FNeVR: Neural Volume Rendering for Face Animation

Bohan Zeng\textsuperscript{1,1}, Boyu Liu\textsuperscript{2,1}, Hong Li\textsuperscript{1}, Xuhui Liu\textsuperscript{1,*}, Jianzhuang Liu\textsuperscript{3}, Dapeng Chen\textsuperscript{4}, Wei Peng\textsuperscript{4}, Baochang Zhang\textsuperscript{1,5,*}

\textsuperscript{1}Institute of Artificial Intelligence, School of Automation Science and Electrical Engineering, Beihang University, Beijing, P.R.China
\textsuperscript{2}Sino-French Engineer School, Beihang University, Beijing, P.R.China
\textsuperscript{3}Huawei Noah’s Ark Lab, Shenzhen, P.R.China
\textsuperscript{4}Huawei, Shenzhen, P.R.China
\textsuperscript{5}Zhongguancun Laboratory, Beijing, P.R.China
\textsuperscript{*}Corresponding author, email: xhliu@buaa.edu.cn, bczhang@buaa.edu.cn
\textsuperscript{1}Equal contributions

Abstract

Face animation, one of the hottest topics in computer vision, has achieved a promising performance with the help of generative models. However, it remains a critical challenge to generate identity preserving and photo-realistic images due to the sophisticated motion deformation and complex facial detail modeling. To address these problems, we propose a Face Neural Volume Rendering (FNeVR) network to fully explore the potential of 2D motion warping and 3D volume rendering in a unified framework. In FNeVR, we design a 3D Face Volume Rendering (FVR) module to enhance the facial details for image rendering. Specifically, we first extract 3D information with a well-designed architecture, and then introduce an orthogonal adaptive ray-sampling module for efficient rendering. We also design a lightweight pose editor, enabling FNeVR to edit the facial pose in a simple yet effective way. Extensive experiments show that our FNeVR obtains the best overall quality and performance on widely used talking-head benchmarks. Our code is available\textsuperscript{1}.

1 Introduction

Aiming at animating a still head, face animation has far-reaching applications, such as photography, social media, movie production and virtual assistance, and has become a popular research topic in computer vision and computer graphics \cite{10,44}. It synthesizes a realistic talking-head video by combining the appearance and identity from one source face image with poses and motions extracted from a given driving video, possibly of a different person’s identity from the source. With the progress of generative models, especially Generative Adversarial Networks (GANs) and Variational Auto-Encoders (VAEs), recent methods have achieved a promising performance in face animation \cite{57,46,51,12}. However, it is still a challenging task because of the difficulty of guaranteeing the motion transform validity and generating authentic images simultaneously.

Face animation methods can be mainly divided into three categories: model-free, landmark-based, and 3D structure-based. Model-free methods \cite{55,45,46,52,7} realize compelling reenactment by modeling the relative motion field from the source to the driving faces. Notably, First Order Motion Model (FOMM) \cite{46} significantly improves the performance of image animation by using self-supervised keypoints with 2D motion warping. Our studies show that 2D warping is effective in generating

\footnote{1https://github.com/zengbohan0217/FNeVR}

high-quality motion transfer. However, it has a limited ability to produce sufficiently realistic images (see Fig. 1(c)). Landmark-based methods [2, 15, 56, 49, 57] introduce 2D facial landmarks and obtain considerable progress in face animation, but they are intrinsically disadvantageous in identity preservation. Motivated by the advanced 3D Morphable Models (3DMMs) [3, 40, 16, 4, 5, 6, 31], researchers tend to integrate the 3D geometric prior to condition synthesis and achieve impressive performance to generate realistic images [30, 12]. Nonetheless, these 3D structure-based methods often focus on introducing 3D facial information in the motion model to improve the authenticity of the results, such as Face vid2vid [51], neglecting the essential advantages of 2D warping in motion transfer. As shown in Fig. 1(d), Face vid2vid produces a clear head profile but with poor facial details.

In this paper, we propose a Face Neural Volume Rendering (FNeVR) network for more realistic face animation, by taking the merits of 2D motion warping [46] on facial expression transformation and 3D volume rendering on high-quality image synthesis in a unified framework. Inspired by the remarkable properties of neural rendering [33, 22] in establishing high fidelity representations, we design a novel 3D Face Volume Rendering (FVR) module for face animation, which enables our FNeVR to learn more realistic facial details efficiently and thus achieve photo-realistic results. Specifically, the proposed FVR module first adopts two convolutional neural networks (CNNs), which we design to extract the 3D color and shape information from the 2D warped feature with the aid of the 3D reconstruction results. We only apply the 3D reconstruction module to supervise the 3D CNNs in the training process without any overhead for inference, which is different from existing 3D structure-based methods [30, 12, 51]. After that, we introduce an orthogonal adaptive ray-sampling module to efficiently calculate the sampled color field and the voxel probability for subsequent image rendering. Instead of using hierarchical sampling as in [33], our orthogonal adaptive ray-sampling only needs one MLP to process the 3D information directly. Additionally, we design a perceptual loss to guarantee the quality of rendering feature maps. All in all, our FNeVR can not only generate more realistic images than 2D-based methods, but also obtain more accurate motion transfer than 3D-based methods (see Fig. 1(e)). Moreover, we present a Lightweight Pose Editing (LPE) module, which is simple yet effective and only takes Euler angle as the input to efficiently realize facial pose editing. We further design a new loss to supervise the face editing model.

The contributions of this paper include:

- A Face Neural Volume Rendering (FNeVR) network is presented to take the merits of 2D motion warping on face expression transformation and 3D volume rendering on high-quality image synthesis in a unified framework for realistic face animation.
- A Face Volume Rendering (FVR) module with orthogonal adaptive ray-sampling is proposed to capture facial details effectively and improve animation performance. Additionally, a simple yet effective Lightweight Pose Editing (LPE) module is presented only based on the Euler angle.
- Extensive experiments are conducted to compare FNeVR with state-of-the-art methods. The results show that our FNeVR obtains the best overall quality and performance on widely used talking-head benchmarks.

2 Related Work

Face Animation. Model-free methods [55, 45, 46, 52, 7, 51] complete the deformation of face images by establishing the motion field without extracting prior facial information in advance. Early models [55] can extract identity and pose information, but they may fail to capture the appropriate deformations in some problematic cases. MonkeyNet [45] handles motion transfer by acquiring
sparse keypoints. Subsequently, the First Order Motion Model (FOMM) [46] significantly improves the performance of face animation with a rigorous first-order mathematical model. While reporting promising synthesized images, most 2D model-free methods have limited ability to model out-of-plane rotations and expressional motions. Face vid2vid [51] further improves FOMM by introducing 3D representations to model a 3D motion field for face generation. Nevertheless, it has a considerable computational cost and still lacks the capability of expression transformation. Landmark-based methods [2; 15; 56; 19; 49; 57] utilize 2D facial landmarks as conditions for reenactment. However, since 2D facial landmarks cause the injection of identity-related information from the driving face to the generated one, these methods often cannot handle identity preservation well during the generation process.

In order to address the above issues, many works [28; 27; 30; 12; 41; 48] focus more on 3D structure-based methods by means of the geometric prior of 3D faces and produce impressive results on subject-agnostic face synthesis. HeadGAN [12] employs the rendered 3D mesh as input to synthesize deformed images, but it does not work well in facial expression transformation. Conditioned on the parameters of the 3D Morphable Model (3DMM) [3], StyleRig [27] and GIF [12] respectively employ pre-trained StyleGAN [25] and StyleGAN2 [26] to warp face images. PIRender [41] controls the face motions and predicts a flow field for deformation. Nonetheless, these methods are intrinsically limited to modeling the specific head components such as hair and teeth.

3D Morphable Face Models. The 3D Morphable Model (3DMM) [3] is a statistical model that provides 3D face representation for rendering face images by parameter modulation. Recent methods [49; 16; 4; 5; 6] extend 3DMM to more precise modeling of shape, expression, or pose on the basis of the original synthesis. Among these models, FLAME [31] additionally models the head pose, joint pose, and geometric expression information, offering more realistic and expressive 3D faces. Benefitting from 3DMM, the face reconstruction methods [11; 14; 1; 54] achieve significant progress. The representative method DECA [13] introduces an encoder to extract the parameters of the face for FLAME and employs another encoder to attain detailed reconstruction. In this paper, we exploit a pre-trained encoder of DECA to extract the parameters required by FLAME and then utilize FLAME to reconstruct a reliable 3D face representation from 2D images.

3D Volume Rendering. 3D volume rendering is used for rendering 2D projections of 3D objects and scenes. NeRF [33] is a breakthrough in view synthesis which can produce impressive rendered results based on the input positional information. Its core idea lies in optimizing a continuous volumetric scene function using a sparse set of input views. Thanks to its excellent performance, many works [43; 36; 37; 38; 8; 18; 59; 22] integrate NeRF with GANs to directly control the pose of synthesized result and generate impressive multi-view images. GRAF [43] introduces conditional GANs on the basis of NeRF essentially, which improves the quality and 3D consistency of the rendering results. GIRAFFE [38] can complete the rendering of a real-world scene and freely add multiple objects to the same scene. Differently, we introduce a lightweight and efficient adaptive volume rendering method, which only requires a single view to enhance the quality of the rendered 2D image.

3 Method

We formulate the face animation task as a talking-head generation with an editable pose. For a given source image $S$ and a driving video $\{D_1, D_2, \cdots, D_N\}$, where $D_i$ denotes the i-th frame in the video and $N$ is the total frame number, the objective of our FNeVR is to generate a photo-realistic talking-head video with the same face identity as $S$ and the motion and facial expressions derived from $\{D_1, D_2, \cdots, D_N\}$. As shown in Fig. 2, the whole framework mainly consists of four modules to achieve this goal: (1) 2D motion estimation, (2) 3D face reconstruction, (3) FVR, and (4) LPE. With $S$ and $D_i$ as input, we first utilize the 2D motion estimation module to estimate 2D facial keypoints and generate a motion field to transform the features. We also employ the 3D reconstruction module to build 3D face representations to provide the rendering network with precise and reliable 3D shape information. Then a novel FVR module is introduced to fully explore the capacity of the 2D motion module on the challenging expression transformation and generate images with high fidelity. In addition, the LPE module is added to extend our model to edit the facial pose efficiently.

3.1 2D Motion Estimation

Despite the trend that most existing methods [51; 21; 12] focus on studying various estimation methods of motion fields, extensive experimental results in [46; 48] show that the strategy of warping
We first extract the warped feature and 3D face reconstruction representation, on which FVR is
where FLAME serves as the decoder, using linear blend skinning (LBS) and additional expression
As shown in Fig. 3(b), we exploit the capability of a pre-trained encoder [13] to extract the parameters
where $f$ vectors), and expression $\psi$ are required by FLAME [31] and use the prior face knowledge of FLAME to reconstruct a 3D face
This process of obtaining the dense motion field and the warped motion fields with related weight masks $\{M_0,M_1,\cdots,M_K\}$ estimated by a U-Net used in FOMM. Furthermore, an additional occlusion map $O$ indicating regions to be inpainted is also predicted. The process of obtaining the dense motion field and the warped feature $F_w$ is:
$$\hat{T}_{S\leftarrow D}(z) = M_0 z + \sum_{k=1}^{K} M_k T_{S\leftarrow D,k}(z),$$
where $f_w(\cdot,\cdot)$ is the warping function, $F_S$ denotes the source feature and $\odot$ is the Hadamard product.

### 3D Face Reconstruction

As shown in Fig. 3(b), we exploit the capability of a pre-trained encoder [13] to extract the parameters required by FLAME [31] and use the prior face knowledge of FLAME to reconstruct a 3D face representation from 2D images. The reconstruction module is a typical encoder-decoder structure, where FLAME serves as the decoder, using linear blend skinning (LBS) and additional expression blend shapes to generate a detailed mesh $v \in \mathbb{R}^{3N}$ with $N = 5023$ vertices. Mathematically, given the parameters of shape $\beta \in \mathbb{R}^{\beta}$, pose $\theta \in \mathbb{R}^{\theta_K}$ (a global rotation vector and $K$ joints’ rotation vectors), and expression $\psi \in \mathbb{R}^{\psi}$, FLAME is represented by a blend skinning function $W$ as:
$$v = W(T_p(\beta, \theta, \psi), J(\beta), \theta, \psi),$$
where $T_p \in \mathbb{R}^{3N}$ depicts a head mesh in zero pose, $J \in \mathbb{R}^{3K}$ denotes the locations of the $K$ joints, and $\psi \in \mathbb{R}^{\psi}$ acts as blend weights. The camera parameter $c$ is further estimated to scale and translate the reconstructed vertices to the camera view. Meanwhile, the shape parameter $\beta$ of the source image is assigned to the driving image as the reconstruction target.
3.3 Face Volume Rendering (FVR)

We design a new neural rendering framework, Face Volume Rendering (FVR), for face animation as illustrated in the red box in Fig. 2. In this framework, we innovatively employ our designed CNNs to effectively extract the 3D color and shape features from the warped feature. We also introduce a new loss function to optimize the 3D shape feature with the 3D reconstruction result, which is only utilized during the training process without any overhead for inference. Moreover, we develop a new orthogonal adaptive ray-sampling module to estimate the sampled color field and voxel probability for image rendering in an efficient way. With these novel designs, our FVR module is capable of capturing more realistic facial details than 2D rendering. We also provide more details about our method in the supplementary materials.

3D Feature Extraction. Based on the warped feature $F_w$, we first need to extract the high dimensional 3D shape feature $F_\sigma$ and 3D color feature $F_{\text{color}}$ for subsequent face volume rendering. To this end, we respectively design two CNNs $P_\sigma$ and $P_{\text{color}}$ (see the supplementary materials), by which richer 3D shape and color features can be achieved than [33].

On the one hand, $P_\sigma$ calculates $F_\sigma = P_\sigma(F_w) \in \mathbb{R}^{H \times W \times D \times N_\sigma}$, where $D$ represents the spatial depth information and $N_\sigma$ is the number of channels representing the 3D shape information, which is set to 16 in this paper. Observing that the spatial mesh $v$ can be well reconstructed by the 3D face reconstruction module, it is thus reliable to utilize the 3D mesh feature $F_m$ derived from $v$ to supervise $P_\sigma$ to achieve a reasonable result. We maximize the inner product between vectorized $F_m$ and $F_\sigma$, which leads to a matching loss function $L_\sigma$:

$$L_\sigma = \exp(-\alpha_1 < F_\sigma \cdot F_m >) - \alpha_2,$$

(4)

where $\alpha_1$ and $\alpha_2$ are parameters to control the shape of the loss function. In practice, we empirically set $\alpha_1 = 10$ and $\alpha_2 = 0.9$. Here, we elaborate on computing $F_m$ below. Firstly, we conduct a down-sampling operation on the 3D vertices $v$ to get $N_{\text{down}}$ sampled vertices $v_{\text{down}}$, which helps to reduce the calculation cost and eliminate redundant vertex information. Additionally, we leverage the Gaussian function to process the $i$-th sampled vertex $v_{\text{down},i}$, forming a Gaussian heatmap $F_{m,i}$ with the size of $H \times W \times D \times 1$:

$$F_{m,i}(x) = \exp\left(-\frac{|x - v_{\text{down},i}|^2}{2\sigma}\right),$$

(5)

where $\sigma$ is the variance and we set $\sigma = 0.01$. Afterward, the 3D mesh feature of the reconstruction result $F_m$ with the size of $H \times W \times D \times 1$ is calculated by a weighted sum of all the Gaussian heatmaps $F_{m,i}$ as:

$$F_m = \sum_{i=1}^{N_{\text{down}}} w_i F_{m,i},$$

(6)

where $w_i$ is the weight for $F_{m,i}$. Notably, $w_i$ is predicted adaptively by an additional CNN. It is worth noting that the 3D reconstruction vertices are not needed during inference, and we can therefore complete the image synthesis in an efficient way.
On the other hand, we need to extract the 3D color feature from the 2D warped feature $F_w$. To this end, we use $P_{\text{color}}$ to estimate the 3D color feature $P_{\text{color}}$ with the size of $H \times W \times D \times N_{\text{color}}$:

$$F_{\text{color}} = P_{\text{color}}(F_w) \in \mathbb{R}^{H \times W \times D \times N_{\text{color}}},$$

where $N_{\text{color}}$ is the number of channels representing the color information.

Orthogonal Adaptive Ray-Sampling. With the extracted 3D features, we further calculate the sampled color field and the voxel probability, which are required for rendering. Different from conventional 3D volume rendering tasks [18, 23, 38], face animation can be considered as a single-view task, which casts rays from the camera into the scene in the direction orthogonal to the image plane. Accordingly, we design an orthogonal ray-sampling module to adaptively estimate the sampled color field $p_{\text{color}}$ and the voxel probability $p_\sigma$, which is achieved by a single MLP-based one-stage process $f_\theta$ as follows:

$$p_{\sigma}, p_{\text{color}} = f_\theta(F_\sigma, F_{\text{color}}) \in \mathbb{R}^{H \times W \times D \times M_{\text{color}}},$$

where $M_{\text{color}}$ is the number of channels representing the color information of $P_{\text{color}}$, and the interval sampling number $D$ is the same as the previous spatial channel number $D$. Note that $p_\sigma$ is the integral of the volume density with an adaptive interval size along the orthogonal ray. In particular, the selection of the interval size is involved in estimating the voxel probability $p_\sigma$ and adaptively adjusted by the MLP.

It is worth noting that unlike the hierarchical volume sampling as in [33], which requires a two-stage network to process multiple views according to the rays of different directions, our orthogonal adaptive ray-sampling only needs one MLP to deal with the assumed ray at the angle orthogonal to the image plane. Furthermore, instead of decoupling the facial attributes as geometric-prior guided sampling methods [22], we directly process the 3D related features to predict the sampled color field and the voxel probability more efficiently. In short, by simplifying the ray-sampling with an adaptive approach, our one-stage method can significantly increase the rendering speed compared with the common two-stage process. More details about the sampling module can refer to the supplementary materials.

Image Rendering. Specifically, we project the sampled color field $P_{\text{color}}$ and the estimated voxel probability $p_\sigma$ onto 2D to generate the rendering feature map $F_r$. Our rendering method is a modified version of that in [32] according to orthogonal adaptive ray-sampling. Also let $i$ be a pixel point in the feature map, and $j$ be a sample interval in each pixel $i$. The rendering method calculates the final color result $F_{r,i}$ for each pixel $i$ according to the corresponding sampled color field $p_{\text{color},i,j}$ and voxel probability $p_{\sigma,i,j}$ as:

$$F_{r,i} = \sum_{j=1}^{D} \tau_j \left(1 - \exp(-p_{\sigma,i,j})\right)p_{\text{color},i,j},$$

where $p_{\sigma,i,j}$ and $p_{\text{color},i,j}$ represent the voxel probability and the sampled color field of the $j$-th interval of the $i$-th pixel respectively, and $\tau_j = \exp(\sum_{k=1}^{j-1} - p_{\sigma,i,k})$ is the transmittance.

Moreover, to ensure the effectiveness of the rendering feature $F_r$, we introduce a two-layer CNN $P_t$ and the VGG perceptual loss [24, 50] in the training process. Specifically, we directly input $F_r$ into the CNN to generate an intermediate image $I_m$, and then compute the perceptual loss $L_R$ between $I_m$ and the driving frame $D_t$ as:

$$L_R = \text{VGG}_{\text{perceptual}}(D_t, I_m).$$

Afterward, we concatenate the volume rendering feature map $F_r$ and the warped feature $F_w$, and feed them into our designed rendering decoder which introduces the SPADE layer [39] to obtain the final generated result. This decoder (see the supplementary materials) can enhance the rendering quality while preserving facial details of the input.

3.4 Lightweight Pose Editing (LPE)

Existing face animation methods, which adopt a 2D warping motion network [46, 48, 21], work well in the feature warping but fail on pose editing. As illustrated in Fig. 3(a), we introduce a lightweight network (LPE) to add a new pose editing function for our framework. Taking the rotation
Table 1: Quantitative comparison of same-identity reconstruction on VoxCeleb [34].

<table>
<thead>
<tr>
<th>Method</th>
<th>$L_1$ ↓</th>
<th>LPIPS ↓</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
<th>AKD ↓</th>
<th>AED ↓</th>
<th>FID ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bilayer [56]</td>
<td>0.1197</td>
<td>0.4247</td>
<td>15.219</td>
<td>0.3968</td>
<td>12.60</td>
<td>129.48</td>
<td>219.8</td>
</tr>
<tr>
<td>FOMM [46]</td>
<td>0.0450</td>
<td>0.1099</td>
<td>23.210</td>
<td>0.7475</td>
<td>1.383</td>
<td>0.244</td>
<td>11.56</td>
</tr>
<tr>
<td>Face vid2vid [51]</td>
<td>0.0485</td>
<td>0.1051</td>
<td>22.642</td>
<td>0.7268</td>
<td>1.616</td>
<td>0.0395</td>
<td>9.142</td>
</tr>
<tr>
<td>Face vid2vid-S</td>
<td>0.0445</td>
<td>0.0901</td>
<td>23.357</td>
<td>0.7437</td>
<td>1.421</td>
<td>0.0243</td>
<td>9.151</td>
</tr>
<tr>
<td>DaGAN [21]</td>
<td>0.0462</td>
<td>0.0981</td>
<td>23.263</td>
<td>0.7536</td>
<td>1.441</td>
<td>0.0247</td>
<td>9.660</td>
</tr>
<tr>
<td>PiRender [41]</td>
<td>0.0566</td>
<td>0.0850</td>
<td>21.040</td>
<td>0.6550</td>
<td>2.186</td>
<td>0.2245</td>
<td>11.88</td>
</tr>
<tr>
<td>FNeVR (ours)</td>
<td>0.0404</td>
<td>0.0804</td>
<td>24.292</td>
<td>0.7773</td>
<td>1.254</td>
<td>0.0231</td>
<td>8.443</td>
</tr>
</tbody>
</table>

angle information $\{yaw \in \mathbb{R}^1, pitch \in \mathbb{R}^1, roll \in \mathbb{R}^1\}$ provided by the 3D face reconstruction module and the keypoint information of the source image as input. $\mathbf{L}$ estimates the target keypoints $\delta_{\text{value}}$ and their Jacobians $\delta_{\text{Jacobian}}$. In practice, we first formulate $yaw$, $pitch$ and $roll$ as a 3D rotation matrix $r_{\text{rotate}} \in \mathbb{R}^{3\times3}$, and then introduce an MLP as $f_{\text{editor}}$ to compute $\delta_{\text{value}}$ and $\delta_{\text{Jacobian}}$ under a specific rotation angle $\theta_{\text{rotate}}$:

$$\delta_{\text{value}}, \delta_{\text{Jacobian}} = f_{\text{editor}}(\theta_{\text{rotate}}, ps, Js) \in \mathbb{R}^{K\times2} \times \mathbb{R}^{K\times2\times2}.$$ (11)

Moreover, we design a loss $L_{\text{editor}}$ to supervise $f_{\text{editor}}$ by aligning $\delta_{\text{value}}$ and $\delta_{\text{Jacobian}}$ with the keypoints and the Jacobians of the driving frame, respectively:

$$L_{\text{editor}} = \lambda_1 L_1(pd, \delta_{\text{value}}) + \lambda_2 L_1(Jd, \delta_{\text{Jacobian}}),$$ (12)

where $\lambda_1$ and $\lambda_2$ are balancing parameters, and $L_1$ is the $l_1$ norm loss. Empirically we set $\lambda_1 = 1$ and $\lambda_2 = 0.5$.

3.5 Training

Our framework is implemented based on self-supervised learning or self-reenactment, where a pair of the source image $S$ and the driving image $D_i$ from the same speaker in a video is used as the input and $D_i$ also acts as the ground truth for supervision. We train our FNeVR by minimizing the following loss $L_{\text{total}}$:

$$L_{\text{total}} = L_P(D_i, x_{\text{generated}}) + L_G(D_i, x_{\text{generated}}) + L_E(pd, Jd) + L_R(D_i, I_m) + L_{\sigma}(F_x, F_m) + L_{\text{editor}}(pd, Jd, \delta_{\text{value}}, \delta_{\text{Jacobian}}),$$ (13)

where $x_{\text{generated}}$ is the output of the whole network FNeVR. $L_{\text{total}}$ consists of four parts: (1) $L_P$ and $L_G$ are used to maintain the quality of the generated image; (2) $L_E$ is used to adjust the 2D motion estimation module to predict reasonable motion fields; (3) $L_R$ and $L_{\sigma}$ are used to optimize the 3D volume rendering module to generate better 3D rendering features; (4) $L_{\text{editor}}$ is used to optimize the pose editor. For (1) and (2), we employ the same loss functions as in [46], including the perceptual loss $L_P$, GAN loss $L_G$ and equivalence constraint loss $L_{\sigma}$. For (3) and (4), the losses mainly provide the quality guarantee of the 3D rendering feature $F_x$.

4 Experiments

In addition to the extensive experiments described in this section, we also provide more results in the supplementary materials.

4.1 Implementation Details

Datasets. Our evaluation is performed on VoxCeleb [34], which contains more than 100,000 videos covering 1,251 speakers of different identities, and VoxCeleb2 [9], which contains about 1M videos of different celebrities. We adopt the same preprocessing strategy of cropping faces from the videos and resizing them to 256x256 as in [46].

Training Details. We train FNeVR with self reconstruction on the training set, in which a pair of source and driving images from the same speaker in a video are used as input, and the driving image also serves as ground truth for supervision. FNeVR is trained for 100 epochs, repeating the video
Figure 4: Qualitative comparison with SOTA baselines. **Top:** same-identity reconstruction; **Bottom:** cross-identity reenactment. For both tasks, our FNeVR can produce more realistic and fine-grained details than the others. For example, FOMM struggles to strike a balance between clarity and facial details, and Face vid2vid-S and DaGAN may cause distortions within some facial areas (eyes, mouth, and teeth).

### Table 2: Quantitative comparison of cross-identity reenactment on VoxCeleb [34] and VoxCeleb2 [9].

<table>
<thead>
<tr>
<th>Method</th>
<th>VoxCeleb</th>
<th>VoxCeleb2</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOMM</td>
<td>106.9</td>
<td>0.5491</td>
</tr>
<tr>
<td>Face vid2vid-S</td>
<td>106.6</td>
<td><strong>0.6447</strong></td>
</tr>
<tr>
<td>DaGAN</td>
<td>110.3</td>
<td>0.5305</td>
</tr>
<tr>
<td>FNeVR (ours)</td>
<td><strong>98.23</strong></td>
<td><strong>0.5505</strong></td>
</tr>
</tbody>
</table>

### Table 3: Quantitative comparison of Flops, memory, and efficiency.

<table>
<thead>
<tr>
<th>Method</th>
<th>Flops(G)</th>
<th>Params(M)</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face vid2vid</td>
<td>231.038</td>
<td>125.216</td>
<td>17.790</td>
</tr>
<tr>
<td>Face vid2vid-S</td>
<td>636.941</td>
<td>173.109</td>
<td>13.219</td>
</tr>
<tr>
<td>DaGAN</td>
<td>75.642</td>
<td>74.660</td>
<td>26.753</td>
</tr>
<tr>
<td>FNeVR (ours)</td>
<td>130.109</td>
<td><strong>61.378</strong></td>
<td><strong>36.568</strong></td>
</tr>
</tbody>
</table>

image set 75 times per epoch. We adopt Adam [29] optimizer with learning rate $\eta = 2 \times 10^{-4}$, $\beta_1 = 0.5$ and $\beta_2 = 0.9$ for each module. Since our model is lightweight, we only use 2 24GB NVIDIA 3090 GPUs during training.

**Evaluation Metrics.** The metrics include (1) reconstruction faithfulness using $\mathcal{L}_1$, PSNR, SSIM [53], and LPIPS [58], (2) output visual quality using FID [20], (3) Average Keypoint Distance (AKD) [46], (4) Average Euclidean Distance (AED) [46], and (5) identity preservation cosine similarity CSIM [42, 23].

### 4.2 Comparison with State-of-the-Art Methods

**Methods.** We compare our FNeVR with five state-of-the-art (SOTA) models: FOMM [46], Face vid2vid [51], Face vid2vid-S [51], Bilayer [56], DaGAN [21], and PIRenderer [41]. We use the official pre-trained models for Bilayer, FOMM, DaGAN and PIRenderer, and a widely recognized unofficial model for Face vid2vid due to the absence of the official code. Face vid2vid-S is obtained by replacing the decoder of Face vid2vid with a SPADE [39] layer-based decoder. All of these models are trained on VoxCeleb.

**Same-Identity Reconstruction.** We conduct a quantitative comparison with SOTA methods on the testing dataset of VoxCeleb when the source image and the driving image belong to the same person.
As shown in Table 1, it is evident that our FNeVR outperforms other SOTA methods on all metrics. Especially, FNeVR achieves pronounced improvements on several metrics, such as image quality (PSNR), semantic consistency (AKD), and image authenticity (FID). Fig. 4(a) shows that FNeVR can generate the most realistic images that are most similar to the driving faces.

Cross-Identity Reenactment. We compare the performance of our FNeVR with SOTA methods on the VoxCeleb and VoxCeleb2 testing datasets when the source image and driving video come from different persons. Concretely, we randomly select 10 source images and 14 driving videos from the testing datasets. Table 2 illustrates that FNeVR produces the best overall performance. Although Face vid2vid-S has a better result in terms of CSIM, it consumes several times more computation and memory cost (FLOPs and Parameters) than FNeVR, as indicated in Table 3, demonstrating FNeVR’s outstanding efficiency in compact computation. Moreover, Fig. 4(b) visualizes some synthesis results from different methods of reenactment and shows that FNeVR can generate more realistic faces whose poses and expressions are closest to the driving faces.

4.3 Ablation Study

We conduct comprehensive experiments on the same-identity reconstruction task to further demonstrate the effectiveness of each module in the proposed FNeVR. Specifically, we employ FOMM [46] as the baseline, and construct four comparison models: (1) replacing FOMM’s decoder with our designed decoder, (2) inserting FVR into FOMM, (3) FNeVR without $F_w$ inputted into the decoder, and (4) FNeVR without $L_\sigma$. The results are shown in Table 4. Both FVR and our designed decoder have significant effectiveness. In particular, FOMM with FVR inserted produces much better results than the baseline. Furthermore, FNeVR without $F_w$ performs worse than the full FNeVR, indicating that not inputting $F_w$ into the decoder will affect the model’s ability to transfer the facial poses and expressions, and lead to the loss of facial details in the generated results. Furthermore, the results show that if $L_\sigma$ is not introduced, all metrics are worse than the full FNeVR, indicating that it is necessary to introduce reliable 3D information. Meanwhile, even without the 3D information, FNeVR still performs well, demonstrating the great effectiveness of the proposed framework. Note that our full model produces the best overall results and surpasses other models a lot in several metrics, fully verifying the efficacy of FNeVR.

4.4 Lightweight Pose Editing

With the pose parameters, we conduct a qualitative experiment with pose editing visualization in Fig. 5 to further evaluate the effectiveness of the proposed LPE. Specifically, according to the Euler angle (yaw, pitch, and roll) provided by the 3D face reconstruction result, we separately adjust the rotation effect of the generated results. Fig. 5 shows that LPE can generate reliable face rotations according to a given Euler angle.
5 Conclusion

In this paper, we propose a Face Neural Volume Rendering (FNeVR) network for face animation, which unifies the 2D motion warping and 3D volume rendering in one framework. We follow the line of 2D motion warping and introduce the 3D face reconstruction module to provide reliable 3D shape information for the rendering process. We innovatively develop a Face Volume Rendering (FVR) module to enhance the facial details of the warped feature and generate high-quality faces. Moreover, we design a Lightweight Pose Editing (LPE) module, which can directly implement pose editing with rotation angles. Extensive experiments illustrate that our FNeVR achieves state-of-the-art performance.

Acknowledgements

This work was supported by “the Fundamental Research Funds for the Central Universities”, and the National Natural Science Foundation of China under Grant 62076016, Beijing Natural Science Foundation-Xiaomi Innovation Joint Fund L223024. Besides, we gratefully acknowledge the support of MindSpore\footnote{https://www.mindspore.cn/}, CANN (Compute Architecture for Neural Networks) and Ascend AI processor used for this research.

References


Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes], please refer to the supplemental material.
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes], please refer to the supplemental material.
   (b) Did you include complete proofs of all theoretical results? [Yes], please refer to the supplemental material.

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]