - \dfrac{1}{K}\right), & \boldsymbol{y}_j \in N(i) \\ \dfrac{A_{ij}}{n}, & \text{otherwise} \end{cases} \tag{27}$$

A_{ij} is the affinity between $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$, which is defined as the form of heat function:

$$A_{ij} = e^{-\frac{\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2}{2\sigma^2}} \tag{28}$$

Details about the inference of the scatter matrices can be found in the original reference. The transformed matrix $\mathbf{T}_{\mathrm{SNFDA}}$ of SNFDA is computed from the generalized eigenvector decomposition problem

$$\mathbf{S}_B\boldsymbol{e} = \lambda\mathbf{S}_w\boldsymbol{e} \tag{29}$$

The top d eigenvectors span to the $\mathbf{T}_{\mathrm{SNFDA}}$ and the transformed feature is obtained by $z = \mathbf{T}_{\mathrm{SNFDA}}\mathbf{y}$. This supervised learning manner successfully introduces the supervision information in a framework to provide a "good" projection from the original data to the low-dimensional. Due to the supervised learning, when the projection is applied on original data, more discriminative features can be obtained for head pose estimation.

In Stage 4, during testing time, the GRNN is applied to map the unseen data point to the low-dimensional embedding and the relevance vector machine (RVM) [30] is adopted to accomplish the pose estimation. Experimental results obtained by the proposed method performing on the database of FacePix [25] and MIT-CBCL [31] show big improvements compared with other state-of-the-art algorithms [23, 26] in Stage 3 and Stages 1 + 2 + 3. This means that this method is more robust for identity and illumination variations.

5. Summary

In this chapter, the head pose estimation, one of the most challenging tasks in the area of computer vision, is introduced, and the main types of methods are demonstrated and compared. Particularly, the manifold learning-based methods are attracted more attention. In reality, data distribution is usually nonlinear in high-dimensional represented space, e.g., the head pose images. Some potential structures are lying on nonlinear but smooth manifolds which are embedded in the original space. The manifold learning algorithms are able to discover and visualize such embedding. Almost all the algorithms are formalized based on the assumption of the local linearity of the nonlinear data. Those algorithms highly benefit the application of head pose estimation, because the face orientations (yaw and pitch) are found to be distributed along some specific manifolds. Promising performance is achieved by the classical manifold learning methods, which, however, are highly improved by the supervised manifold learning. It proves that the supervised information represented as angles of head poses is helpful in head pose estimation. However, there are still hurdles to take. Most of the methods are tested in different settings, e.g., different database is used in different method. A common framework could help to offer fair justifications. Other feature instead of the simple color space can be considered to better represent the face images. Some useful tools are available online to help better understand the work [16, 32, 33].

Author details

Chao Wang[1,†*], Yuanhao Guo[2,†] and Xubo Song[3]

*Address all correspondence to: chaocharleswang@gmail.com

1 Online Search Team, The Home Depot, Atlanta, GA, USA

2 Imaging & BioInformatics, Leiden Institute of Advanced Computer Science, Leiden University, Leiden, The Netherlands

3 Department of Biomedical Engineering, Oregon Health & Science University, Portland, OR, USA

† These authors contributed equally to this work

References

[1] Murphy-Chutorian, Erik and Trivedi, Mohan Manubhai. Head pose estimation in computer vision: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2009;**31**(4):607–626.

[2] Gourier, Nicolas, Hall, Daniela and Crowley, James L. Estimating face orientation from robust detection of salient facial structures. In: 2004 ICPR Workshop on Visual Observation of Deictic Gestures. Cambridge, UK: IEEE; 2004.

[3] Taigman, Yaniv, Yang, Ming, Ranzato, Marc'Aurelio and Wolf, Lior. Deepface: Closing the gap to human-level performance in face verification. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA: IEEE; 2014. pp. 1701–1708.

[4] Beymer, David James. Face recognition under varying pose. In: 1994 IEEE Conference on Computer Vision and Pattern Recognition. Seattle, WA, USA: IEEE; 1994. pp. 756–761.

[5] Niyogi, Sourabh and Freeman, William T. Example-based head tracking. In: 2nd IEEE International Conference on Automatic Face and Gesture Recognition. Killington, VT, USA: IEEE; 1996. pp. 374–378.

[6] Viola, Paul and Jones, Michael. Rapid object detection using a boosted cascade of simple features. In: 2001 IEEE Conference on Computer Vision and Pattern Recognition. Kauai, HI, USA: IEEE; 2001. pp. I-511–I-518.

[7] Felzenszwalb, Pedro F and Huttenlocher, Daniel P. Pictorial structures for object recognition. International Journal of Computer Vision. 2005;**61**(1):55–79.

[8] Cootes, Timothy F, Edwards, Gareth J and Taylor, Christopher J. Active appearance models. In: 1998 European Conference on Computer Vision. Freiburg, Germany: Springer; 1998. pp. 484–498.

[9] Matthews, Iain and Baker, Simon. Active appearance models revisited. International Journal of Computer Vision. 2004;**60**(2):135–164.

[10] Tenenbaum, Joshua B, De Silva, Vin and Langford, John C. A global geometric framework for nonlinear dimensionality reduction. Science. 2000;**290**(5500):2319–2323.

[11] Roweis, Sam T and Saul, Lawrence K. Nonlinear dimensionality reduction by locally linear embedding. Science. 2000;**290**(5500):2323–2326.

[12] Belkin, Mikhail and Niyogi, Partha. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation. 2003;**15**(6):1373–1396.

[13] Maaten, Laurens van der and Hinton, Geoffrey. Visualizing data using t-SNE. Journal of Machine Learning Research. 2008;**9**:2579–2605.

[14] Dalal, Navneet and Triggs, Bill. Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition; San Diego, CA, USA: IEEE; 2005. pp. 886–893.

[15] Wang, Chao and Song, Xubo. Robust frontal view search using extended manifold learning. Journal of Visual Communication and Image Representation. 2013;**24**(7):1147–1154.

[16] Available from: http://isomap.stanford.edu/

[17] Balasubramanian, Mukund and Schwartz, Eric L. The isomap algorithm and topological stability. Science. 2002;295(5552):7–7.

[18] Cox, Trevor F and Cox, Michael AA. Multidimensional Scaling. USA: CRC Press; 2000.

[19] Weinberger, Kilian Q and Saul, Lawrence K. Unsupervised learning of image manifolds by semidefinite programming. International Journal of Computer Vision. 2006;**70**(1):77–90.

[20] Sherrah, Jamie, Gong, Shaogang and Ong, Eng-Jon. Face distributions in similarity space under varying head pose. Image and Vision Computing. 2001;**19**(12):807–819.

[21] Li, Stan Z, Fu, Qingdong, Gu, Lie, Scholkopf, B, Cheng, Yimin and Zhang, Hongjiag. Kernel machine based learning for multi-view face detection and pose estimation. In: 2001 IEEE International Conference on Computer Vision. Vancouver, Canada: IEEE; 2001.

[22] Raytchev, Bisser, Yoda, Ikushi and Sakaue, Katsuhiko. Head pose estimation by nonlinear manifold learning. In: The 17th International Conference on Pattern Recognition; Cambridge, UK: IEEE; 2004. pp. 462–466.

[23] Balasubramanian, Vineeth Nallure, Ye, Jieping and Panchanathan, Sethuraman. Biased manifold embedding: A framework for person-independent head pose estimation. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. Minneapolis, MN, USA: IEEE; 2007. pp. 1–7.

[24] Specht, Donald F. A general regression neural network. IEEE Transactions on Neural Network. 1991;**2**(6):568–576.

[25] Little, Danny, Krishna, Sreekar, John Jr, A and Panchanathan, Sethuraman. A methodology for evaluating robustness of face recognition algorithms with respect to variations in pose angle and illumination angle. In: ICASSP (2); Citeseer; 2005. pp. 89–92.

[26] BenAbdelkader, Chiraz. Robust head pose estimation using supervised manifold learning. In: European Conference on Computer Vision; Crete, Greece: Springer; 2010. p. 518–531.

[27] Wang, Chao and Song, Xubo. Robust head pose estimation via supervised manifold learning. Neural Networks. 2014;**53**:15–25.

[28] Wang, Chao and Song, Xubo. Robust head pose estimation using supervised manifold projection. In: 2012 19th IEEE International Conference on Image Processing. Orlando, FL, USA: IEEE; 2012. pp. 161–164.

[29] He, Xiaofei, Cai, Deng, Yan, Shuicheng and Zhang, Hong-Jiang. Neighbourhood preserving embedding. In: 2005 IEEE International Conference on Computer Vision. Beijing, China: IEEE; 2005. pp. 1208–1213.

[30] Tipping, Michael E. Sparse Bayesian learning and the relevance vector machine. Journal of Machine Learning Research. 2001;**12**:11–244.

[31] Huang, Jennifer, Heisele, Bernd and Blanz, Volker. Component-based face recognition with 3D morphable models. In: International Conference on Audio-and Video-Based Biometric Person Authentication. Guildford, UK: Springer; 2003. pp. 27–34.

[32] Available from: https://www.cs.nyu.edu/~roweis/lle/code.html

[33] Available from: https://lvdmaaten.github.io/drtoolbox/