Rotate-and-Render: Unsupervised Photorealistic Face Rotation from Single-View Images

Hang Zhou[1*] Jihao Liu[2*] Ziwei Liu[1] Yu Liu[1†] Xiaogang Wang[1]
[1]The Chinese University of Hong Kong [2]SenseTime Research
{zhouhang@link,yuliu@ee,xgwang@ee}.cuhk.edu.hk liujihao@sensetime.com zwliu.hust@gmail.com

Figure 1: **The rotated faces synthesized by our approach on CelebA-HQ dataset.** The first row is the input and the second row is our result. It can be observed that our unsupervised framework produces near photo-realistic results even under extreme poses and expressions.

Abstract

Though face rotation has achieved rapid progress in recent years, the lack of high-quality paired training data remains a great hurdle for existing methods. The current generative models heavily rely on datasets with multi-view images of the same person. Thus, their generated results are restricted by the scale and domain of the data source. To overcome these challenges, we propose a novel unsupervised framework that can synthesize photo-realistic rotated faces using only single-view image collections in the wild. Our key insight is that ***rotating*** *faces in the 3D space back and forth, and re-****rendering*** *them to the 2D plane can serve as a strong self-supervision. We leverage the recent advances in 3D face modeling and high-resolution GAN to constitute our building blocks. Since the 3D rotation-and-render on faces can be applied to arbitrary angles without losing details, our approach is extremely suitable for in-the-wild scenarios (*i.e. *no paired data are available), where existing methods fall short. Extensive experiments demonstrate that our approach has superior synthesis quality as well as identity preservation over the state-of-the-art methods, across a wide range of poses and domains. Furthermore, we validate that our rotate-and-render framework naturally can act as an effective data augmentation engine for boosting modern face recognition systems even on strong baseline models.*

1. Introduction

Face rotation, or more generally speaking multi-view face synthesis, has long been a topic of great research interests, due to its wide applications in computer graphics, augmented reality and particularly, face recognition. It is also an ill-posed task with inherent ambiguity that can not be well solved by existing methods. Traditionally, this problem is addressed by using 3D models such as 3DMM [1]. A common challenge here is that invisible areas would appear when rotating a 3D-fitted face. Previous researchers propose to solve this problem through symmetric editing and invisible region filling [48]. However, this filling process usually introduces visible artifacts which lead to non-realistic results.

With the rapid progress of deep learning and generative adversarial networks (GANs), reconstruction-based methods have been widely applied to face frontalization and ro-

*Equal contribution.
†Corresponding author.

tation [42, 14, 33, 12, 32]. Due to the information loss when encoding the given face to bottleneck embeddings, reconstruction-based methods often suffer from the loss of local details such as known facial textures and shapes. It further leads to the confusion of identity information. Moreover, the most notable drawback of existing reconstruction-based methods is that multi-view data of the same person has to be provided as direct supervisions in most cases. To this end, the datasets used for training are constraint to ones in controlled environments such as Multi-PIE [7], and synthetic ones such as 300W-LP [48]. Models trained on controlled datasets can only generate results within a specific domain, lacking the desired generalization ability. Also, their generated resolutions are normally limited to under 128×128, far from perceptually satisfying.

To overcome these challenges, we propose a novel unsupervised framework that can synthesize photorealistic rotated faces using only single-view image collections in the wild, and can achieve arbitrary-angle face rotation. While details and ID information tend to degrade during the encoding process of 2D-based methods, we propose to keep as much known information about the given face as possible with 3D model.

*Our key insight is that **rotating** faces in the 3D space back and forth, and re-**rendering** them to the 2D plane can serve as a strong self-supervision.* We take the advantage of both 3D face modeling and GANs by using off-the-shelf 3D-fitting network 3DDFA [49] and the neural renderer [18]. Invisible parts would appear to one fixed 2D view when a face is rotated from one pose to another. While previous methods require both images to form an {input, ground truth} pair, we use the self-supervision of one single image. The key is to create and then eliminate the artifacts caused by rotations. Given one face at pose P_a, we **rotate** its 3D-mesh firstly to another arbitrary pose P_b, and **render** it to the 2D space to get a 2D-rendered image Rd_b. Then we **rotate** it back to its original position and **render** it to be $\mathrm{Rd}_{a'}$ using textures extracted from Rd_b. Finally, we use an image-to-image translation network to fill the invisible parts and map the rendered image to real image domain. The overview of our pipeline is shown in Fig. 2. In this way, existing local texture information can be preserved while GAN is responsible for fixing the occluded parts. As the whole pipeline rotates and renders a face forward and backward, we term it **Rotate-and-Render** framework.

Remarkably, our proposed framework does not rely on paired data or any kind of label, thus any face image can be used as our training source. With unlimited training data, our model can be leveraged to boost large-scale face recognition, providing augmentations and alignments for profile faces. While previous methods are often evaluated on small datasets with moderate baselines, we validate the effectiveness of our approach for large-scale face recognition on strong baseline models.

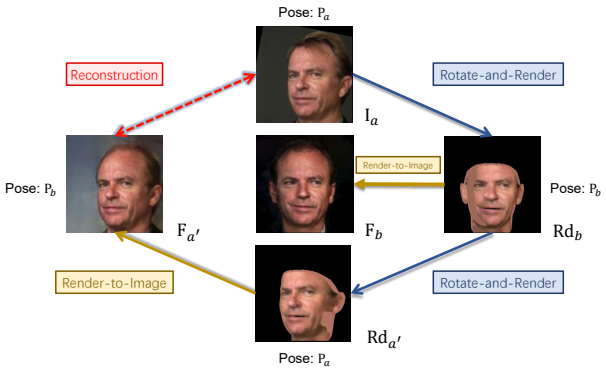

Figure 2: **Overview of our unsupervised face rotation framework from only single-view image collections.** We **rotate** the 3D-mesh firstly to an arbitrary pose P_b, and **render** it to the 2D space to get a 2D-rendered image Rd_b. Then we **rotate** it back to its original position P_a and **render** it to be $\mathrm{Rd}_{a'}$ using textures extracted from Rd_b. Finally we use an render-to-image translation network to fill the invisible parts and map the rendered image to real image domain.

Our contributions are summarized as follows: **1)** We propose a novel Rotate-and-Render framework for training face rotation in a fully unsupervised manner under in-the-wild scenarios. No paired data or any label is needed. **2)** We convert incomplete rendered images to real images using an image-to-image translation network, with which photo-realistic face rotation results can be generated. **3)** We validate that our generation results benefit large-scale face recognition even on strong baseline models.

2. Related Work

2.1. Face Rotation and Multi-View Synthesis

The problem of face rotation aims to synthesize multi-view faces given a single face image regardless of its viewpoint. Among all views, the frontal view particularly attracts much more research interests. Traditionally, this problem is tackled by building 3D models and warping textures on 3D or 2D [9, 48]. OpenGL can also be used to also easier cases [24]. However, their synthesized results are usually blurry and not photorealistic.

Reconstruction-based Methods. Recently, with the progress of deep learning [22] and GANs [6], reconstruction-based models have revolutionized the field of face frontalization [33, 14, 12, 34, 29, 30]. DR-GAN [33, 34], for the first time, adopts GAN to generate frontal faces with an encoder-decoder architecture. Although they do not use multi-view data, the generated results are not satisfying and have perceptually-visible artifacts. Then TP-GAN [14] utilizes global and local networks together with a multi-task learning strategy to

frontalize faces. CAPG-GAN [12] uses face heatmap to guide the generation of multi-view faces. Most of the methods are trained on Multi-PIE dataset, which makes them overfit the dataset's environment and cannot generalize well to unseen data. Even FNM [30] which proposes to combine both labeled and unlabeled data can only normalize faces to a standard MultiPIE-like frontal-view. The common shortcoming that almost all of them share is the requirement of paired multi-view training data.

3D Geometry-based Methods. Several attempts have been made to incorporate 3D prior knowledge into GAN-based frontalization pipeline. FF-GAN [44] proposes to integrate the regression of 3DMM coefficients into the network and employ them for generation. But its generation results are of low-quality. UV-GAN [3] proposes to complete UV-map using image-to-image translation, which is similar to our work. However, their pipeline requires high precision 3D fitting and ground-truth UV-maps, which are both difficult to obtain. Besides, their generated results are not photorealistic under in-the-wild scenarios. Recently, HF-PIM [2] achieves high-quality face frontalization results using facial texture map and correspondence fields. However, their method also requires paired data for training. In this work, our proposed approach not only gets rid of the requirement for paired training data, but also has the capacity to generate high-quality results which preserve texture and identity information.

2.2. Image-to-Image Translation

Image-to-image translation aims at translating an input image to a corresponding output image, typically from a different modality or domain. The core idea is to predict pixel values directly with encoder-decoder architecture and low-level connections. GANs are widely adopted in this field [16, 47, 35, 26, 37, 5, 45, 43, 21, 20], since the adversarial loss can alleviate the blurry issue by L_1 reconstruction loss. Pix2Pix framework [16] firstly uses image-conditional GANs for this task. Pix2PixHD [35] used stacked structures to produce high-quality images. Recently, SPADE [26] and MaskGAN [20] discovers that semantic information can be infused via conditional batch normalization to further improve the results.

Cycle Consistency in Unsupervised Image-to-Image Translation. Cycle consistency has been proven useful on various tasks [47, 28, 38, 36], particularly for image translation without paired data. For example, CycleGAN [47] achieves unpaired image-to-image translation, and GANimation [28] proposes to generation animation without supervision. Our idea shares similar intuition with cycle consistency. However, The difference is that for most unpaired translation papers, they focus on mapping across domain by training neural networks, while our proposed Rotate-and-Render operation is performed off-line.

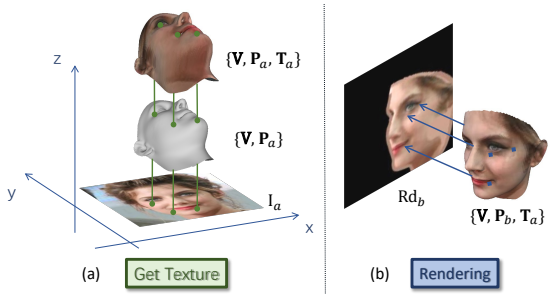

Figure 3: The process of getting texture and rendering. 3D points lying on the same lines will correspond to the same texture in the 2D space. The subscript a and b are associated with pose a and b in Fig 4.

Inpainting with GAN. After the rendering step, our target is to synthesize photorealistic images from renders with artifacts, which is substantially a combination of image translation and inpainting. Recently, advanced inpainting techniques also benefit from the image-to-image translation pipeline with GAN [27, 40, 37, 46]. Therefore we adopt the recent advances in image-to-image translation to realize render-to-image generation.

3. Our Approach

Overall Framework. Our whole face rotation framework consists of three parts: 3D face fitting, the rotate-and-render strategy for training data preparation, and the render-to-image translation module. We elaborate each component as follows.

3.1. 3D Face Modeling and Render

3D Face Fitting. Our method relies on a rough 3D parametric fitting of a given face, where all kinds of 3D models and predicting methods are applicable. Here we briefly introduce the notations and concepts of 3D face model we use as an instruction for the following sections.

Given one definition of 3D face model with n vertices, the shape prediction of a face $\mathbf{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n]$ represents the normalized position of its vertices in the 3D space with $\boldsymbol{v}_i = [x_i, y_i, z_i]^{\mathrm{T}}$. The projection of a 3D shape onto the 2D image can be written as:

$$\Pi(\mathbf{V}, \mathbf{P}) = f * \mathbf{p_r} * \mathbf{R} * \mathbf{V} + \mathbf{h_{2d}}, \tag{1}$$

where Π is the projection function that maps model vertices to their 2D positions. The matrix multiplication is denoted by "$*$". f is the scale factor, $\mathbf{p_r}$ is the orthographic projection matrix, $\mathbf{R}$ is the rotation matrix and $\mathbf{h_{2d}}$ is the 2D shift. Here we regard all the above defined projection related parameters to be a joint representation of the face's relative pose $\mathbf{P} = \{f, \mathbf{R}, \mathbf{h_{2d}}\}$.

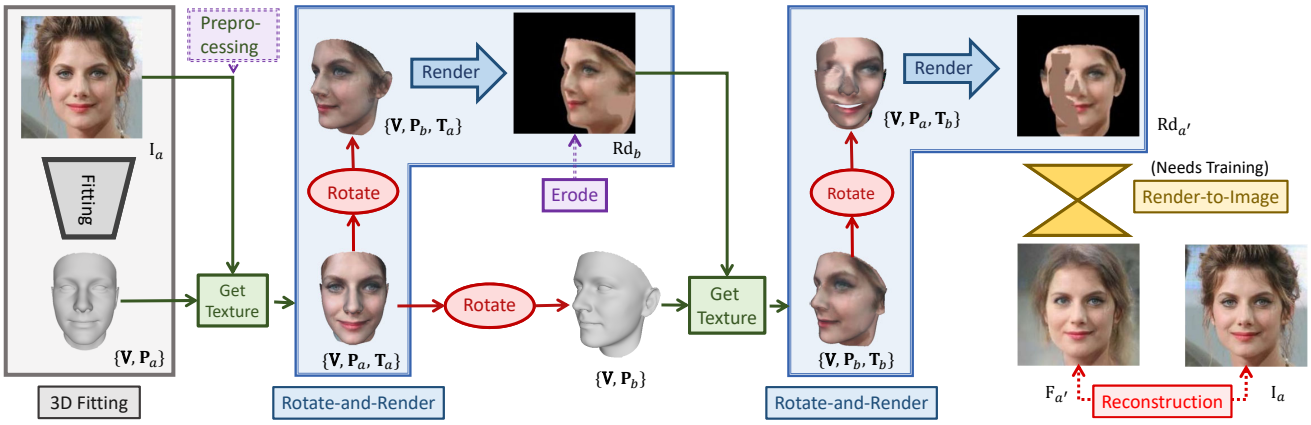

Figure 4: **Our framework for unsupervised photorealistic face rotation.** Our key insight is that **rotating** faces in the 3D space back and forth, and re-**rendering** them to the 2D plane can serve as a strong self-supervision. After two rotate-and-render processes, we can create a rendered image $\text{Rd}_{a'}$ which has the artifacts of a face rotated to the position a from any position b. So that the input image I_a itself can serve as ground truth for training. Only the Render-to-Image module needs training during the whole process.

Acquiring Textures. Textures also play a crucial role when transforming a complete 3D representation to 2D space. For each vertex $\boldsymbol{v}_i$, there exists an associated texture $\boldsymbol{t}_i = [r_i, g_i, b_i]^{\text{T}}$ on a colored face. We use the simplest vertical projection to get the colors of the vertices from the original image I. The color of each vertex can be written as :

$$\boldsymbol{t}_i = \text{I}(\Pi(\boldsymbol{v}_i, \mathbf{P})), \tag{2}$$

where $\Pi(\boldsymbol{v}_i, \mathbf{P})$ is the projected 2D coordinate of the vertice $\boldsymbol{v}_i$. In this way we get all corresponding textures $\mathbf{T} = [\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n]$. This process can be easily depicted in Fig 3. We refer to the whole process of getting textures uniformly as:

$$\mathbf{T} = \text{GetTex}(\text{I}, \{\mathbf{V}, \mathbf{P}\}). \tag{3}$$

The projected result $\Pi(\mathbf{V}, \mathbf{P})$ is irrelevant to the vertices' z coordinate due to the orthographic matrix $\mathbf{p_r}$. For each position (x_j, y_j) on the 2D space, there might exist multiple rotated vertices on the line $\{x = x_j \text{ and } y = y_j\}$ in 3D space, then the same texture will be assigned to every one of them. For all $\boldsymbol{v}_k \in \{\boldsymbol{v} \mid (x_j, y_j) = \Pi(\boldsymbol{v}_k, \mathbf{P})\}$, only the outermost one with the largest z axis value gets the correct texture. Its index is

$$K_j = \arg\max_k([0, 0, 1] * \boldsymbol{v}_k). \tag{4}$$

The rest are actually invisible vertices due to occlusion in the 2D space. We keep the wrongly acquired textures and regard them as artifacts that we aim to deal with.

Rendering. Given a set of 3D representation of a face $\{\mathbf{V}, \mathbf{P}, \mathbf{T}\}$, rendering is to map it to the 2D space and generate an image. The rendering process is the reverse of acquiring texture as depicted in Fig 3 (b). Same as equation 4, it is known that K_j is the index of outermost vertice given a 2D point (x_j, y_j). The rendered color image Rd can be calculated as:

$$\text{Rd}(x_j, y_j) = \begin{cases} \mathbf{T}\{K_j\}, & \exists K_j \in \mathbb{N}, \\ 0, & \nexists K_j \in \mathbb{N}. \end{cases} \tag{5}$$

Finally, we denote the whole rendering process as:

$$\text{Rd} = \text{Render}(\{\mathbf{V}, \mathbf{P}, \mathbf{T}\}). \tag{6}$$

We use the open-sourced Neural Mesh Renderer [18] to perform rendering without any training.

3.2. Rotate-and-Render Training Strategy

It can be discovered that with an accurately fitted 3D model, our aim is to fill the invisible vertices with the correct textures for getting another view of a face. However, existing works with similar ideas [2, 3] require ground truth supervision from multi-view images which are difficult to get.

Here we propose a simple strategy to create training pairs called Rotate-and-Render (R&R) which consists of two rotate-and-render operations. The key idea is to create the artifacts caused by rotating occluded facial surface to the front and eliminate them. Thus we can leverage only self-supervision to train networks. The whole pipeline and visualizations of each step are all illustrated in Fig 4.

Given an input image I_a, we firstly get the 3D model parameters by a 3D-face fitting model:

$$\{\mathbf{V}, \mathbf{P}_a\} = \text{Fitting}(\text{I}_a), \tag{7}$$

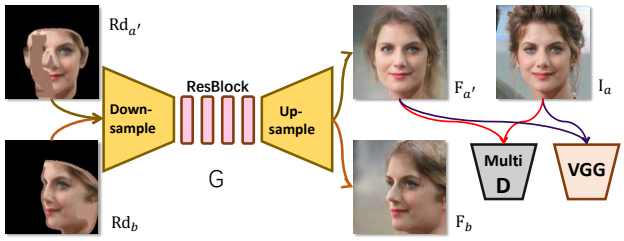

Figure 5: Our render-to-image module. Rendered images $\text{Rd}_{a'}$ and Rd_b are sent into the generator G to get the results $\text{F}_{a'}$ and F_b. Losses are conducted between generated $\text{F}_{a'}$ and the ground truth I_a through the discriminator and the pretrained VGG network.

where a denotes the current view of this face in the 2D space, with $\mathbf{P}_a = \{f, \mathbf{R}_a, \mathbf{h_{2d}}\}$. The textures of its vertices can be acquired as:

$$\mathbf{T}_a = \text{GetTex}(\text{I}_a, \{\mathbf{V}, \mathbf{P}_a\}). \tag{8}$$

We then rotate the 3D representation of this face to another random 2D view b by multiplying $\mathbf{R}_a$ with another rotation matrix $\mathbf{R}_{random}$ to get $\mathbf{P}_b = \{f, \mathbf{R}_b, \mathbf{h_{2d}}\}$. And we render the current 3D presentation to $\text{Rd}_b = \text{Render}(\{\mathbf{V}, \mathbf{P}_b, \mathbf{T}_a\})$. This completes the first rotate-and-render operation.

Under this circumstance, another set of textures can be acquired as:

$$\mathbf{T}_b = \text{GetTex}(\text{Rd}_b, \{\mathbf{V}, \mathbf{P}_b\}). \tag{9}$$

We discover that the vertice set whose textures are correct under view b is the subset of the vertice set whose textures are correct under view a. So unlike previous works rely on a ground truth image I_b as supervision to recover a face under view b given view a, we propose to recover $\mathbf{T}_a$ regarding $\mathbf{T}_b$ as input.

Specifically, we rotate the 3D position $\mathbf{P}_b$ back to $\mathbf{P}_a$ and render it back to its original 2D position with

$$\text{Rd}_{a'} = \text{Render}(\{\mathbf{V}, \mathbf{P}_a, \mathbf{T}_b\}). \tag{10}$$

This $\text{Rd}_{a'}$ is basically a rendered image with artifacts caused by rotating a face from view b to a in the 2D space. In this way, we get our input/ground truth pair $\{\text{Rd}_{a'}, \text{I}_a\}$ for training.

3.3. Render-to-Image Generation

In order to eliminate the artifacts and map the rendered images Rd_b and $\text{Rd}_{a'}$ from the rendered domain to real image domain, we propose the render-to-image generation module to create $\text{F}_{a'} = \text{G}(\text{Rd}_{a'})$ and $\text{F}_b = \text{G}(\text{Rd}_b)$ using generator G, as shown in Fig 5.

The basic generator G is adopted from CycleGAN [47], which is enough to handle most images in our datasets.

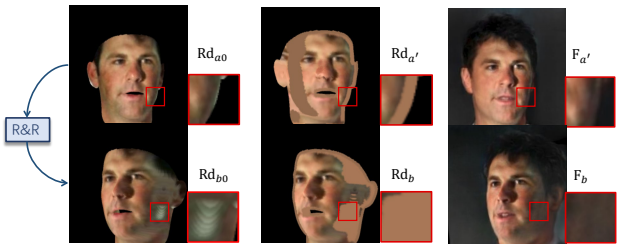

Figure 6: Illustration of the miss-alignment effect and the results of the erosion preprocessing.

The multi-layer discriminator and perceptual loss from Pix2PixHD [35] are borrowed directly. The loss function of the discriminator includes the adversarial loss

$$\mathcal{L}_{\text{GAN}}(\text{G}, \text{D}) = \mathbb{E}_{\text{I}}[\log \text{D}(\text{I}_a)] + \mathbb{E}_{\text{Rd}}[\log(1 - \text{D}(\text{G}(\text{Rd}_{a'}))], \tag{11}$$

and a feature matching loss. The feature matching loss is realized by extracting features from multiple layers of the discriminator and regularizing the distance between input and generated images. We use $F_D^{(i)}(\text{I})$ to denote the feature extracted from the ith layer of the discriminator for an input I. For total N_D layers, the feature matching loss can be written as:

$$\mathcal{L}_{FM}(G, D) = \frac{1}{N_D} \sum_{i=1}^{N_D} \|F_D^{(i)}(\text{I}_a) - F_D^{(i)}(\text{G}(\text{Rd}_{a'}))\|_1. \tag{12}$$

Perceptual loss is achieved by using ImageNet pretrained VGG network. It is used to regularize both the generation results and the generated identity. The perceptual loss is very similar to that of $\mathcal{L}_{FM}$, with the features denoted by $F_{vgg}^{(i)}$, the loss function is:

$$\mathcal{L}_{vgg}(\text{G}, \text{D}) = \frac{1}{N_{vgg}} \sum_{i=1}^{N_{vgg}} \|F_{vgg}^{(i)}(\text{I}_a) - F_{vgg}^{(i)}(\text{G}(\text{Rd}_{a'}))\|_1. \tag{13}$$

Our full objective function can be written as:

$$\mathcal{L}_{total} = \mathcal{L}_{\text{GAN}} + \lambda_1 \mathcal{L}_{FM} + \lambda_2 \mathcal{L}_{vgg}. \tag{14}$$

During testing, our desired output can be directly generated by assigning the target view $\mathbf{P_c}$ to form $\text{Rd}_c = \text{Render}(\{\mathbf{V}, \mathbf{P}_c, \mathbf{T}_a\})$, and send the rendered Rd_c into the trained generator G.

3.4. Building Block Details

3D Fitting Tool. Our choice of 3D face model is the 3D Morpohable Model (3DMM) [1]. Its flattened vertice matrix $\mathbf{V}$ can be denoted by $\mathbf{S} = [\boldsymbol{v}_1^{\text{T}}, \boldsymbol{v}_2^{\text{T}}, \cdots, \boldsymbol{v}_n^{\text{T}}]^{\text{T}}$. Its description of 3D faces is based on PCA:

$$\mathbf{S} = \overline{\mathbf{S}} + \mathbf{A}_{id}\boldsymbol{\alpha}_{id} + \mathbf{A}_{exp}\boldsymbol{\alpha}_{exp}. \tag{15}$$

Here $\overline{\mathbf{S}}$ is the mean shape, $\mathbf{A}_{id}$ and $\mathbf{A}_{exp}$ are the principle axes for identities and expressions, respectively.

We use the open-sourced implementation and the pretrained model of 3DDFA [49] for 3D face fitting. It is a deep learning based model for regressing the parameters $[\mathbf{P}^{\mathrm{T}}, \boldsymbol{\alpha}_{id}, \boldsymbol{\alpha}_{exp}]$, from a single 2D face image. Therefore the face's 3D shape and its relative position in 2D space $\{\mathbf{V}, \mathbf{P}\}$ can be predicted. Please be noted that we do not train the 3DDFA. The usage of it is only for fast application, and its miss-alignment on 3D shape prediction can cast certain problems to the final results. An alternative way is to tune the 3D model using identity labels. However, in a fully unsupervised setting, we propose to solve it using a novel eroding preprocessing.

Pre-Render and Erosion. Due to the inaccuracy of 3D fitting methods, the projection of wrongly fitted vertices sometimes lies outside the true edges of a face. When this kind of 3D miss alignment happens, background pixels would be assigned to these vertices. During the rotate-and-render process, those wrongly acquired textures will be rendered onto the rotated Rd_b (see Fig. 6 left column). However, such artifacts are difficult to create directly on $\mathrm{Rd}_{a'}$ by the rotate-and-render process, which means that they do not exist in training input-output pairs. Thus they cannot be handled by our generator.

The way to solve it is to pre-render the fitted 3D representation $\{\mathbf{V}, \mathbf{P}_a, \mathbf{T}_a\}$ to Rd_{a0}, and erode the rendered image by certain pixels. The erosion is performed basing on the projected edge of $\mathbf{V}$ with an average color of all vertices. Then texture $\mathbf{T}_a$ is renewed to

$$\mathbf{T}_a = \mathrm{GetTex}(\mathrm{erode}(\mathrm{Rd}_a), \{\mathbf{V}, \mathbf{P}_a\}). \tag{16}$$

So that Rd_b can only contain artifacts that exist in $\mathrm{Rd}_{a'}$. The output after the erosion can be found at Fig. 6.

4. Experiments

4.1. Experimental Settings

Implementation Details. We firstly run 3DDFA across all datasets to get the parameters $\{\mathbf{V}, \mathbf{P}\}$ for all images. With known $\mathbf{V}$, we are able to know the facial key points and perform alignment for faces according to their eye centers and noses. The generator G contains 4 downsample and upsample blocks and 9 residual blocks. Spectral Normalization [23] and Batch Normalization [15] are applied to all layers. The discriminator consists of two scales.

Our models are trained using Pytorch on 8 Tesla V100 GPUs with 16 GB memory. Two graphical cards are used for rendering and the others for training. The time for rotating one single image is about 0.1s. The weights λ_1 and λ_2 are both set to 10. Please refer to our code and models for more details.[1]

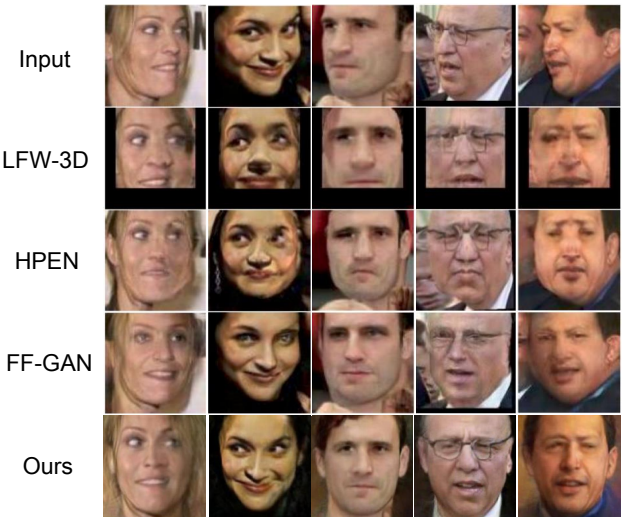

Figure 7: Frontalization results with 3D-based Methods. the first row is the input. From top to down rows are results from: LFW-3D [9]; HPEN [48]; FF-GAN [44] and the last is ours. The samples are selected from LFW [13].

Datasets. Using our rotate-and-render strategy, we do not rely on any paired multi-view data or supervision, so there is no problem of over-fitting in evaluation. Theoretically, we have unlimited number of data for training our system. As datasets in controlled environments are not always applicable to real-world scenarios, we focus more on large-scale in-the-wild problems.

CASIA-WebFace [41] and **MS-Celeb-1M** [8] are selected as training sets for both our render-to-image module and our face recognition network. Specifically, we adopt the version named **MS1MV2** cleaned in [4]. For evaluating our boost on strong face recognition systems, we test on the standard **LFW** [13], **IJBA** [19] which contain profile faces in videos, and **MegaFace 1-million Challenge** [25]. MegaFace is the most challenging and wildly applied dataset for evaluating face recognition results.

4.2. Qualitative Results

As most papers do not release their code and models, we directly borrow the results reported from some of the papers and perform frontalization with the corresponding image in the reported dataset. Our results are cropped for better visualization. It is recommended to zoom-in for a better view. Results on **CelebA-HQ** [17] are shown in Fig 1 to validate that we can generate high-quality results under extreme poses and expressions

Comparison with 3D-based methods. Fig 7 illustrates the results of 3D-based methods and FFGAN [44], which com-

[1] https://github.com/Hangz-nju-cuhk/Rotate-and-Render.

Figure 8: Frontalization results comparing with GAN-based methods. The samples are selected from LFW.

bines 3D and GAN. It can be seen from the figure that while pure 3D methods attempt to fill the missing part with symmetric priors, they would create great artifacts when coming to large poses. This figure is extracted from FFGAN [44]. However, FFGAN fails to rotate faces to the front fully with serious loss of details. Our results seem much appealing comparing to theirs.

Comparison with GAN-based methods. Fig 8 depicts our comparisons with GAN-based methods. TP-GAN [14], CAPG-GAN [12], and FNM [30] are all purely reconstruction-based methods trained on a constrained dataset. As a consequence, the generated results lie only in the domain where their networks are trained. This limits their applications in the field of entertainment. Ours, on the other hand, preserves the illumination and environment satisfyingly. Besides, the results of TP-GAN [14] change the facial shape, CAPG-GAN [12] suffers from identity change. HF-PIM [2] is the previous state-of-the-art method. However, the results reported seem to change more details of identity than that of ours. For example, nose and jaw shapes on the first person (Keanu Reeves) in their results have been changed. FNM [30] is a GAN-based method that takes both constraint and in-the-wild data. Their results tend to generate the standard frontal face with neutral expressions in even the same background.

Numerical Results. We have conducted an additional numerical experiments on VidTIMIT dataset [31] to directly compare frontalization results on pixel level. The multi-view frames part is used. We select frames with yaw angle larger than 45° as input and frontal face as ground truth. We unify the scale to 128×128 to compute the PSNR. Only open-sourced methods are used for comparison. The results of ours, FNM [30] and CR-GAN [32] are 28.8, 27.9 and 28.2 (dB), respectively. This is only for reference because it is almost impossible for frontalized results to match the ground truth. However, we can still find our results outperform others.

Training Data	Method	ACC(%)	AUC(%)
CASIA	TP-GAN[14]	96.13	99.42
CASIA	FF-GAN[44]	96.42	99.45
-	CAPG-GAN[12]	99.37	99.90
-	LightCNN in [2]	99.39	99.87
-	HF-PIM[2]	99.41	99.92
CASIA	Res18	98.77	99.90
CASIA+rot.	**Res18(ours)**	**98.95**	**99.91**
MS1MV2	Res18	99.63	99.91
MS1MV2+rot.	**Res18(ours)**	**99.68**	**99.92**

Table 1: Verification performance on LFW dataset. Our method can still boost the performance on a strong baseline.

Training Data	Method	@FAR=.01	@FAR=.001
CASIA	FF-GAN	85.2±1.0	66.3±3.3
CASIA	DR-GAN	87.2±1.4	78.1±3.5
CASIA	FNM[30]	93.4±0.9	83.8±2.6
-	HF-PIM[2]	95.2±0.7	89.7±1.4
CASIA	Res18	90.57±1.2	80.0±4.1
CASIA+rot.	**Res18(ours)**	**91.98±0.7**	**82.48±2.5**
MS1MV2	Res18	97.28±0.6	95.39±0.9
MS1MV2+rot.	**Res18(ours)**	**97.30±0.6**	**95.63±0.7**

Table 2: Verification performance on IJB-A dataset [19].

4.3. Face Recognition Results

Face Recognition Settings. Normally, previous works report face recognition results to validate the success of their identity preserving quality after frontalization. However, clear descriptions about their settings are seldom given. Moreover, they report results on the baseline of LightCNN [39] with different structures and baseline scores, which are non-comparable. As a result, we will only report their final numbers as a reference.

Meant for proposing a real-world applicable system, we use the state-of-the-art loss function ArcFace [4] and standard ResNet18 [10] backbone which has slightly smaller parameter size than LightCNN29 [39] for face recognition. Both results trained on CASIA and MS1MV2 are provided. We propose a different way of using frontalized data to boost the performance of face recognition by augmenting the original datasets with our 3D generated ones. Then the networks are trained on our augmented datasets. For fair comparison, best results are provided for baselines.

Quantitative Results. We do a comprehensive study on the LFW datatset. The 1:1 face verification accuracy and area-under-curve (AUC) results are reported. It can be discovered that regardless of both weak (CAISA) or strong (MS1M) baselines, augmenting the training set with our frontalized results can boost the performance. Particularly, HF-PIM improves less on their baseline than our proposed method. Improvements can be found on IJB-A as listed

Angle	±30°	±45°	±60°	±75°	±90°
Res18	100	99.7	96.0	76.4	39.0
FF-GAN[44]	92.5	89.7	85.2	77.2	61.2
TP-GAN[14]	98.1	95.4	87.7	77.4	64.6
CAPG-GAN[12]	99.6	97.3	90.3	76.4	66.1
HF-PIM[2]	100	99.9	99.1	96.4	92.3
Res18(ours)	**100**	**100**	**99.7**	**99.3**	**94.4**

Table 3: Rank-1 recognition rates (%) across views on Multi-PIE dataset.

Training Data	Method	Id%
MS1MV2	R100 (ArcFace)	98.35
MS1MV2	R18	97.00
MS1MV2+rotate	**R18**	**97.48**
MS1MV2	R50	98.26
MS1MV2+rotate	**R50**	**98.44**

Table 4: Evalution on MegaFace dataset, "Id" refers to the rank-1 face identification accuracy with 1M distractors.

in Table 2 with the verification experiments conducted on ResNet18. MultiPIE [7] is also evaluated with the same settings as [14] and all works listed in Table 3. As stated before, controlled datasets are not our focus. However, with our strong baseline, we can improve the recognition results on MultiPIE as well.

Finally, we conduct experiments on MegaFace dataset, which is wildly used in the field of face recognition. However, no face frontalization paper has reported results on this dataset. Rank-1 face identification accuracy with 1 million distractors has been evaluated. We use the modified version of the 18 and 50 layers ResNet introduced by the ArcFace [4], and compare with their state-of-the-art result on R100. With our rotated augmentation, we can boost the performance of R50 to outperform the ArcFace R100 model, reaching 98.44%.

4.4. Ablation Study

We train the same network with the same set of parameters and training strategy additionally without (i) the VGG loss; (ii) using multi-scale discriminator and the feature matching loss, as they are proposed together. Instead, we change it to the PatchGAN in the original Pix2Pix paper [16]. An additional L_1 loss on image level is added for regularization.

Quantitative Results. Frechet Inception Distance (FID) [11] is widely used to measure the distances between generated and real image domains. The results of FID scores are listed in Table 5. We can observe that the performance reaches the best FID scores with our full model.

Qualitative Results. We show two sets of frontalization results on the LFW dataset. It can be seen that without the multi-scale D, it is difficult for the network to discriminate the real domain, so the results contain certain GAN artifacts. Without the VGG loss, the results tend to be smooth. They are both crucial when it comes to large poses.

Note that the modules we use are the least ones to generate reasonable results. Our method can still create visible artifacts, but there is great potential for improvements.

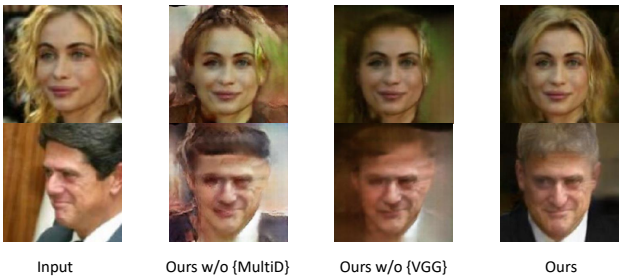

Figure 9: Ablation study of frontalization results.

Approach \ Dataset	LFW	IJB-A
Ours w/o {VGG}	93.1	126.5
Ours w/o {MultiD}	83.9	132.3
Ours	**83.1**	**70.9**

Table 5: Ablation study on loss functions with FID metric. For FID scores, the lower the better.

5. Conclusion

In this work, we take the advantage of 3D face priors and propose a novel strategy called Rotate-and-Render for unsupervised face rotation. Our key insight is to create self-supervision signal by rotating and rendering the roughly-predicted 3D representation to a random pose and back to its original place. So that self-supervision can be leveraged to create photo-realistic results by translating rendered image to real-image domain. Through comprehensive experiments, the following strengths of our pipeline have been validated: **1)** No multi-view or paired-data, nor any kind of labels, including identities are needed for training our method. **2)** Instead of purely frontalization a single-view face can be rotated to any desired angle. **3)** Near photorealistic face rotation results which preserve details and illumination condition can be generated. Visualization indicates the superiority of our method. **4)** It can be used to augment face datasets and boost recognition results on large-scale benchmarks with strong baseline models.

Acknowledgements. We thank Hao Shao for helpful discussions. This work is supported in part by SenseTime Group Limited, and in part by the General Research Fund through the Research Grants Council of Hong Kong under Grants CUHK14202217, CUHK14203118, CUHK14205615, CUHK14207814, CUHK14208619, CUHK14213616.

References

[1] Volker Blanz, Thomas Vetter, et al. A morphable model for the synthesis of 3d faces. In *Siggraph*, 1999. 1, 5

[2] Jie Cao, Yibo Hu, Hongwen Zhang, Ran He, and Zhenan Sun. Learning a high fidelity pose invariant model for high-resolution face frontalization. In *NeurIPS*, 2018. 3, 4, 7, 8

[3] Jiankang Deng, Shiyang Cheng, Niannan Xue, Yuxiang Zhou, and Stefanos Zafeiriou. Uv-gan: Adversarial facial uv map completion for pose-invariant face recognition. In *CVPR*, 2018. 3, 4

[4] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019. 6, 7, 8

[5] Yixiao Ge, Zhuowan Li, Haiyu Zhao, Guojun Yin, Shuai Yi, Xiaogang Wang, et al. Fd-gan: Pose-guided feature distilling gan for robust person re-identification. In *Advances in neural information processing systems*, pages 1222–1233, 2018. 3

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 2

[7] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. *Image and Vision Computing*. 2, 8

[8] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016. 6

[9] Tal Hassner, Shai Harel, Eran Paz, and Roee Enbar. Effective face frontalization in unconstrained images. In *CVPR*, 2015. 2, 6

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 8

[12] Yibo Hu, Xiang Wu, Bing Yu, Ran He, and Zhenan Sun. Pose-guided photorealistic face rotation. In *CVPR*, 2018. 2, 3, 7, 8

[13] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. 2008. 6

[14] Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *ICCV*, 2017. 2, 7, 8

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 6

[16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 3, 8

[17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ICLR*, 2018. 6

[18] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *CVPR*, 2018. 2, 4

[19] Brendan F Klare, Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In *CVPR*, 2015. 6, 7

[20] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 3

[21] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, and Hongsheng Li. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. In *Advances in Neural Information Processing Systems*, pages 568–578, 2019. 3

[22] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 2

[23] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 6

[24] Joel Ruben Antony Moniz, Christopher Beckham, Simon Rajotte, Sina Honari, and Chris Pal. Unsupervised depth estimation, 3d face rotation and replacement. In *Advances in Neural Information Processing Systems*, pages 9736–9746, 2018. 2

[25] Aaron Nech and Ira Kemelmacher-Shlizerman. Level playing field for million scale face recognition. In *CVPR*, 2017. 6

[26] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 3

[27] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016. 3

[28] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 818–833, 2018. 3

[29] Shengju Qian, Kwan-Yee Lin, Wayne Wu, Yangxiaokang Liu, Quan Wang, Fumin Shen, Chen Qian, and Ran He. Make a face: Towards arbitrary high fidelity face manipulation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10033–10042, 2019. 2

[30] Yichen Qian, Weihong Deng, and Jiani Hu. Unsupervised face normalization with extreme pose and expression in the wild. In *CVPR*, 2019. 2, 3, 7

[31] Conrad Sanderson and Brian C Lovell. Multi-region probabilistic histograms for robust and scalable identity inference.

In *International conference on biometrics*, pages 199–208. Springer, 2009. 7

[32] Yu Tian, Xi Peng, Long Zhao, Shaoting Zhang, and Dimitris N Metaxas. Cr-gan: learning complete representations for multi-view generation. *IJCAI*, 2018. 2, 7

[33] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017. 2

[34] Luan Quoc Tran, Xi Yin, and Xiaoming Liu. Representation learning by rotating your faces. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 2

[35] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 3, 5

[36] Xiaolong Wang, Allan Jabri, and Alexei A Efros. Learning correspondence from the cycle-consistency of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2566–2576, 2019. 3

[37] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *Advances in neural information processing systems*, 2018. 3

[38] Wayne Wu, Yunxuan Zhang, Cheng Li, Chen Qian, and Chen Change Loy. Reenactgan: Learning to reenact faces via boundary transfer. In *ECCV*, 2018. 3

[39] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 2018. 7

[40] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5485–5493, 2017. 3

[41] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014. 6

[42] Junho Yim, Heechul Jung, ByungIn Yoo, Changkyu Choi, Dusik Park, and Junmo Kim. Rotating your face using multi-task deep neural network. In *CVPR*, 2015. 2

[43] Weidong Yin, Ziwei Liu, and Chen Change Loy. Instance-level facial attributes transfer with geometry-aware flow. In *AAAI*, 2019. 3

[44] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Towards large-pose face frontalization in the wild. In *ICCV*, 2017. 3, 6, 7, 8

[45] Hang Zhou, Yu Liu, Ziwei Liu, Ping Luo, and Xiaogang Wang. Talking face generation by adversarially disentangled audio-visual representation. In *AAAI*, 2019. 3

[46] Hang Zhou, Ziwei Liu, Xudong Xu, Ping Luo, and Xiaogang Wang. Vision-infused deep audio inpainting. In *ICCV*, 2019. 3

[47] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 3, 5

[48] Xiangyu Zhu, Zhen Lei, Junjie Yan, Dong Yi, and Stan Z Li. High-fidelity pose and expression normalization for face recognition in the wild. In *CVPR*, 2015. 1, 2, 6

[49] Xiangyu Zhu, Xiaoming Liu, Zhen Lei, and Stan Z Li. Face alignment in full pose range: A 3d total solution. *IEEE transactions on pattern analysis and machine intelligence*, 41(1):78–92, 2017. 2, 6