Towards fitting a 3D dense facial model to a 2D image: A landmark-free approach

Yuhang Wu, Xiang Xu, Shishir K. Shah, Ioannis A. Kakadiaris
Computational Biomedicine Lab
Department of Computer Science, University of Houston, Houston, TX, USA
{ywu36,xxu21,sshah,ioannisk}@uh.edu

Abstract

Head pose estimation helps to align a 3D face model to a 2D image, which is critical to research requiring dense 2D-to-2D or 3D-to-2D correspondence. Traditional pose estimation relies strongly on the accuracy of landmarks, so it is sensitive to missing or incorrect landmarks. In this paper, we propose a landmark-free approach to estimate the pose projection matrix. The method can be used to estimate this matrix in unconstrained scenarios and we demonstrate its effectiveness through multiple head pose estimation experiments.

1. Introduction

Recently, head pose estimation, which can be formulated as a 3D-to-2D model fitting problem, has drawn much attention. It is a fundamental and essential module in most systems that require knowledge of pose of the human face. Popular application domains include 3D face reconstruction from a single 2D image [7, 13, 18], and 3D-aided face recognition [2, 15, 31, 14, 17].

Depending on their output, pose estimation systems can be divided into two categories. In the first, the system returns the head orientation parameters with respect to the image plane [9, 19, 28, 29, 16, 3, 11]. The methods in this case exploit machine learning approaches to learn a mapping between a 2D image and pose parameters directly so that the only input is the region of interest (ROI) of the face. However, these landmark-free pose estimation approaches have two limitations. First, they are constrained by the available poses in the quantized space, and they are difficult to generalize to new poses unavailable during training. Second, they cannot estimate an accurate projection matrix that aligns a 3D model on a 2D image. In the second category of approaches, the system finds a decomposable projection matrix that can be used to project a 3D head model to a given 2D image [2, 8, 13, 18]. Since these methods exploit the classical 3D-to-2D projective geometry [12], they require several landmarks (always more than four) to be accurately annotated at identical positions on both 3D and 2D models. The merit of these approaches is that every 3D vertex will correspond to a location on the 2D face through the 3D-to-2D projection matrix, which provides an accurate alignment process for recovering point-wise 3D information from the 2D image space [7, 13, 18], as well as a solution for aligning two facial images depicting different poses into the same canonical space. This class of approaches has been widely used in 3D-aided face recognition and has been shown to achieve very competitive performance [15, 31].

There is much research in both categories of approaches, but little research has been performed at their intersection. Consider the task of estimating the head pose in an image captured by a low-resolution surveillance camera. Assuming that (i) automatic landmark annotation approaches all fail to report the precise fiducial points on the face, (ii) a personalized 3D model for texture lifting is not available, and (iii) an estimated face ROI is available, *how can the pose be estimated accurately?* In addressing this problem, we propose a method that provides accurate estimates of the yaw and pitch angle as well as a 3D-to-2D projection matrix without landmark annotation and discretized pose constraints.

Our method takes advantage of the robustness of sparse coding, the generality of boosted regression and 3D model fitting, and the precision of template matching. The contributions of our paper are: (i) a method to estimate a 3D-to-2D projection matrix that does not require annotated landmarks, and (ii) a method that estimates poses not available in the training set.

The rest of the paper is organized as follows: Section 2 reviews the literature in pose estimation. Our method is presented in Section 3. Experiments and results are presented in Section 4. Section 5 discusses the results and the conclusions are offered in Section 6.

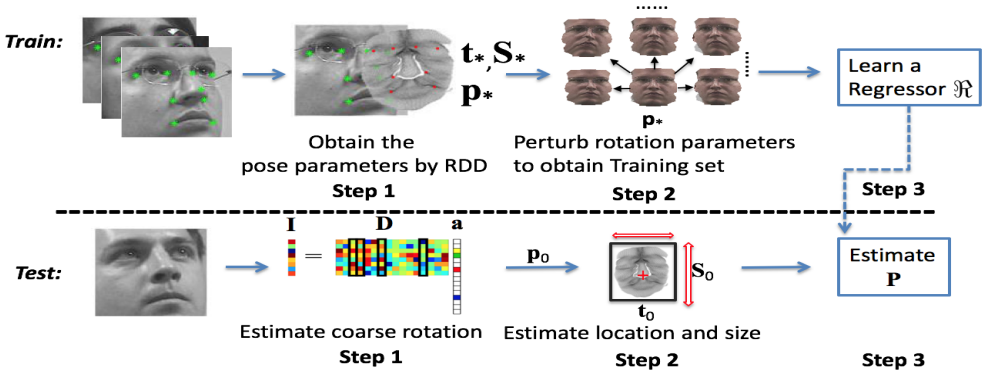


Figure 1. The overview of our method.

2. Related Work

The classical pose estimation methods [8, 12] which have been adopted widely in 3D related research [2, 7, 13, 14, 18, 17] rely on landmark annotations. Even though the pose is accurately estimated when the landmarks are annotated accurately, these methods do not work well when the image resolution is low or multiple landmarks are missing. Another drawback of these methods is that the estimated projection matrix can be decomposed into infinite combinations of rotation and translation matrices, which may result in an unconstrained head orientation space. Although the rotation matrix is numerically correct when working jointly with a 3D translation vector, the corresponding pitch and yaw angles may not be consistent with the facial orientation appearing in the query image. This intrinsically creates a gap between the classical pose estimation and appearance-based pose estimation methods. In this work, we build a bridge between the two domains through rotation-determined-decomposition (RDD). In training, with the help of annotated images and RDD, we compute the intrinsic rotation matrix that corresponds to the actual pitch and yaw angles observed in an image. In testing, without the help of annotations, we estimate the pitch and yaw angle of the face first, and then reconstruct the full projection matrix iteratively.

The landmark-free (appearance-based) methods can generally be divided into two categories: classification-based methods [9, 19, 28, 29] and regression-based methods [11, 4]. Classification-based approaches are robust to small appearance noise but are constrained by the discretization of training space. Thus, they are not suitable for accurate 3D model fitting. In addition, the local manifolds between different pitch and yaw angles have large variances, which implies that an accurate parameter estimation method needs to learn multiple subspaces to adapt to the subspace changes [16]. This, however, undermines the robustness of the method since the selection of local model and corresponding pose estimation is always an ill-posed problem. Geng and Xia [9] proposed a soft-labeling approach to approximate the pose label by neighborhood voting, which attained state-of-the-art performance. However, this approach still suffers from the constraint of training grids. The regression methods [11, 4] are able to eliminate the limitation of training poses and report on angles lying between the grid samples. However, traditional regressors may not work well to solve the extreme non-linear N-to-1 dimension regression problem [16]. Recently, boosted regression [5, 30] approaches have been demonstrated to work well to solve the non-linear facial landmark regression problem. In this paper, we adopted the method proposed by Xiong and Torre [30] and extend it by adding a metric term. Instead of extracting the features in the original image, the feature map is computed on the texture of annotated facial model (T-AFM) [17], which is a pose-normalized space. The unique property of this space helps us to learn a series of general regression steps instead of pose-specific local manifolds [16] in parameter regression, thus circumventing the chicken-and-egg dilemma while at the same time increasing the robustness of pose estimation.

3. Method

The key to solving the landmark-free 3D-to-2D model fitting problem in the proposed approach is to learn a method to estimate the orientation and position of a 3D model so that it is registered (with minimum error) to the 2D image. To accomplish this objective, in training, we learn a regressor $\Re$ to move the inaccurately located 3D model to an approximately correct position based on the values of T-AFM. In testing, given a face ROI, we first estimate a raw position of the 3D model with our 2D sparse coding (2dSC) approach for initializing $\Re$, and then move the 3D model to the correct position by $\Re$. The output of our method is an estimated 3D-to-2D projection matrix $\mathbf{P}$. The overview of our method is depicted in Fig. 1.

3.1. Training

First, we propose to train a regressor that is able to compute the 3D model's location based on the values of T-AFM. In the following, we illustrate why we need the T-AFM, and describe our method for learning the regressor.

Texture of Annotated Face Model (T-AFM), which is a pose-normalized space (Fig. 3), is defined by Toderici *et al*. [25]. It is acquired in two steps. First, we align a 3D AFM to a 2D image, so that the fiducial points of the 3D model are projected at the corresponding position in the 2D image. Then, we lift the texture from the original image to the T-AFM based on the vertices of the aligned 3D model. Any texture lying on the same mesh of the aligned 3D model will be mapped to the same position on the T-AFM. With the geometric constraints of the 3D mesh, even though the faces may have different poses in the original images, in T-AFM the pose factor is eliminated and they all face the viewer. Recently proposed methods [2, 14] are intrinsically similar to the T-AFM, as both employ a pose-normalized space for verification/identification.

Specifically, to align the AFM to a 2D image, we first need a 3D-to-2D projection matrix $\mathbf{P}$. Traditionally, this is acquired by solving a least square minimization problem with the help of 2D and 3D landmarks. After we obtain, $\mathbf{P}$, we map the pixels of the original image into the T-AFM according to the 3D mesh in $\mathbb{M}$ through a texture lifting process g (see [17] for more details). Let $\mathbf{\Gamma}^\mu$ denote the texture of a T-AFM of image μ, then:

$$\mathbf{\Gamma}^\mu = g(\mathbf{I}^\mu, \mathbf{P}^\mu, \mathbb{M}^\mu), \tag{1}$$

where $\mathbb{M}^\mu$ is an AFM, and $\mathbf{I}^\mu$ is the face ROI for image. An accurate $\mathbf{\Gamma}^\mu$ can be obtained if we have both a personalized AFM $\mathbb{M}^\mu$, and a projection matrix $\mathbf{P}_*^\mu$ estimated from manually annotated 2D and 3D landmarks (all notations with a star are estimated from 2D and 3D landmarks). Unfortunately, in real applications, we usually do not have access to a personalized 3D model $\mathbb{M}^\mu$. An approximate $\mathbf{\Gamma}^\mu$ can be obtained by aligning a generic 3D AFM $\bar{\mathbb{M}}$ to the 2D image if we have the 2D landmark positions. However, in both circumstances, the accuracy of $\mathbf{\Gamma}^\mu$ relies on the accuracy of $\mathbf{P}_*^\mu$, which is used to project the 3D model to the 2D image. Since the 2D landmarks are occasionally not available, in this paper we propose a method to leverage the current observation of $\mathbf{\Gamma}^\mu$ to update $\mathbf{P}^\mu$ and formulate the problem as a non-linear regression problem.

Due to the unique pose-invariant property of T-AFM, in ideal circumstances, given an $\mathbb{M}^\mu$, the distortion of T-AFM is only dependent on $\Delta\mathbf{P}^\mu$, the difference of $\mathbf{P}^\mu$ from the correct $\mathbf{P}_*^\mu$ rather than the current $\mathbf{P}$ itself. By learning a mapping through T-AFM to $\Delta\mathbf{P}^\mu$, we bypass learning multiple sensitive manifolds to adapt the local change of $\mathbf{P}^\mu$ in training, and learn a *general* way to regress $\mathbf{P}$ as well. There are additionally two benefits to exploiting T-AFM. First, we leverage more information lying on the non-fiducial areas of the face. Second, we implicitly reduce the inter-class distance between the representations of different poses in T-AFM, which helps the regressor to converge more compactly. Next, we illustrate Alg. 1 and Alg. 2, which we use to learn the regressor: $\Re$.

Algorithm 1 Perturb the parameters for learning $\Re$

Input: $\mathbf{Y}^\mu$, $\mathbf{X}^\mu$ $(\mu = 1 : Z)$
Output: $\mathbf{t}_*^\mu$, $\mathbf{S}_*^\mu$, $\mathbf{q}_0^{\mu,\nu}$
Step 1: Obtain the parameters: $\mathbf{q}_0^{\mu,\nu}$, $\mathbf{t}_*^\mu$, $\mathbf{S}_*^\mu$ by RDD
Step 2: Perturb $\mathbf{q}_0^\mu$ to obtain $\mathbf{q}_0^{\mu,\nu}$ $(\nu = 1 : N^2)$

Step 1[1]**:** To train a regressor from $\mathbf{\Gamma}$ to $\Delta\mathbf{P}$ given the $\mathbf{P}_*$, we first generate T-AFM with different $\Delta\mathbf{P}$ to simulate the potential *inaccurate* 3D registration by perturbing $\mathbf{P}_*$. Since $\mathbf{P}$ is a matrix, we decompose it into pose parameters and perturb them individually.

Let $\mathbf{Y}$ denote an n-by-2 matrix representing the array of n 2D landmarks in an original image, $i \in \{1, \ldots, n\}$, $\mathbf{X}$ denotes an n-by-3 matrix representing the corresponding n 3D points on AFM. In the classical pose estimation framework, we will decompose the 3×4 projection matrix $\mathbf{P}$ into the following pose parameters: an upper-triangular intrinsic matrix $\mathbf{K}$, an orthogonal 3×3 rotation matrix $\mathbf{R}$, and a 3×1 translation vector $\mathbf{e}$ as Eq. 3 represents:

$$\mathbf{Y} = \mathbf{PX}; \tag{2}$$

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{e}]. \tag{3}$$

However, in this decomposition, since the $\mathbf{K}$ is unknown, the results of numerical decomposition of $\mathbf{P}$ (e.g., QR-decomposition) are not unique; if we modify any column of $\mathbf{K}$ and the corresponding column in the $[\mathbf{R}|\mathbf{e}]$, the $\mathbf{P}$ remains the same. If we hope to train a stable model, we must have a stable decomposition of $\mathbf{P}$. There exists a way to rotate an AFM to a pose where it shares the same look-up direction with the provided 2D image. We call this rotation matrix the intrinsic rotation matrix ($\mathbf{O}$), and the way to obtain $\mathbf{O}_*$ is through RDD. This idea has been used for frontal face image generation [2, 18], and in our work we generalize this method and use it to obtain the pose parameters for perturbation.

The RDD can be formed as the following equation:

$$\mathbf{Y} = \mathbf{CX} + \mathbf{t} = \mathbf{SOX} + \mathbf{t}. \tag{4}$$

The 3D model first being rotated by $\mathbf{O}$, then, projected into a 2D space using $\mathbf{S}$, which is a 2×3 matrix. Finally, the 2D projection is translated into the same position as the input image based on the 2D translation vector $\mathbf{t}$. Compared with Eq. 2, this model is straightforward and similar

[1] In the description of step 1, we omit index μ for clarity.

to template matching. The term **SOX** can be considered as a deformable template, $\mathbf{t}$ helps determine the position of the template in a 2D image, and (**SO**) is a deformation coefficient of the latent model $\mathbf{X}$. Different from the upper-triangular intrinsic camera matrix $\mathbf{K}$ in the classical model, a 2×3 intrinsic projection matrix $\mathbf{S}$ is used to project the rotated model from 3D space to 2D space. Ideally, the values $\mathbf{S}(1,1)$ and $\mathbf{S}(2,2)$ are nonzero. They are the scalar values measuring the stretching affect along pitch and yaw.

Since the roll angle can be eliminated by face detectors through image rotation, we assume the images provided are without roll. We denote the yaw and pitch angle of face as q_x and q_y, and we concatenate them to form a vector $\mathbf{q}$. The 3×3 matrix $\mathbf{O}$, then can be decomposed into a combination of two basic matrices by Eq. 5 according to classical rotation matrix decomposition:

$$\mathbf{O}(\mathbf{q}) = \mathbf{O}_x(q_x)\mathbf{O}_y(q_y). \tag{5}$$

In the following, we use $\mathbf{O}(\mathbf{q})$ to denote the $\mathbf{O}$ computed from Eq. 5 based on $\mathbf{q}$, and use $\mathbf{O}(\mathbf{q}_*)$ to denote $\mathbf{O}_*$.

When solving an inverse problem to obtain $\mathbf{O}(\mathbf{q}_*)$ in training, since $\mathbf{Y}$ and $\mathbf{X}$ are available, we first compute the mean points of any two points in $\mathbf{Y}$ and $\mathbf{X}$ to suppress the potential Gaussian noise in annotation [8]. We denote the mean point sets as $\bar{\mathbf{Y}}$ and $\bar{\mathbf{X}}$. Let $\bar{\mathbf{y}}$ denote a single 2D point in $\bar{\mathbf{Y}}$. Then, we compute $\bar{\mathbf{y}}^i - \bar{\mathbf{y}}^j (i \neq j, i, j \in \{1, \ldots, n\})$ for every pair of landmarks in $\mathbf{Y}$ to generate $\mathbf{J}$. Here, i and j are the indices of the landmarks. Correspondingly, we also obtain the $\mathbf{L}$ on 3D through the pair-wise subtraction operation mentioned above. Hereby, we simplify Eq. 4 into Eq. 6:

$$\mathbf{J} = \mathbf{CL} = \mathbf{SO}(\mathbf{q})\mathbf{L}. \tag{6}$$

The term (**SO**) is denoted by $\mathbf{C}$, which can be obtained using a standard least squares minimization method. To decompose the 2×3 intrinsic projection matrix $\mathbf{S}$ and the orthogonal matrix $\mathbf{O}(\mathbf{q}_*)$ from $\mathbf{C}$, we first add a row to $\mathbf{C}$ and set its values to be the cross product between the first two rows to obtain matrix $\mathbf{C}'$. It is used to recover the missing information on the z-axis. Then, by computing SVD, we decompose the extended matrix $\mathbf{C}'$ into $\mathbf{C}' = \mathbf{U}\mathbf{\Omega}\mathbf{V}^T$, whereby we are able to compute the following parameters. $\mathbf{O}(\mathbf{q}_*) = \mathbf{U}\mathbf{V}^T$ and $\mathbf{S}_* = \mathbf{C}(\mathbf{O}(\mathbf{q}_*))^{-1}$. Finally, we obtain $\mathbf{t}_* = \bar{\mathbf{Y}} - \mathbf{SR}\bar{\mathbf{X}}$. In Eq. (6), the decomposition of $\mathbf{C}'$ is independent of $\mathbf{t}$ so the $\mathbf{O}(\mathbf{q}_*)$ can be determined.

Step 2: Since $\mathbf{t}$ and $\mathbf{S}$ are mainly determined by the location and size of face ROI ($\mathbf{I}$), $\mathbf{q}$ is the principal element that affects $\mathbf{P}$. Next, we concentrate on learning a mapping from the features extracted on $\mathbf{\Gamma}$ to update $\mathbf{q}$.

After we obtain the $\mathbf{q}_*$, we need to generate an attraction around $\mathbf{q}_*$ and train a way to reach $\mathbf{q}_*$ from any potentially inaccurate initialization in testing. With this in mind, we generate training samples to simulate the potentially inaccurate $\mathbf{q}$ through perturbations of $\mathbf{q}_*$. Specifically, since the $\mathbf{q}_*$ is composed of a pose and a yaw angle, around the $\mathbf{q}_*$, we perturb both angles N times and obtain $\mathbf{q}_0^{\mu,\nu}$, where μ is the image index ranging from 1 to Z (Z is the number of images in the training set), and ν is the index of perturbation ranging from 1 to N^2. The subscript 0 indicates the initialization for Step 3. Steps 1 and 2 are summarized in Alg. 1.

Algorithm 2 Learn the parameters of $\Re$ (Step 3)

Input: $\mathbf{I}^\mu$, $\mathbb{M}^\mu$, $\mathbf{Y}^\mu$, $\mathbf{X}^\mu$, $\mathbf{q}_0^{\mu,\nu}$, $\mathbf{t}_*^\mu$, $\mathbf{S}_*^\mu$
Output: $\mathbf{Q}_k$, $\mathbf{b}_k$ $(k = 1 : L)$

```
while Σ_μ Σ_ν ||q_k^{μ,ν} − q_{k−1}^{μ,ν}|| ≤ threshold do
    Synthesize the 2D landmarks Ŷ_k^{μ,ν}
    Estimate the P_k^{μ,ν} using Eq. 2
    Obtain the T-AFM using Γ_k^{μ,ν} = g(I^μ, P_k^{μ,ν}, M^μ)
    Compute the FISH feature Λ_k^{μ,ν} on Γ_k^{μ,ν}
    Estimate the Q_k and b_k for ℜ using Eq. 7
    Update q_k^{μ,ν} using Eq. 8
    k=k+1
end while
```

Step 3: In this step, a regressor is learned to update $\mathbf{q}$ based on the features extracted on $\mathbf{\Gamma}$. The engine of this regressor is the supervised descent method (SDM) [30]. SDM has been widely used in landmark localization and shown to be effective in solving non-linear regression problems. Let k denote the index of iteration and let L denote the maximum number of iterations. The regression parameters $\mathbf{Q}_k$ and $\mathbf{b}_k$ are learned by solving the linear regression Eq. 7 based on multivariate linear regression, and the parameter $\mathbf{q}_k^{\mu,\nu}$ is updated according to Eq. 8:

$$\arg\min_{\mathbf{Q}_k, \mathbf{b}_k} \sum_\mu \sum_\nu ||\mathbf{q}_*^\mu - \mathbf{q}_k^{\mu,\nu} - \mathbf{Q}_k \mathbf{\Lambda}_k^{\mu,\nu} - \mathbf{b}_k||^2, \tag{7}$$

$$\mathbf{q}_k^{\mu,\nu} = \mathbf{q}_{k-1}^{\mu,\nu} + \Delta\mathbf{q}_{k-1}^{\mu,\nu} = \mathbf{q}_{k-1}^{\mu,\nu} + \mathbf{Q}_{k-1}\mathbf{\Lambda}_{k-1}^{\mu,\nu} + \mathbf{b}_{k-1}, \tag{8}$$

where $\mathbf{\Lambda}_k^{\mu,\nu}$ denotes the features extracted from $\mathbf{\Gamma}_k^{\mu,\nu}$. To obtain $\mathbf{\Gamma}_k^{\mu,\nu}$, according to Eq. 1, the 3D-to-2D projection matrix $\mathbf{P}_k^{\mu,\nu}$ is needed. To compute $\mathbf{P}_k^{\mu,\nu}$ from the $\mathbf{q}_k^{\mu,\nu}$, we first estimate the $\mathbf{O}(\mathbf{q}_k^{\mu,\nu})$ according to Eq. 5, then we synthesize virtual 2D landmarks $\hat{\mathbf{Y}}_k^{\mu,\nu}$ using Eq. 4 based on $\mathbf{S}_*^\mu$, $\mathbf{t}_*^\mu$ and $\mathbf{X}^\mu$. Finally, $\mathbf{P}_k^{\mu,\nu}$ is estimated by solving a least squares minimization problem $\hat{\mathbf{Y}}_k^{\mu,\nu} = \mathbf{P}_k^{\mu,\nu}\mathbf{X}$. $\mathbf{\Gamma}_k^{\mu,\nu}$ is computed using Eq. 1. We then extract the feature $\mathbf{\Lambda}_k^{\mu,\nu}$ as follows: we first divide $\mathbf{\Gamma}_k^{\mu,\nu}$ into two parts symmetric along the horizontal axis. The part that is fully visible to the observer in the current pose is called as the "visible component", which can be inferred by $\mathbf{q}_{k-1}^{\mu,\nu}$. We divide each component into overlapping patches and compute the sharpness

score FISH [26] as a measure of local distortion. We concatenate the scores on each face component into one feature vector, perform PCA to reduce its dimension, and then compute the difference of the feature vector with the features extracted from the same subject's T-AFM in the reference database to finally obtain $\mathbf{\Lambda}_k^{\mu,\nu}$. Note that when regressing the pitch, we exploit the feature vector in the visible component, while regressing the yaw, we use the concatenated feature vector of both components. This configuration provides the best performance in our experiments.

3.2. Testing

Algorithm 3 Align an AFM model to a 2D image

Input: $\mathbf{I}$, $\mathbb{M}$, $\mathbf{X}$, $\mathbf{Q}_k$, $\mathbf{b}_k$
Output: $\mathbf{P}$

Step 1: Estimate $\mathbf{q}_0$ by 2dSC based on $\mathbf{I}$
Step 2: Estimate $\mathbf{S}_0$ and $\mathbf{t}_0$ using $\mathbf{q}_0$ and $\mathbf{I}$
Step 3: Apply the Alg. 4 and obtain $\mathbf{P}$

Step 1: Given a face ROI, we first need to estimate a initial $\mathbf{q}$, denoted by $\mathbf{q}_0$. Our starting point is based on the conclusion of recent research [9] in facial pitch and yaw angle estimation that uses multiple adjacent soft labels to determine that actual pose is better than predicting a single label. In this paper, we use an efficient approach to obtain an initial pose as depicted in Fig. 1.

The sparse coding (SC) approach has been widely used in face recognition [27]. However, there is little work extending it to solve a two-dimensional classification problem (e.g., pose estimation). Similar to Masi *et al*. [20], we design a dictionary $\mathbf{D} \in \mathbb{R}^{\theta \times wh}$, where each column is a θ-dimension feature vector extracted from the whole ROI in training. The columns are represented by $\mathbf{d}_s^{\sigma}$, where s is the element that belongs to the s^{th} subject, and σ is the σ^{th} pose in all available $w \cdot h$ poses in the training set; w denotes the available yaw poses, and h denotes the pitch poses in training. Given a query sample ROI $\mathbf{I}$, we estimate the sparse coefficients $\mathbf{a}$ with Lasso [23]. Since the dictionary lies on the 2D space, we name this method **2dSC**:

$$\arg\min_{\mathbf{a}} |\mathbf{I} - \mathbf{aD}|_2^2 + \beta|\mathbf{a}|_1. \tag{9}$$

After we solve Eq. 9, the sparse coefficients $\mathbf{a}$ can be obtained. Let a be a single element in $\mathbf{a}$; we sum the elements across subjects $\sum_s a_s$ and obtain a 1D vector. We rearrange this vector to a $w \cdot h$ 2D coefficient matrix $\mathbf{A}$, where columns represent the yaw angle and rows represent the pitch angle in the training set. A filter $\mathbf{F}$ is defined as follows: [0,1,0;1,1,1;0,1,0]. We compute the response of filter $\mathbf{F}$ on $\mathbf{A}$ and find the maximum response position annotated as $\mathbf{A}_{(\widetilde{w},\widetilde{h})}$. The $\widetilde{w}$ and $\widetilde{h}$ are the corresponding yaw and pitch angle of this position. Along with $\mathbf{A}_{(\widetilde{w},\widetilde{h})}$, we place the four neighboring elements of $\mathbf{A}_{(\widetilde{w},\widetilde{h})}$ into an active set. Then, we determine the estimated pitch and yaw angle in $\mathbf{q}_0$ by computing the dot product of the elements in the active set and the corresponding yaw/pitch angle in $w \cdot h$ poses, and sum them. Finally, we regress the $\mathbf{q}_0$ to a nearest pose in $w \cdot h$.

Step 2: We estimate $\mathbf{S}_k$ and $\mathbf{t}_k$ using $\mathbf{q}_k$ as follows. The matrix $\mathbf{S}_k$ is obtained through first finding a set of training samples whose $\mathbf{q}_*$ is similar to $\mathbf{q}_0$ through a nearest-neighbor algorithm. Here we use ϕ to indicate the index of a selected sample. We denote the scaling matrix $\mathbf{S}$ in the selected sample to be $\hat{\mathbf{S}}^{\phi}$. Then, we divide the first and second columns of each $\hat{\mathbf{S}}^{\phi}$ by the width and height of the ϕ^{th} image's bounding box and obtain a normalized $\hat{\mathbf{S}}^{\phi}$, denoted as $\hat{\mathbf{S}}_{\delta}^{\phi}$. We compute a mean $\hat{\mathbf{S}}_{\delta}^{\phi}$ of all the selected samples and multiply the mean $\hat{\mathbf{S}}_{\delta}^{\phi}$ by the $\mathbf{I}$'s width and height and obtain our estimate of $\mathbf{S}_k$. To obtain $\mathbf{t}_k$, we first compute an average offset between the ϕ^{th} bounding box's center and the $\hat{\mathbf{t}}^{\phi}$ among the selected samples, then add this average offset to the center of the $\mathbf{I}$ and finally obtain an approximate $\mathbf{t}_k$.

Step 3: We iteratively update $\mathbf{q}_0$ with the help of $\Re$ according to Alg. 4 and the whole process of testing is shown in Alg. 3.

Algorithm 4 Estimate $\mathbf{P}$

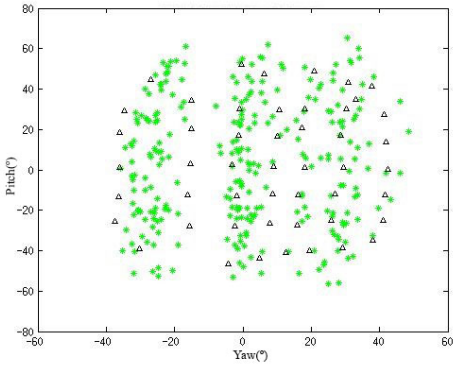
Input: $\mathbf{I}$, $\mathbb{M}$, $\mathbf{X}$, $\mathbf{Q}_k$, $\mathbf{b}_k$, $\mathbf{q}_0$, $\mathbf{S}_0$, $\mathbf{t}_0$, reference T-AFM
Output: 3D-to-2D projection matrix $\mathbf{P}$

1: Synthesize $\hat{\mathbf{Y}}_0$ using Eq. 4 and $\mathbf{q}_0$, $\mathbf{S}_0$, $\mathbf{t}_0$
2: Compute $\mathbf{P}_0$ through Eq. 2
3: **for** *k = 1:L* **do**
4: Extract $\mathbf{\Lambda}_k$ on $\mathbf{\Gamma}_k$
5: Update $\mathbf{q}_k$ by Eq. 8 with $\mathbf{Q}_k$, $\mathbf{b}_k$
6: Update $\mathbf{S}_k$ and $\mathbf{t}_k$ using $\mathbf{q}_k$ and $\mathbf{I}$
7: Synthesize $\hat{\mathbf{Y}}_k$ using Eq. 4 with $\mathbf{q}_k$, $\mathbf{S}_k$, $\mathbf{t}_k$
8: Compute $\mathbf{P}_k$ using Eq. 2
9: **end for**
10: Output the final $\mathbf{P}_k$

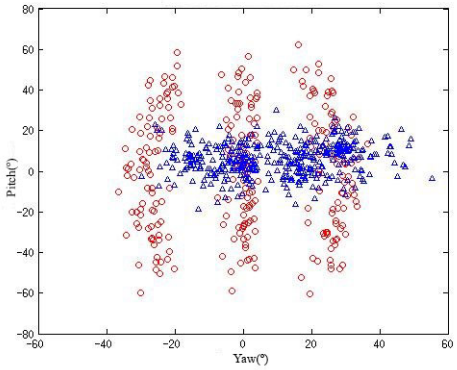
4. Experiments

4.1. Methodology

Training: To implement the Step 1 in testing, we need to provide enough exemplars to cover the possible variations of head pose to provide a coarse estimate. With this in mind, we selected a symmetric subset of the Point 04 database [10], which contains seven yaw and pitch angles with the labels $\{-60°, -30°, -15°, 0°, 15°, 30°, 60°\}$ in both yaw and pitch. We manually annotated each image with nine 2D landmarks and re-estimated the pose of each group through fitting a mean 3D model obtained by averaging the corresponding 3D sparse landmarks in the FRGC



(a)



(b)

Figure 2. The pose distribution in training and testing sets. (a) The black triangles denote the poses we used in the Point 04 database. The green stars denote the poses used to train the regressor in UHDB31-A. (b) The red circles denote the testing poses in UHDB31-B, and the blue triangles denote the testing poses in UHDB11.

database [21]. We used this re-estimated pose as the label of Point 04. To train the regressor in the second stage, we exploited our own database, UHDB31-A. This annotated database is captured using 21 high resolution 3dMD [1] cameras with an average 800*600 facial ROI size. The original dataset named UHDB31 includes data from 77 subjects. A personalized 3D model is available for each subject. We selected a subset of the database that includes nine pitch variations and three yaw angles (27 views), 540 images in total. We used 10 subjects for training (270 images) and another 10 for testing (270 images). These two datasets are named UHDB31-A and UHDB31-B, respectively. The pose distribution for training is shown in Fig. 2(a). All the bounding boxes are either annotated manually (UHDB11) or based on the landmark positions (UHDB31). To train the regressor, we perturbed both pitch and yaw angle 19 times with the following offsets: $\{-17°, -15°, -13°, -10°, -7°, -5°, -3°, -2°, -1°, 0°, 1°, 2°, 3°, 5°, 7°, 10°, 13°, 15°, 17°\}$.

Testing: We tested our method on two databases. The first database is UHDB31-B. The second database is a subset of UHDB11 [24], a published database that aims to validate the 3D-aided 2D face recognition algorithm. The pose distribution of the two databases is shown in Fig. 2(b). We selected all of the 408 images without significant roll angle in UHDB11 to validate our algorithm. For each subject, we manually select the face ROI and used one frontal image and a personal AFM to generate the reference T-AFM database. We selected the RBF-SVR as our baseline to evaluate our model, trained on the selected Point 04 database with our newly estimated pitch and yaw angle.

In training, we used a person-specific AFM for texture lifting. In testing, since we cannot access the person-specific 3D model, as is typical for in-the-wild scenarios, we report the accuracy with and without a personal 3D model. In feature extraction of 2dSC, the HOG feature is configured as a default parameter [6] and extracted from the face ROI on the Point 04 dataset and the query image. In the regression, the size of the block for FISH is 8×8 and the overlap is 2 pixels. The size of the T-AFM is 90×90. We computed the FISH feature on an effective region with the sizes 53×38 (half face) and 53×74 (whole face) for pitch and yaw, respectively, which gives the best performance. Before extraction of the FISH, the Tan and Triggs [22] illumination normalization algorithm was applied on T-AFM to minimize the effects of lighting.

Table 1. Pose estimation result on UHDB11 and UHDB31-B; the error of pose estimation is measured in degrees.

Methods	UHDB11		UHDB31-B	
	pitch	yaw	pitch	yaw
RBF-SVR(Baseline)	7.26	7.98	10.40	6.98
2dSD(Initialization)	9.50	5.68	12.28	4.06
2dSC+SDM+PS3D	7.08	5.86	10.63	4.72
2dSC+SDM+G3D	7.12	5.75	9.23	4.66
2dSC+SDM(pitch)+PS3D	6.98	5.68	9.12	4.06
2dSC+SDM(pitch)+G3D	**6.98**	**5.68**	**9.00**	**4.06**

4.2. Results

We present the qualitative results of pose estimation in Table 1. The 2dSC+SDM is the standard algorithm setting mentioned above. In 2dSC+SDM(pitch), we regressed the pitch angle using SDM and left the yaw as initialized by 2dSC. The PS3D denotes that we use a personalized 3D AFM for texture lifting, and G3D means that we used a general 3D AFM for initialization (e.g., a mean 3D model in FRGC 3D dataset). We also provide the quantitative results of 3D model fitting in Fig. 3.

(a) (b) (c) (d) (e) (f)

Figure 3. Examples of results (best viewed in color). Results depicted in columns (a-c) are generally acceptable. Results (d-f) indicate failures. In the first row, each dot represents the projected 3D vertex position on the 2D image. Red dots are estimated by our proposed algorithm, and represent the 2D projected position of 3D landmarks estimated by traditional landmark-based model fitting approaches. The second row depicts the T-AFM, while the third row depicts the estimated T-AFM after the algorithm terminated.

5. Discussion

The results indicate that the regressor is more effective to address the pitch misalignment. However, 2dSC appears to work better to estimate the yaw. Even though this result is surprising, it is consistent with the phenomenon mentioned by Guo *et al*. [11], in which the author shows that the optimum algorithms to estimate the pitch and yaw are not the same. Unfortunately, the author has not explained why this is the case. In this paper, we attempt to provide several explanations based on the design of our experiments.

(i) The exemplar-based algorithm (2dSC) is more suitable for estimating the yaw, as these are continuous changes of facial occlusions, and represents them well. Since there is not much occlusion variance in the change of pitch, and therefore a lack of critical discriminative information, the exemplar based algorithm treats them as more or less identical and fails to tell the difference. This can be observed in the comparison between 2dSC and RBF-SVR, and the data in Table 1.

(ii) The regression based algorithm disclosed the continuous change of face along the vertical direction so that it is more suitable for pitch estimation on T-AFM. However, the algorithm could not discriminate whether the feature's change is caused by misalignment or the change of subject. Hence, it is not accurate for yaw regression. Since the regression is trained on all poses together, from the success of the pitch regression we could conclude that the regression of pitch is general and identical regardless of the actual pose, but the change of yaw needs to be modeled on a case-by-case basis if we hope to formulate it as a regression problem based on T-AFM, or formulate it as a symmetric detection problem done by Ma *et al*. [19].

(iii) The personalized AFM is not necessary in testing; using a generic AFM appears to work better in pose parameter estimation. We also can observe that a more fine-grained approach needs to be proposed to deal with the error in the scale and translation parameters (Fig. 3(d-f)).

6. Conclusions

We proposed a novel framework to fit a 3D model to 2D image based on sparse coding and T-AFM-based regression. This is the first attempt to fit a dense 3D model to 2D image without the assistance of 2D landmarks.

7. Acknowledgment

This research was funded in part by the US Army Research Lab (W911NF-13-1-0127) and the UH Hugh Roy and Lillie Cranz Cullen Endowment Fund. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of the sponsors.

References

[1] 3dMD. 3dMD: 3D imaging systems and Software, 2012.

[2] R. Abiantun, U. Prabhu, and M. Savvides. Sparse feature extraction for pose-tolerant face recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 2061–2073, 2014.

[3] S. O. Ba and J.-M. Odobez. Recognizing visual focus of attention from head pose in natural meetings. *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(1):16–33, 2009.

[4] V. Balasubramanian, J. Ye, and S. Panchanathan. Biased manifold embedding: A framework for person-independent head pose estimation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, Minneapolis, MN, Jun. 18-23 2007.

[5] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. *International Journal of Computer Vision*, 107(2):177–190, 2014.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, San Diego, CA, Jun 20–26 2005.

[7] P. Dou, Y. Wu, S. K. Shah, and I. A. Kakadiaris. Robust 3D face shape reconstruction from single images via two-fold coupled structure learning. In *Proc. British Machine Vision Conference*, Nottingham, UK, September 1-5 2014.

[8] P. Dou, Y. Wu, S. K. Shah, and I. A. Kakadiaris. Benchmarking 3D pose estimation for face recognition. In *Proc. IEEE International Conference on Pattern Recognition*, pages 190–195, Stockholm, Sweden, Aug. 24-28 2014.

[9] X. Geng and Y. Xia. Head pose estimation based on multivariate label distribution. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1837–1842, Columbus, OH, June 24-27 2014.

[10] N. Gourier, D. Hall, and J. Crowley. Estimating face orientation from robust detection of salient facial structures. In *Proc. International Workshop on Visual Observation of Deictic Gestures*, pages 1–9, Cambridge, England, UK, Aug. 23-26 2004.

[11] G. Guo, Y. Fu, C. R. Dyer, and T. Huang. Head pose estimation: classification or regression? In *Proc.* 19^{th} *International Conference on Pattern Recognition*, pages 1–4, Tampa, FL, Dec. 8-11 2008.

[12] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[13] T. Hassner. Viewing real-world faces in 3D. In *Proc. IEEE International Conference on Computer Vision*, pages 3607 – 3614, Sydney, Australia, Dec. 1-8 2013.

[14] G. Hsu and H. Peng. Face recognition across poses using a single 3D reference model. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 869–874, Portland, Oregon, June 25-27 2013.

[15] G. Hu, Chi-Ho-Chan, F. Yan, W. Christmas, and J. Kittler. Robust face recognition by an albedo based 3D morphable model. In *Proc.* 2^{nd} *International Joint Conference on Biometrics*, Clearwater, FL, September 29 - October 2 2014.

[16] D. Huang, M. Storer, F. De la Torre, and H. Bischof. Supervised local subspace learning for continuous head pose estimation. In *Proc.* 24^{th} *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2928, Colorado Springs, CO, June 21-23 2011.

[17] I. A. Kakadiaris, G. Passalis, G. Toderici, M. Murtuza, Y. Lu, N. Karampatziakis, and T. Theoharis. Three-dimensional face recognition in the presence of facial expressions: An annotated deformable model approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(4):640–649, 2007.

[18] I. Kemelmacher-Shlizerman and S. Seitz. Face reconstruction in the wild. In *Proc.* 13^{th} *IEEE International Conference on Computer Vision*, pages 1746–1753, Barcelona, Spain, Nov. 6-13 2011.

[19] B. Ma, A. Li, X. Chai, and S. Shan. CovGa: A novel descriptor based on symmetry of regions for head pose estimation. *Neurocomputing*, 143:97–108, 2014.

[20] I. Masi, G. Lisanti, A. Bagdanov, P. Pala, and A. Bimbo. Using 3D models to recognize 2D faces in the wild. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 775 – 780, Portland, Oregon, June 25-27 2013.

[21] P. J. Phillips, W. T. Scruggs, A. J. O'Toole, P. J. Flynn, K. W. Bowyer, C. L. Schott, and M. Sharpe. FRVT 2006 and ICE 2006 large-scale experimental results. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(5):831–846, May 2010.

[22] X. Tan and B. Triggs. Real-time human pose and shape estimation for virtual try-on using a single commodity depth camera. *IEEE Trans. on Image Processing*, 19(6):1635–1650, June, 2010.

[23] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[24] G. Toderici, G. Evangelopoulos, T. Fang, T. Theoharis, and I. A. Kakadiaris. UHDB11 database for 3D-2D face recognition. In *Proc.* 6^{th} *Pacific-Rim Symposium on Image and Video Technology*, pages 73–86, Guanajuato, Mexico, Oct. 28–Nov. 1 2013.

[25] G. Toderici, G. Passalis, S. Zafeiriou, G. Tzimiropoulos, M. Petrou, T. Theoharis, and I. A. Kaka diaris. Bidirectional relighting for 3D-aided 2D face recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 2721–2728, San Francisco, CA, June 13-18 2010.

[26] P. V. Vu and D. Chandler. A fast wavelet-based algorithm for global and local image sharpness estimation. *IEEE Signal Processing Letters*, 19(7):423–426, 2012.

[27] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma. Toward a practical face recognition system: Robust alignment and illumination by sparse representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(2):372 –386, Feb 2012.

[28] C. Wang and X. Song. Robust head pose estimation via supervised manifold learning. *Neural Networks*, 53:15–25, 2014.

[29] Q. Wang, Y. Wu, Y. Shen, Y. Liu, and Y. Lei. Supervised sparse manifold regression for head pose estimation in 3D space. *Signal Processing*, pages 34 – 42, 2015.

[30] X. Xiong and F. De la Torre. Supervised descent method and its applications to face alignment. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 532–539, Portland, Oregon, June 25-27 2013.

[31] D. Yi, Z. Lei, and Z. Li. Towards pose robust face recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 3539–3545, Portland, Oregon, June 25-27 2013.