On Learning 3D Face Morphable Model from In-the-wild Images

Luan Tran, and Xiaoming Liu, *Member, IEEE*

Abstract—As a classic statistical model of 3D facial shape and albedo, 3D Morphable Model (3DMM) is widely used in facial analysis, e.g., model fitting, image synthesis. Conventional 3DMM is learned from a set of 3D face scans with associated well-controlled 2D face images, and represented by two sets of PCA basis functions. Due to the type and amount of training data, as well as, the linear bases, the representation power of 3DMM can be limited. To address these problems, this paper proposes an innovative framework to learn a nonlinear 3DMM model from a large set of in-the-wild face images, without collecting 3D face scans. Specifically, given a face image as input, a network encoder estimates the projection, lighting, shape and albedo parameters. Two decoders serve as the nonlinear 3DMM to map from the shape and albedo parameters to the 3D shape and albedo, respectively. With the projection parameter, lighting, 3D shape, and albedo, a novel analytically-differentiable rendering layer is designed to reconstruct the original input face. The entire network is end-to-end trainable with only weak supervision. We demonstrate the superior representation power of our nonlinear 3DMM over its linear counterpart, and its contribution to face alignment, 3D reconstruction, and face editing.
Source code and additional results can be found at our project page: http://cvlab.cse.msu.edu/project-nonlinear-3dmm.html

Index Terms—morphable model, 3DMM, face, nonlinear, weakly supervised, in-the-wild, face reconstruction, face alignment.

1 Introduction

The 3D Morphable Model (3DMM) is a statistical model of 3D facial shape and texture in a space where there are explicit correspondences [1]. The morphable model framework provides two key benefits: first, a point-to-point correspondence between the reconstruction and all other models, enabling morphing, and second, modeling underlying transformations between types of faces (male to female, neutral to smile, etc.). 3DMM has been widely applied in numerous areas including, but not limited to, computer vision [1]–[3], computer graphics [4]–[7], human behavioral analysis [8], [9] and craniofacial surgery [10].

Traditionally, 3DMM is learnt through *supervision* by performing dimension reduction, typically Principal Component Analysis (PCA), on a training set of co-captured 3D face scans and 2D images. To model highly variable 3D face shapes, a large amount of high-quality 3D face scans is required. However, this requirement is expensive to fulfill as acquiring face scans is very laborious, in both data capturing and post-processing stage. The first 3DMM [1] was built from scans of 200 subjects with a similar ethnicity/age group. They were also captured in well-controlled conditions, with only neutral expressions. Hence, it is fragile to large variances in the face identity. The widely used Basel Face Model (BFM) [11] is also built with only 200 subjects in neutral expressions. Lack of expression can be compensated using expression bases from FaceWarehouse [12] or BD-3FE [13], which are learned from the offsets to the neutral pose. After more than a decade, almost all existing models use no more than 300 training scans. Such small training sets are far from adequate to describe the full variability of human faces [14]. Until recently, with a significant effort as well as a novel automated and robust model construction pipeline, Booth *et al.* [14] build the first large-scale 3DMM from scans of $\sim 10,000$ subjects.

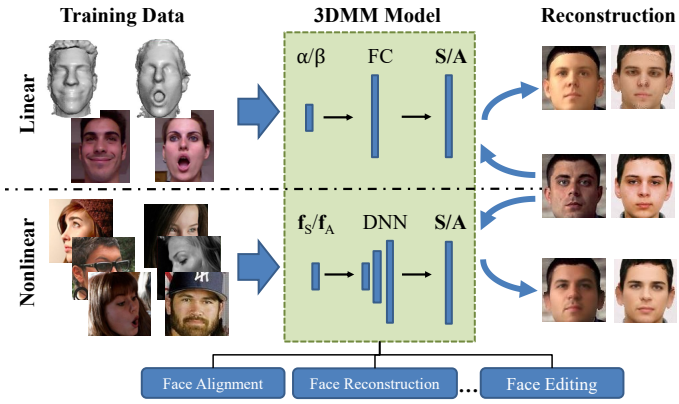


Fig. 1: Conventional 3DMM employs linear bases models for shape/albedo, which are trained with 3D face scans and associated controlled 2D images. We propose a nonlinear 3DMM to model shape/albedo via deep neural networks (DNNs). It can be trained from in-the-wild face images without 3D scans, and also better reconstruct the original images due to the inherent nonlinearity.

Second, the texture model of 3DMM is normally built with a small number of 2D face images *co-captured* with 3D scans, under well-controlled conditions. Despite there is a considerable improvement of 3D acquisition devices in the last few years, these devices still cannot operate in arbitrary in-the-wild conditions. Therefore, all the current 3D facial datasets have been captured in the laboratory environment. Hence, such models are only learnt to represent the facial texture in similar, rather than in-the-wild, conditions. This substantially limits its application scenarios.

Finally, the representation power of 3DMM is limited by not only the size or type of training data but also its *formulation*. The facial variations are nonlinear in nature. E.g., the variations in

• *L. Tran and X. Liu are with the Department of Computer Science and Engineering, Michigan State University.*
E-mail: tranluan@msu.edu, liuxm@cse.msu.edu

Manuscript received Aug 28, 2018.

different facial expressions or poses are nonlinear, which violates the linear assumption of PCA-based models. Thus, a PCA model is unable to interpret facial variations sufficiently well. This is especially true for facial texture. For all current 3DMM models, their low-dimension albedo subspace faces the same problem of lacking facial hair, e.g., beards. To reduce the fitting error, it compensates unexplainable texture by alternating surface normal, or shrinking the face shape [15]. Either way, linear 3DMM-based applications often degrade their performances when handling out-of-subspace variations.

Given the barrier of 3DMM in its data, supervision and linear bases, this paper aims to revolutionize the paradigm of learning 3DMM by answering a fundamental question:

> *Whether and how can we learn a nonlinear 3D Morphable Model of face shape and albedo from a set of in-the-wild 2D face images, without collecting 3D face scans?*

If the answer were yes, this would be in sharp contrast to the conventional 3DMM approach, and remedy all aforementioned limitations. Fortunately, we have developed approaches to offer positive answers to this question. With the recent development of deep neural networks, we view that it is the right time to undertake this new paradigm of 3DMM learning. Therefore, the core of this paper is regarding how to learn this new 3DMM, what is the representation power of the model, and what is the benefit of the model to facial analysis.

We propose a novel paradigm to *learn a nonlinear 3DMM model from a large in-the-wild 2D face image collection, without acquiring 3D face scans*, by leveraging the power of deep neural networks captures variations and structures in complex face data. As shown in Fig. 1, starting with an observation that the linear 3DMM formulation is equivalent to a single layer network, using a deep network architecture naturally increases the model capacity. Hence, we utilize two convolution neural network decoders, instead of two PCA spaces, as the shape and albedo model components, respectively. Each decoder will take a shape or albedo parameter as input and output the dense 3D face mesh or a face skin reflectant. These two decoders are essentially the nonlinear 3DMM.

Further, we learn the fitting algorithm to our nonlinear 3DMM, which is formulated as a CNN encoder. The encoder network takes a face image as input and generates the shape and albedo parameters, from which two decoders estimate shape and albedo.

The 3D face and albedo would *perfectly* reconstruct the input face, if the fitting algorithm and 3DMM are well learnt. Therefore, we design a differentiable rendering layer to generate a reconstructed face by fusing the 3D face, albedo, lighting, and the camera projection parameters estimated by the encoder. Finally, the end-to-end learning scheme is constructed where the encoder and two decoders are learnt jointly to minimize the difference between the reconstructed face and the input face. Jointly learning the 3DMM and the model fitting encoder allows us to leverage the large collection of *in-the-wild* 2D images without relying on 3D scans. We show significantly improved shape and facial texture representation power over the linear 3DMM. Consequently, this also benefits other tasks such as 2D face alignment, 3D reconstruction, and face editing.

A preliminary version of this work was published in 2018 IEEE Conference on Computer Vision and Pattern Recognition [16]. We extend it in numerous ways: 1) Instead of having lighting embedded in texture, we split texture into albedo and shading. Truthfully modeling the lighting help to improve the shape modeling as it can help to guide the surface normal learning. This results in better performance in followed tasks: alignment and reconstruction, as demonstrated in our experiment section. 2) We propose to present the shape component in the 2D UV space, which helps to reserve spatial relation among its vertices. This also allows us to use a CNN, rather than an expensive multi-layer perceptron, as the shape decoder. 3) To ensure plausible reconstruction, we employ multiple constraints to regularize the model learning.

In summary, this paper makes the following contributions:

- We learn a *nonlinear* 3DMM model, fully models shape, albedo and lighting, that has greater representation power than its traditional linear counterpart.
- Both shape and albedo are represented as 2D images, which help to maintain spatial relations as well as leverage CNN power in image synthesis.
- We jointly learn the model and the model fitting algorithm via *weak supervision*, by leveraging a large collection of 2D images without 3D scans. The novel rendering layer enables the end-to-end training.
- The new 3DMM further improves performance in related tasks: face alignment, face reconstruction and face editing.

2 Prior Work

Linear 3DMM. Blanz and Vetter [1] propose the first generic 3D face model learned from scan data. They define a linear subspace to represent shape and texture using principal component analysis (PCA) and show how to fit the model to data. Since this seminal work, there has been a large amount of effort on improving 3DMM modeling mechanism. In [1], the dense correspondence between facial mesh is solved with a regularised form of optical flow. However, this technique is only effective in a constrained setting, where subjects share similar ethnicities and ages. To overcome this challenge, Patel and Smith [17] employ a Thin Plate Splines (TPS) warp [18] to register the meshes into a common reference frame. Alternatively, Paysan *et al.* [11] use a Nonrigid Iterative Closest Point [19] to directly align 3D scans. In a different direction, Amberg *et al.* [8] extended Blanz and Vetter's PCA-based model to emotive facial shapes by adopting an additional PCA modeling of the residuals from the neutral pose. This results in a single linear model of both identity and expression variation of 3D facial shape. Vlasic *et al.* [20] use a multilinear model to represent the combined effect of identity and expression variation on the facial shape. Later, Bolkart and Wuhrer [21] show how such a multilinear model can be estimated directly from the 3D scans using a joint optimization over the model parameters and groupwise registration of 3D scans.

Improving Linear 3DMM. With PCA bases, the statistical distribution underlying 3DMM is Gaussian. Koppen *et al.* [22] argue that single-mode Gaussian can't well represent real-world distribution. They introduce the Gaussian Mixture 3DMM that models the global population as a mixture of Gaussian subpopulations, each with its own mean, but shared covariance. Booth *et al.* [23] aim to improve texture of 3DMM to go beyond controlled settings by learning in-the-wild feature-based texture model. On another direction, Tran *et al.* [24] learn to regress robust and discriminative 3DMM representation, by leveraging multiple images from

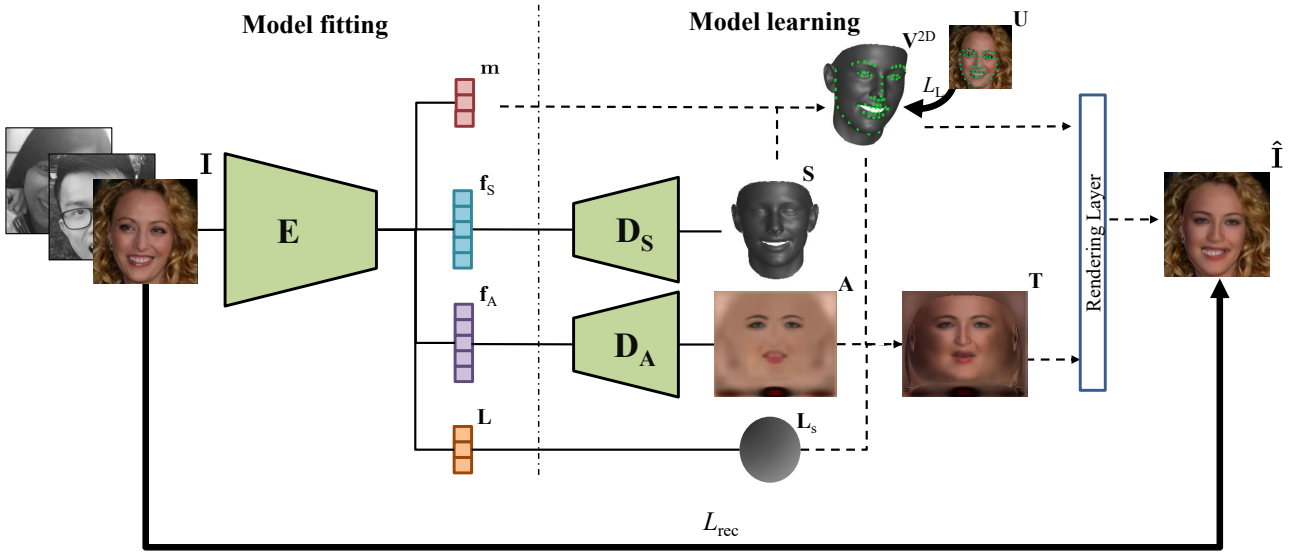


Fig. 2: Jointly learning a nonlinear 3DMM and its fitting algorithm from unconstrained 2D in-the-wild face image collection, in a weakly supervised fashion. $\mathbf{L}_S$ is a visualization of shading on a sphere with lighting parameters $\mathbf{L}$.

the same subject. However, all works are still based on statistical PCA bases. Duong *et al.* [25] address the problem of linearity in face modeling by using Deep Boltzmann Machines. However, they only work with 2D face and sparse landmarks; and hence cannot handle faces with large-pose variations or occlusion well. Concurrent to our work, Tewari *et al.* [26] learn a (potentially non-linear) corrective model on top of a linear model. The final model is a summation of the base linear model and the learned corrective model, which contrasts to our unified model. Furthermore, our model has an advantage of using 2D representation of both shape and albedo, which maintains spatial relations between vertices and leverages CNN power for image synthesis. Finally, thanks for our novel rendering layer, we are able to employ perceptual, adversarial loss to improve the reconstruction quality.

2D Face Alignment. 2D Face Alignment [27], [28] can be cast as a regression problem where 2D landmark locations are regressed directly [29]. For large-pose or occluded faces, strong priors of 3DMM face shape have been shown to be beneficial [30]. Hence, there is increasing attention in conducting face alignment by fitting a 3D face model to a single 2D image [31]–[36]. Among the prior works, iterative approaches with cascade of regressors tend to be preferred. At each cascade, there is a single [30], [37] or even two regressors [38] used to improve its prediction. Recently, Jourabloo and Liu [35] propose a CNN architecture that enables the end-to-end training ability of their network cascade. Contrasted to aforementioned works that use a fixed 3DMM model, our model and model fitting are learned jointly. This results in a more powerful model: a single-pass encoder, which is learned jointly with the model, achieves state-of-the-art face alignment performance on AFLW2000 [32] benchmark dataset (see Tab. 5)

3D Face Reconstruction. Face reconstruction creates a 3D face model from an image collection [39], [40] or even with a single image [41], [42]. This long-standing problem draws a lot of interest because of its wide applications. 3DMM also demonstrates its strength in face reconstruction, especially in the monocular case. This problem is a highly under-constrained, as with a single image, present information about the surface is limited. Hence, 3D face reconstruction must rely on prior knowledge like 3DMM [43]. Statistical PCA linear 3DMM is the most commonly used approach. Besides 3DMM fitting methods [44]–[49], recently, Richardson *et al.* [50] design a refinement network that adds facial details on top of the 3DMM-based geometry. However, this approach can only learn 2.5D depth map, which loses the correspondence property of 3DMM. The follow up work by Sela *et al.* [42] try to overcome this weakness by learning a correspondence map. Despite having some impressive reconstruction results, both these methods are limited by training data synthesized from the linear 3DMM model. Hence, they fail to handle out-of-subspace variations, e.g., facial hair.

Unsupervised learning in 3DMM. Collecting large-scale 3D scans with detailed labels for learning 3DMM is not an easy task. A few work try to use large-scale synthetic data as in [41], [51], but they don't generalize well as there still be a domain gap with real images. Tewari *et al.* [48] is among the first work attempting to learn 3DMM fitting from unlabeled images. They use an unsupervised loss which compares projected textured face mesh with the original image itself. The sparse landmark alignment is also used as an auxiliary loss. Genova *et al.* [52] further improve this approach by comparing reconstructed images and original input using higher-level features from a pretrained face recognition network. Compared to these work, our work has a different objective of learning a *nonlinear* 3DMM.

3 The Proposed Nonlinear 3DMM

In this section, we start by introducing the traditional linear 3DMM and then present our novel nonlinear 3DMM model.

3.1 Conventional Linear 3DMM

The 3D Morphable Model (3DMM) [1] and its 2D counterpart, Active Appearance Model [53]–[55], provide parametric models for synthesizing faces, where faces are modeled using two components: shape and albedo (skin reflectant). In [1], Blanz *et al.* propose to describe the 3D face space with PCA:

$$\mathbf{S} = \bar{\mathbf{S}} + \mathbf{G}\alpha, \tag{1}$$

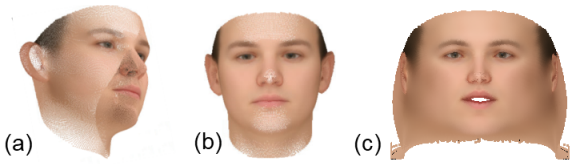


Fig. 3: Three albedo representations. (a) Albedo value per vertex, (b) Albedo as a 2D frontal face, (c) UV space 2D unwarped albedo.

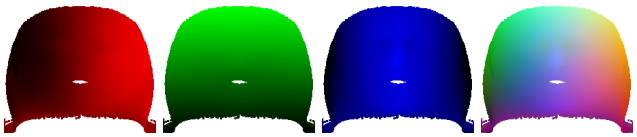


Fig. 4: UV space shape representation. From left to right: individual channels for x, y and z spatial dimension and final combined shape image.

where $\mathbf{S} \in \mathbb{R}^{3Q}$ is a 3D face mesh with Q vertices, $\bar{\mathbf{S}} \in \mathbb{R}^{3Q}$ is the mean shape, $\alpha \in \mathbb{R}^{l_S}$ is the shape parameter corresponding to a 3D shape bases $\mathbf{G}$. The shape bases can be further split into $\mathbf{G} = [\mathbf{G}_{id}, \mathbf{G}_{exp}]$, where $\mathbf{G}_{id}$ is trained from 3D scans with neutral expression, and $\mathbf{G}_{exp}$ is from the offsets between expression and neutral scans.

The albedo of the face $\mathbf{A} \in \mathbb{R}^{3Q}$ is defined within the mean shape $\bar{\mathbf{S}}$, which describes the R, G, B colors of Q corresponding vertices. $\mathbf{A}$ is also formulated as a linear combination of basis functions:

$$\mathbf{A} = \bar{\mathbf{A}} + \mathbf{R}\beta, \tag{2}$$

where $\bar{\mathbf{A}}$ is the mean albedo, $\mathbf{R}$ is the albedo bases, and $\beta \in \mathbb{R}^{l_T}$ is the albedo parameter.

The 3DMM can be used to synthesize novel views of the face. Firstly, a 3D face is projected onto the image plane with the weak perspective projection model:

$$\mathbf{V} = \mathbf{R} * \mathbf{S}, \tag{3}$$

$$g(\mathbf{S}, \mathbf{m}) = \mathbf{V}^{\text{2D}} = f * \mathbf{Pr} * \mathbf{V} + \mathbf{t}_{2d} = M(\mathbf{m}) * \begin{bmatrix} \mathbf{S} \\ \mathbf{1} \end{bmatrix}, \tag{4}$$

where $g(\mathbf{S}, \mathbf{m})$ is the projection function leading to the 2D positions $\mathbf{V}^{\text{2D}}$ of 3D rotated vertices $\mathbf{V}$, f is the scale factor, $\mathbf{Pr} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$ is the orthographic projection matrix, $\mathbf{R}$ is the rotation matrix constructed from three rotation angles (pitch, yaw, roll), and $\mathbf{t}_{2d}$ is the translation vector. While the project matrix M is of the size of 2×4, it has six degrees of freedom, which is parameterized by a 6-dim vector $\mathbf{m}$. Then, the 2D image is rendered using texture and an illumination model such as Phong reflection model [56] or Spherical Harmonics [57].

3.2 Nonlinear 3DMM

As mentioned in Sec. 1, the linear 3DMM has the problems such as requiring 3D face scans for supervised learning, unable to leverage massive in-the-wild face images for learning, and the limited representation power due to the linear bases. We propose to learn a nonlinear 3DMM model using only large-scale in-the-wild 2D face images.

3.2.1 Problem Formulation

In linear 3DMM, the factorization of each of components (shape, albedo) can be seen as a matrix multiplication between coefficients and bases. From a neural network's perspective, this can be viewed as a shallow network with only *one fully connected layer* and no activation function. Naturally, to increase the model's representation power, the shallow network can be extended to a deep architecture. In this work, we design a novel learning scheme to joint learn a deep 3DMM model and its inference (or fitting) algorithm.

Specifically, as shown in Fig. 2, we use two deep networks to decode the shape, albedo parameters into the 3D facial shape and albedo respectively. To make the framework end-to-end trainable, these parameters are estimated by an encoder network, which is essentially the fitting algorithm of our 3DMM. Three deep networks join forces for the ultimate goal of reconstructing the input face image, with the assistant of a physically-based rendering layer. Fig. 2 visualizes the architecture of the proposed framework. Each component will be present in following sections.

Formally, given a set of K 2D face images $\{\mathbf{I}_i\}_{i=1}^{K}$, we aim to learn an encoder E: $\mathbf{I} \rightarrow \mathbf{m}, \mathbf{L}, \mathbf{f}_S, \mathbf{f}_A$ that estimates the projection $\mathbf{m}$, lighting parameter $\mathbf{L}$, shape parameters $\mathbf{f}_S \in \mathbb{R}^{l_S}$, and albedo parameter $\mathbf{f}_A \in \mathbb{R}^{l_A}$, a 3D shape decoder D_S: $\mathbf{f}_S \rightarrow \mathbf{S}$ that decodes the shape parameter to a 3D shape $\mathbf{S} \in \mathbb{R}^{3Q}$, and an albedo decoder D_A: $\mathbf{f}_A \rightarrow \mathbf{A}$ that decodes the albedo parameter to a realistic albedo $\mathbf{A} \in \mathbb{R}^{3Q}$, with the objective that the rendered image with $\mathbf{m}$, $\mathbf{L}$, $\mathbf{S}$, and $\mathbf{A}$ can well approximate the original image. Mathematically, the objective function is:

$$\begin{aligned} &\arg\min_{E, D_S, D_A} \sum_{i=1}^{K} \left\| \hat{\mathbf{I}}_i - \mathbf{I}_i \right\|_1, \\ &\hat{\mathbf{I}} = \mathcal{R}\left(E_m(\mathbf{I}), E_L(\mathbf{I}), D_S(E_S(\mathbf{I})), D_A(E_A(\mathbf{I}))\right), \end{aligned} \tag{5}$$

where $\mathcal{R}(\mathbf{m}, \mathbf{L}, \mathbf{S}, \mathbf{A})$ is the rendering layer (Sec. 3.2.3).

3.2.2 Albedo & Shape Representation

Fig. 3 illustrates three possible albedo representations. In traditional 3DMM, albedo is defined per vertex (Fig. 3(a)). This representation is also adopted in recent work such as [26], [48]. There is an albedo intensity value corresponding to each vertex in the face mesh. Despite widely used, this representation has its limitations. Since 3D vertices are not defined on a 2D grid, this representation is mostly parameterized as a vector, which not only loses the spatial relation of its vertices, but also prevents it to leverage the convenience of deploying CNN on 2D albedo. In contrast, given the rapid progress in image synthesis, it is desirable to choose a 2D image, e.g., a frontal-view face image in Fig. 3(b), as an albedo representation. However, frontal faces contain little information of two sides, which would lose many albedo information for side-view faces.

In light of these consideration, we use an unwrapped 2D texture as our texture representation (Fig. 3(c)). Specifically, each 3D vertex $\mathbf{v}$ is projected onto the UV space using cylindrical unwarp. Assuming that the face mesh has the top pointing up the y axis, the projection of $\mathbf{v} = (x, y, z)$ onto the UV space $\mathbf{v}^{\text{uv}} = (u, v)$ is computed as:

$$v \rightarrow \alpha_1.\arctan\left(\frac{x}{z}\right) + \beta_1, \quad u \rightarrow \alpha_2.y + \beta_2, \tag{6}$$

where $\alpha_1, \alpha_2, \beta_1, \beta_2$ are constant scale and translation scalars to place the unwrapped face into the image boundaries. Here, per-

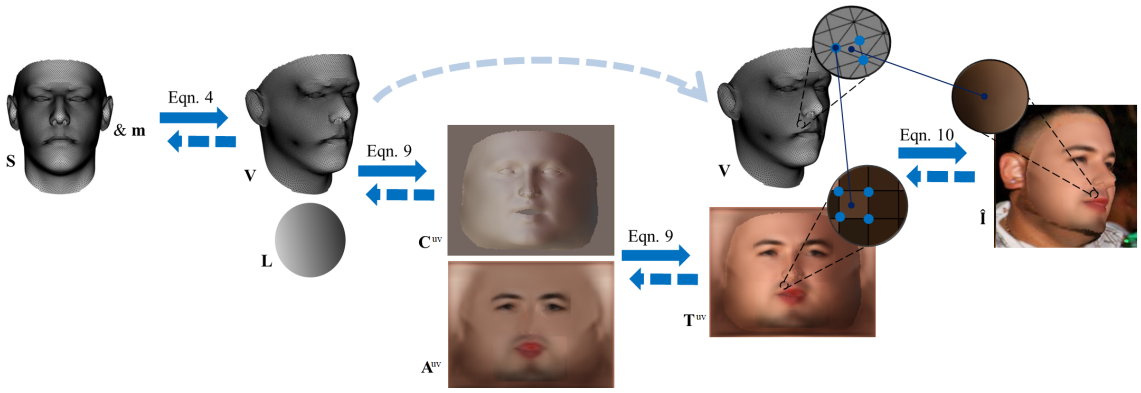


Fig. 5: Forward and backward pass of the rendering layer.

vertex albedo $\mathbf{A} \in \mathbb{R}^{3Q}$ could be easily computed by sampling from its UV space counterpart $\mathbf{A}^{\text{uv}} \in \mathbb{R}^{U \times V}$:

$$\mathbf{A}(\mathbf{v}) = \mathbf{A}^{\text{uv}}(\mathbf{v}^{\text{uv}}). \tag{7}$$

Usually, it involves sub-pixel sampling via bilinear interpolation:

$$\mathbf{A}(\mathbf{v}) = \sum_{\substack{u' \in \{\lfloor u \rfloor, \lceil u \rceil\} \\ v' \in \{\lfloor v \rfloor, \lceil v \rceil\}}} \mathbf{A}^{\text{uv}}(u', v')(1 - |u - u'|)(1 - |v - v'|), \tag{8}$$

where $\mathbf{v}^{\text{uv}} = (u, v)$ is the UV space projection of $\mathbf{v}$ via Eqn. 6.

Albedo information is naturally expressed in the UV space but spatial data can be embedded in the same space as well. Here, a 3D facial mesh can be represented as a 2D image with three channels, one for each spatial dimension x, y and z. Fig 4 gives an example of this UV space shape representation $\mathbf{S}^{\text{uv}} \in \mathbb{R}^{U \times V}$.

Representing 3D face shape in UV space allow us to use a CNN for shape decoder D_S instead of using a multi-layer perceptron (MLP) as in our preliminary version [16]. Avoiding using wide fully-connected layers allow us to use deeper network for D_S, potentially model more complex shape variations. This results in better fitting results as being demonstrated in our experiment (Sec. 4.1.2).

Also, it is worth to note that different from our preliminary version [16] where the reference UV space, for texture, is build upon projection of the mean shape with neutral expression; in this version, the reference shape used has the mouth open. This change helps the network to avoid learning a large gradient near the two lips' borders in the vertical direction when the mouth is open.

To regress these 2D representation of shape and albedo, we can employ CNNs as shape and albedo networks respectively. Specifically, D_S, D_A are CNN constructed by multiple fractionally-strided convolution layers. After each convolution is batchnorm and eLU activation, except the last convolution layers of encoder and decoders. The output layer has a $tanh$ activation to constraint the output to be in the range of $[-1, 1]$. The detailed network architecture is presented in Tab. 1.

3.2.3 *In-Network Physically-Based Face Rendering*

To reconstruct a face image from the albedo $\mathbf{A}$, shape $\mathbf{S}$, lighting parameter $\mathbf{L}$, and projection parameter $\mathbf{m}$, we define a rendering layer $\mathcal{R}(\mathbf{m}, \mathbf{L}, \mathbf{S}, \mathbf{A})$ to render a face image from the above parameters. This is accomplished in three steps, as shown in Fig. 5. Firstly, the facial texture is computed using the albedo $\mathbf{A}$ and the surface normal map of the rotated shape $N(\mathbf{V}) = N(\mathbf{m}, \mathbf{S})$. Here, following [58], we assume distant illumination and a purely *Lambertian* surface reflectance. Hence the incoming radiance can be approximated using spherical harmonics (SH) basis functions $H_b : \mathbb{R}^3 \to \mathbb{R}$, and controlled by coefficients $\mathbf{L}$. Specifically, the texture in UV space $\mathbf{T}^{\text{uv}} \in \mathbb{R}^{U \times V}$ is composed of albedo $\mathbf{A}^{\text{uv}}$ and shading $\mathbf{C}^{\text{uv}}$:

$$\mathbf{T}^{\text{uv}} = \mathbf{A}^{\text{uv}} \odot \mathbf{C}^{\text{uv}} = \mathbf{A}^{\text{uv}} \odot \sum_{b=1}^{B^2} L_b H_b(N(\mathbf{m}, \mathbf{S}^{\text{uv}})), \tag{9}$$

where B is the number of spherical harmonics bands. We use $B = 3$, which leads to $B^2 = 9$ coefficients in $\mathbf{L}$ for each of three color channels. Secondly, the 3D shape/mesh $\mathbf{S}$ is projected to the image plane via Eqn. 4. Finally, the 3D mesh is then rendered using a Z-buffer renderer, where each pixel is associated with a single triangle of the mesh,

$$\begin{aligned} \hat{\mathbf{I}}(m, n) &= \mathcal{R}(\mathbf{m}, \mathbf{L}, \mathbf{S}^{\text{uv}}, \mathbf{A}^{\text{uv}})_{m,n} \\ &= \mathbf{T}^{\text{uv}}\Big(\sum_{\mathbf{v}_i \in \Phi^{\text{uv}}(g,m,n)} \lambda_i \mathbf{v}_i \Big), \end{aligned} \tag{10}$$

where $\Phi(g, m, n) = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ is an operation returning three vertices of the triangle that encloses the pixel (m, n) after projection g; $\Phi^{\text{uv}}(g, m, n)$ is the same operation with resultant vertices mapped into the referenced UV space using Eqn. 6. In order to handle occlusions, when a single pixel resides in more than one triangle, the triangle that is closest to the image plane is selected. The final location of each pixel is determined by interpolating the location of three vertices via barycentric coordinates $\{\lambda_i\}_{i=1}^3$.

There are alternative designs to our rendering layer. If the texture representation is defined per vertex, as in Fig. 3(a), one may warp the input image $\mathbf{I}_i$ onto the vertex space of the 3D shape $\mathbf{S}$, whose distance to the per-vertex texture representation can form a reconstruction loss. This design is adopted by the recent work of [26], [48]. In comparison, our rendered image is defined on a 2D grid while the alternative is on top of the 3D mesh. As a result, our rendered image can enjoy the convenience of applying the perceptual loss or adversarial loss, which is shown to be critical in improving the quality of synthetic texture. Another design for rendering layer is image warping based on the spline interpolation, as in [59]. However, this warping is continuous: every pixel in the input will map to the output. Hence this warping operation fails in the occluded region. As a result, Cole *et al.* [59] limit their scope to only synthesizing frontal-view faces by warping from normalized faces.

The CUDA implementation of our rendering layer is publicly available at https://github.com/tranluan/Nonlinear_Face_3DMM.

TABLE 1: The architectures of E, D_A and D_S networks.

E			D_A/D_S		
Layer	Filter/Stride	Output Size	Layer	Filter/Stride	Output Size
			FC		6×7×320
Conv11	7×7/2	112×112×32	FConv52	3×3/2	12×14×160
Conv12	3×3/1	112×112×64	FConv51	3×3/1	12×14×256
Conv21	3×3/2	56×56×64	FConv43	3×3/2	24×28×256
Conv22	3×3/1	56×56×64	FConv42	3×3/1	24×28×128
Conv23	3×3/1	56×56×128	FConv41	3×3/1	24×28×192
Conv31	3×3/2	28×28×128	FConv33	3×3/2	48×56×192
Conv32	3×3/1	28×28×96	FConv32	3×3/1	48×56×96
Conv33	3×3/1	28×28×192	FConv31	3×3/1	48×56×128
Conv41	3×3/2	14×14×192	FConv23	3×3/2	96×112×128
Conv42	3×3/1	14×14×128	FConv22	3×3/1	96×112×64
Conv43	3×3/1	14×14×256	FConv21	3×3/1	96×112×64
Conv51	3×3/2	7×7×256	FConv13	3×3/2	192×224×64
Conv52	3×3/1	7×7×160	FConv12	3×3/1	192×224×32
Conv53	3×3/1	7×7×(l_S+l_A+64)	FConv11	3×3/1	192×224×3
AvgPool	7×7/1	1×1×(l_S+l_A+64)			
FC_m	64×6	6			
FC_L	64×27	27			

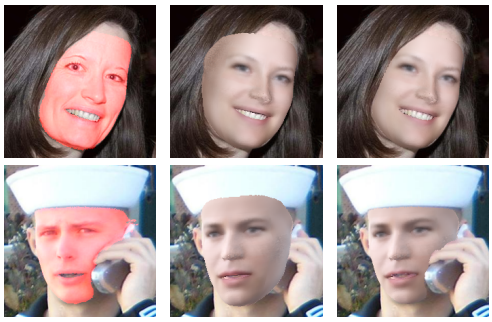


Fig. 6: Rendering with segmentation masks. Left to right: segmentation results, naive rendered images, occulusion-aware rendered images.

3.2.4 Occlusion-aware Rendering

Very often, in-the-wild faces are occluded by glasses, hair, hands, etc. Trying to reconstruct abnormal occluded regions could make the model learning more difficult or result in an model with external occlusion baked in. Hence, we propose to use a segmentation mask to exclude occluded regions in the rendering pipeline:

$$\hat{\mathbf{I}} \leftarrow \hat{\mathbf{I}} \odot \mathbf{M} + \mathbf{I} \odot (1 - \mathbf{M}). \tag{11}$$

As a result, these occluded regions won't affect our optimization process. The foreground mask $\mathbf{M}$ is estimated using the segmentation method given by Nirkin*et al.* [60]. Examples of segmentation masks and rendering results can be found in Fig. 6.

3.2.5 Model Learning

The entire network is end-to-end trained to reconstruct the input images, with the loss function:

$$L = L_{\text{rec}} + \lambda_L L_{\text{L}} + \lambda_{\text{reg}} L_{\text{reg}}, \tag{12}$$

where the reconstruction loss L_{rec} enforces the rendered image $\hat{\mathbf{I}}$ to be similar to the input $\mathbf{I}$, the landmark loss L_L enforces geometry constraint, and the regularization loss L_{reg} encourages plausible solutions.

Reconstruction Loss. The main objective of the network is to reconstruct the original face via disentangle representation. Hence, we enforce the reconstructed image to be similar to the original input image:

$$L^i_{\text{rec}} = \frac{1}{|\mathcal{V}|} \sum_{q \in \mathcal{V}} ||\hat{\mathbf{I}}(q) - \mathbf{I}(q)||_2 \tag{13}$$

where $\mathcal{V}$ is the set of all pixels in the images covered by the estimated face mesh. There are different norms can be used to measure the closeness. To better handle outliers, we adopt the robust $l_{2,1}$, where the distance in the 3D RGB color space is based on l_2 and the summation over all pixels enforces sparsity based on l_1-norm [7], [61].

To improve from blurry reconstruction results of l_p losses, in our preliminary work [16], thanks for our rendering layer, we employ adversarial loss to enhance the image realism. However, adversarial objective only encourage the reconstruction to be close to the real image distribution but not necessary the input image. Also, it's known to be not stable to optimize. Here, we propose to use a perceptual loss to enforce the closeness between images $\hat{\mathbf{I}}$ and $\mathbf{I}$, which overcomes both of adversarial loss's weaknesses. Besides encouraging the pixels of the output image $\hat{\mathbf{I}}$ to exactly match the pixels of the input $\mathbf{I}$, we encourage them to have similar feature representations as computed by the loss network φ.

$$L^f_{\text{rec}} = \frac{1}{|\mathcal{C}|} \sum_{j \in \mathcal{C}} \frac{1}{W_j H_j C_j} ||\varphi_j(\hat{\mathbf{I}}) - \varphi_j(\mathbf{I})||_2^2. \tag{14}$$

We choose VGG-Face [62] as our φ to leverage its face-related features and also because of simplicity. The loss is summed over $\mathcal{C}$, a subset of layers of φ. Here $\varphi_j(\mathbf{I})$ is the activations of the j-th layer of φ when processing the image $\mathbf{I}$ with dimension $W_j \times H_j \times C_j$. This feature reconstruction loss is one of perceptual losses widely used in different image processing tasks [63].

The final reconstruction loss is a weighted sum of two terms:

$$L_{\text{rec}} = L^i_{\text{rec}} + \lambda_f L^f_{\text{rec}}. \tag{15}$$

Sparse Landmark Alignment. To help achieving better model fitting, which in turn helps to improve the model learning itself, we employ the landmark alignment loss, measuring Euclidean distance between estimated and groundtruth landmarks, as an auxiliary task,

$$L_{\text{L}} = \left\| M(\mathbf{m}) * \begin{bmatrix} \mathbf{S}(:, \mathbf{d}) \\ \mathbf{1} \end{bmatrix} - \mathbf{U} \right\|_2^2, \tag{16}$$

where $\mathbf{U} \in \mathbb{R}^{2\times 68}$ is the manually labeled 2D landmark locations, $\mathbf{d}$ is a constant 68-dim vector storing the indexes of 68 3D vertices corresponding to the labeled 2D landmarks. Different from traditional face alignment work where the shape bases are fixed, our work jointly learns the bases functions (i.e., the shape decoder D_S) as well. Minimizing the landmark loss while updating D_S only moves a tiny subsets of vertices. If the shape $\mathbf{S}$ is represented as a vector and D_S is a MLP consisting of fully connected layers, vertices are independent. Hence L_{L} only adjusts 68 vertices. In case $\mathbf{S}$ is represented in the UV space and D_S is a CNN, local neighbor region could also be modified. In both cases, updating D_S based on L_{L} only moves a subsets of vertices, which could lead to implausible shapes. Hence, when optimizing the landmark loss, we fix the decoder D_S and only update the encoder.

Also, it is worth noting that different from some related work [64], our network only requires ground truth landmarks during training. It is able to predict landmarks via $\mathbf{m}$ and $\mathbf{S}$ during the test time.

Regularizations. To ensure plausible reconstruction, we add a few regularization terms:

Albedo Symmetry As the face is symmetry, we enforce the albedo symmetry constraint,

$$L_{\text{sym}} = \left\| \mathbf{A}^{\text{uv}} - \text{flip}(\mathbf{A}^{\text{uv}}) \right\|_1 . \tag{17}$$

Employing on 2D albedo, this constraint can be easily implemented via a horizontal image flip operation flip().

Albedo Constancy Using symmetry constraint can help to correct the global shading. However, symmetrical details, i.e., dimples, can still be embedded in the albedo channel.

To further remove shading from the albedo channel, following Retinex theory [22] which assumes albedo to be piecewise constant, we enforce sparsity in two directions of its gradient, similar to [65], [66]:

$$L_{\text{const}} = \sum_{\mathbf{v}_j^{\text{uv}} \in \mathcal{N}_i} \omega(\mathbf{v}_i^{\text{uv}}, \mathbf{v}_j^{\text{uv}}) \left\| \mathbf{A}^{\text{uv}}(\mathbf{v}_i^{\text{uv}}) - \mathbf{A}^{\text{uv}}(\mathbf{v}_j^{\text{uv}}) \right\|_2^p , \tag{18}$$

where $\mathcal{N}_i$ denotes a set of 4-pixel neighborhood of pixel $\mathbf{v}_i^{\text{uv}}$. With the assumption that pixels with the same chromaticity (i.e., $\mathbf{c}(x) = \mathbf{I}(x)/|\mathbf{I}(x)|$) are more likely to have the same albedo, we set the constant weight $\omega(\mathbf{v}_i^{\text{uv}}, \mathbf{v}_j^{\text{uv}}) = \exp\left(-\alpha \left\| \mathbf{c}(\mathbf{v}_i^{\text{uv}}) - \mathbf{c}(\mathbf{v}_j^{\text{uv}}) \right\| \right)$, where the color is referenced from the input image using the current estimated projection. Following [65], we set $\alpha = 15$ and $p = 0.8$ in our experiment.

Shape Smoothness For shape component, we impose the smoothness by adding the Laplacian regularization on the vertex locations for the set of all vertices.

$$L_{\text{smooth}} = \sum_{\mathbf{v}_i^{\text{uv}} \in \mathbf{S}^{\text{uv}}} \left\| \mathbf{S}^{\text{uv}}(\mathbf{v}_i^{\text{uv}}) - \frac{1}{|\mathcal{N}_i|} \sum_{\mathbf{v}_j^{\text{uv}} \in \mathcal{N}_i} \mathbf{S}^{\text{uv}}(\mathbf{v}_j^{\text{uv}}) \right\|_2 . \tag{19}$$

Intermediate Semi-Supervised Training. Fully unsupervised training using only the reconstruction and adversarial loss on the rendered images could lead to a degenerate solution, since the initial estimation is far from ideal to render meaningful images. Therefore, we introduce intermediate loss functions to guide the training in the early iterations.

With the face profiling technique, Zhu *et al.* [32] expand the 300W dataset [67] into $122,450$ images with fitted 3DMM shapes $\widetilde{\mathbf{S}}$ and projection parameters $\widetilde{\mathbf{m}}$. Given $\widetilde{\mathbf{S}}$ and $\widetilde{\mathbf{m}}$, we create the pseudo groundtruth texture $\widetilde{\mathbf{T}}$ by referring every pixel in the UV space back to the input image, i.e., the backward of our rendering layer. With $\widetilde{\mathbf{m}}$, $\widetilde{\mathbf{S}}$, $\widetilde{\mathbf{T}}$, we define our intermediate loss by:

$$L_0 = L_{\text{S}} + \lambda_T L_{\text{T}} + \lambda_m L_{\text{m}} + \lambda_L L_{\text{L}} + \lambda_{\text{reg}} L_{\text{reg}}, \tag{20}$$

where:

$$L_{\text{S}} = \left\| \mathbf{S} - \widetilde{\mathbf{S}} \right\|_2^2 , \tag{21}$$

$$L_{\text{T}} = \left\| \mathbf{T} - \widetilde{\mathbf{T}} \right\|_1 , \tag{22}$$

$$L_{\text{m}} = \left\| \mathbf{m} - \widetilde{\mathbf{m}} \right\|_2^2 . \tag{23}$$

It's also possible to provide pseudo groundtruth to the SH coefficients $\mathbf{L}$ and followed by albedo $\mathbf{A}$ using least square optimization with a constant albedo assumption, as has been done in [58], [66]. However, this estimation is not reliable for in-the-wild images with occlusion regions. Also empirically, with proposed regularizations, the model is able to explore plausible solutions for these components by itself. Hence, we decide to refrain from supervising $\mathbf{L}$ and $\mathbf{A}$ to simplify our pipeline.

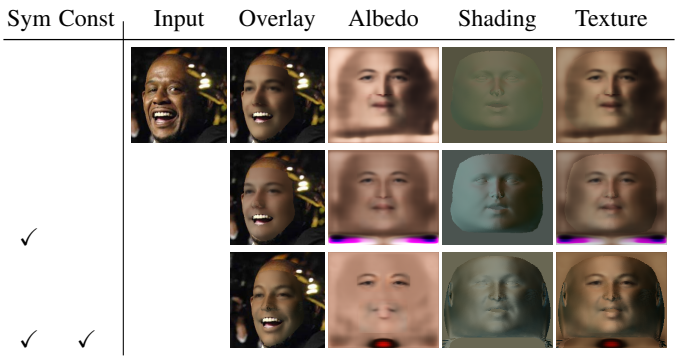


Fig. 7: Effect of albedo regularizations: albedo symmetry (sym) and albedo constancy (const). When there is no regularization being used, shading is mostly baked into the albedo. Using the symmetry property helps to resolve the global lighting. Using constancy constraint futher removes shading from the albedo channel, which results in a better 3D shape.

Due to the pseudo groundtruth, using L_0 may run into the risk that our solution learns to mimic the linear model. Thus, we switch to the loss of Eqn. 12 after L_0 converges. Note that the estimated groundtruth of $\widetilde{\mathbf{m}}$, $\widetilde{\mathbf{S}}$, $\widetilde{\mathbf{T}}$ and the landmarks are the only supervision used in our training, for which our learning is considered as *weakly* supervised.

4 Experimental Results

The experiments study three aspects of the proposed nonlinear 3DMM, in terms of its expressiveness, representation power, and applications to facial analysis. Using facial mesh triangle definition by Basel Face Model (BFM) [11], we train our 3DMM using 300W-LP dataset [32], which contains $122,450$ in-the-wild face images, in a wide pose range from $-90°$ to $90°$. Images are loosely square croped around the face and scale to 256×256. During training, images of size 224×224 are randomly cropped from these images to introduce translation variations.

The model is optimized using Adam optimizer with a learning rate of 0.001 in both training stages. We set the following parameters: $Q = 53,215$, $U = 192, V = 224$, $l_S = l_T = 160$. λ values are set to make losses to have similar magnitudes.

4.1 Ablation Study

4.1.1 Effect of Regularization

Albedo Regularization. In this work, to regularize albedo learning, we employ two constraints to efficiently remove shading from albedo namely albedo symmetry and constancy. To demonstrate the effect of these regularization terms, we compare our full model with its partial variants: one without any albedo requlariztion and one with the symmetry constraint only. Fig. 7 shows visual comparison of these models. Learning without any constraints results in the lighting is totally explained by the albedo, meanwhile is the shading is almost constant (Fig. 7(a)). Using symmetry help to correct the global lighting. However, symmetric geometry details are still baked into the albedo (Fig. 7(b)). Enforcing albedo constancy helps to further remove shading from it (Fig. 7(c)). Combining these two regularizations helps to learn plausible albedo and lighting, which improves the shape estimation.

Shape Smoothness Regularization. We also evaluate the need in shape regularization. Fig. 8 shows visual comparisons between

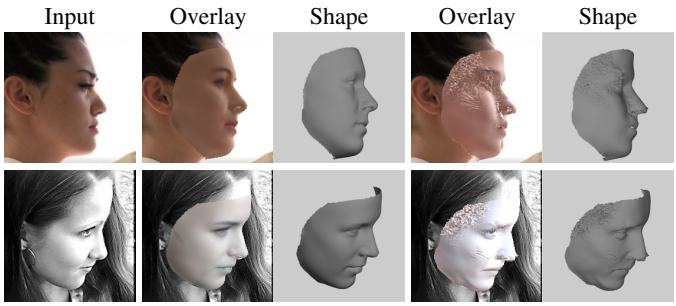


Fig. 8: Effect of shape smoothness regularization.

TABLE 2: Face alignment performance on ALFW2000.

Method	Lighting	UV shape	NME
Our [16]			4.70
Our	✓		4.30
Our	✓	✓	**4.12**

our model and its variant without the shape smoothness constraint. Without the smoothness term the learned shape becomes noisy especially on two sides of the face. The reason is that, the hair region is not completely excluded during training because of imprecise segmentation estimation.

4.1.2 Modeling Lighting and Shape Representation

In this work, we make two major algorithmic differences with our preliminary work [16]: incorporating lighting into the model and changing the shape representation.

Our previous work [16] models the texture directly, while this work disentangles the shading from the albedo. As argued, modeling the lighting should have a positive impact on shape learning. Hence we compare our models with results from [16] in face alignment task.

Also, in our preliminary work [16], as well as in traditional 3DMM, shape is represented as a vector, where vertices are independent. Despite this shortage, this approach has been widely adopted due to its simplicity and sampling efficiency. In this work, we explore an alternative to this representation: represent the 3D shape as a position map in the 2D UV space. This representation has three channels: one for each spatial dimension. This representation maintains the spatial relation among facial mesh's vertices. Also, we can use CNN as the shape decoder replacing an expensive MLP. Here we also evaluate the performance gain by switching to this representation.

Tab. 2 reports the performance on the face alignment task of different variants. As a result, modeling lighting helps to reduce the error from 4.70 to 4.30. Using the 2D representation, with the convenience of using CNN, the error is further reduced to 4.12.

4.1.3 Comparison to Autoencoders

We compare our model-based approach with a convolutional autoencoder in Fig. 9. The autoencoder network has a similar depth and model size as ours. It gives blurry reconstruction results as the dataset contain large variations on face appearance, pose angle and even diversity background. Our model-based approach obtains sharper reconstruction results and provides semantic parameters allowing access to different components including 3D shape, albedo, lighting and projection matrix.

Fig. 9: Comparison to convolutional autoencoders (AE). Our approach produces results of higher quality. Also it provides access to the 3D facial shape, albedo, lighting, and projection matrix.

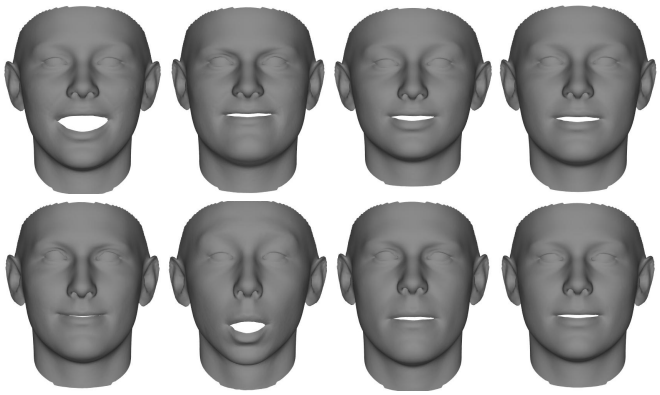


Fig. 10: Each column shows shape changes when varying one element of $\mathbf{f}_S$, by 10 times standard deviations, in opposite directions. Ordered by the magnitude of shape changes.

Fig. 11: Each column shows albedo changes when varying one element of $\mathbf{f}_A$ in opposite directions.

4.2 Expressiveness

Exploring feature space. We feed the entire CelebA dataset [68] with $\sim$200k images to our network to obtain the empirical distribution of our shape and texture parameters. By varying the mean parameter along each dimension proportional to its standard deviation, we can get a sense how each element contribute to the final shape and texture. We sort elements in the shape parameter $\mathbf{f}_S$ based on their differences to the mean 3D shape. Fig. 10 shows four examples of shape changes, whose differences rank No.1, 40, 80, and 120 among 160 elements. Most of top changes are expression related. Similarly, in Fig. 11, we visualize different texture changes by adjusting only one element of $\mathbf{f}_A$ off the mean parameter $\bar{\mathbf{f}}_A$. The elements with the same 4 ranks as the shape counterpart are selected.

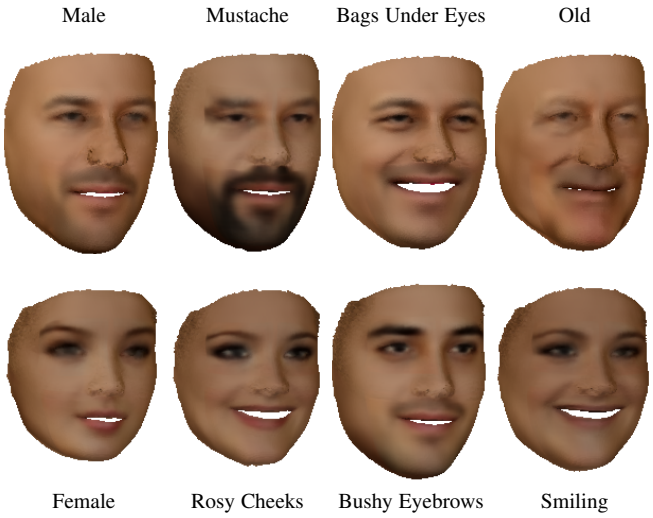


Fig. 12: Nonlinear 3DMM generates shape and albedo embedded with different attributes.

TABLE 3: Quantitative comparison of texture representation power (Average reconstruction error on non-occluded face portion.)

Method	Linear	Nonlinear w. Grad De.	Nonlinear w. Network
L_1	0.097	0.053	0.057

Attribute Embedding. To better understand different shape and albedo instances embedded in our two decoders, we dig into their attribute meaning. For a given attribute, e.g., male, we feed images with that attribute $\{\mathbf{I}_i\}_{i=1}^n$ into our encoder E to obtain two sets of parameters $\{\mathbf{f}_S^i\}_{i=1}^n$ and $\{\mathbf{f}_A^i\}_{i=1}^n$. These sets represent corresponding empirical distributions of the data in the low dimensional spaces. Computing the mean parameters $\bar{\mathbf{f}}_S, \bar{\mathbf{f}}_A$ and feed into their respective decoders, also using the mean lighting parameter, we can reconstruct the mean shape and texture with that attribute. Fig. 12 visualizes the reconstructed textured 3D mesh related to some attributes. Differences among attributes present in both shape and texture. Here we can observe the power of our nonlinear 3DMM to model small details such as "bag under eyes", or "rosy cheeks", etc.

4.3 Representation Power

We compare the representation power of the proposed nonlinear 3DMM vs. traditional linear 3DMM.

Albedo. Given a face image, assuming we know the groundtruth shape and projection parameters, we can unwarp the texture into the UV space, as we generate "pseudo groundtruth" texture in the weakly supervision step. With the groundtruth texture, by using gradient descent, we can jointly estimate, a lighting parameter $\mathbf{L}$ and an albedo parameter $\mathbf{f}_A$ whose decoded texture matches with the groundtruth. Alternatively, we can minimize the reconstruction error in the image space, through the rendering layer with the groundtruth $\mathbf{S}$ and $\mathbf{m}$. Empirically, two methods give similar performances but we choose the first option as it involves only one warping step, instead of doing rendering in every optimization iteration. For the linear model, we use the fitting results of Basel texture and Phong illumination model [56] given by [32]. As in Fig. 13, our nonlinear texture is closer to the groundtruth than the linear model. This is expected since the linear model is trained with controlled images. Quantitatively, our nonlinear model has significantly lower averaged L_1 reconstruction error than the linear model (0.053 vs. 0.097, as in Tab. 3).

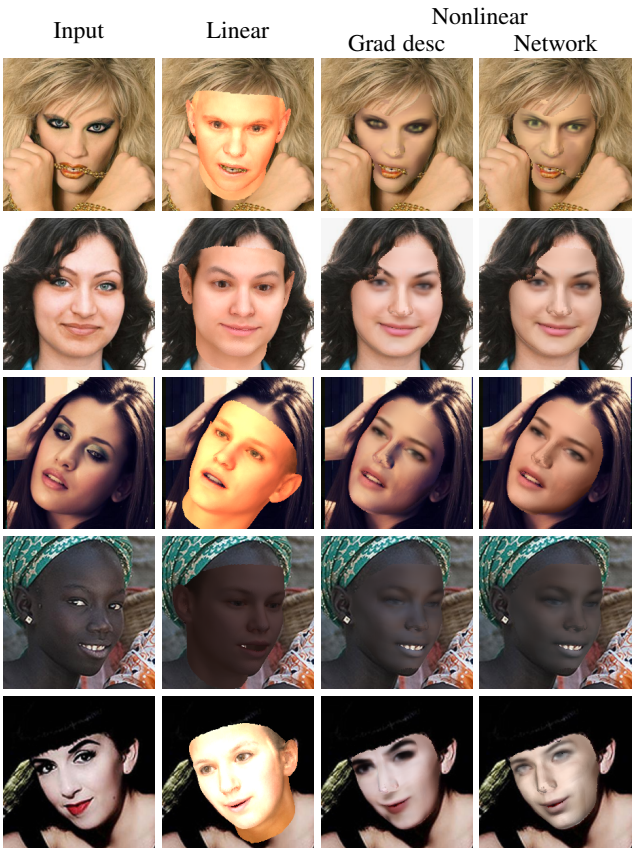


Fig. 13: Texture representation power comparison. Our nonlinear model can better reconstruct the facial texture.

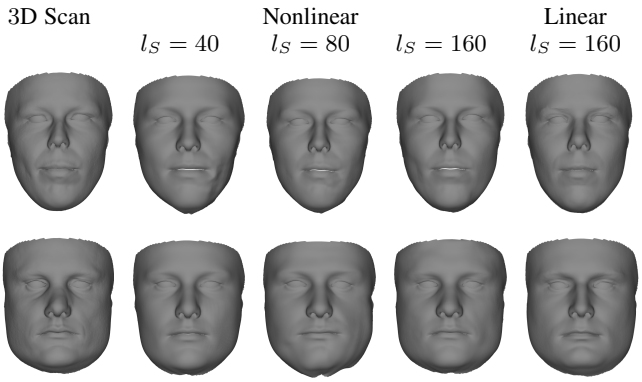


Fig. 14: Shape representation power comparison.

TABLE 4: 3D scan reconstruction comparison (NME).

l_S	40	80	160
Linear	0.0321	0.0279	0.0241
Nonlinear [16]	0.0277	0.0236	0.0196
Nonlinear	0.0268	0.0214	**0.0146**

3D Shape. We also compare the power of nonlinear and linear

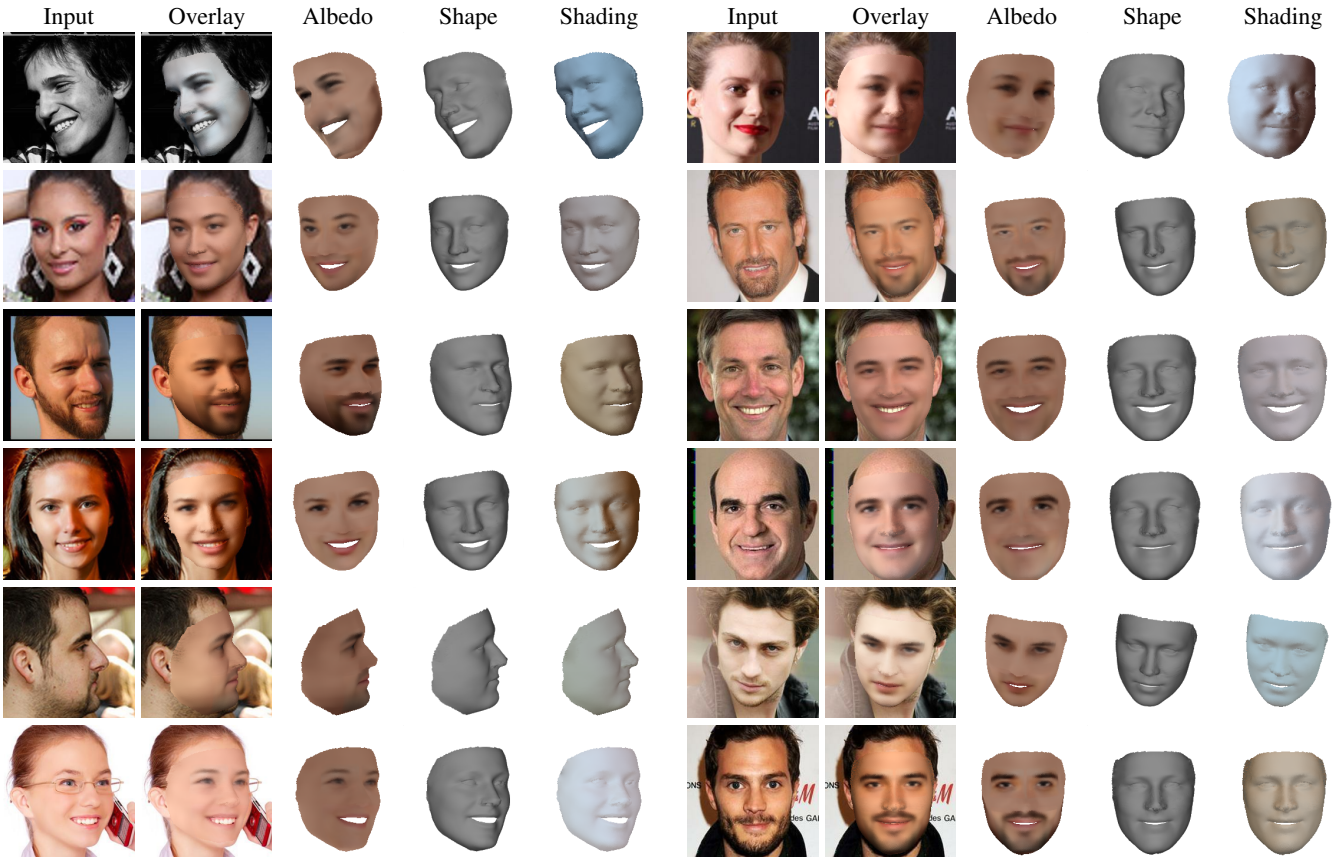


Fig. 15: 3DMM fits to faces with diverse skin color, pose, expression, lighting, facial hair, and faithfully recovers these cues. Left half shows results from AFLW2000 dataset, right half shows results from CelebA.

3DMMs in representing real-world 3D scans. We compare with BFM [11], the most commonly used 3DMM at present. We use ten 3D face scans provided by [11], which are not included in the training set of BFM. As these face meshes are already registered using the same triangle definition with BFM, no registration is necessary. Given the groundtruth shape, by using gradient descent, we can estimate a shape parameter whose decoded shape matches the groundtruth. We define matching criterion on both vertex distances and surface normal direction. This empirically improves fidelity of final results compared to only optimizing vertex distances. Also, to emphasize the compactness of nonlinear models, we train different models with different latent space sizes. Fig. 14 shows the visual quality of two models' reconstruction. Our reconstructions closely match the face shapes details.

To quantify the difference, we use NME, averaged per-vertex errors between the recovered and groundtruth shapes, normalized by inter-ocular distances. Our nonlinear model has a significantly smaller reconstruction error than the linear model, 0.0146 vs. 0.0241 (Tab. 4). Also, the nonlinear models are more compact. They can achieve similar performances as linear models whose latent spaces sizes doubled.

4.4 Applications

Having shown the capability of our nonlinear 3DMM (i.e., two decoders), now we demonstrate the applications of our entire network, which has the additional encoder. Many applications of 3DMM are centered on its ability to fit to 2D face images. Similar to linear 3DMM, our nonlinear 3DMM can be utilized for model fitting, which decomposes a 2D face into its shape, albedo and lighting. Fig. 15 visualizes our 3DMM fitting results on AFLW2000 and CelebA dataset. Our encoder estimates the shape $\mathbf{S}$, albedo $\mathbf{A}$ as well as lighting $\mathbf{L}$ and projection parameter $\mathbf{m}$. We can recover personal facial characteristic in both shape and albedo. Our albedo can present facial hair, which is normally hard to be recovered by linear 3DMM.

TABLE 5: Face alignment performance on ALFW2000.

Method	Linear	SDM [69]	3DDFA [32]	DeFA [70]	Our
NME	5.61	6.12	5.42	4.50	**4.12**

4.4.1 2D Face Alignment

Face alignment is a critical step for many facial analysis tasks such as face recognition [71], [72]. With enhancement in the modeling, we hope to improve this task (Fig. 16). We compare face alignment performance with state-of-the-art methods, SDM [69], 3DDFA [32] and DeFA [70], on AFLW2000 dataset. The accuracy is evaluated by the Normalized Mean Error (NME), the average of visible landmark error normalized by the bounding box size [31].

Here, current state-of-the-art DeFA [70] is trained in a corpus consisting of five datasets including our training set of 300W-LP and employs multiple constrains namely landmark, contour, SIFT features. 3DDFA [32] is a cascade of CNNs that iteratively refines its estimation in multiple steps, meanwhile ours is a single-pass of E and D_S. Comparing to our method, DeFA has an advantages

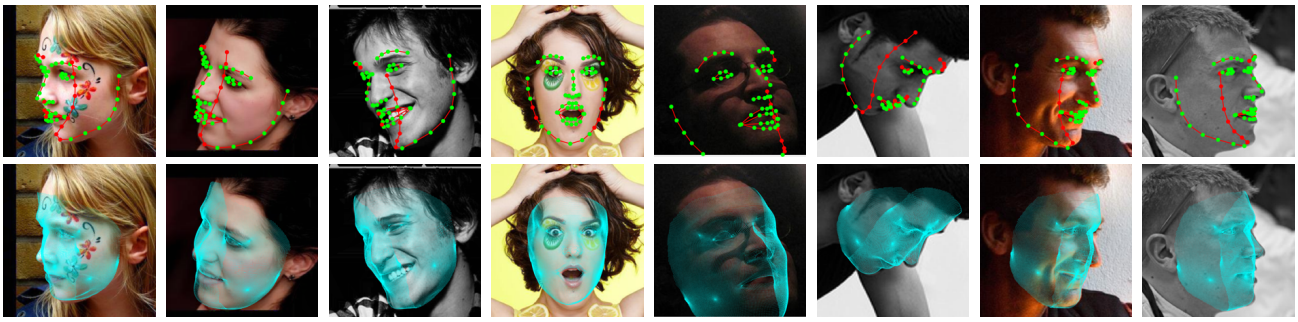


Fig. 16: Our 2D face alignment results. Invisible landmarks are marked as red. We can well handle extreme pose, lighting and expression.

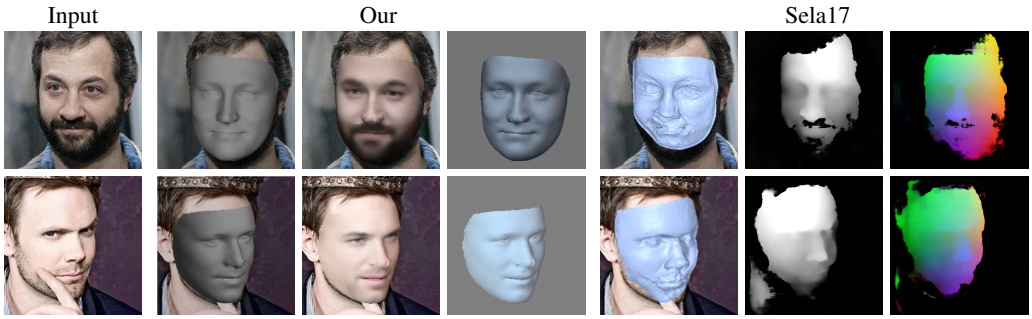


Fig. 17: 3D reconstruction results comparison to Sela *et al.* [50]. Besides showing the shape, we also show their estimated depth and correspondence map. Facial hair or occlusion can cause serious problems in their output maps.

on training set, 3DDFA has an edge on the network architecture. However, by jointly learning model fitting with 3DMM, our network can significantly surpass their performance, as in Tab. 5. Another perspective is that in conventional 3DMM fitting [32], the texture is used as the input to regress the shape parameter, while ours adopts an analysis-by-synthesis scheme and texture is the output of the synthesis. Further, for a more fair comparison of nonlinear vs. linear models, we train an encoder with the same architecture as our E, whose output parameter will multiple with the linear shape bases $\mathbf{G}$, and train with the landmark loss function (Eqn. 16). Again we observe the higher error from the linear model-based fitting.

4.4.2 3D Face Reconstruction

We compare our approach to three recent representative face reconstruction work: 3DMM fitting networks learned in supervised (Sela *et al.* [42]) or unsupervised fashion (Tewari *et al.* [48]) and also a non-3DMM approach (Jackson *et al.* [73]).

The high-quality 3D reconstruction work by Richardson *et al.* [41], [50], Sela *et al.* [42] obtains impressive results on adding fine-level details to the face shape when images are within the span of the used synthetic training corpus or the employed 3DMM model. However, their performance significantly degrades when dealing with variations not in its training data span, e.g., facial hair. Our approach is not only robust to facial hair and make-up, but also automatically learns to reconstruct such variations based on the jointly learned model. We provide comparisons with them in Fig. 17, using the code provided by the author.

The current state-of-art 3D monocular face reconstruction method by Sela *et al.* [42] consisting of three steps: an image-to-image network estimating a depth map and a correspondence map, non-rigid registration and a fine detail reconstruction. Their image-to-image network is trained on synthetic data generated by

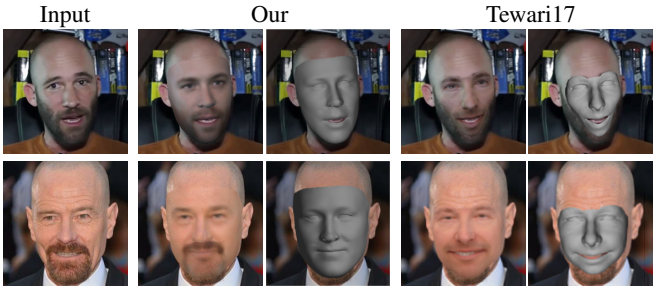


Fig. 18: 3D reconstruction results comparison to Tewari *et al.* [48] on 300VW dataset [74] (first row) and CelebA [68] (second row). Their reconstructed shapes suffer from the surface shrinkage when dealing with challenging texture, i.e., facial hair. Meanwhile, our nonlinear model is more robust to these variations.

the linear model. Besides domain gap between synthetic and real images, this network faces a more serious problem of lacking facial hair in the low-dimension texture subspace of the linear model. This network's output tends to ignore these unexplainable region (Fig. 17), which leads to failure in later steps. Our network is more robust in handing these in-the-wild variations. Furthermore, our approach is orthogonal to Sela *et al.* [42]'s fine detail reconstruction module or Richardson *et al.* [50]'s finenet. Employing these refinement on top of our fitting could lead to promising further improvement.

MoFA, the monocular reconstruction work by Tewari *et al.* [48], is relevant to us as they also learn to fit 3DMM in an unsupervised fashion. Even being trained on in-the-wild images, their method is still limited to the linear bases. Hence there reconstructions suffer the surface shrinkage when dealing with challenging texture, i.e., facial hair (Fig. 18). Our

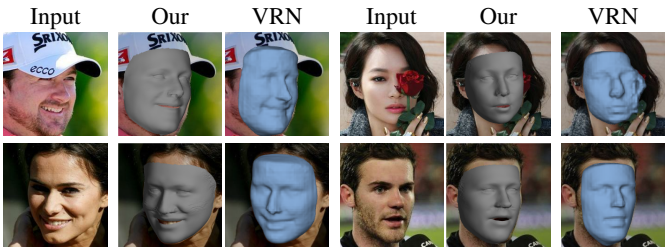


Fig. 19: 3D reconstruction results comparison to VRN by Jackson *et al.* [73] on CelebA dataset. Volumetric shape representation results in non-smooth 3D shape and loses correspondence between reconstructed shapes.

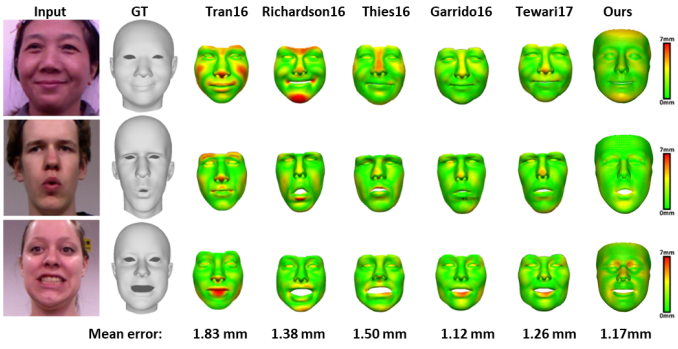


Fig. 20: Quantitative evaluation of 3D reconstruction. We obtain a low error that is comparable to optimization-based methods.

network faithfully models these in-the-wild texture, which leads to better 3D shape reconstruction.

We also compare our approach with a non-3DMM apporach VRN by Jackson *et al.* [73]. To avoid using low-dimension subspace of the linear 3DMM, it directly regresses a 3D shape volumetric representation via an encoder-decoder network with skip connection. This potentially helps the network to explore a larger solution space than the linear model, however with a cost of losing correspondence between facial meshes. Fig. 19 shows 3D reconstruction visual comparison between VRN and ours. In general, VRN robustly handles in-the-wild texture variations. However, because of the volumetric shape representation, the surface is not smooth and is partially limited to present medium-level details as ours. Also, our model further provides projection matrix, lighting and albedo, which is applicable for more applications.

Following the same setting in [48], we also quantitatively compare our method with prior works on 9 subjects of FaceWarehouse database [12]. Visual and quantitative comparisons are shown in Fig. 20. We achieve on-par results with Garrido *et al.* [64], an offline optimization method, while surpass all other regression methods [24], [48], [50].

4.4.3 Face editing

Decomposing face image into individual components give us ability to edit the face by manipulating any component. Here we show two examples of face editing using our model.

Relighting. First we show an application to replacing the lighting of a target face image using lighting from a source face (Fig. 21). After estimating the lighting parameters $\mathbf{L}_{\text{source}}$ of the source image, we render the transfer shading using the target shape $\mathbf{S}_{\text{target}}$ and the source lighting $\mathbf{L}_{\text{source}}$. This transfer shading can be used to

Fig. 21: Lighting transfer results. We transfer the lighting of source images (first row) to target images (first column). We have similar performance compare to the state-of-the-art method of Shu *et al.* [75] despite being orders of magnitude faster (150 ms vs. 3 min per image).

replace the original source shading. Alternatively, value of $\mathbf{L}_{\text{source}}$ can be arbitrarily chosen based on the SH lighting model, without the need of source images. Also, here we use the original texture instead of the output of our decoder to maintain image details.

Attribute Manipulation. Given faces fitted by 3DMM model, we can edit images by naive modifying one or more elements in the albedo or shape representation. More interestingly, we can even manipulate the semantic attribute, such as growing beard, smiling, etc. The approach is similar to learning attribute embedding in Sec. 4.2. Assuming, we would like to edit appearance only. For a given attribute, e.g., beard, we feed two sets of images with and without that attribute $\{\mathbf{I}_i^p\}_{i=1}^n$ and $\{\mathbf{I}_i^n\}_{i=1}^n$ into our encoder to obtain two average parameters $\mathbf{f}_A^p$ and $\mathbf{f}_A^n$. Their difference $\mathbf{\Delta f}_A = \mathbf{f}_A^p - \mathbf{f}_A^n$ is the direction to move from the distribution of negative images to positive ones. By adding $\mathbf{\Delta f}_A$ with different magnitudes, we can generate modified images with different degree of changes. To achieve high-quality editing with identity-preserved, the final editing result is obtained by adding the residual, the different between the modified image and our reconstruction, to the original input image. This is a critical difference to Shu *et al.* [66] to improve results quality (Fig. 22).

4.5 Runtime

In this section, we compare running time for multiple 3D reconstruction approaches. Since different methods implemented in different frameworks/languages; this comparison aims to only provide relative comparisons between them. Sela *et al.* [42] and VRN [73] both use an encoder-decoder network with skip con-

Fig. 22: Growing mustache editing results. The first collumn shows original images, the following collumns show edited images with increasing magnitudes. Comparing to Shu *et al.* [66] results (last row), our edited images are more realistic and identity preserved.

TABLE 6: Running time of various 3D face reconstruction methods.

Method	Encoder	Decoder	Post-processing	Rendering
Sela *et al.* [42]	$\sim$ 10 ms		$\sim$ 180s	-
VRN [73]	$\sim$ 10 ms		-	-
MoFA [48]	$\sim$ 4ms	Neglectable	-	-
Our	2.7ms	5.5 ms	-	140 ms

nections with similar runtime. However, Sela *et al.* [42] requires an expensive nonrigid registration step as well as an refinement module. We get a comparable encoder running time with 3DMM regression network of MoFA [48]. However, since they directly use liner bases, the decoding step is trivial as a single multiplication; our model requires decoding features via two CNNs for shape and texture, respectively. We also note that the running time for the rendering layer is significantly higher than other components. Luckily, rendering to reconstruct input has no value and it is not required during testing.

5 Conclusions

Since its debut in 1999, 3DMM has became a cornerstone of facial analysis research with applications to many problems. Despite its impact, it has drawbacks in requiring training data of 3D scans, learning from controlled 2D images, and limited representation power due to linear bases for both shape and texture. These drawbacks could be formidable when fitting 3DMM to unconstrained faces, or learning 3DMM for generic objects such as shoes. This paper demonstrates that there exists an alternative approach to 3DMM learning, where a nonlinear 3DMM can be learned from a large set of in-the-wild face images without collecting 3D face scans. Further, the model fitting algorithm can be learnt jointly with 3DMM, in an end-to-end fashion.

Our experiments cover a diverse aspects of our learnt model, some of which might need the subjective judgment of the readers. We hope that both the judgment and quantitative results could be viewed under the context that, unlike linear 3DMM, no genuine 3D scans are used in our learning. Finally, we believe that unsupervisedly or weak-supervisedly learning 3D models from large-scale in-the-wild 2D images is one promising research direction. This work is one step along this direction.

References

[1] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3D faces," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 1999.

[2] R. Yu, S. Saito, H. Li, D. Ceylan, and H. Li, "Learning dense facial correspondences in unconstrained images," in *ICCV*, 2017.

[3] A. T. Tran, T. Hassner, I. Masi, E. Paz, Y. Nirkin, and G. Medioni, "Extreme 3D face reconstruction: Looking past occlusions," in *CVPR*, 2018.

[4] O. Aldrian and W. A. Smith, "Inverse rendering of faces with a 3D morphable model," *TPAMI*, 2013.

[5] F. Shi, H.-T. Wu, X. Tong, and J. Chai, "Automatic acquisition of high-fidelity facial performances using monocular videos," *ACM TOG*, 2014.

[6] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt, "Real-time expression transfer for facial reenactment." *ACM Trans. Graph.*, vol. 34, no. 6, pp. 183:1–183:14, 2015.

[7] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of RGB videos," in *CVPR*, 2016.

[8] B. Amberg, R. Knothe, and T. Vetter, "Expression invariant 3D face recognition with a morphable model," in *FG*, 2008.

[9] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, "Towards large-pose face frontalization in the wild," in *ICCV*, 2017.

[10] F. C. Staal, A. J. Ponniah, F. Angullia, C. Ruff, M. J. Koudstaal, and D. Dunaway, "Describing crouzon and pfeiffer syndrome based on principal component analysis," *Journal of Cranio-Maxillofacial Surgery*, 2015.

[11] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, "A 3D face model for pose and illumination invariant face recognition," in *AVSS*, 2009.

[12] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou, "Facewarehouse: A 3D facial expression database for visual computing," *TVCG*, 2014.

[13] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato, "A 3D facial expression database for facial behavior research," in *FGR*, 2006.

[14] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniah, and D. Dunaway, "A 3D morphable model learnt from 10,000 faces," in *CVPR*, 2016.

[15] M. Zollhöfer, J. Thies, D. Bradley, P. Garrido, T. Beeler, P. Péerez, M. Stamminger, M. Nießner, and C. Theobalt, "State of the art on monocular 3D face reconstruction, tracking, and applications," *Eurographics*, 2018.

[16] L. Tran and X. Liu, "Nonlinear 3D morphable model," in *CVPR*, 2018.

[17] A. Patel and W. A. Smith, "3D morphable face models revisited," in *CVPR*, 2009.

[18] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *TPAMI*, 1989.

[19] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid ICP algorithms for surface registration," in *CVPR*, 2007.

[20] D. Vlasic, M. Brand, H. Pfister, and J. Popović, "Face transfer with multilinear models," in *ACM TOG*, 2005.

[21] T. Bolkart and S. Wuhrer, "A groupwise multilinear correspondence optimization for 3D faces," in *ICCV*, 2015.

[22] P. Koppen, Z.-H. Feng, J. Kittler, M. Awais, W. Christmas, X.-J. Wu, and H.-F. Yin, "Gaussian mixture 3D morphable face model," *Pattern Recognition*, 2017.

[23] J. Booth, E. Antonakos, S. Ploumpis, G. Trigeorgis, Y. Panagakis, and S. Zafeiriou, "3D face morphable models "In-the-wild"," in *CVPR*, 2017.

[24] A. T. Tran, T. Hassner, I. Masi, and G. Medioni, "Regressing robust and discriminative 3D morphable models with a very deep neural network," in *CVPR*, 2017.

[25] C. Nhan Duong, K. Luu, K. Gia Quach, and T. D. Bui, "Beyond principal components: Deep Boltzmann Machines for face modeling," in *CVPR*, 2015.

[26] A. Tewari, M. Zollhöfer, P. Garrido, F. Bernard, H. Kim, P. Pérez, and C. Theobalt, "Self-supervised multi-level face model learning for monocular reconstruction at over 250 Hz," in *CVPR*, 2018.

[27] H. Wu, X. Liu, and G. Doretto, "Face alignment via boosted ranking models," in *CVPR*, 2008.

[28] X. Liu, "Discriminative face alignment," *TPAMI*, 2009.

[29] P. Dollár, P. Welinder, and P. Perona, "Cascaded pose regression," in *CVPR*, 2010.

[30] A. Jourabloo and X. Liu, "Pose-invariant 3D face alignment," in *ICCV*, 2015.

[31] ——, "Large-pose face alignment via CNN-based dense 3D model fitting," in *CVPR*, 2016.

[32] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, "Face alignment across large poses: A 3D solution," in *CVPR*, 2016.

[33] F. Liu, D. Zeng, Q. Zhao, and X. Liu, "Joint face alignment and 3D face reconstruction," in *ECCV*, 2016.

[34] J. McDonagh and G. Tzimiropoulos, "Joint face detection and alignment with a deformable Hough transform model," in *ECCV*, 2016.

[35] A. Jourabloo, X. Liu, M. Ye, and L. Ren, "Pose-invariant face alignment with a single CNN," in *ICCV*, 2017.

[36] A. Jourabloo and X. Liu, "Pose-invariant face alignment via CNN-based dense 3D model fitting," *IJCV*, 2017.

[37] S. Tulyakov and N. Sebe, "Regressing a 3D face shape from a single image," in *ICCV*, 2015.

[38] Y. Wu and Q. Ji, "Robust facial landmark detection under significant head poses and occlusion," in *ICCV*, 2015.

[39] J. Roth, Y. Tong, and X. Liu, "Unconstrained 3D face reconstruction," in *CVPR*, 2015.

[40] ——, "Adaptive 3D face reconstruction from unconstrained photo collections," in *CVPR*, 2016.

[41] E. Richardson, M. Sela, and R. Kimmel, "3D face reconstruction by learning from synthetic data," in *3DV*, 2016.

[42] M. Sela, E. Richardson, and R. Kimmel, "Unrestricted facial geometry reconstruction using image-to-image translation," in *ICCV*, 2017.

[43] J. Roth, Y. Tong, and X. Liu, "Adaptive 3D face reconstruction from unconstrained photo collections," *TPAMI*, 2017.

[44] V. Blanz and T. Vetter, "Face recognition based on fitting a 3D morphable model," *TPAMI*, 2003.

[45] L. Gu and T. Kanade, "A generative shape regularization model for robust face alignment," in *ECCV*, 2008.

[46] L. Zhang and D. Samaras, "Face recognition from a single training image under arbitrary unknown lighting using spherical harmonics," *TPAMI*, 2006.

[47] P. Dou, S. K. Shah, and I. A. Kakadiaris, "End-to-end 3D face reconstruction with deep neural networks," in *CVPR*, 2017.

[48] A. Tewari, M. Zollhöfer, H. Kim, P. Garrido, F. Bernard, P. Pérez, and C. Theobalt, "MoFA: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction," in *ICCV*, 2017.

[49] F. Liu, R. Zhu, D. Zeng, Q. Zhao, and X. Liu, "Disentangling features in 3D face shapes for joint face reconstruction and recognition," in *CVPR*, 2018.

[50] E. Richardson, M. Sela, R. Or-El, and R. Kimmel, "Learning detailed face reconstruction from a single image," in *CVPR*, 2017.

[51] H. Kim, M. Zollhöfer, A. Tewari, J. Thies, C. Richardt, and C. Theobalt, "Inversefacenet: Deep single-shot inverse face rendering from a single image," in *CVPR*, 2018.

[52] K. Genova, F. Cole, A. Maschinot, A. Sarna, D. Vlasic, and W. Kyle, "Unsupervised training for 3D morphable model regression," in *CVPR*, 2018.

[53] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *TPAMI*, 2001.

[54] X. Liu, P. Tu, and F. Wheeler, "Face model fitting on low resolution images," in *BMVC*, 2006.

[55] X. Liu, "Video-based face model fitting using adaptive active appearance model," *Image and Vision Computing*, 2010.

[56] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, 1975.

[57] R. Ramamoorthi and P. Hanrahan, "An efficient representation for irradiance environment maps," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001.

[58] Y. Wang, L. Zhang, Z. Liu, G. Hua, Z. Wen, Z. Zhang, and D. Samaras, "Face relighting from a single image under arbitrary unknown lighting conditions," *TPAMI*, 2009.

[59] F. Cole, D. Belanger, D. Krishnan, A. Sarna, I. Mosseri, and W. T. Freeman, "Face synthesis from facial identity features," in *CVPR*, 2017.

[60] Y. Nirkin, I. Masi, A. T. Tran, T. Hassner, and G. M. Medioni, "On face segmentation, face swapping, and face perception," in *FG*, 2018.

[61] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "FaceVR: Real-time facial reenactment and eye gaze control in virtual reality," *arXiv:1610.03151*, 2016.

[62] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *BMVC*, 2015.

[63] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*, 2016.

[64] P. Garrido, M. Zollhöfer, D. Casas, L. Valgaerts, K. Varanasi, P. Pérez, and C. Theobalt, "Reconstruction of personalized 3D face rigs from monocular video," *ACM TOG*, 2016.

[65] A. Meka, M. Zollhöfer, C. Richardt, and C. Theobalt, "Live intrinsic video," *ACM TOG*, 2016.

[66] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras, "Neural face editing with intrinsic image disentangling," in *CVPR*, 2017.

[67] C. Sagonas, E. Antonakos, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "300 faces in-the-wild challenge: Database and results," *Image and Vision Computing*, 2016.

[68] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *ICCV*, 2015.

[69] J. Yan, Z. Lei, D. Yi, and S. Li, "Learn to combine multiple hypotheses for accurate face alignment," in *CVPRW*, 2013.

[70] Y. Liu, A. Jourabloo, W. Ren, and X. Liu, "Dense face alignment," in *ICCVW*, 2017.

[71] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning GAN for pose-invariant face recognition," in *CVPR*, 2017.

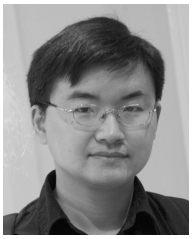
[72] ——, "Representation learning by rotating your faces," *TPAMI*, 2018.

[73] A. S. Jackson, A. Bulat, V. Argyriou, and G. Tzimiropoulos, "Large pose 3D face reconstruction from a single image via direct volumetric cnn regression," in *ICCV*, 2017.

[74] J. Shen, S. Zafeiriou, G. G. Chrysos, J. Kossaifi, G. Tzimiropoulos, and M. Pantic, "The first facial landmark tracking in-the-wild challenge: Benchmark and results," in *ICCVW*, 2015.

[75] Z. Shu, S. Hadap, E. Shechtman, K. Sunkavalli, S. Paris, and D. Samaras, "Portrait lighting transfer using a mass transport approach," *ACM TOG*, 2018.

Luan Tran received his B.S. in Computer Science from Michigan State University with High Hornors in 2015. He is now pursuing his Ph.D. also at Michigan State University in the area of deep learning and computer vision. His research areas of interest include deep learning and computer vision, in particular, face modeling and face recognition. He is a member of the IEEE.



Xiaoming Liu is an Associate Professor at the Department of Computer Science and Engineering of Michigan State University. He received the Ph.D. degree in Electrical and Computer Engineering from Carnegie Mellon University in 2004. Before joining MSU in Fall 2012, he was a research scientist at General Electric (GE) Global Research. His research interests include computer vision, machine learning, and biometrics. As a co-author, he is a recipient of Best Industry Related Paper Award runner-up at ICPR 2014, Best Student Paper Award at WACV 2012 and 2014, Best Poster Award at BMVC 2015, and Michigan State University College of Engineering Withrow Endowed Distinguished Scholar Award. He has been the Area Chair for numerous conferences, including FG, ICPR, WACV, ICIP, and CVPR. He is the program co-chair of WACV 2018 and BTAS 2018. He is an Associate Editor of Neurocomputing journal. He has authored more than 100 scientific publications, and has filed 26 U.S. patents.