Face Alignment in Full Pose Range: A 3D Total Solution

Xiangyu Zhu, Xiaoming Liu, Member, IEEE, Zhen Lei, Senior Member, IEEE, and Stan Z. Li, Fellow, IEEE

Abstract—Face alignment, which fits a face model to an image and extracts the semantic meanings of facial pixels, has been an important topic in the computer vision community. However, most algorithms are designed for faces in small to medium poses (yaw angle is smaller than 45°), which lack the ability to align faces in large poses up to 90°. The challenges are three-fold. Firstly, the commonly used landmark face model assumes that all the landmarks are visible and is therefore not suitable for large poses. Secondly, the face appearance varies more drastically across large poses, from the frontal view to the profile view. Thirdly, labelling landmarks in large poses is extremely challenging since the invisible landmarks have to be guessed. In this paper, we propose to tackle these three challenges in an new alignment framework termed 3D Dense Face Alignment (3DDFA), in which a dense 3D Morphable Model (3DMM) is fitted to the image via Cascaded Convolutional Neural Networks. We also utilize 3D information to synthesize face images in profile views to provide abundant samples for training. Experiments on the challenging AFLW database show that the proposed approach achieves significant improvements over the state-of-the-art methods.

Index Terms—Face Alignment, 3D Morphable Model, Convolutional Neural Network, Cascaded Regression

1 INTRODUCTION

Face alignment is the process of moving and deforming a face model to an image, so as to extract the semantic meanings of facial pixels. It is an essential preprocessing step for many face analysis tasks, e.g. recognition [1], animation [2], tracking [3], attributes classification [4] and image restoration [5]. Traditionally, face alignment is approached as a landmark detection problem that aims to locate a sparse set of facial fiducial points, some of which include “eye corner”, “nose tip” and “chin center”. In the past two decades, a number of effective frameworks have been proposed such as ASM [6], AAM [7] and CLM [8]. Recently, with the introduction of Cascaded Regression [9], [10], [11] and Convolutional Neural Networks [12], [13], face alignment has observed significant improvements in accuracy. However, most of the existing methods are designed for medium poses, under the assumptions that the yaw angle is smaller than 45° and all the landmarks are visible. When the range of yaw angle is extended up to 90°, significant challenges emerge. These challenges can be differentiated in three main ways:

Modelling: Landmark shape model [6] implicitly assumes that each landmark can be robustly detected by its distinctive visual patterns. However, when faces deviate from the frontal view, some landmarks become invisible due to self-occlusion [14]. In medium poses, this problem can be addressed by changing the semantic positions of face contour landmarks to the silhouette, which is termed landmark marching [15]. However, in large poses where half of face is occluded, some landmarks are inevitably invisible and show no detectable appearance. In turn, landmarks can lose their semantic meanings, which may cause the shape model to fail.

Fitting: Another challenge in full-pose face alignment is derived from the dramatic appearance variations from front to profile. Cascaded Linear Regression [11] and traditional non-linear models [16], [17], [10] are not flexible enough to cover these complex variations in a unified way. Another framework demonstrates more flexibility by adopting different landmark and fitting models for differing view categories [14], [18], [19], [20]. Unfortunately, since the nature of this framework must test every view, computational cost is likely to significantly increase. More recently, Convolutional Neural Network (CNN) based methods have demonstrated improved performance over traditional methods in many applications. For effective large-pose face alignment, CNN should be combined with the Cascaded Regression framework. However, most existing methods adopt a single network to complete fitting [13], which limits its performance.

Training Data: Labelled data is the basis for any supervised
learning based algorithms. However, manual labelling of landmarks on large-pose faces is very tedious since the occluded landmarks have to be “guessed” which is impossible for most of people. As a result, almost all the public face alignment databases such as AFW [19], LFPW [21], HELEN [22] and IBUG [23] are collected in medium poses. Few large-pose databases such as AFLW [24] only contain visible landmarks, which could be ambiguous in invisible landmarks, makes it hard to train a unified face alignment model.

In this paper, we aim to solve the problem of face alignment in full pose range, where the yaw angle is allowed to vary between $\pm 90^\circ$. We believe that face alignment is not barely a 2D problem since self-occlusion and large appearance variations are caused by the face rotation in the 3D space, which can be conveniently addressed by incorporating 3D information. More specifically, we improve the face model from 2D sparse landmarks to a dense 3D Morphable Model (3DMM) [25] and consider face alignment as a 3DMM fitting task. The optimization concept therein will change accordingly from landmark positions to pose (scale, rotation and translation) and morphing (shape and expression) parameters. We call this novel face alignment framework 3D Dense Face Alignment (3DDFA). To realize 3DDFA, we propose to combine two achievements in recent years, namely, Cascaded Regression and the Convolutional Neural Network (CNN). This combination requires the introduction of a new input feature which fulfills the “cascade manner” and “convolution manner” simultaneously (see Sec. 3.2) and a new cost function which can model the priority of 3DMM parameters (see Sec. 3.4). Besides to provide enough data for training, we find that given a face image and its corresponding 3D model, it is possible to rotate the image out of plane with high fidelity. This rotation enables the synthesis of a large number of training samples in large poses.

In general, we propose a novel face alignment framework to address the three challenges of modelling, fitting and training data in large poses. The main contributions of the paper are summarized as follows:

1) To address the self-occlusion challenge, we assert that in large poses, fitting a 3DMM is more suitable than detecting 2D landmarks. The visibility estimated from 3DMM enables us to only fit the vertices with detected image patterns. The landmarks, if needed, can be sampled from the fitted 3D face afterwards. See the samples in Fig. 1.

2) To handle appearance variations across large poses, we propose a novel Cascaded Convolutional Neural Network as the regressor, in which two specially designed input features called Projected Normalized Coordinate Code (PNCC) and Pose Adaptive Feature (PAF) are introduced to connect CNNs in a cascade manner. Besides, a novel cost function named Optimized Weighted Parameter Distance Cost (OWPDC) is proposed to formulate the priority of 3DMM parameters during training.

3) To enable the training of the 3DDFA, we construct a face database consisting of pairs of 2D face images and 3D face models. We further elucidate a face profiling method to synthesize $60k$ training samples across large poses. The synthesized samples well simulate the face appearances in large poses and boost the performance of both previous and the proposed face alignment approaches.

This paper is an extension of our previous work [26] the following four aspects: 1) Traditional 3DMM uses Euler angles to represent the 3D rotation, which shows ambiguity when the yaw angle reaches $90^\circ$. In this paper, quaternions are used instead as the rotation formulation to eliminate the ambiguity. 2) A new input feature called Pose Adaptive Feature (PAF) is utilized to remedy the drawbacks of PNCC to further boost the performance. 3) We improve the cost function in [26] through the OWPDC which not only formulates the importance but also the priority of 3DMM parameters during training. 4) Additional experiments are conducted to better analyze the motivation behind the design of the input features and the cost function.

2 RELATED WORKS

Face alignment can be summarized as fitting a face model to an image. As such, there are two basic problems involved with this task: how to model the face shape and how to estimate the model parameters. In this section, we motivate our approach by discussing related works with respect to these two problems.

2.1 Face Model

Traditionally, face shape is represented by a sparse set of 2D facial fiducial points. Cootes et al. [6], [7] show that shape variations can be modeled with subspace analysis such as Principal Components Analysis (PCA). Although, this 2D-subspace model can only cope with shape variations from a narrow range of face poses, since the non-linear out-of-plane rotation cannot be well represented with the linear subspace. To deal with the pose variations, some modifications like Kernel PCA [27] and Bayesian Mixture Model [14] are proposed to introduce non-linearity into the subspace models. Recently, Cao et al. [10] propose to abandon any explicit shape constraints and directly use landmark coordinates as the shape model, which called 2D Non-Parametric Model (2D-NPM). 2D-NPM considerably improves the flexibility of the shape model at the cost of losing any shape priors and increasing the difficulty of model fitting. Besides 2D shape model, Blanz et al. [28], [25] propose the 3D Morphable Model (3DMM) which applies PCA on a set of 3D face scans. By incorporating 3D information, 3DMM disentangles the non-linear out-of-plane transformation from the PCA subspace. The remaining shape and expression variations have shown high linearity [25], [2], which can be well modeled with PCA. Compared with 2D models, 3DMM separates rigid (pose) and non-rigid (shape and expression) transformations, enabling it to cover diverse shape variations and keep shape prior at the same time. Additionally, points visibility can be easily estimated by 3DMM [26], which can provide important clues to handle self-occlusion in profile views.

2.2 Model Fitting

Most fitting methods can be divided into two categories: the template fitting based [7], [29], [30] and regression based [31], [9], [16], [11], [32]. The template fitting methods always maintain a face appearance model to fit images. For example, Active Appearance Model (AAM) [7] and Analysis-by-Synthesis 3DMM Fitting [25] simulate the process of face image generation and achieve alignment by minimizing the difference between the model appearance and the input image. Active Shape Model (ASM) [6] and Constrained Local Model (CLM) [8], [33] build a template model for each landmark and use a PCA shape model to constrain the fitting results. TSPM [19] and CDM [18] employ part
Different from 2D methods, 3D face alignment [46] aims to fit a 3D face from given 2D landmarks. Interestingly, 2D/3D face alignment results can be mutually transformed, where 3D to 2D is made by sampling landmark vertices and 2D to 3D is made by 3D face reconstruction.

In this work, we propose a framework to combine three major achievements—3DMM, Cascaded Regression and CNN—to solve the large-pose face alignment problem.

3 3D Dense Face Alignment (3DDFA)

In this section, we introduce how to combine Cascaded Regression and CNNs to realize 3DDFA. By applying a CNN as the regressor in Eqn. 1, Cascaded CNN can be formulated as:

$$p^{k+1} = p^k + \text{Net}^k(\text{Fea}(I, p^k)).$$

There are four components in this framework: the regression objective $p$ (Sec. 3.1), the image features $\text{Fea}$ (Sec. 3.2), the CNN structure $\text{Net}$ (Sec. 3.3) and the cost function to train the framework (Sec. 3.4).

3.1 3D Morphable Model

Blanz et al. [25] propose the 3D Morphable Model (3DMM) to describe the 3D face space with PCA:

$$S = S + A_{id} \alpha_{id} + A_{exp} \alpha_{exp},$$

where $S$ is a 3D face, $\bar{S}$ is the mean shape, $A_{id}$ is the principle axes trained on the 3D face scans with neutral expression and $\alpha_{id}$ is the shape parameter, $A_{exp}$ is the principle axes trained on the offsets between expression scans and neutral scans and $\alpha_{exp}$ is the expression parameter. In this work, the $A_{id}$ and $A_{exp}$ come from BFM [55] and FaceWarehouse [56] respectively. After the 3D face is constructed, it can be projected onto the image plane with scale orthographic projection:

$$V(p) = f \ast \text{Pr} \ast R \ast (S + A_{id} \alpha_{id} + A_{exp} \alpha_{exp}) + t_{2d},$$

where $V(p)$ is the model construction and projection function, leading to the 2D positions of model vertices, $f$ is the scale factor, $\text{Pr}$ is the orthographic projection matrix $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$, $R$ is the rotation matrix and $t_{2d}$ is the translation vector. The collection of all the model parameters is $p = [f, R, t_{2d}, \alpha_{id}, \alpha_{exp}]^T$. 

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2778152, IEEE Transactions on Pattern Analysis and Machine Intelligence
3.1.1 Rotation Formulation

Face rotation is traditionally formulated with the Euler angles \([57]\) including \(p\)itch, \(y\)aw and \(r\)oll. However, when faces are close to the profile view, there is ambiguity in Euler angles termed gimbal lock \([58]\), see Fig. 3 as an example.

The ambiguity in Euler angles will confuse the regressor and affect the fitting performance. Therefore we adopt a four nodding face.

In our implementation, we merge the scale parameter \(f\) into \([q_0, q_1, q_2, q_3]\) through dividing the quaternion by \(\sqrt{f}\) and do not constrain the quaternion to be unit. As a result, the fitting objective will be \(p = [q_0, q_1, q_2, q_3, l_{2d}, \alpha_{id}, \alpha_{exp}]^T\).

3.2 Feature Design

As the conjunction point of Cascaded Regression and CNN, the input feature should fulfill the requirements from both frameworks, which can be summarized as the following three aspects: Firstly, the convolvable property requires that the convolution operation on the input feature should make sense. As the CNN input, the feature should be a smooth 2D map reflecting the accuracy of current fitting. Secondly, to enable the cascade manner, the feedback property requires the input feature to depend on the CNN output \([9]\). Finally, to guarantee the cascade to converge at the ground truth parameter, the convergence property requires the input feature to be discriminative when the fitting is complete.

Fig. 2. An overview of the two-stream network in 3DDFA. With an intermediate parameter \(p^k\), in the first stream we construct a novel Projected Normalized Coordinate Code (PNCC), which is stacked with the input image and sent to the CNN. In the second stream, we get some feature anchors with consistent semantics and conduct Pose Adaptive Convolution (PAC) on them. The outputs of the two streams are merged with an additional fully connected layer to predict the parameter update \(\Delta p^k\).

Fig. 3. An example of gimbal lock. We assume the rotation sequence is from \(p\)itch to \(y\)aw to \(r\)oll. In the first row, the face is firstly rotated 20° around the \(p\)itch axis and then 90° around the \(y\)aw axis, whose Euler angles are \([0°, 90°, 0°]\). In the second row, the face is firstly rotated 90° around the \(y\)aw axis and then 20° around the \(r\)oll axis, whose Euler angles are \([0°, 90°, 20°]\). However the two different Euler angles correspond to the same rotation matrix, generating the profile view of a nodding face.

Besides the three requirements, we find that the input features of face alignment can be divided into two categories. The first category is the image-view feature, where the original image is directly sent to the regressor. For example, \([12]\), \([13]\), \([40]\) use the input image as the CNN input and \([59]\), \([60]\) stack the image with a landmark response map as the input. These kind of features does not lose any information provided by the image but require the regressor to cover any face appearances.

The second category is the model-view feature, where image pixels are rearranged based on the model condition. For example, AAM \([7]\) warps the face image to the mean shape and SDM \([11]\) extract SIFT features at landmark locations. This kind of features aligns the face appearance with current fitting, which simplifies the alignment task progressively during optimization. However, they do not cover the pixels beyond the face model, leading to a bad description of context. As such, fitting with model-view features is easily trapped in local minima \([39]\). In this paper, we propose a model-view feature called Pose Adaptive Feature (PAF) and a image-view feature called Projected Normalized Coordinate Code (PNCC). We further demonstrate that optimal results can be achieved by combining both features.

3.2.1 Pose Adaptive Convolution

Traditional convolutional layers convolve along a 2D map from pixel to pixel, while we intend to convolve at some semantically consistent locations on the face, called Pose Adaptive Convolution (PAC). Considering human face can be roughly approximated with...
a cylinder [61], we compute the cylindrical coordinate of each vertex and sample $64 \times 64$ feature anchors with constant azimuth and height intervals, see Fig. 4(a).

![Fig. 4. Pose Adaptive Convolution (PAC): (a) The $64 \times 64$ feature anchors on the 3D face model. (b) The projected feature anchors $V(p)_{\text{anchor}}$ (the blue/red ones indicate visible/invisible anchors). (c) The feature patch map concatenated by the patches cropped at $V(p)_{\text{anchor}}$. (d) Conducting convolution, whose stride and filter size are the same with the patch size, on the feature patch map and shrinking the responses at invisible points, leading to the Pose Adaptive Feature (PAF).](image)

Given a current model parameter $p$, we first project 3DMM and sample the feature anchors on the image plane, getting $64 \times 64 \times 2$ projected feature anchors $V(p)_{\text{anchor}}$ (Fig. 4(b)). Second we crop $d \times d$ (5 in our implementation) patch at each feature anchor and concatenate the patches into a $(64 \times d) \times (64 \times d)$ patch map according to their cylindrical coordinates (Fig. 4(c)). Finally we conduct $d \times d$ convolutions at the stride of $d$ on the patch map, generating $64 \times 64$ response maps (Fig. 4(d)). The convolutional filters are learned with a common convolutional layer, jointly with other CNN layers as described in Sec. 3.3.

Note that this process is equivalent to directly conducting $d \times d$ convolutions on the projected feature anchors $V(p)_{\text{anchor}}$, which implicitly localize and frontalize the face, making the convolution pose invariant. In order to shrink the features at the occluded region, we consider the vertices whose normal points to minus $z$ as self-occluded and divide the responses at occluded region by two, generating the Pose Adaptive Feature (PAF). We do not eliminate occluded features as [48] since this information is still valuable prior to perfect fitting.

3.2.2 Projected Normalized Coordinate Code

The proposed image-view feature depends on a new type of vertex index, which is introduced as follows: we normalize the 3D mean face to $0 - 1$ in $x, y, z$ axis as Eqn. 5:

$$NCC_d = \frac{S_d - \min(S_d)}{\max(S_d) - \min(S_d)} \quad (d = x, y, z), \tag{5}$$

where the $S$ is the mean shape of 3DMM. After normalization, the 3D coordinate of each vertex uniquely distributes between $[0, 0, 0]$ and $[1, 1, 1]$, so it can be considered as a vertex index, which we call Normalized Coordinate Code (NCC) (Fig. 5(a)). Since NCC has three channels as RGB, we can also show NCC as the face texture. It can be seen as different from the traditional vertex index (from 1 to the number of vertices), NCC is smooth along the face surface.

In the fitting process, with a model parameter $p$, we adopt Z-Buffer to render the projected 3D face colored by NCC (Fig. 5(b)) as in Eqn. 6:

$$\text{PNCC} = Z-\text{Buffer}(V_{3d}(p), \text{NCC}), \quad \tag{6}$$

$$V_{3d}(p) = R \ast (S + A_{id} \alpha_{id} + A_{exp} \alpha_{exp}) + [k_{2d}, 0]^T,$$

where $Z-\text{Buffer}(\nu, \tau)$ renders the 3D mesh $\nu$ colored by $\tau$ and $V_{3d}(p)$ is the projected 3D face. We call the rendered image Projected Normalized Coordinate Code (PNCC). Afterwards, PNCC is stacked with the input image and sent to the CNN.

![Fig. 5. The Normalized Coordinate Code (NCC) and the Projected Normalized Coordinate Code (PNCC). (a) The normalized mean face, which is also demonstrated with NCC as its texture (NCC$_x = R$, NCC$_y = G$, NCC$_z = B$). (b) The generation of PNCC, the projected 3D face is rendered by Z-Buffer with NCC as its colormap.](image)

Comparing PAF and PNCC, we can see that PAF is a model-view feature since it implicitly warps the image with feature anchors and PNCC is an image-view feature it sends the original image into a CNN. Regarding the three properties, they fulfill the feedback property since they both depend on $p$ which is updated by the output of the CNN. As for the convolvable property, PAC is the convolution on the continuous locations indicated by the feature anchors and its result PAF is a smooth 2D map. PNCC is also smooth in 2D and the convolution indicates the linear combination of NCCs on a local patch. As for the convergence property, when the CNN detects that in PAF the face is aligned to front and in PNCC each NCC supersedes its corresponding image pattern, the cascade will converge.

3.3 Network Structure

Unlike existing CNN methods [12], [59] that apply different network structures for different fitting stages, 3DDFA employs a unified network structure across the cascade. In general, at iteration $k (k = 0, 1, \ldots, K)$, given an initial parameter $p^0$, we construct PNCC and PAF with $p^k$ and train a two-stream CNN $Net^k$ to conduct fitting. The output features from two streams are merged to predict the parameter update $\Delta p^k$:

$$\Delta p^k = Net^k(\text{PAF}(p^k, I), \text{PNCC}(p^k, I)). \tag{7}$$

Afterwards, a better intermediate parameter $p^{k+1} = p^k + \Delta p^k$ becomes the input of the next network $Net^{k+1}$ which has the same structure but different weights with $Net^k$. Fig. 2 shows the network structure. In the PNCC stream, the input is the $200 \times 200 \times 3$ color image stacked by the $200 \times 200 \times 3$ PNCC. The network contains five convolutional layers, four pooling layers and one fully connected layer. In the PAF stream, the input is the $200 \times 200 \times 3$ color image and $64 \times 64$ feature anchors. The image is processed with the pose adaptive convolution, followed by three pooling layers, three convolutional layers and one fully connected layer. The outputs of the two streams are merged with an additional fully connected layer to predict the 234-dimensional parameter update including 6-dimensional pose parameters $[q_0, q_1, q_2, q_3, t_{2dx}, t_{2dy}]$, 199-dimensional shape parameters $\alpha_{id}$ and 29-dimensional expression parameters $\alpha_{exp}$.

3.4 Cost Function

Different from the landmark shape model, the parameters in 3D-MM contribute to the fitting accuracy with very different impacts, giving parameters different priorities. As a result, regression-based
methods suffer from the inequivalence between parameter error and alignment error \cite{10}. In this section, we will discuss this problem with two baseline cost functions and propose our own ways to model the parameter priority.

### 3.4.1 Parameter Distance Cost (PDC)

Take the first iteration as an example. The purpose of the CNN is to predict the parameter update $\Delta p$ so as to move the initial parameter $p_0^g$ closer to the ground truth $p^g$. Intuitively, we can minimize the distance between the ground truth and the current parameter with the Parameter Distance Cost (PDC):

$$ E_{\text{pdc}} = \| \Delta p - (p^g - p_0^g) \|^2. \quad (8) $$

PDC has been traditionally used in regression based model fitting \cite{35, 36, 62}. However, different dimension in $p$ has different influences on the resultant 3D face. For example, with the same deviation, the yaw angle will bring a larger alignment error than a shape parameter, while PDC optimizes them equally, leading to sub-optimal results.

### 3.4.2 Vertex Distance Cost (VDC)

Since 3DDFA aims to morph the 3DMM to the ground truth 3D face, we can optimize $\Delta p$ by minimizing the vertex distances between the current and the ground truth 3D face:

$$ E_{\text{vdc}} = \| V(p_0^g + \Delta p) - V(p^g) \|^2, \quad (9) $$

where $V(\cdot)$ is the face construction and projection as Eqn. 4. We call this cost Vertex Distance Cost (VDC). Compared with PDC, VDC better models the fitting error by explicitly considering parameter semantics. However, VDC is not convex itself, the optimization is not guaranteed to converge to the ground truth parameter $p^g$. Furthermore, we observe that VDC exhibits pathological curvature \cite{63} since the directions of pose parameters always exhibit much higher curvatures than the PCA coefficients. As a result, optimizing VDC with gradient descent converges very slowly due to the “zig-zagging” problem. Second-order optimizations are preferred to handle the pathological curvature but they are expensive and hard to be implemented on GPU.

### 3.4.3 Weighted Parameter Distance Cost (WPDC)

In our previous work \cite{26}, we propose a cost function named Weighted Parameter Distance Cost (WPDC). The motivation is explicitly weighting parameter error by its importance:

$$ E_{\text{wpdc}} = \| \Delta p - (p^g - p_0^g) \|^T \text{diag}(w) \| \Delta p - (p^g - p_0^g) \| Z. \quad (10) $$

where $w$ is the parameter importance vector, which is defined as follows:

$$ w = (w_1, w_2, \ldots, w_i, \ldots, w_p), $$

$$ w_i = \| V(p_{\text{deg},i}) - V(p^g) \| / Z, $$

$$ p_{\text{deg},i} = (p_1^g, \ldots, p_{i-1}^g, (p_0^g + \Delta p), p_{i+1}^g, \ldots, p_p^g), \quad (11) $$

where $p$ is the number of parameter, $p_{\text{deg},i}$ is the $i$-degated parameter whose $i$th element comes from the predicted parameter $(p_0^g + \Delta p)$ and the others come from the ground truth parameter $p^g$. $Z$ is a regular term which is the maximum of $w$. $\| V(p_{\text{deg},i}) - V(p^g) \|$ models the alignment error brought by miss-predicting the $i$th model parameter, which is indicative of its importance. In the training process, the CNN firstly concentrates on the parameters with larger $\| V(p_{\text{deg},i}) - V(p^g) \|$ such as rotation and translation. As $p_{\text{deg},i}$ is closer to $p^g$, the weights of these parameters begin to shrink and the CNN will optimize less important parameters while simultaneously keeping the high-priority parameters sufficiently good. Compared with VDC, WPDC makes sure the parameter is optimized toward $p^g$ and it remedies the pathological curvature issue at the same time.

However, the weight in WPDC only models the “importance” but not the “priority”. In fact, parameters become important sequentially. Take Fig. 6 as an example, when WPDC evaluates a face image with open mouth and large pose, it will assign both expression and rotation high weights. We can observe that attempting to estimate expression makes little sense before the pose is accurate enough, see Fig. 6(b). One step further, if we force the CNN to only concentrate on pose parameters, we obtain a better fitting result, see Fig. 6(c). Consequently for this sample, even though pose and expression are both important, pose has higher priority than expression, but WPDC misses that.

![Image](image.png)

(a) Image (b) Error = 10.72 (c) Error = 4.39

Fig. 6. (a) An open-mouth face in near-profile view. (b) The fitting result of WPDC in the first iteration. (c) The fitting result when the CNN is restricted to only regress the $i$-dimensional pose parameters. Errors are measured by Normalized Mean Error.

### 3.4.4 Optimized Weighted Parameter Distance Cost (OW-PDC)

We can observe that “priority” is a between-parameter relationship which can only be modeled by treating all the parameters as a whole rather than evaluating them separately as WPDC. In this paper, we propose to find the best weights through optimization:

$$ E_{\text{owpdc}} = (\Delta p - (p^g - p_0^g))^T \text{diag}(w^*) (\Delta p - (p^g - p_0^g)), $$

$$ w^* = \arg \min_w \| V(p^c + \text{diag}(w) * (p^g - p^c)) - V(p^g) \|^2 $$

$$ + \lambda \| \text{diag}(w) * (p^g - p^c) \|^2, \quad (12) $$

subject to $0 \leq w \leq 1$, where $w$ is the weights vector, $\Delta p$ is the CNN output, $p^c = p_0^g + \Delta p$ is the current predicted parameter, 0 and 1 are the zeros and ones vectors respectively and $\leq$ is the element-wise less than. In Eqn. 12, by adding a weighted parameter update $\text{diag}(w) * (p^g - p^c)$ to the current parameter $p^c$, we hope the new face is closer to the ground truth face with limited updating. Note that $\| \text{diag}(w) * (p^g - p^c) \|^2$ is the square sum of the gradient of OW-PDC, which models how much CNN weights need to be tuned to predict each parameter. We use this penalty term to choose the parameters which are most beneficial to the fitting and are easiest to learn. The range of $w$ is constrained to be $[0, 1]$ to make sure the parameter is optimized to $p^g$. Obviously, when the $\lambda$ is set to 0, there will be a trivial solution that $w = 1$ and OW-PDC will deteriorate to PDC.

In the training process, directly optimizing Eqn. 12 for each sample is computationally intensive. We expand $V(p^c + $
\[ \text{diag}(w)(p^g - p^c) \] at \( p^g \) with the Taylor formula and let \( \Delta p^c = p^g - p^c \), Eqn. 12 will be:
\[
\|V'(p^g) * \text{diag}(w - 1) * \Delta p^c\|^2 + \lambda \|\text{diag}(w) * \Delta p^c\|^2, \tag{13}
\]
where \( V'(p^g) \) is the Jacobian. Expanding Eqn. 13 and removing the constant terms, we get:
\[
w^T \left( \text{diag}(\Delta p^c) V'(p^g) V'(p^g) \text{diag}(\Delta p^c) \right) w
- 2 * 1^T \left( \text{diag}(\Delta p^c) V'(p^g) V'(p^g) \text{diag}(\Delta p^c) \right) w
+ \lambda * w^T \text{diag}(\Delta p^c) * \Delta p^c w, \tag{14}
\]
where \( .^T \) is the element-wise multiplication. Let \( H = V'(p^g) \text{diag}(\Delta p^c) \) which is a \( 2n \times p \) matrix where \( n \) is the number of vertices and \( p \) is the number of parameters, the optimization will be:
\[
\arg \min_w w^T \left( H^T * H + \lambda * \text{diag}(\Delta p^c) * \Delta p^c \right) * w
+ 2 * 1^T * H^T * H * w, \tag{15}
\]
s.t. \[ 0 \leq w \leq 1, \]
which is a standard quadratic programming problem with the unique solution. The most consuming component in Eqn. 15 is the computation of \( V'(p^g) \). Fortunately, \( p^g \) is constant during training and \( V'(p^g) \) can be pre-computed offline. As a result, the computation of \( w^c \) can be reduced to a \( p \)-dimensional quadratic programming which can be efficiently solved. The only parameter in OWPDC is the \( \lambda \). It directly determines which parameter is valid during training. We set \( \lambda = 0.17 * \| V'(p^c) - V'(p^g) \|^2 \) in our implementation.

4 FACE PROFILING

All the regression based methods rely on training data, especially for CNNs which have thousands of parameters to learn. Therefore, massive labelled faces in large poses are crucial for 3DDFA. However, few of the released face alignment databases contain large-pose samples \([19], [21], [22], [23]\) since labelling standardized landmarks on them is very challenging. In this work, we demonstrate that profile faces can be well synthesized from existing training samples with the help of 3D information. Inspired by the recent achievements in face frontalization \([15], [64]\) which generates the frontal view of faces, we propose to invert this process to synthesize the profile view of faces from medium-pose samples, which is called face profiling. Different from the face synthesizing in recognition \([65]\), face profiling is not required to keep the identity information but to make the synthesizing results realistic. However, current synthesizing methods do not keep the external face region \([66], [65]\), which contains important context information for face alignment. In this section, we elucidate a novel face synthesizing method to generate the profile views of face image with out-of-plane rotation, providing abundant realistic training samples for 3DDFA.

4.1 3D Image Meshing

The depth estimation of a face image can be conducted on the face region and the external region respectively, with different requirements of accuracy. On the face region, we fit a 3DMM through the Multi-Features Framework (MFF) \([47]\) (see Fig. 7(b)). With the ground truth landmarks as a solid constraint throughout the fitting process, MFF can always get accurate results. Few difficult samples can be easily adjusted manually. On the external region, we follow the 3D meshing method proposed by Zhu et al. \([15]\) to mark some anchors beyond the face region and simulate their depth, see Fig. 7(c). Afterwards the whole image can be tuned into a 3D object through triangulation (see Fig. 7(c)/d)).

Fig. 7. 3D Image Meshing. (a) The input image. (b) The fitted 3D face through MFF. (c) The depth image from 3D meshing. (d) A different view of the depth image.

4.2 3D Image Rotation

The simulated depth information enables the 2D image to rotate out of plane to generate the appearances in larger poses. However, as shown in Fig. 8(b), the 3D rotation squeezes the external face region and loses the background. As a result, we need to further adjust the anchors to keep the background relatively unchanged and preserve the smoothness simultaneously. Inspired by our previous work \([15]\), we propose to adjust background anchors by solving an equation list about their relative positions.

In the source image as shown in Fig. 8(a), the triangulated anchors build up a graph where the anchors are the vertices and the mesh lines are the edges. In the graph, each edge represents an anchor-to-anchor relationship:
\[
x_{a \rightarrow src} - x_{b \rightarrow src} = \Delta x_{src}, \quad y_{a \rightarrow src} - y_{b \rightarrow src} = \Delta y_{src}, \tag{16}
\]
where \((x_{a \rightarrow src}, y_{a \rightarrow src})\) and \((x_{b \rightarrow src}, y_{b \rightarrow src})\) are two connecting anchors, \(\Delta x_{src}\) and \(\Delta y_{src}\) are the spatial offsets in \(x, y\) axes, which should be preserved in synthesizing. After profiling, we keep the face contour anchors (the magenta points in Fig. 8(b)) consistent and predicting other anchors with the unchanged anchor offsets:
\[
x_{a \rightarrow adj} - x_{b \rightarrow adj} = \Delta x_{src}, \quad y_{a \rightarrow adj} - y_{b \rightarrow adj} = \Delta y_{src}, \tag{17}
\]
Specifically, if $a$ is a face contour anchor, we set $(x_{a_{\text{adj}}}, y_{a_{\text{adj}}})$ to the positions after profiling $(x_{a_{\text{pro}}}, y_{a_{\text{pro}}})$, otherwise $(x_{a_{\text{adj}}}, y_{a_{\text{adj}}})$ are two unknowns need to be solved. By collecting Eqn. 17 for each graph edge, we form an equation list whose least square solution is the adjusted anchors (as seen in Fig. 8(c)).

In this work, we enlarge the yaw angle of image at the step of 5° until 90°, see Fig. 9. Different from face frontalization, with larger rotation angles the self-occluded region can only be expanded. As a result, we avoid the troubling invisible region filling which may produce large artifacts [15]. Through face profiling, we not only obtain face samples in large poses but also augment the dataset to a large scale.

![Fig. 9. 2D and 3D view of face profiling. (a) The original yaw angle $yaw_0$. (b) $yaw_0 + 20°$. (c) $yaw_0 + 30°$. (d) $yaw_0 + 40°$.](image)

5 IMPLEMENTATION

### Training Strategy

With a huge number of parameters, the CNN tends to overfit the training set and the deeper cascade might learn nothing with overfitted samples. Therefore we regenerate $p^k$ at each iteration using a nearest neighbor strategy. By observing that the fitting error highly depends on the ground truth face posture (FP), we perturb a training sample based on a set of similar-FP validation samples. In this paper, we define the face posture as the rotated 3D face without scaling and translation:

$$FP = R^g * (\mathbf{S} + A_{id}x_{id}^g + A_{exp}x_{exp}^g),$$

where $R^g$ is constructed from the normalized ground truth quaternion, $x_{id}^g$ and $x_{exp}^g$ are the ground truth shape and expression parameters respectively. Before training, we select two folds of samples as the validation set and for each training sample we construct a validation subset $\{v_1, ..., v_m\}$ whose members share similar FP with the training sample. At iteration $k$, we regenerate the initial parameter by:

$$p^k = p^g - (p^g_{v_i} - p^k_{v_i}),$$

where $p^k$ and $p^g$ are the initial and ground truth parameter of a training sample, $p^g_{v_i}$ and $p^k_{v_i}$ come from a validation sample $v_i$ which is randomly chosen from the corresponding validation subset. Note that $v_i$ is never used in training.

### Initialization

Besides the face profiling, we also augment the training data (10 times) by randomly in-plane rotating images (up to 30 degrees) and perturbing bounding boxes. Specifically, the bounding boxes are randomly perturbed by a multivariate normal distribution whose mean vector and covariance matrix are obtained by the difference between ground truth bounding boxes and automated detected face rectangles using FTF [67]. This augmentation is quite effective in improving the robustness of the model. During testing, to get $p^0$ we first set $\alpha_{id}, \alpha_{exp}$ to zero and the quaternion to $[1, 0, 0, 0]$, getting a frontal 3D mean face. Then we calculate $\varepsilon_2$ by moving the mean point of the 3D face to the center of the bounding box. Finally, we scale the 3D face, which is equivalent to scaling the quaternion, to make the bounding box enclose the whole face region.

### Running Time

During testing, 3DDFA takes 21.3ms for each iteration, among which PAF and PNCC take 11.6ms and 6.8ms respectively on 3.40GHZ CPU and CNN forward propagation takes 2.9ms on GTX TITAN X GPU. In our implementation, 3DDFA has three iterations and takes 63.9ms (15.65fps) for each sample. Note that the efficiency is mainly limited by the input features, which can be further improved by GPU implementation.

6 EXPERIMENTS

6.1 Datasets

Three databases are used in our experiments, i.e. 300W-LP, AFLW [24] and a specifically constructed AFLW2000-3D.

**300W-LP**: 300W [68] standardises multiple face alignment databases with 68 landmarks, including AFW [19], LFPW [69], HELEN [70], IBUG [68] and XM2VTS [71]. With 300W, we adopt the proposed face profiling to generate 61,225 samples across large poses (1,786 from IBUG, 5,207 from AFW, 16,556 from LFPW and 37,676 from HELEN, XM2VTS is not used), which is further flipped to 122,450 samples. We call the synthesized database as 300W Across Large Poses (300W-LP).

**AFLW**: AFLW [24] contains 21,080 in-the-wild faces with large pose variations (yaw from $-90°$ to $90°$). Each image is annotated up to 21 visible landmarks. The database is very suitable for evaluating face alignment performance in large poses.

**AFLW2000-3D**: Evaluating 3D face alignment in the wild is difficult due to the lack of pairs of 2D image and 3D scan. Considering the recent achievements in 3D face reconstruction which can construct a 3D face from 2D landmarks [52], [15], we assume that a 3D model can be accurately fitted if sufficient 2D landmarks are provided. Therefore the evaluation can be degraded to 2D landmark evaluation which also makes it possible to compare 3DDFA with other 2D face alignment methods. While AFLW is not suitable for this task since only visible landmarks may lead to serious ambiguity in 3D shape, as reflected by the fake good alignment phenomenon in Fig. 10. In this work, we construct a database called AFLW2000-3D for 3D face alignment evaluation, which contains the ground truth 3D faces and the corresponding 68 landmarks of the first 2,000 AFLW samples. More details about the construction of AFLW2000-3D are given in supplemental material.

In all the following experiments, we follow [39] and regard the 300W-LP samples synthesized from the training part of LFPW, HELEN and the whole AFW as the training set (101, 144 images in total). The testing are conducted on three databases: the 300W testing part for general face alignment, the AFLW for large-pose face alignment and the AFLW2000-3D for 3D face alignment. The alignment accuracy is evaluated by the Normalized Mean Error (NME).

6.2 Performance with Different Input Features

As described in Sec. 3.2, the input features of face alignment methods can be divided into two categories, the image-view...
As the alternative to PNCC, we propose two alternative input features which miss these two properties respectively. Corresponding to PNCC and PAF, we divide the network in Fig. 2 into PNCC stream and PAF stream by removing the last fully connected layer and regress the 256-dimensional output of each stream to the parameter update respectively. The combined two-stream network is also reported to demonstrate the improvements. As shown in Fig. 11, PNCC performs better than PAF when used individually and the improvement is enlarged as the pose becomes larger. Besides, PNCC and PAF achieve better performance when combined, which may infer a complementary relationship. This complementary relationship might be because PNCC covers the whole image and contains rich context information, enabling it to fit large scale facial components like the face contour. While PAF is more adept at fitting facial features due to the implicit frontalization, which can well assist PNCC.

6.3 Analysis of Feature Properties

In Sec. 3.2, we introduce three requirements of the input feature: feedback, convolvable and convergence. Among them, the benefits from convolvable and convergence may not be obvious and are further evaluated here. Corresponding to PNCC and PAF, we propose two alternative input features which miss these two properties respectively.

Convolvable Property: As the alternative to PNCC, we propose the Projected Index (PIndex) which renders the projected 3D face with the 1-channel vertex index (from 1 to 53, 490 in BFM [55]) rather than the 3-channel NCC, see Fig. 12. Note that even though PIndex provides the semantic meaning of each pixel, it is not smooth and the convolution of vertex indexes on a local patch is hard to be interpreted by the CNN. As a result, PIndex violates the convolvable requirement. Using the PNCC stream as the network, we adopt PNCC and PIndex as the input feature respectively. As shown in Table 1, by violating the convolvable requirement, the performance drops since the learning task becomes more difficult.

Convergence Property: As the alternative to PAF, we propose the Texture Mapping (TM) [52] which rearranges the pixels on the projected feature anchors to a $64 \times 64$ image, see Fig. 13. Compared with PAF, the main drawback of TM is the weak description beyond the model region. As shown in Fig. 13(a) and Fig. 13(b), TM cannot discriminate whether the projected 3D model occludes the face in the image completely [72]. As a result, whether the fitting is complete is not discriminative for TM, which means the convergence requirement is not fulfilled. On the contrary, PAF can better describe the context information with the convolution on the face contour vertices. As shown in Fig. 13(c) and Fig. 13(d), PAF shows different appearances before and after the face contour is fitted.
TM which use the PAF stream as the network. We can see that PAF outperforms TM by over 6% which verifies the effectiveness of the convergence property.

6.4 Analysis of Cost Function

Performance with Different Cost: We demonstrate the errors along the cascade with different cost functions including PDC, VDC, WPDC and OWPDC. Fig. 14 demonstrates the testing error at each iteration. All the networks are trained until convergence. It is shown that PDC cannot well model the fitting error and converges to an unsatisfied result. VDC is better than PDC, but the pathological curvature problem makes it only concentrate on a small set of parameters and limits its performance. WPDC models the importance of each parameter and achieves a better result. Finally OWPDC further models the parameter priority, leading to faster convergence and the best performance.

Weights of OWPDC: Since the weights of OWPDC reflect the priority of parameters, how the priority changes along the training process is also an interesting point to investigate. In this experiment, for each mini-batch during training, we record the mean weights of the mini-batch and plot the mini-batch weight in Fig. 15. It can be seen that at beginning, the pose parameters (rotation and translation) show much higher priority than morphing parameters (shape and expression). As the training proceeds with error reducing, the pose weights begin to decrease and the CNN deals out its concentration to morphing parameters.

6.5 Error Reduction in Cascade

To analyze the overfitting problem in Cascaded Regression and evaluate the effectiveness of initialization regeneration, we divide 300W-LP into 97, 967 samples for training and 24, 483 samples for testing, without identity overlapping. Fig. 16 shows the training and testing errors at each iteration, without and with initialization regeneration. As observed, in traditional Cascaded Regression the training and testing errors converge fast after two iterations. While with initialization regeneration, the training error is updated at the beginning of each iteration and the testing error continues descending. Considering both effectiveness and efficiency we choose three iterations in 3DDFA.

6.6 Comparison Experiments

In this paper, we evaluate the performance of 3DDFA on three different tasks: the large-pose face alignment on AFLW, the 3D face alignment on AFLW2000-3D and the medium-pose face alignment on 300W.

6.6.1 Large Pose Face Alignment on AFLW

Protocol: In this experiment, we regard the whole AFLW as the testing set and divide it into three subsets according to their absolute yaw angles: $[0^\circ, 30^\circ]$, $[30^\circ, 60^\circ]$, and $[60^\circ, 90^\circ]$ with 11, 596, 5, 457 and 4, 027 samples respectively. The alignment accuracy is evaluated by the Normalized Mean Error (NME), which is the average of landmarks error normalised by face size [48]. The face size is defined as the $\sqrt{\text{width} \times \text{height}}$ of the bounding box (the rectangle hull of all the 68 landmarks). Besides, we report the standard deviation of NMEs across testing subsets to measure the pose robustness. During training, we use the projected 3D landmarks as the ground truth to train 2D methods. For convenient comparison, the ground truth bounding boxes are used for initialization.

Methods: Since little experiment has been conducted on the whole AFLW, we choose some baselines with released training codes,
including RCPR [45], ESR [10], LBF [38], CFSS [39], SDM [73], MDM [32], RMFA [74] and TSPM [19]. Among them RCPR is an occlusion-robust method with the potential to deal with self-occlusion and we train it with landmark visibility computed by 3D information [64]. ESR, SDM, LBF and CFSS are popular Cascaded Regression based methods, among which SDM [73] is the winner of ICCV2013 300W face alignment challenge. MDM is a deep learning based method which adopts CNNs to extract image features. TSPM and RMFA adopt the multi-view framework which can deal with large poses. Besides the state-of-the-art methods, we introduce a Deep Convolutional Network (DCN) as a CNN based baseline. DCN directly regresses raw image pixels to the feature. TSPM and RMFA adopt the multi-view framework which to pose variations.

Table 2 shows the comparison results and Fig. 18 shows the corresponding CED curves. Each 2D method is trained on 300W and 300W-LP respectively to demonstrate the boost from face profiling. For DCN, 3DDFA and TSPM which depend on large training sets, we also compare with our previous work [26] but we do not adopt the SDM based landmark refinement here.

<table>
<thead>
<tr>
<th>Method</th>
<th>AFLW Dataset (21 pts)</th>
<th>AFLW2000-3D Dataset (68 pts)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[0, 30]</td>
<td>[30, 60]</td>
</tr>
<tr>
<td>LBF(300W-LP)</td>
<td>8.43</td>
<td>8.15</td>
</tr>
<tr>
<td>ESR(300W)</td>
<td>5.38</td>
<td>7.71</td>
</tr>
<tr>
<td>ESR(300-WLP)</td>
<td>5.66</td>
<td>8.15</td>
</tr>
<tr>
<td>CFSS(300W)</td>
<td>4.68</td>
<td>8.15</td>
</tr>
<tr>
<td>CFSS(300W-LP)</td>
<td>5.42</td>
<td>9.49</td>
</tr>
<tr>
<td>RCPR(300W)</td>
<td>5.40</td>
<td>7.71</td>
</tr>
<tr>
<td>RCPR(300W-LP)</td>
<td>5.43</td>
<td>7.71</td>
</tr>
<tr>
<td>MDM(300W)</td>
<td>5.14</td>
<td>10.34</td>
</tr>
<tr>
<td>MDM(300-WLP)</td>
<td>5.14</td>
<td>10.34</td>
</tr>
<tr>
<td>SDM(300W)</td>
<td>4.67</td>
<td>6.17</td>
</tr>
<tr>
<td>SDM(300W-LP)</td>
<td>4.75</td>
<td>6.17</td>
</tr>
<tr>
<td>TSPM(300W-LP)</td>
<td>5.51</td>
<td>6.87</td>
</tr>
<tr>
<td>RMFA</td>
<td>5.67</td>
<td>6.70</td>
</tr>
<tr>
<td>DCN(300-WLP)</td>
<td>4.99</td>
<td>6.17</td>
</tr>
<tr>
<td>3DDFA(Pre)</td>
<td>3.00</td>
<td>5.60</td>
</tr>
<tr>
<td>Proposed</td>
<td>4.11</td>
<td>4.38</td>
</tr>
</tbody>
</table>

Fig. 20. Typical failure reasons of 3DDFA, including (a) complicated shadow and occlusion, (b) extreme pose and expression, (c) extreme illumination and (d) limited shape variations of 3DDM on nose.

The NME(%) of face alignment results on AFLW and AFLW2000-3D with the first and the second best results highlighted. The brackets show the training sets.

Results: Firstly, the results indicate that all the methods benefit substantially from face profiling when dealing with large poses. The improvements in [60°, 90°] exceed 40% for all the methods. This is especially impressive since the alignment models are trained on the synthesized data and tested on real samples, which well demonstrates the fidelity of face profiling. Secondly, in near frontal view, most of methods show very similar performance as shown in Fig 18(a). As the yaw angle increases in Fig 18(b) and Fig 18(c), most of 2D methods begin to degrade but 3DDFA could still maintain its performance. Finally, 3DDFA reaches the state of the art above all the 2D methods especially beyond medium poses. The minimum standard deviation also demonstrates its robustness to pose variations.

In Fig. 17, we demonstrate some alignment results of 3DDFA and representative 2D methods. Besides, Fig. 20 show some typical failure cases.

6.6.2 3D Face Alignment in AFLW2000-3D
As described in Section 6.1, 3D face alignment evaluation can be degraded to full-landmarks evaluation considering both visible and invisible ones. Using AFLW2000-3D as the testing set, this experiment follows the same protocol as AFLW, except all the 68 landmarks are used for evaluation. There are 1,306 samples in [0°, 30°], 462 samples in [30°, 60°] and 232 samples in [60°, 90°]. The results are demonstrated in Table 2 and the CED curves are plotted in Fig. 19. We do not report the performance of TSPM models since they do not detect invisible landmarks.

Compared with the results in AFLW, we can see that the standard deviation is dramatically increased, meaning that it is more difficult to keep pose robustness when considering all the landmarks. Besides, the improvement of 3DDFA over the best 2D method DCN is increased from 26.49% in AFLW to 30.33% in AFLW2000-3D, which demonstrates the superiority of 3DDFA in 3D face alignment.
6.6.3 Medium Pose Face Alignment

As a face alignment approach to deal with full pose range, 3DDFA also shows competitive performance on the medium-pose 300W database, using the common protocol in [39]. The alignment accuracy is evaluated by the standard landmark mean error normalized by the inter-pupil distance (NME). For 3DDFA, we sample the 68 landmarks from the fitted 3D face and refine them with SDM to reduce the labelling bias. Table 3 shows that even in medium poses 3DDFA performs competitively, especially on the challenging set.

6.6.4 Robustness to Initialization

The alignment performance can be greatly affected by the bounding boxes used for initialization. In this experiment, we initialize alignment methods with detected bounding boxes by FTF face detector [67] rather than the ground truth bounding boxes. We drop the bad boxes whose IOU with ground truth bounding boxes are less than 0.6 and generate the bounding boxes of undetected faces by random perturbation used in training. Table 4 shows the comparison results with the best two competitors DCN and SDM. Firstly, it can be seen that our method still outperforms others when initialized with face detectors. Besides, by comparing the performance drop brought by replacing bounding boxes, our method demonstrates best robustness to initialization.

7 CONCLUSIONS

Most of face alignment methods tend to fail in profile view since the self-occluded landmarks cannot be detected. Instead of the traditional landmark detection framework, this paper fits a dense 3D Morphable Model to achieve pose-free face alignment. By proposing two input features of PNCC and PAF, we cascade a couple of CNNs as a strong regressor to estimate model parameters. A novel OWPDC cost function is also proposed to consider the priority of parameters. To provide abundant samples for training, we propose a face profiling method to synthesize face appearances in profile views. Experiments show the state-of-the-art performance on AFLW, AFLW2000-3D and 300W.

REFERENCES

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2778152, IEEE Transactions on Pattern Analysis and Machine Intelligence

A. Jourabloo and X. Liu, “Pose-invariant 3D face alignment,” in Computer Vision and Pattern Recognition (CVPR), 2015 IEEE International Conference on. IEEE, 2015. 3, 5, 10


J. Roth, Y. Tong, and X. Liu, “Adaptive 3D face reconstruction from unconstrained photo collections,” in Proceedings of IEEE International Conference on Computer Vision & Pattern Recognition (CVPR16), Las Vegas, NV, USA, 2016. 3


V. Lepeit and P. Fua, Monocular model-based 3D tracking of rigid objects. Now Publishers Inc, 2005. 4


Xiangyu Zhu received the B.S. degree in Sichuan University (SCU) in 2012. He is currently working toward the Ph.D. degree with National Laboratory of Pattern Recognition in the Institute of Automation, Chinese Academy of Sciences. His research interests include pattern recognition and computer vision, in particular, 3D morphable model and face analysis.

Xiaoming Liu received the BE degree from Beijing Information Technology Institute, China, in 1997 and the ME degree from Zhejiang University, China, in 2000. His research interests include pattern recognition, biometrics, human computer interface, object tracking/recognition, online learning, computer vision, and pattern recognition.

Zhen Lei received the BS degree in automation from the University of Science and Technology of China, in 2005, and the PhD degree from the Institute of Automation, Chinese Academy of Sciences, in 2010, where he is currently an associate professor. He has published over 100 papers in international journals and conferences. His research interests are in computer vision, pattern recognition, image processing, and face recognition in particular.

Stan Z. Li received his B.Eng from Hunan University, China, M.Eng from National University of Defense Technology, China, and PhD degree from Surrey University, UK. He is currently a professor and the director of Center for Biometrics and Security Research (CBSR), Institute of Automation, Chinese Academy of Sciences (CASIA). He worked at Microsoft Research Asia as a researcher from 2000 to 2004. Prior to that, he was an associate professor at Nanyang Technological University, Singapore. He was elevated to IEEE Fellow for his contributions to the fields of face recognition, pattern recognition and computer vision. His research interests include pattern recognition and machine learning, image and vision processing, face recognition, biometrics, and intelligent video surveillance.