Fast and Precise Face Alignment and 3D Shape Reconstruction from a Single 2D Image

Ruiqi Zhao, Yan Wang, C. Fabian Benitez-Quiroz, Yaojie Liu, and Aleix M. Martinez(✉)

The Ohio State University, Columbus, USA
{zhao.823,wang.9021,benitez-quiroz.1,liu.4002,martinez.158}@osu.edu

Abstract. Many face recognition applications require a precise 3D reconstruction of the shape of the face, even when only a single 2D image is available. We present a novel regression approach that learns to detect facial landmark points and estimate their 3D shape rapidly and accurately from a single face image. The main idea is to regress a function $f(.)$ that maps 2D images of faces to their corresponding 3D shape from a large number of sample face images under varying pose, illumination, identity and expression. To model the non-linearity of this function, we use a deep neural network and demonstrate how it can be efficiently trained using a large number of samples. During testing, our algorithm runs at more than 30 frames/s on an i7 desktop. This algorithm was the top 2 performer in the 3DFAW Challenge.

Keywords: 3D modeling and reconstruction of faces · Fine-grained detection · 3D shape from a single 2D image · Precise and detailed detections

1 Introduction

Humans can readily and accurately estimate the 3D shape of a face by simply observing a single 2D image example of it. Recent results demonstrate that humans use this shape information to infer identity, expression, facial actions and other properties from such 3D reconstructions [1]. Several computer vision approaches have been developed over the years that attempt to replicate this outstanding ability, e.g., [2–6] provide 3D shape estimates of a set of 2D landmark points on a single image, [7,8] use 2D landmark points over several images, and [9–11] provide 3D shape reconstructions from 2D images. Unfortunately, the results of these algorithms are not yet comparable to those of humans [12].

The present paper describes a novel algorithm that provides a fast and precise estimation of the 3D shape of a face from a single 2D image. The major idea of the paper is to define a mapping function $f(.)$ that identifies the 3D shape of a face from the shading patterns observed in a 2D image. This is illustrated in Fig. 1. As seen in this image, the goal is to define a function $\mathbf{s} = f(\mathbf{a})$ that, given an image $\mathbf{a} \in \mathbb{R}^p$ (p the number of pixels), yields the 3D coordinates of the l landmark points defining the shape of the face, $\mathbf{s} \in \mathbb{R}^{3l}$.

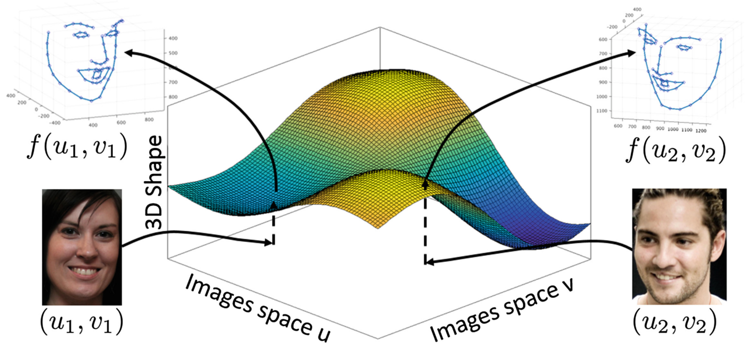

Fig. 1. Conceptual illustration of the proposed approach. The u and v axes correspond to the face image space (pixel values), while the z axis corresponds to the 3D shape. A finite set of face image samples and their associated 2D shape landmarks are used to estimate the parameters of a deep network. This network defines a mapping $f(.)$ from a face image sample to its associated 3D shape.

Given the large number of possible identities, illuminations, poses and expressions, this functional mapping $f(.)$ is difficult to estimate. To resolve this problem, we use a deep neural network. A deep neural network is a regression approach to estimate non-linear mappings of the form $\mathbf{s} = f(\mathbf{a})$, where $\mathbf{a}$ is the input and $\mathbf{s}$ is the output. This means that the network must have p input and $3l$ output nodes. The number of hidden layers and non-linear functions between layers are what allow us to learn the complex 2D image to 3D shape mapping. This is in sharp contrast to linear regression methods attempted before [2–6] as well as non-linear attempts to model 2D shape from a single image [13–15] or 3D shape from multiple images [6,16].

Compared to previous approaches, our algorithm is also able to learn from a small and large number of 3D sample shapes. A small number of samples might not seem sufficient to learn our regressor, but we define data augmentation methods that allow us to circumvent this problem. This is done by using a camera model to generate multiple views of the same 3D shape and the matching 2D landmark point on the original sample image. We demonstrate how this approach is able to successfully and accurately recover the 3D shape of faces from a single view.

We submitted the results of the herein defined algorithm to the 3D Face Alignment in the Wild (3DFAW) challenge. Our algorithm yielded an accuracy of 3.97 %. This was the second best result (with the top algorithm only slightly better at 3.47 % accuracy). We provide additional comparative results with the other 3DFAW participants and algorithms defined in the literature.

It is also important to mention that our derived multilayer neural network can be trained very quickly and testing runs faster than real-time (> 30 frames/s).

2 Related Work

Three-dimensional (3D) reconstruction from a single face image can be roughly divided into two approaches: dense 3D estimation using synthesis and 3D landmark estimation.

With respect to dense 3D face modeling, the main challenge is locating a dense set of corresponding face features in a variety of face images [9]. A parametric morphable face model is generally used to generate arbitrary synthetic images under different poses and illumination, using a 3D Point Distribution Model (PDM) on the morphing function to constrain the face space [9]. Basel Face Model (BFM) [10] improves the texture accuracy while reducing correspondence artifacts by improving the scanning methodology and the registration model. To learn a model from a set of 3D scans of faces, an automatic framework was designed to estimate 3D face shape and texture of faces under varying pose and illumination [16]. In [17], a 3D morphable model was constructed from $> 9,000$ 3D facial scans, by using a novel and almost fully automated construction pipeline. And, in [18], a single template face is used as a reference prior to reconstruct the 3D surface of test faces using a shape-from-shading algorithm. Other approaches are designed to combine 2D and 3D Active Appearance Models (AAM) by constraining a 2D AAM with the equivalent 3D shape modes, which has advantages in both fitting speed and ease of model construction [19,20]. In [11] a monocular face shape reconstruction is formulated as a 2-fold Coupled Structure Learning process, which consists of the regression between two subspaces spanned by 3D and 2D sparse landmarks, and a coupled dictionary of 3D dense and sparse shapes. These models tend to be computational expensive and their model complexity typically yields subpar alignments and reconstructions.

The alternative approach is 3D landmark estimation where we use an image to infer a set of points describing the contour of a set of facial features, e.g., eyes, eyebrows, nose, mouth, etc. These methods are directly related with our proposed algorithm. In [21], a fully automatic method to estimate 3D face shape from a single image is proposed without resorting to manual annotations. This is done by first computing gradient features to get a rough initial estimate of the 2D landmarks. These initial positions are iteratively refined by estimating the 3D shape and pose using an EM-like algorithm. Recently, Cascaded Regression Approach (CRA) has been employed to detect 3D face landmarks from a single image [22,23]. The general idea of CRA is to start from an initial estimate and then learn a sequence of regressors to gradually reduce the distance between this estimate and the actual ground-truth. Specifically, in [22] the authors assume that 3D face shapes can be modeled using PDM. In [23], a direct 3D landmark detection approach is proposed. Here, from an initial set of 3D landmark points, tree-based regressors are used to improve the estimate of the 3D shape of the face. The authors argue that a two steps approaches, i.e., 2D landmark detection and 3D estimation, is generally computationally expensive and needs to be avoided. We prove otherwise. A contribution of our work is to demonstrate that the step of *upgrading* from 2D to 3D landmark points is computationally efficient (running at $> 1,000$ images/s) and yields better accuracies than previous algorithms.

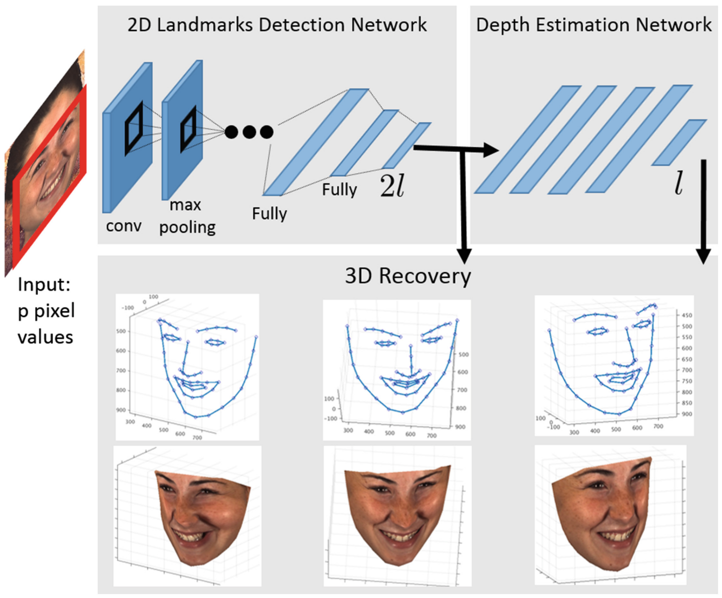

Fig. 2. Proposed network architecture. The face is detected using a standard face detector. The first layers of the network detect the 2D coordinates of l landmarks, x and y coordinates of the landmark points. The latter layers of our network then add the depth information to these 2D landmark points, z values.

3 Method Overview

Our proposed method is illustrated in Fig. 1. As described earlier, our goal is to learn (regress) the non-linear mapping function $\mathbf{s} = f(\mathbf{a})$ from an image $\mathbf{a}$ to a set of 3D landmark points $\mathbf{s}$ defining the 3D shape of the face.

We derive a deep neural network to regress this function $f(.)$. Deep neural networks allow us to model complex, non-linear functions from large numbers of samples. Our samples include 2D images of faces $\mathbf{a}_i$, $i = 1, \ldots, n$, and $n = n_1 + n_2$, with the first n_1 images with their corresponding 2D and 3D shapes, $\mathbf{s}_i$, and the second n_2 images with just 2D shapes. Our proposed network architecture is illustrated in Fig. 2. As can be seen in this figure, we have p entry nodes, representing the p image pixels of the face, and $3l$ output nodes, defining the 3D shape of the face. To facilitate the learning of the function $f(.)$, the entry p nodes must only define the face and, hence, this needs to be aligned. To this end, we use [24] to detect the face, so that the bounding box is resized to have p pixels.

Next, we need to define the optimization criteria of our neural network. The proposed approach requires us to define two optimization criteria. First, we need to derive a criterion for the accurate detection of the 2D landmark points on the aligned image. Second, we must define a criterion for converting these 2D landmark points to 3D. These two criteria are illustrated in Fig. 2. Note that the first criterion is used to optimize the parameters of the first several layers of our deep network, while the second criterion optimizes the parameters of the latter layers. Since our goal is to achieve accurate 3D reconstructions, we will use gradient descent to optimize the parameters of the network until the second criterion (i.e., the 3D shape reconstruction) is as accurate as possible. While it is generally true that this means that the 2D landmark detections needs to be accurate too, we do not directly account for this because the goal of the 3DFAW competition is only to provide an accurate 3D reconstruction of the face.

In the sections to follow, we derive these two criteria for the detection of the 2D fiducial points and their 3D reconstruction and define the details of the architecture of the proposed deep neural network.

4 2D Landmark Points

We define a deep convolutional neural network with p input nodes, $2l$ output nodes and 6 layers (Fig. 2). This includes four convolutional layers and two fully-connected layers. Next, we derive an optimization criterion to learn to detect 2D landmark points accurately.

4.1 Optimization Criterion

Let us define the image samples and their corresponding 2D output variable (i.e., 2D landmark points) as the set $\{(\mathbf{a}_1, \mathbf{o}_1), \ldots, (\mathbf{a}_n, \mathbf{o}_n)\}$, where $\mathbf{o}_i$ is the true (desirable) location of the 2D landmark points of the face. Note that $\mathbf{o}_i$ is a vector of $2l$ image coordinates, $\mathbf{o}_i = (u_{i1}, v_{i1}, \ldots, u_{il}, v_{il})^T$, where $(u_{ij}, v_{ij})^T$ is the j^{th} landmark point.

The goal of a computer vision system is to identify the vector of mapping functions $\mathbf{f}(\mathbf{a}_i, \mathbf{w}) = (f_1(\mathbf{a}_i, w_1), \ldots, f_r(\mathbf{a}_i, w_l))^T$ that converts the input image $\mathbf{a}_i$ to an output vector $\mathbf{o}_i$ of detections, and $\mathbf{w} = (w_1, \ldots, w_l)^T$ is the vector of parameters of these mapping functions. Hence, $f_j(\mathbf{a}_i, w_j) = (\hat{u}_{ij}, \hat{v}_{ij})^T$ are the estimates of the 2D image coordinates u_{ij} and v_{ij}, and w_j are the parameters of the function f_j.

For a fixed mapping function $\mathbf{f}(\mathbf{a}_i, \mathbf{w})$ (e.g., a ConvNet), the goal is to optimize $\mathbf{w}$; formally,

$$\mathcal{J}(\widetilde{\mathbf{w}}) = \min_{\mathbf{w}} \mathcal{L}_{\text{local}}(\mathbf{f}(\mathbf{a}_i, \mathbf{w}), \mathbf{o}_i), \tag{1}$$

where $\mathcal{L}_{\text{local}}(.)$ denotes the loss function. Specifically, we use the L^2-loss defined as,

$$\mathcal{L}_{\text{local}}(\mathbf{f}(\mathbf{a}_i, \mathbf{w}), \mathbf{o}_i) = l^{-1} \sum_{j=1}^{l} \left(f_j(\mathbf{a}_i, w_j) - \mathbf{o}_{ij}\right)^2, \tag{2}$$

where $\mathbf{o}_{ij}$ is the j^{th} element of $\mathbf{o}_i$, i.e., $\mathbf{o}_{ij} \in \mathbb{R}^2$.

Without loss of generality, and to simplify notation, we will use $\mathbf{f}_i$ in lieu of $\mathbf{f}(\mathbf{a}_i, \mathbf{w})$ and f_{ij} instead of $f_j(\mathbf{a}_i, w_j)$. Note that the functions f_{ij} are the same for all i, but may be different for distinct values of j.

The above derivations correspond to a *local* fit. That is, (1) and (2) attempt to optimize the fit of each one of the outputs *independently* and then take the average fit over all outputs. This approach has several solutions, even for a fixed fitting error. For example, the error can be equally distributed across *all* outputs $\|f_{ij} - \mathbf{o}_{ij}\|_2 \approx \|f_{ik} - \mathbf{o}_{ik}\|_2$, $\forall j, k$, where $\|.\|_2$ is the 2-norm of a vector. Or, most of the error is in one (or a few) of the estimates: $\|f_{ij} - \mathbf{o}_{ij}\|_2 >> \|f_{ik} - \mathbf{o}_{ik}\|_2$ and $\|f_{ik} - \mathbf{o}_{ik}\|_2 \approx 0$, $\forall k \neq j$. In general, for a fixed fitting error, the latter example is less preferable, because it leads to large errors in one of the output variables. When this happens we say that the algorithm did not converge as expected.

One solution to this problem is to add an additional constraint to minimize

$$\frac{2}{r(r+1)} \sum_{1 \leq j < k \leq r} \left| (f_{ij} - \mathbf{o}_{ij}) - (f_{ik} - \mathbf{o}_{ik}) \right|^c, \tag{3}$$

with $c \geq 1$. However, this typically results in very slow training, limiting the amount of training data that can be efficiently used. By reducing the number of training samples, we generalize worse and typically obtain less accurate detections [25]. Another typical problem of this equation is that it sometime leads to non-convergence (or convergence with very large fitting error), because the constraint is not flexible enough for current optimization algorithms. We solve these problems by adding a *global* fitting criterion that instead of slowing or halting desirable convergences, it speeds them up.

To do this, it is key to note that the constraint in (2) is local because it measures the fit of each element of $\mathbf{o}_i$ (i.e., $\mathbf{o}_{ij}$) independently. By *local*, we mean that we only care about that one local result. The same criterion can nonetheless be used to measure the fit of pairs of points; formally,

$$\mathcal{L}_{\text{pairs}}(\mathbf{f}_i, \mathbf{o}_i) = \frac{2}{l(l+1)} \sum_{1 \leq j < k \leq l} \left(g\left(f_{ij}, f_{ik}\right) - g\left(\mathbf{o}_{ij}, \mathbf{o}_{ik}\right) \right)^2, \tag{4}$$

where $g(\mathbf{d}, \mathbf{e}) = \|\mathbf{d} - \mathbf{e}\|_b$ is the b-norm of $\mathbf{d} - \mathbf{e}$ (e.g., the 2-norm, $g(\mathbf{d}, \mathbf{be}) = \sqrt{(\mathbf{d} - \mathbf{e})^T (\mathbf{d} - \mathbf{e})}$).

Key to these derivations is to realize that (4) is no longer local, since it takes into account the *global* structure of each pair of elements. This resolves the problems of (2) enumerated above, yielding accurate detections of landmark points and fast training.

4.2 Implementation Details

We set $l = 66$ and use $n_1 + n_2 = 18600$ samples. As mentioned above, we use four convolutional layers, two max pooling layers and two fully connected layers. Following [26], we apply normalization, dropout, and rectified linear units (ReLU) at the end of each convolutional layer. One advantage of our proposed algorithm is that learning can be efficiently performed with very large datasets. Since we wish to have a landmark detector invariant to any affine transformation and partial occlusions, we also use a data augmentation approach. Specifically, we generated an additional $80,000$ images by applying two-dimensional affine transformations to the existing training set, i.e., scale, reflection, translation and rotation; scale was between 2 and .5, rotation was $-10°$ to $10°$, and translation and reflection were randomly generated. To make the network more robust to partial occlusions, we added random occluding boxes of $d \times d$ pixels as in [27], with d between .2 and .4 times the inter-eye distance; 25 % of our training images have partial occlusions.

5 3D Shape

Next, we describe how to recover the 3D information (i.e., the depth value) of the 2D landmark points detected above. We start by writing the n 2D landmark points on the i^{th} image in matrix from as

$$\mathbf{U}_i = \begin{pmatrix} u_{i1} & u_{i2} & \cdots & u_{in} \\ v_{i1} & v_{i2} & \cdots & v_{in} \end{pmatrix} \in \mathbb{R}^{2\times n}. \tag{5}$$

Our goal is to recover the 3D coordinates of these 2D landmark points,

$$\mathbf{S}_i = \begin{pmatrix} x_{i1} & x_{i2} & \cdots & x_{in} \\ y_{i1} & y_{i2} & \cdots & y_{in} \\ z_{i1} & z_{i2} & \cdots & z_{in} \end{pmatrix} \in \mathbb{R}^{3\times n}, \tag{6}$$

where $(x_{ij}, y_{ij}, z_{ij})^T$ are the 3D coordinates of the j^{th} face landmark.

Assuming a weak-perspective camera model, with calibrated camera matrix $\mathcal{M} = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \end{pmatrix}$, the weak-perspective projection of the face 3D landmark points is given by

$$\mathbf{U}_i = \mathcal{M}\mathbf{S}_i. \tag{7}$$

This result is of course defined up to scale, since $\boldsymbol{u}_i = \lambda \boldsymbol{x}_i$ and $\boldsymbol{v}_i = \lambda \boldsymbol{y}_i$, where $\mathbf{x}_i^T = (x_{i1}, x_{i2}, ..., x_{in})$, $\mathbf{y}_i^T = (y_{i1}, y_{i2}, ..., y_{in})$, $\mathbf{z}_i^T = (z_{i1}, z_{i2}, ..., z_{in})$, $\mathbf{u}_i^T = (u_{i1}, u_{i2}, ..., u_{in})$ and $\mathbf{v}_i^T = (v_{i1}, v_{i2}, ..., v_{in})$. This will require that we standardize our variables when deriving our algorithm.

5.1 Deep 3D Shape Reconstruction from 2D Landmarks

Proposed Neural Network. Given a training set with l 3D landmark points $\{\mathbf{S}_i\}_{i=1}^{l}$, we aim to learn the function $f : \mathbb{R}^{2l} \rightarrow \mathbb{R}^n$, that is,

$$\widehat{\mathbf{z}}_i = f(\widehat{\mathbf{x}}_i, \widehat{\mathbf{y}}_i), \tag{8}$$

where $\widehat{\mathbf{x}}_i$, $\widehat{\mathbf{y}}_i$ and $\widehat{\mathbf{z}}_i$ are obtained by standardizing $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_i$ as follows,

$$\begin{aligned} \widehat{x}_{ij} &= \frac{x_{ij} - \overline{\mathbf{x}}_i}{(\sigma(\mathbf{x}_i) + \sigma(\mathbf{y}_i))/2}, \\ \widehat{y}_{ij} &= \frac{y_{ij} - \overline{\mathbf{y}}_i}{(\sigma(\mathbf{x}_i) + \sigma(\mathbf{y}_i))/2}, \\ \widehat{z}_{ij} &= \frac{z_{ij} - \overline{\mathbf{z}}_i}{(\sigma(\mathbf{x}_i) + \sigma(\mathbf{y}_i))/2}, \end{aligned} \tag{9}$$

where $\overline{\mathbf{x}}_i$, $\overline{\mathbf{y}}_i$ and $\overline{\mathbf{z}}_i$ are mean values, and $\sigma(\mathbf{x}_i)$, $\sigma(\mathbf{y}_i)$ and $\sigma(\mathbf{z}_i)$ are the standard deviation of the elements in $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_i$, respectively.

We standardize $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_i$ to eliminate the effect of scaling and translation of the 3D face, as noted above. Herein, we model the function $f(.)$ using a multilayer neural network.

Figure 2 depicts the overall architecture of our neural network. It contains M layers. The m^{th} layer is defined by,

$$\boldsymbol{a}^{(m+1)} = \tanh\left(\boldsymbol{\Omega}^{(m)}\boldsymbol{a}^{(m)} + \boldsymbol{b}^{(m)}\right), \tag{10}$$

where $\boldsymbol{a}^{(m)} \in \mathbb{R}^d$ is an input vector, $\boldsymbol{a}^{(m+1)} \in \mathbb{R}^r$ is the output vector, d and r specify the number of input and output nodes, respectively, and $\boldsymbol{\Omega} \in \mathbb{R}^{r\times d}$ and $\boldsymbol{b} \in \mathbb{R}^r$ are network parameters, with the former a weighting matrix and the latter a basis vector. Our neural network uses a Hyperbolic Tangent function, tanh(.).

Our objective is to minimize the sum of the Euclidean distances between the predicted depth location $\boldsymbol{a}_i^{(m)}$ and the ground-truth $\widehat{\boldsymbol{z}}_i$ of our l 3D landmark points, formally,

$$\min \sum_{i=1}^{l} \|\widehat{\boldsymbol{z}}_i - \boldsymbol{a}_i^{(m)}\|_2, \tag{11}$$

with $\|.\|_2$ the Euclidean distance of two vectors. We utilize the RMSProp algorithm [28] to optimize our model parameters.

Testing. When testing on the t^{th} face, we have $\mathbf{u}_t$ and $\mathbf{v}_t$, and want to estimate $\mathbf{x}_t$, $\mathbf{y}_t$ and $\mathbf{z}_t$. From Eq. (7) we have $\boldsymbol{u}_t = \lambda\boldsymbol{x}_t$ and $\boldsymbol{v}_t = \lambda\boldsymbol{y}_t$. Thus, we first standardize the data,

$$\begin{aligned} \widehat{u}_{tj} &= \frac{u_{tj} - \overline{\boldsymbol{u}}_t}{(\sigma(\boldsymbol{u}_t) + \sigma(\boldsymbol{v}_t))/2}, \\ \widehat{v}_{tj} &= \frac{v_{tj} - \overline{\boldsymbol{v}}_t}{(\sigma(\boldsymbol{u}_t) + \sigma(\boldsymbol{v}_t))/2}. \end{aligned} \tag{12}$$

This yields $\widehat{\boldsymbol{x}}_t = \widehat{\boldsymbol{u}}_t$ and $\widehat{\boldsymbol{y}}_t = \widehat{\boldsymbol{v}}_t$. Therefore, we can directly feed $(\widehat{\boldsymbol{u}}_t, \widehat{\boldsymbol{v}}_t)$ into the trained neural network to obtain its depth $\widehat{\boldsymbol{z}}_t$. Then, the 3D shape of the face can be recovered as $(\widehat{\boldsymbol{u}}_t^T, \widehat{\boldsymbol{v}}_t^T, \widehat{\boldsymbol{z}}_t^T)^T$, a result that is defined up to scale.

Implementation Details. Our feed-forward neural network contains six layers. The number of nodes in each layer is $2n$, $2n$, $2n$, $2n$, $2n$, and n. We divide our training data into a training and a validation set. In each of these two sets, we perform data augmentation. Specifically, we use the weak-perspective camera model defined above to generate new 2D views of the 3D landmark points given in the training set. This process helps the algorithm learn how each 3D shape is seen from a large variety of 2D views (translation, rotation, scale). We use Keras library [29] on top of Theano [30] to develop our multilayer neural network. Early stopping is enabled to prevent overfitting and accelerate the training process.

We stop the training process if the validation error does not decrease after 10 iterations. We set the learning rate at .01.

6 Experimental Results

We report the result of our experiment on the data of the 3D Face Alignment in the Wild Challenge (3DFAW). Three of the four datasets in the challenge are subsets of MultiPIE [31], BU-4DFE [32] and BP4D-Spontaneous [33] databases respectively. Another dataset TimeSlice3D that contains annotated 2D images are extracted from online videos. The depth has been recovered using a model-based Structure from Motion technique [34]. In total, there are 18,694 training images. Each image has 66 labeled 3D fiducial points and a face bounding box centered around the mean 2D projection of the landmarks. The 2D to 3D correspondence assumes a weak-perspective projection. The depth values have been normalized to have zero mean. Another 4,912 images are used for testing. Participants in the challenging only had access to the testing images and their bounding box, but not the 3D landmarks.

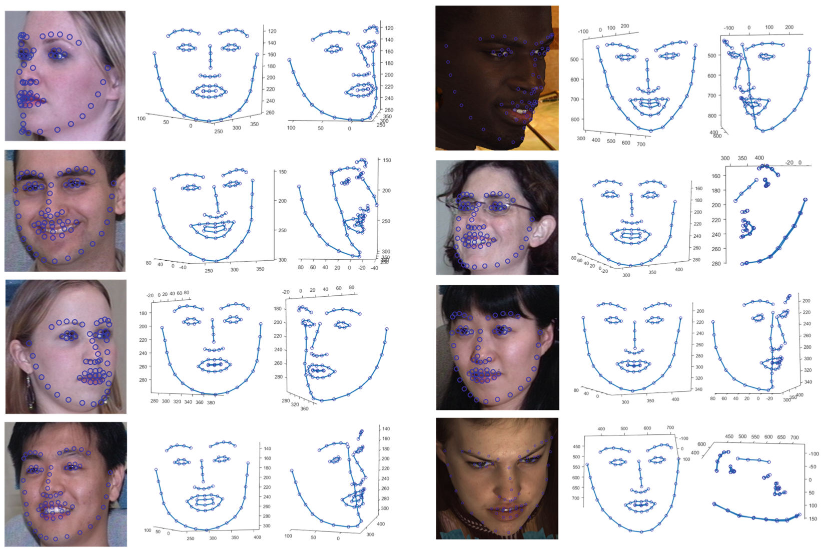

Fig. 3. Qualitative results on the testing set of the challenge. Our approach can detect 3D landmarks of face with large head pose precisely.

Detection error is evaluated using Ground Truth Error (GTE) and Cross View Ground Truth Consistency Error (CVGTCE). GTE is the average point-to-point Euclidean error between prediction and ground truth normalized by the Euclidean distance between the outer corners of the eyes. Formally,

$$E_{gte}(\mathbf{S}, \widetilde{\mathbf{S}}) = \frac{1}{n} \sum_{k=1}^{n} \frac{\|\mathbf{s}_k - \widetilde{\mathbf{s}}_k\|}{d}, \tag{13}$$

where $\|.\|$ is the L_2-norm, $\mathbf{S}$ and $\widetilde{\mathbf{S}}$ are the 3D prediction and ground truth, $\mathbf{s}_k$ and $\tilde{\mathbf{s}}_k$ are the k^{th} 3D point of $\mathbf{S}$ and $\widetilde{\mathbf{S}}$ respectively, and d is the Euclidean distance between the outer corners of the eyes.

CVGTCE is a measurement that evaluates cross-view consistency of the predicted landmarks by comparing the prediction and ground truth from a different view of the same target. Formally,

$$E_{cvgtce}(\mathbf{S}, \widetilde{\mathbf{S}}, P) = \frac{1}{n} \sum_{k=1}^{n} \frac{\|(c\mathbf{R}\mathbf{s}_k + \mathbf{t}) - \widetilde{\mathbf{s}}_k\|}{d}, \tag{14}$$

where $P = \{c, \mathbf{R}, \mathbf{t}\}$ encodes the rigid transformation, i.e., scale (c), rotation ($\mathbf{R}$), and translation ($\mathbf{t}$) between $\mathbf{S}$ and $\widetilde{\mathbf{S}}$. These can be obtained by optimizing the following:

$$\{c, \mathbf{R}, \mathbf{t}\} = \underset{c,\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_{k=1}^{n} \|\widetilde{\mathbf{s}}_k - (c\mathbf{R}\mathbf{s}_k + \mathbf{t})\|$$

Our GTE and CVGTCE for testing images are 5.88% and 3.97%, respectively. Figure 3 shows the qualitative results on the testing set of the challenge. Additionally, we performed another test on the training set of the challenge. We randomly select 13,694 images from training set to train the multi-layer neural network for 3D shape estimation from 2D landmarks. We test on the other 5,000 images in the training set with ground truth 2D face landmarks. The GTE is 2.00%. Comparison of our method with other top ranked methods on 3DFAW challenge dataset is shown in Table 1.

6.1 Across Database Testing

To compare with the state-of-the-art method, we performed another experiment on the images of the BP4D-S database [33]. Note that we tested the proposed approach using the pre-trained model on the 3DFAW dataset of the previous section. That is, no images or 3D data from BP4D-S are used as part of our

Table 1. Comparisons of the GTE and CVGTCE on 3DFAW challenge dataset.

Participant	CVGTCE	GTE
psxab5	3.4767	4.5623
Ours	3.9700	5.8835
rpiisl	4.9488	6.2071
trigeorgis	5.4595	7.6403
olgabellon	5.9093	10.8001

training procedures, i.e., *the experiment is across datasets*. For fair comparison, we followed the procedure in [22]. We randomly selected 100 images with yaw angle between 0° and 10°, 500 images with yaw angle between 10° and 20° and other 500 images with yaw angle between 20° and 30° for a total of 1100 images. Since the landmarks in BP4D-S database are different from the challenge database, we selected the 45 overlapping landmarks to test our algorithm. The reported error in [22] was calculated using the average of point-wise estimation error (APE) as follows:

$$E_{ape}(\mathbf{S}, \widetilde{\mathbf{S}}) = \frac{1}{n} \sum_{k=1}^{n} \|\mathbf{s}_k - \widetilde{\mathbf{s}}_k\| \tag{15}$$

As shown in Table 2, our pre-trained model achieves the smallest APE compared with [22] and the baseline (i.e., using the 3D mean face of the samples in [33]). Figure 4 shows the qualitative results of the proposed approach on samples from BP4D-S.

Table 2. Comparisons of the APE on BP4D-S database.

Ours	PIFA [22]	Baseline
4.14	4.75	5.02

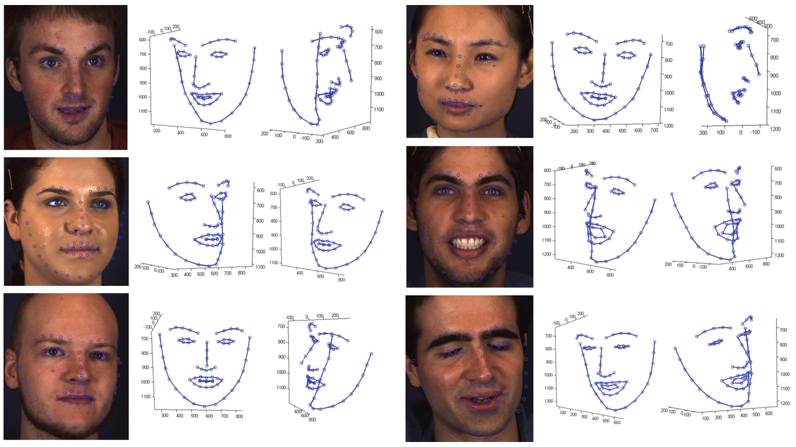

Fig. 4. Qualitative results on the BP4D-S database. Our pre-train model can detect 3D landmarks of face with large head pose and facial expressions precisely.

7 Conclusions

We have presented an algorithm for the reconstruction of the 3D shape of a face from a single 2D image. The proposed algorithm yields very low reconstruction errors and was the top 2 in the 3DFAW competition. Our approach is based on the idea of learning a mapping function from an image of a face to its 3D shape. Herein, we proposed to use a feed-forward neural network to learn this mapping. We defined two criteria, one to learn to detect important shape landmark points on the image and another to recover their depth information. We also presented a data augmentation approach that utilizes camera models to aid the learning of this complex, non-linear mapping function. The derived deep architecture and optimization criteria can be efficiently learned using a large number of samples and testing runs at $>$ 30 frames/s on an i7 desktop.

Acknowledgements. Supported by the National Institutes of Health, grants R01-EY-020834 and R01-DC-014498, and a Google Faculty Research Award.

References

1. Martinez, A., Du, S.: A model of the perception of facial expressions of emotion by humans: research overview and perspectives. J. Mach. Learn. Res. **13**(1), 1589–1608 (2012)
2. Zhou, X., Leonardos, S., Hu, X., Daniilidis, K.: 3D shape estimation from 2D landmarks: a convex relaxation approach. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4447–4455 (2015)
3. Ramakrishna, V., Kanade, T., Sheikh, Y.: Reconstructing 3D human pose from 2D image landmarks. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 573–586. Springer, Heidelberg (2012)
4. Lin, Y.-L., Morariu, V.I., Hsu, W., Davis, L.S.: Jointly optimizing 3D model fitting and fine-grained classification. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part IV. LNCS, vol. 8692, pp. 466–480. Springer, Heidelberg (2014)
5. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Category-specific object reconstruction from a single image. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1966–1974 (2015)
6. Hamsici, O.C., Gotardo, P.F.U., Martinez, A.M.: Learning spatially-smooth mappings in non-rigid structure from motion. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 260–273. Springer, Heidelberg (2012)
7. Fayad, J., Russell, C., Agapito, L.: Automated articulated structure and 3D shape recovery from point correspondences. In: The IEEE International Conference on Computer Vision (ICCV), pp. 431–438 (2011)
8. Gotardo, P.F.U., Martinez, A.M.: Kernel non-rigid structure from motion. In: IEEE International Conference on Computer Vision (ICCV), pp. 802–809 (2011)
9. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3D faces. In: 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), pp. 187–194 (1999)

10. Paysan, P., Knothe, R., Amberg, B., Romdhani, S., Vetter, T.: A 3D face model for pose and illumination invariant face recognition. In: Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 296–301 (2009)
11. Dou, P., Wu, Y., Shah, S., Kakadiaris, I.: Robust 3D face shape reconstruction from single images via two-fold coupled structure learning and off-the-shelf landmark detectors. In: the British Machine Vision Conference, BMVA Press (2014)
12. Ding, L., Martinez, A.: Features versus context: an approach for precise and detailed detection and delineation of faces and facial features. IEEE Trans. Pattern Anal. Mach. Intell. **28**(8), 1274–1286 (2006)
13. Rivera, S., Martinez, A.M.: Learning deformable shape manifolds. Pattern Recogn. **45**(4), 1792–1801 (2012)
14. Xiong, X., De la Torre, F.: Supervised descent method and its applications to face alignment. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
15. Xiong, X., la Torre, F.D.: Global supervised descent method. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
16. Blanz, V., Vetter, T.: Face recognition based on fitting a 3D morphable model. IEEE Trans. Pattern Anal. Mach. Intell. **25**(9), 1063–1074 (2003)
17. Booth, J., Roussos, A., Zafeiriou, S., Ponniah, A., Dunaway, D.: A 3D morphable model learnt from 10,000 faces. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016
18. Kemelmacher-Shlizerman, I., Basri, R.: 3D face reconstruction from a single image using a single reference face shape. IEEE Trans. Pattern Anal. Mach. Intell. **33**(2), 394–405 (2011)
19. Hamsici, O.C., Martinez, A.M.: Active appearance models with rotation invariant kernels. In: 12th International Conference on Computer Vision (ICCV), pp. 1003–1009 (2009)
20. Xiao, J., Baker, S., Matthews, I., Kanade, T.: Real-time combined 2D+3D active appearance models. In: The IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 535–542 (2004)
21. Gu, L., Kanade, T.: 3D alignment of face in a single image. In: The IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1305–1312 (2006)
22. Jourabloo, A., Liu, X.: Pose-invariant 3D face alignment. In: The International Conference on Computer Vision (ICCV) (2015)
23. Tulyakov, S., Sebe, N.: Regressing a 3D face shape from a single image. In: The International Conference on Computer Vision (ICCV) (2015)
24. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (2001)
25. Martínez, A.M., Kak, A.C.: PCA versus LDA. IEEE Trans. Pattern Anal. Mach. Intell. **23**(2), 228–233 (2001)
26. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (2012)
27. Martínez, A.M.: Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. IEEE Trans. Pattern Anal. Mach. Intell. **24**(6), 748–763 (2002)

28. Tieleman, T., Hinton, G.: Lecture 6.5-RmsProp: Divide the gradient by a running average of its recent magnitude. In: COURSERA: Neural Networks for Machine Learning (2012)
29. Chollet, F.: keras (2015). https://github.com/fchollet/keras
30. Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I., Bergeron, A., Bouchard, N., Warde-Farley, D., Bengio, Y.: Theano: new features and speed improvements. arXiv preprint arXiv:1211.5590 (2012)
31. Gross, R., Matthews, I., Cohn, J., Kanade, T., Baker, S.: Multi-pie. Image Vis. Comput. **28**(5), 807–813 (2010)
32. Yin, L., Chen, X., Sun, Y., Worm, T., Reale, M.: A high-resolution 3D dynamic facial expression database. In: 8th IEEE International Conference On Automatic Face & Gesture Recognition, FG 2008, pp. 1–6. IEEE (2008)
33. Zhang, X., Yin, L., Cohn, J.F., Canavan, S., Reale, M., Horowitz, A., Liu, P., Girard, J.M.: BP4D-spontaneous: a high-resolution spontaneous 3D dynamic facial expression database. Image Vis. Comput. **32**(10), 692–706 (2014)
34. Jeni, L.A., Cohn, J.F., Kanade, T.: Dense 3D face alignment from 2D video for real-time use. Image and Vision Computing (2016)