FNeVR: Neural Volume Rendering for Face Animation

Bohan Zeng[1,†], Boyu Liu[2,†], Hong Li[1], Xuhui Liu[1,*], Jianzhuang Liu[3], Dapeng Chen[4], Wei Peng[4], Baochang Zhang[1,5,*]

[1]Institute of Artificial Intelligence, School of Automation Science and Electrical Engineering, Beihang University, Beijing, P.R.China

[2]Sino-French Engineer School, Beihang University, Beijing, P.R.China

[3]Huawei Noah's Ark Lab, Shenzhen, P.R.China

[4]Huawei, Shenzhen, P.R.China

[5]Zhongguancun Laboratory, Beijing, P.R.China

[*]Corresponding author, email: xhliu@buaa.edu.cn, bczhang@buaa.edu.cn

[†]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[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 [14; 50]. 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 [65; 52; 59; 16]. 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 [63; 51; 52; 60; 9] realize compelling reenactment by modeling the relative motion field from the source to the driving faces. Notably, First Order Motion Model (FOMM) [52] 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

[1]https://github.com/zengbohan0217/FNeVR

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

(a) Source (b) Driving (c) FOMM (d) Face vid2vid (e) FNeVR

Figure 1: Illustration of face synthesis results. FOMM with 2D warping produces a blurred head profile but with promising pose transformations. Face vid2vid produces a clear head profile but with poor facial details. Our FNeVR can combine their merits and generate photo-realistic images.

high-quality motion transfer. However, it has a limited ability to produce sufficiently realistic images (see Fig. 1(c)). Landmark-based methods [2; 19; 64; 57; 65] 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) [4; 46; 20; 5; 6; 7; 36], researchers tend to integrate the 3D geometric prior to condition synthesis and achieve impressive performance to generate realistic images [35; 16]. 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 [59], 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 [52] 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 [39; 26] 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 [35; 16; 59]. 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 [39], 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 [63; 51; 52; 60; 9; 59] complete the deformation of face images by establishing the motion field without extracting prior facial information in advance. Early models [63] can extract identity and pose information, but they may fail to capture the appropriate deformations in some problematic cases. MonkeyNet [51] handles motion transfer by acquiring

sparse keypoints. Subsequently, the First Order Motion Model (FOMM) [52] 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 [59] 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; 19; 64; 23; 57; 65] 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 [33; 32; 35; 16; 47; 56] 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 [16] 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) [4], StyleRig [55] and GIF [21] respectively employ pre-trained StyleGAN [30] and StyleGAN2 [31] to warp face images. PIRenderer [47] 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) [4] is a statistical model that provides 3D face representation for rendering face images by parameter modulation. Recent methods [46; 20; 5; 6; 7] extend 3DMM to more precise modeling of shape, expression, or pose on the basis of the original synthesis. Among these models, FLAME [36] 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 [15; 18; 1; 62] achieve significant progress. The representative method DECA [17] 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 [39] 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 [49; 42; 43; 44; 10; 22; 67; 26] integrate NeRF with GANs to directly control the pose of synthesized result and generate impressive multi-view images. GRAF [49] introduces conditional GANs on the basis of NeRF essentially, which improves the quality and 3D consistency of the rendering results. GIRAFFE [44] 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 [59; 25; 16] focus on studying various estimation methods of motion fields, extensive experimental results in [52; 56] show that the strategy of warping

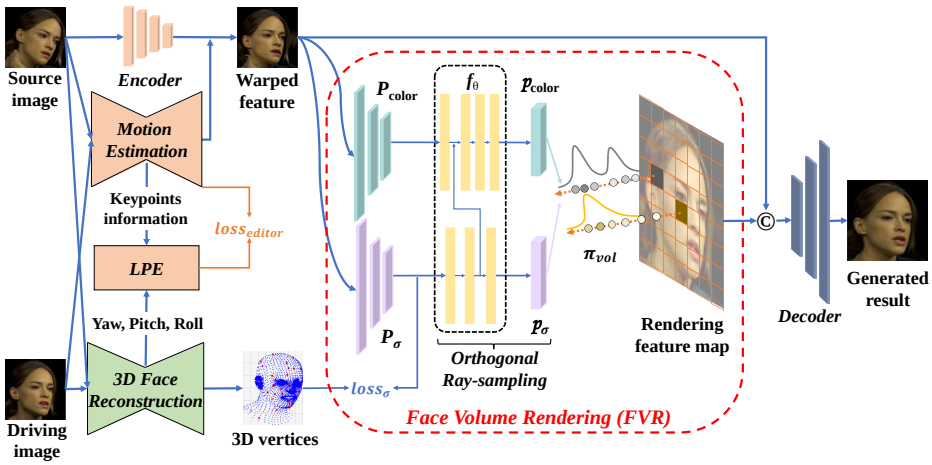


Figure 2: Overview of the proposed FNeVR, where © indicates the channel-wise concatenation. We first extract the warped feature and 3D face reconstruction representation, on which FVR is then performed to obtain the 3D shape and color information for face rendering. The 3D face reconstruction result supervises LPE and FVR by providing pose angles and 3D face vertices.

2D features achieves a consistent motion transfer of facial details. Based on this observation, we utilize the 2D motion estimation module proposed in FOMM [52] to complete the warping process. As shown in Fig. 3(a), we use an Hourglass Network [41] as the keypoint detector to extract $K = 10$ keypoints $\{p_{S,k}, p_{D,k} \in \mathbb{R}^2\}$ and their Jacobians $\{J_{S,k}, J_{D,k} \in \mathbb{R}^{2\times 2}\}$ of both source and driving images. For each keypoint, we utilize the first-order Taylor expansion to build an affine approximation motion filed according to [52]:

$$\mathcal{T}_{S\leftarrow D,k}(z) \approx p_{S,k} + J_{S,k}J_{D,k}^{-1}(z - p_{D,k}), \tag{1}$$

where $z \in \mathbb{R}^2$ indicates a location in the image. Apart from K sparse motion fields, an extra motion field is further introduced to preserve the static background information. To generate the final dense motion field, we aggregates these $K+1$ 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:

$$\begin{aligned} \hat{\mathcal{T}}_{S\leftarrow D}(z) &= M_0 z + \sum_{k=1}^{K} M_k \mathcal{T}_{S\leftarrow D,k}(z), \\ F_w &= O \odot f_w(F_S, \hat{\mathcal{T}}_{S\leftarrow D}), \end{aligned} \tag{2}$$

where $f_w(\cdot,\cdot)$ is the warping function, F_S denotes the source feature and $\odot$ is the Hadamard product.

3.2 3D Face Reconstruction

As shown in Fig. 3(b), we exploit the capability of a pre-trained encoder [17] to extract the parameters required by FLAME [36] 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 $\boldsymbol{\beta} \in \mathbb{R}^{|\boldsymbol{\beta}|}$, pose $\boldsymbol{\theta} \in \mathbb{R}^{3(K+1)}$ (a global rotation vector and K joints' rotation vectors), and expression $\boldsymbol{\psi} \in \mathbb{R}^{|\boldsymbol{\psi}|}$, FLAME is represented by a blend skinning function W as:

$$v = W(\boldsymbol{T_p}(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}), \boldsymbol{J}(\boldsymbol{\beta}), \boldsymbol{\theta}, \boldsymbol{\mathcal{W}}), \tag{3}$$

where $\boldsymbol{T_p} \in \mathbb{R}^{3N}$ depicts a head mesh in zero pose, $\boldsymbol{J} \in \mathbb{R}^{3K}$ denotes the locations of the K joints, and $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{K\times N}$ acts as blend weights. The camera parameter $\boldsymbol{c}$ is further estimated to scale and translate the reconstructed vertices to the camera view. Meanwhile, the shape parameter $\boldsymbol{\beta}$ of the source image is assigned to the driving image as the reconstruction target.

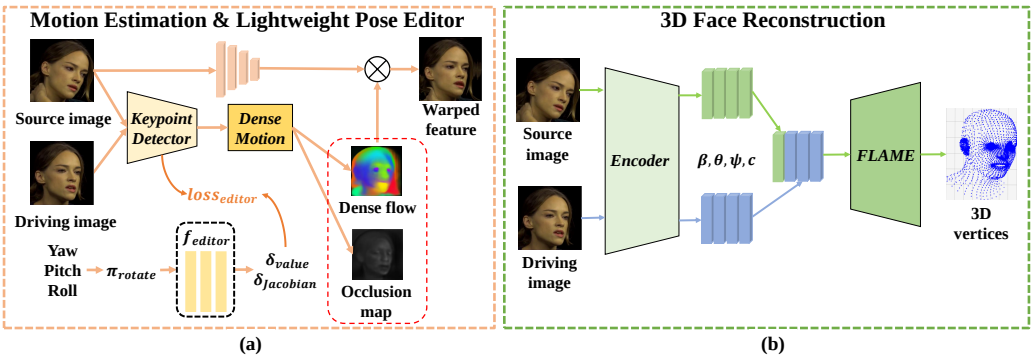


Figure 3: Illustration of Motion Estimation and LPE modules, and 3D Face Reconstruction module.

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_σ and 3D color feature F_{color} for subsequent face volume rendering. To this end, we respectively design two CNNs P_σ and P_{color} (see the supplementary materials), by which richer 3D shape and color features can be achieved than [39].

On the one hand, P_σ 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_σ 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_σ to achieve a reasonable result. We maximize the inner product between vectorized F_m and F_σ, which leads to a matching loss function $\mathcal{L}_\sigma$:

$$\mathcal{L}_\sigma = \exp(-\alpha_1 < F_\sigma \cdot F_m >) - \alpha_2, \tag{4}$$

where α_1 and α_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_{down} sampled vertices v_{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 $v_{down,i}$, forming a Gaussian heatmap $F_{m,i}$ with the size of $H \times W \times D \times 1$:

$$F_{m,i}(x) = \exp(-\frac{\| x - v_{down,i} \|^2}{2\sigma}), \tag{5}$$

where σ 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:

$$F_m = \sum_{i=1}^{N_{down}} w_i F_{m,i}, \tag{6}$$

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_{color} to estimate the 3D color feature F_{color} with the size of $H \times W \times D \times N_{color}$:

$$F_{color} = P_{color}(F_w) \in \mathbb{R}^{H \times W \times D \times N_{color}}, \tag{7}$$

where N_{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 [22; 49; 44], 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_{color} and the voxel probability p_σ, which is achieved by a single MLP-based one-stage process f_θ as follows:

$$p_\sigma, p_{color} = f_\theta(F_\sigma, F_{color}) \in \mathbb{R}^{H \times W \times D \times 1} \times \mathbb{R}^{H \times W \times D \times M_{color}}, \tag{8}$$

where M_{color} is the number of channels representing the color information of p_{color}, and the interval sampling number D is the same as the previous spatial channel number D. Note that p_σ 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_σ and adaptively adjusted by the MLP.

It is worth noting that unlike the hierarchical volume sampling as in [39], 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 [26], 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_{color} and the estimated voxel probability p_σ onto 2D to generate the rendering feature map F_r. Our rendering method is a modified version of that in [38] 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 $p_{color,i,j}$ and voxel probability $p_{\sigma,i,j}$ as:

$$F_{r,i} = \sum_{j=1}^{D} \tau_j \left(1 - \exp(-p_{\sigma,i,j})\right) p_{color,i,j}, \tag{9}$$

where $p_{\sigma,i,j}$ and $p_{color,i,j}$ 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 [28; 58] 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 $\mathcal{L}_R$ between I_m and the driving frame D_i as:

$$\mathcal{L}_R = VGG_{perceptual}(D_i, I_m). \tag{10}$$

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 [45] 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 [52; 56; 25], 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 [40].

Method	$\mathcal{L}_1 \downarrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	AKD $\downarrow$	AED $\downarrow$	FID $\downarrow$
Bilayer [64]	0.1197	0.4247	15.219	0.3968	12.60	0.0546	219.8
FOMM [52]	0.0450	0.1099	23.210	0.7475	1.383	0.0244	11.56
Face vid2vid [59]	0.0485	0.1051	22.642	0.7268	1.616	0.0395	9.142
Face vid2vid-S [59]	0.0445	0.0901	23.357	0.7473	1.421	0.0243	9.151
DaGAN [25]	0.0462	0.0981	23.263	0.7536	1.441	0.0247	9.660
PIRender [47]	0.0566	0.0850	21.040	0.6550	2.186	0.2245	11.88
FNeVR (ours)	**0.0404**	**0.0804**	**24.292**	**0.7773**	**1.254**	**0.0231**	**8.443**

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, LPE (f_{editor}) estimates the target keypoints δ_{value} and their Jacobians $\delta_{Jacobian}$. In practice, we first formulate yaw, $pitch$ and $roll$ as a 3D rotation matrix $r_{rotate} \in \mathbb{R}^{3\times3}$, and then introduce an MLP as f_{editor} to compute δ_{value} and $\delta_{Jacobian}$ under a specific rotation angle θ_{rotate}:

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

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

$$\mathcal{L}_{editor} = \lambda_1 L_1(p_D, \delta_{value}) + \lambda_2 L_1(J_D, \delta_{Jacobian}), \tag{12}$$

where λ_1 and λ_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 $\mathcal{L}_{total}$:

$$\begin{aligned}\mathcal{L}_{total} = \mathcal{L}_P(D_i, x_{generated}) + \mathcal{L}_G(D_i, x_{generated}) + \mathcal{L}_E(p_D, J_D) + \\ \mathcal{L}_R(D_i, I_m) + \mathcal{L}_\sigma(F_\sigma, F_m) + \mathcal{L}_{editor}(p_D, J_D, \delta_{value}, \delta_{Jacobian}),\end{aligned} \tag{13}$$

where $x_{generated}$ is the output of the whole network FNeVR. $\mathcal{L}_{total}$ consists of four parts: (1) $\mathcal{L}_P$ and $\mathcal{L}_G$ are used to maintain the quality of the generated image; (2) $\mathcal{L}_E$ is used to adjust the 2D motion estimation module to predict reasonable motion fields; (3) $\mathcal{L}_R$ and $\mathcal{L}_\sigma$ are used to optimize the 3D volume rendering module to generate better 3D rendering features; (4) $\mathcal{L}_{editor}$ is used to optimize the face pose editor. For (1) and (2), we employ the same loss functions as in [52], including the perceptual loss $\mathcal{L}_P$, GAN loss $\mathcal{L}_G$ and equivalence constraint loss $\mathcal{L}_E$. For (3) and (4), the losses mainly provide the quality guarantee of the 3D rendering feature F_r.

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 [40], which contains more than 100,000 videos covering 1,251 speakers of different identities, and VoxCeleb2 [12], which contains about 1M videos of different celebrities. We adopt the same preprocessing strategy of cropping faces from the videos and resizing them to 256×256 as in [52].

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 comparision of cross-identity reenactment on VoxCeleb [40] and VoxCeleb2 [12].

Method	VoxCeleb		VoxCeleb2	
	FID↓	CSIM ↑	FID↓	CSIM ↑
FOMM	106.9	0.5491	138.1	0.5228
Face vid2vid-S	106.6	**0.6447**	148.6	**0.6290**
DaGAN	110.3	0.5305	139.6	0.4932
FNeVR (ours)	**98.23**	0.5505	**133.9**	0.5282

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

Method	Flops(G)	Params(M)	FPS
Face vid2vid	231.038	125.216	17.790
Face vid2vid-S	636.941	173.109	13.219
DaGAN	**75.642**	74.660	26.753
FNeVR (ours)	130.109	**61.378**	**36.568**

image set 75 times per epoch. We adopt Adam [34] 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 [61], and LPIPS [66], (2) output visual quality using FID [24], (3) Average Keypoint Distance (AKD) [52], (4) Average Euclidean Distance (AED) [52], and (5) identity preservation cosine similarity CSIM [48; 27].

4.2 Comparison with State-of-the-Art Methods

Methods. We compare our FNeVR with five state-of-the-art (SOTA) models: FOMM [52], Face vid2vid [59], Face vid2vid-S [59], Bilayer [64], DaGAN [25], and PIRenderer [47]. 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 [45] layer-based decoder. All of these models are trained on VoxCeleb.

Figure 5: Visualization of the pose editing results produced by LPE.

Table 4: Ablation study for same-identity reconstruction on the VoxCeleb testing set.

Configurations	$\mathcal{L}_1 \downarrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	AKD $\downarrow$	AED $\downarrow$	FID $\downarrow$
FOMM baseline	0.0450	0.1099	23.210	0.7475	1.383	0.0244	11.56
FOMM + SPADE	0.0426	0.0889	24.006	0.7710	1.316	0.0246	11.13
FOMM + FVR	**0.0404**	0.0875	**24.374**	**0.7783**	1.290	0.0233	10.76
FNeVR w/o F_w	0.0460	0.0854	22.653	0.7423	1.367	0.0262	12.07
FNeVR w/o $\mathcal{L}_\sigma$	0.0412	0.0869	24.163	0.7705	1.284	0.0239	9.024
FNeVR (ours)	**0.0404**	**0.0804**	24.292	0.7773	**1.254**	**0.0231**	**8.443**

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 (PNSR), 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 [52] 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 $\mathcal{L}_\sigma$. 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 $\mathcal{L}_\sigma$ 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[2], CANN (Compute Architecture for Neural Networks) and Ascend AI processor used for this research.

References

[1] Riza Alp Guler, George Trigeorgis, Epameinondas Antonakos, Patrick Snape, Stefanos Zafeiriou, and Iasonas Kokkinos. Densereg: Fully convolutional dense shape regression in-the-wild. In *CVPR*, 2017.
[2] Hadar Averbuch-Elor, Daniel Cohen-Or, Johannes Kopf, and Michael F Cohen. Bringing portraits to life. *TOG*, 2017.
[3] Tadas Baltrušaitis, Peter Robinson, and Louis-Philippe Morency. Openface: an open source facial behavior analysis toolkit. In *WACV*, 2016.
[4] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.
[5] Timo Bolkart and Stefanie Wuhrer. A groupwise multilinear correspondence optimization for 3d faces. In *ICCV*, 2015.
[6] Timo Bolkart and Stefanie Wuhrer. A robust multilinear model learning framework for 3d faces. In *CVPR*, 2016.
[7] James Booth, Anastasios Roussos, Stefanos Zafeiriou, Allan Ponniah, and David Dunaway. A 3d morphable model learnt from 10,000 faces. In *CVPR*, 2016.
[8] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, 2017.
[9] Egor Burkov, Igor Pasechnik, Artur Grigorev, and Victor Lempitsky. Neural head reenactment with latent pose descriptors. In *CVPR*, 2020.
[10] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021.
[11] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv:2003.04297*, 2020.
[12] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. *INTERSPEECH*, 2018.
[13] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019.
[14] Zhigang Deng and Junyong Noh. Computer facial animation: A survey. In *Data-Driven 3D Facial Animation*. Springer, 2008.
[15] Pengfei Dou, Shishir K Shah, and Ioannis A Kakadiaris. End-to-end 3d face reconstruction with deep neural networks. In *CVPR*, 2017.
[16] Michail Christos Doukas, Stefanos Zafeiriou, and Viktoriia Sharmanska. Headgan: One-shot neural head synthesis and editing. In *ICCV*, 2021.
[17] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *TOG*, 2021.
[18] Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, and Xi Zhou. Joint 3d face reconstruction and dense alignment with position map regression network. In *ECCV*, 2018.
[19] Jiahao Geng, Tianjia Shao, Youyi Zheng, Yanlin Weng, and Kun Zhou. Warp-guided gans for single-photo facial animation. *TOG*, 2018.

[2] https://www.mindspore.cn/

[20] Thomas Gerig, Andreas Morel-Forster, Clemens Blumer, Bernhard Egger, Marcel Luthi, Sandro Schönborn, and Thomas Vetter. Morphable face models-an open framework. In *FG*, 2018.

[21] Partha Ghosh, Pravir Singh Gupta, Roy Uziel, Anurag Ranjan, Michael J Black, and Timo Bolkart. Gif: Generative interpretable faces. In *3DV*, 2020.

[22] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv:2110.08985*, 2021.

[23] Sungjoo Ha, Martin Kersner, Beomsu Kim, Seokjun Seo, and Dongyoung Kim. Marionette: Few-shot face reenactment preserving identity of unseen targets. In *AAAI*, 2020.

[24] 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. *Asian Journal of Applied Science and Engineering*, 2017.

[25] Fa-Ting Hong, Longhao Zhang, Li Shen, and Dan Xu. Depth-aware generative adversarial network for talking head video generation. In *CVPR*, 2022.

[26] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *CVPR*, 2022.

[27] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. Curricularface: adaptive curriculum learning loss for deep face recognition. In *CVPR*, 2020.

[28] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

[29] James T. Kajiya and Brian Von Herzen. Ray tracing volume densities. In *SIGGRAPH*, 1984.

[30] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[31] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

[32] Hyeongwoo Kim, Mohamed Elgharib, Michael Zollhöfer, Hans-Peter Seidel, Thabo Beeler, Christian Richardt, and Christian Theobalt. Neural style-preserving visual dubbing. *TOG*, 2019.

[33] Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Niessner, Patrick Pérez, Christian Richardt, Michael Zollhöfer, and Christian Theobalt. Deep video portraits. *TOG*, 2018.

[34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[35] Mohammad Rami Koujan, Michail Christos Doukas, Anastasios Roussos, and Stefanos Zafeiriou. Head2head: Video-based neural head synthesis. In *FG*, 2020.

[36] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *TOG*, 2017.

[37] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017.

[38] Nelson Max. Optical models for direct volume rendering. *TVCG*, 1995.

[39] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[40] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *INTERSPEECH*, 2017.

[41] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[42] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *ICCV*, 2019.

[43] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. In *NeurIPS*, 2020.

[44] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021.

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

[46] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *AVSS*, 2009.

[47] Yurui Ren, Ge Li, Yuanqi Chen, Thomas H Li, and Shan Liu. Pirenderer: Controllable portrait image generation via semantic neural rendering. In *ICCV*, 2021.

[48] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *CVPR*, 2021.

[49] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *NeurIPS*, 2020.

[50] Tong Sha, Wei Zhang, Tong Shen, Zhoujun Li, and Tao Mei. Deep person generation: A survey from the perspective of face, pose and cloth synthesis. *arXiv:2109.02081*, 2021.

[51] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. Animating arbitrary objects via deep motion transfer. In *CVPR*, 2019.

[52] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *NeurIPS*, 2019.

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[54] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

[55] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. In *CVPR*, 2020.
[56] Qiulin Wang, Lu Zhang, and Bo Li. Safa: Structure aware face animation. In *3DV*, 2021.
[57] Ting-Chun Wang, Ming-Yu Liu, Andrew Tao, Guilin Liu, Bryan Catanzaro, and Jan Kautz. Few-shot video-to-video synthesis. In *NeurIPS*, 2019.
[58] 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.
[59] Ting-Chun Wang, Arun Mallya, and Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *CVPR*, 2021.
[60] Yaohui Wang, Di Yang, Francois Bremond, and Antitza Dantcheva. Latent image animator: Learning to animate images via latent space navigation. *arXiv:2203.09043*, 2022.
[61] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004.
[62] Huawei Wei, Shuang Liang, and Yichen Wei. 3d dense face alignment via graph convolution networks. *arXiv:1904.05562*, 2019.
[63] Olivia Wiles, A Koepke, and Andrew Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *ECCV*, 2018.
[64] Egor Zakharov, Aleksei Ivakhnenko, Aliaksandra Shysheya, and Victor Lempitsky. Fast bi-layer neural synthesis of one-shot realistic head avatars. In *ECCV*, 2020.
[65] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *ICCV*, 2019.
[66] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.
[67] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis. *arXiv:2110.09788*, 2021.

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]

Appendix

A Derivation of the Volume Rendering Formula

This section provides the derivation of our 3D volume rendering formula Eq. 9 in the paper. Previous methods adopt a two-stage architecture to realize multi-view synthesis discretely, which is very time-consuming with redundant information. Hence we propose an orthogonal adaptive ray-sampling module to adaptively estimate the attributes required for image rendering of single-view face animation.

A.1 Classical Discrete Volume Rendering Formula

Generally speaking, the volume rendering formula needs to be transformed from continuous integration to discrete summation, for which [39] designs a two-stage sampling method to realize the multi-view synthesis of a single object. Specifically, the classical volume rendering [29] calculates the pixel color along the camera ray as follows:

$$\begin{aligned} C &= \int_{t_s}^{t_e} \tau(t)\sigma(t)c(t)dt, \\ \tau(t) &= \exp(-\int_{t_s}^{t} \sigma(s)ds), \end{aligned} \tag{14}$$

where σ represents the volume density, c is the emitted color, $[t_s, t_e]$ is the integral interval, and τ is the transmittance. To make it implementable, the continuous integration is transformed to discrete summation with D intervals. With some manipulations, [38] shows:

$$C = \sum_{i=1}^{D} c_i \tau_i (1 - \exp(-\sigma_i \delta_i)), \tag{15}$$

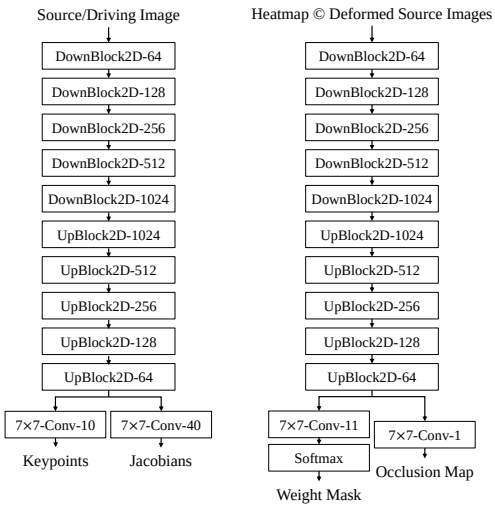
where $\delta_i = t_{i+1} - t_i$ is the i-th interval size, σ_i and c_i are constants to approximate the volume density and color values in $[t_i, t_{i+1}]$, respectively, and $\tau_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$.

A.2 Volume Rendering of FNeVR

We introduce an orthogonal ray adaptive sampling module for single-view face animation. Following [39], we perform ray casting in the normalized device coordinate (NDC) space. For each image, we define $N = H \times W$ rays, where W and H are the width and height of the input image, respectively. Each ray is orthogonal to the near clipping plane, and the ray path from the pixel centre is calculated. According to Eq. 15 above, conventional neural rendering first uses a "coarse" network to roughly estimate the computational contribution of each uniform sampling location, and then further performs non-uniform discretization for a more informed sampling of points, which are inputted to another "fine" network for accurate estimation. Instead, our FVR only introduces one MLP network to directly estimate the voxel probability p_σ of each voxel, which is the integral of the volume density σ within a suitable interval size δ. Therefore, for our model, the selection of the interval size δ is actually involved in the estimation of the voxel probability p_σ and adaptively adjusted by the MLP. Finally, we rewrite Eq. 15 as the weighted sum of all sampled colors p_{color} for each ray as:

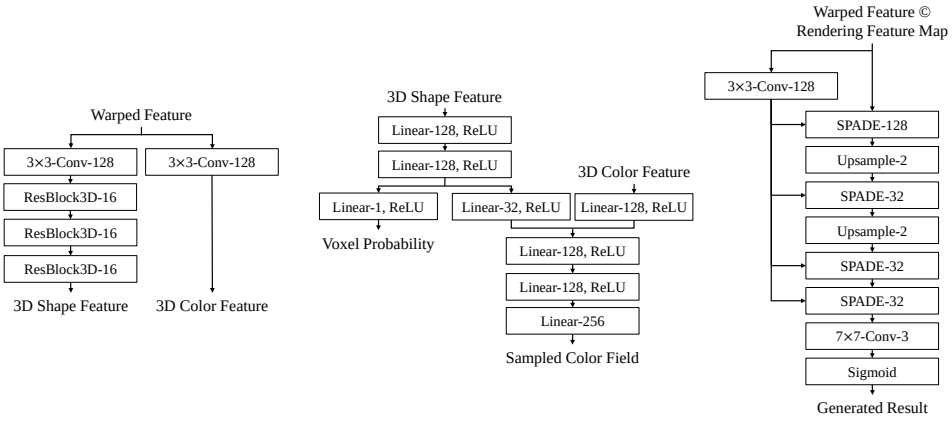
$$\begin{aligned} C &= \sum_{i=1}^{D} \tau_i (1 - \exp(-p_{\sigma,i})) p_{color,i} \\ \tau_i &= \exp(-\sum_{j=1}^{i-1} p_{\sigma,j}), \end{aligned} \tag{16}$$

which is Eq. 9 in the main paper. It is worth noting that our FVR can be applied to more generation tasks as long as appropriate 3D shape and color information is provided. For example, if we introduce reliable 3D body information to FVR, FVR can be applied to generate photo-realistic images of the body.



(a) Keypoint Detector **(b) Dense Motion Network**

Figure 6: Architecture of the motion estimation in our framework. ©: channel-wise concatenation. For the basic blocks UpBlock2D and DownBlock2D, please refer to Fig. 8(c) and Fig. 8(d), respectively.



(a) 3D Feature Extractors **(b) Orthogonal Adaptive Ray-Sampler** **(c) Rendering Decoder**

Figure 7: Architecture of the rendering component in our framework. ©: channel-wise concatenation. For the basic blocks ResBlock3D and SPADE, please refer to Fig. 8(e) and Fig. 8(b), respectively.

B Implementation Details

B.1 Architecture Details

Keypoint Detector. We employ the keypoint detector proposed by [52], which takes one image as input and adopts a UNet-based encoder-decoder architecture to extract the feature for keypoints prediction. Since we adopt a 2D warping strategy, only 2D CNN is required. Its detailed structure is shown in Fig. 6(a).

Dense Motion Network. We use the dense motion network proposed by [52] to predict an occlusion map indicating the regions to be inpainted and a group of weight masks related to the K sparse motion fields, which are calculated by each pair of keypoints $\{p_{S,k}, p_{D,k}\}$ with their Jacobians. Then we can aggregate the sparse motions by a weighted sum to obtain the final dense motion field. The detailed structure of the dense motion network is shown in Fig. 6(b). In particular, the input to the network is the concatenation of K heatmaps and $K+1$ deformed source images, where the heatmaps are calculated by the keypoints, and the deformed images are transformed by $K+1$ motion fields.

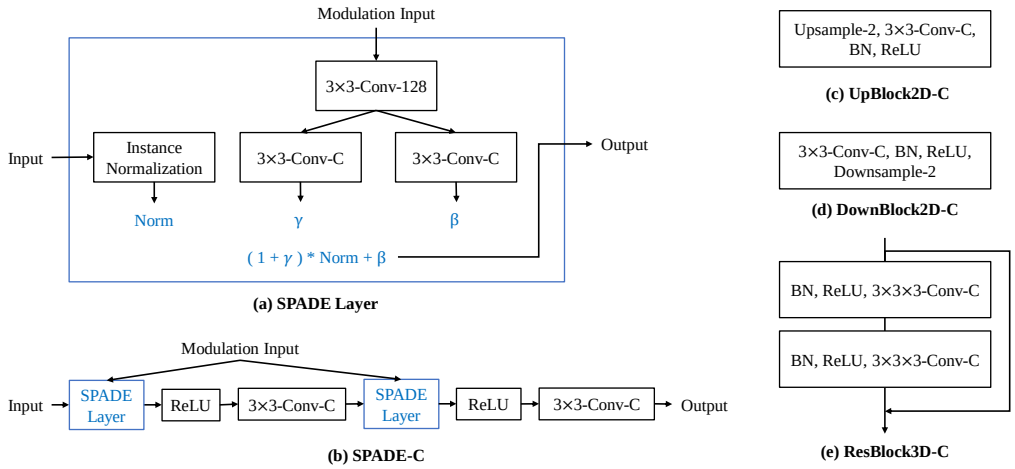


Figure 8: Architectures of the basic blocks.

3D Feature Extractor. Since we need to extract the 3D information of the warped feature for subsequent image rendering, we design a color extractor and a shape extractor, both of which first refine the 2D feature with the size of $256 \times 64 \times 64$ into 3D features with the size of $16 \times 8 \times 64 \times 64$. The detailed structure of the 3D feature extractors is shown in Fig. 7(a). N_σ and N_{color} are the final dimensions of the outputs of the color extractor and the shape extractor, respectively, and they are both set to 16.

Orthogonal Adaptive Ray-Sampler. Based on the extracted 3D features above, we can further calculate the sampled color field and the voxel probability, which are required for volume rendering. We adopt an MLP-based network to estimate the sampled color field with the size of $64 \times 64 \times 8 \times 256$ and the voxel probability with the size of $64 \times 64 \times 8 \times 1$. The detailed structure of the orthogonal adaptive ray-sampler is shown in Fig. 7(b).

Rendering Decoder. We design a rendering decoder based on the SPADE layer [45]. We adopt a series of the SPADE layers and upsampling layers to transform the warped feature and the volume rendering results into a photo-realistic image. The detailed structure of this decoder is shown in Fig. 7(c).

B.2 Details of the Loss Functions

Perceptual Loss. During the training process, we use the VGG perceptual loss [28] to align the high-level semantic feature of the generated image with that of the ground truth. Specifically, we use the pre-trained VGG-19 [53] model f_{VGG} to extract L features of the generated image $x_{generated}$ and the driving frame D_i and use the L_1 norm loss for similarity evaluation. Following [52], we first adopt the multi-scale pyramid by downsampling the generated image and ground truth to obtain four different resolution results, and then we aggregate the losses of the four resolutions to obtain the final result:

$$\mathcal{L}_P = \sum_{j=1}^{4} \sum_{l=1}^{L} L_1(f_{VGG,l}(D_{i,j}), f_{VGG,l}(x_{generated,j})). \tag{17}$$

GAN Loss. We use a discriminator D_{GAN} with N feature extractors to distinguish the generated results from the ground truth in M scales. To be specific, we use the same generator-discriminator GAN loss as [52], including the least square loss [37] and the feature matching loss [58] to train an adversarial generative network:

$$\mathcal{L}_{G,generator,square} = \sum_{m=1}^{M} \mathbb{E}[(1 - D_{GAN,m}(x_{generated,m}))^2], \tag{18}$$

$$\mathcal{L}_{G,generator,matching} = \mathbb{E} \sum_{m=1}^{M} \sum_{n=1}^{N} L_1(D^{(n)}_{GAN,m}(D_{i,m}), D^{(n)}_{GAN,m}(x_{generated,m})), \tag{19}$$

$$\mathcal{L}_{G,discriminator} = \sum_{m=1}^{M} \mathbb{E}[(1 - D_{GAN,m}(D_{i,m}))^2] + \mathbb{E}[(D_{GAN,m}(x_{generated,m}))^2], \tag{20}$$

where we denote the m-th scale discriminator as $D_{GAN,m}$ and its n-th layer feature extractor as $D^{(n)}_{GAN,m}$. With the GAN loss, we further improve the authenticity of the generated images.

Equivariance Loss. To ensure the validity and consistency of the unsupervised keypoint detector, we use a known geometric transformation $\mathcal{T}$ to provide the equivariance constraints as stated in [52]. Concretely, for the keypoints $\{p_{\mathcal{T}(D),k}\}$, which are extracted from the transformed image $\mathcal{T}(D)$, and the keypoints $\{p_{D,k}\}$ of the original image D, we have:

$$\mathcal{L}_{E,keypoints} = \sum_{k=1}^{K} L_1(p_{D,k} - \mathcal{T}^{-1}(p_{\mathcal{T}(D),k})), \tag{21}$$

where $\mathcal{T}^{-1}$ is the inverse transformation of $\mathcal{T}$. Furthermore, we optimize the Jacobians $\{J_{D,k}\}$ by:

$$\mathcal{L}_{E,Jacobians} = \sum_{k=1}^{K} L_1(\mathbb{1} - J^{-1}_{D,k} \cdot J_{\mathcal{T}^{-1}} \cdot J_{\mathcal{T}(D),k}), \tag{22}$$

where $\mathbb{1}$ denotes the 2×2 identity matrix, $J_{\mathcal{T}^{-1}}$ is the Jacobian of the inverse geometric transformation $\mathcal{T}^{-1}$, and $\{J_{\mathcal{T}(D),k}\}$ are the Jacobians detected from the transformed image $\mathcal{T}(D)$.

C Experiment Details

C.1 Evaluation Metrics

We quantitatively compare our FNeRV with other state-of-the-art methods using the following metrics:

$\mathcal{L}_1$ Distance. We compute the averaged $\mathcal{L}_1$ distance between the generated and ground-truth images.

Learned Perceptual Image Patch Similarity (LPIPS). LPIPS [66] estimates the perceived distance of the generated image from the ground truth image by computing the cosine distances between the VGG network [53] features of the two images layer by layer and averaging them.

Peak Signal-to-Noise Ratio (PSNR). Using the ratio of the maximum possible power of the ground truth image to the mean square error between the reconstructed image and the ground truth to measure the image reconstruction quality.

Structure Similarity Index Measure (SSIM). From the perspective of image distortion modelling discussed in [61], the structural similarity of two images is measured by considering three different factors, brightness, contrast, and structure, where the mean is used as an estimate of brightness, the standard deviation as an estimate of contrast, and the covariance as a measure of structural similarity.

Average Keypoint Distance (AKD). We use a facial landmark detector [8] to detect facial landmarks from the reconstructed image and the ground-truth image and calculate the average distance between the two groups of landmarks. AKD can measure the accuracy of the facial posture of the generated frame and driving frame.

Average Euclidean Distance (AED). We use the face recognition network [3] to calculate the feature representations of the ground-truth image and the generated video frame and calculate their averaged Euclidean distance.

Source Sample 1 Sample 2 Sample 3 Sample 4 Sample 5

Figure 9: Illustration of same identity with different poses and expressions.

Table 5: Comparison of CSIM for the same identity with different poses and expressions.

Sample	MoCo CSIM ↑	Arc CSIM ↑	Curricular CSIM ↑
Sample 1	0.9853	0.8327	0.9131
Sample 2	0.9962	0.9655	0.9663
Sample 3	0.9203	0.7725	0.6524
Sample 4	0.9034	0.6101	0.7626
Sample 5	0.9327	0.7307	0.7074

Frechet Inception Distance score (FID). FID [24] is a metric to evaluate the image quality created by the generated model by imitating human perception of image similarity. It uses inception V3 [54] to compare the distribution of the generated image with the distribution of the real image used to train the generator.

Cosine Similarity (CSIM). The cosine similarity measures the difference between two individuals by using the cosine value of the angle between two vectors in a vector space. We use the pre-trained face recognition model [27] to generate embedded vectors to compute the cosine similarity to assess the quality of identity preservation.

In the same-identity case, we perform an experiment on all test videos from the Voxceleb [40] dataset. We treat the first frame as the source portrait and all the other frames as the driving video. Hence, we can get the synthesized frames with corresponding ground truth frames. We can directly calculate the evaluation metrics of $\mathcal{L}_1$, LPIPS, PSNR, SSIM, AKD, AED and FID between the real images and the synthetic images. For the cross-identity case, we randomly select 10 source images and 14 driving videos from each testing set of the Voxceleb and Voxceleb2 [12]. In this case, we do not have ground-truth images, so we can only calculate FID between the driving images and the synthetic images, and CSIM between the source images and the synthetic images.

C.2 Reenactment Analysis

In the reenactment experiment, our FNeVR outperforms other SOTA methods on FID but has a certain gap with Face vid2vid [59] on CSIM. CSIM verifies the identity preservation according to the feature distance between the generated image and the source image, and is adopted in a number of studies [16; 59; 25]. According to our analysis, there are two reasons accounting for why CSIM of Face vid2vid is better than our FNeVR. On the one hand, 3D warping indeed provides a relatively stable face pose transformation with a high computational cost. On the other hand, we argue that it is not entirely reasonable to use CSIM to examine the preservation of face identity information, since 3D warping often leads to poor facial expression transformation, whereas it still produces good CSIM.

We think that expression and pose variation have a significant impact on CSIM, which can be proved by a simple experiment. Specially, we use three feature extraction networks (MoCo v2 [11], Arcface[13], and Curricularface [27]) to calculate the CSIM values between one source image and five sample images with the same person identity as the source image but different poses or expressions. As shown in Fig. 9, compared with the source image, Sample 1 and Sample 2 are slightly different in expressions, while Sample 3 has a great difference in pose. Meanwhile, Sample 4 and Sample 5 are disparate from the source image in both poses and expressions. The comparison of CSIM is shown in Table 5. Note that only Arc CSIM and Curricular CSIM focus on facial feature extraction. Obviously, despite the same identity, the variations in both poses and expressions have a great impact on CSIM,

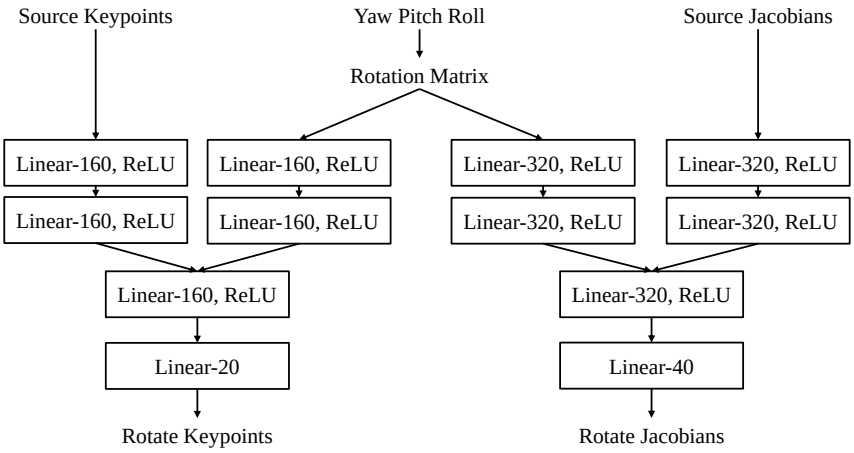


Figure 10: Architecture of the lightweight pose editor.

Figure 11: Additional experimental results of reconstruction.

especially for detectors focusing on face features. Especially, there are pronounced decreases in CSIM for Sample 3, Sample 4 and Sample 5, indicating the considerable influence of facial poses. In other words, rich variations in facial poses and expressions make it difficult to maintain relatively high CSIM. Consequently, it is understandable that our model obtains lower CSIM than Face vid2vid. Finding more reasonable metrics to verify the preservation of identity information is part of our future research.

D Additional Experimental Results

D.1 Pose Editing

We design an MLP-based lightweight pose editor (LPE), and its detailed structure is shown in Fig. 10. In the training process, we input the deviations of the Euler angles yaw, $pitch$, and $roll$ between the source image and the driving frame provided by the 3D reconstruction network into LPE, together with the keypoint information of the source image. In the inference process, we can directly control the face pose with specific yaw, $pitch$, and $roll$ angles.

D.2 Additional Comparisons with State-of-the-Art Methods

Reconstruction. In Fig. 11, we show three animation examples from the VoxCeleb [40] dataset, in which there is significant expression variation between each pair of two faces. From the generated

Figure 12: Additional experimental results of reenactment without the relative motion transfer.

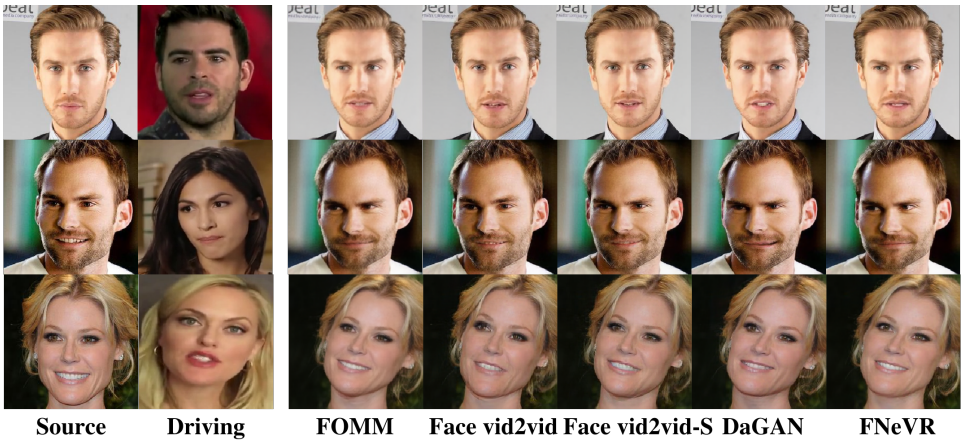


Figure 13: Additional experimental results of reenactment with the relative motion transfer.

results, we see that FOMM [52] can generate promising expression transformed images but with blurred head profiles, while Face vid2vid [59] and DaGAN [25] generate the images with clear head profiles but poor expression transformation. For instance, Face vid2vid and DaGAN cannot transfer the visual directions or the corners of the mouth very well. Our FNeVR can combine their merits and generate promising expression transformed images with clear head profiles.

Reenactment. To enrich the reenactment experimental results shown in the main paper, we conduct additional experiments and show more samples in Fig. 12. Notably, in the cross-identity reenactment experiment of the main paper, all FOMM-based models [52; 59; 25] do not employ the relative motion transfer approach [52], in which the frame D_{best} in the driving video whose pose is most similar to the source image S is selected as the basis, and the motion fields of other driving images D_i are calculated by the motion differences from D_{best}. Utilizing the relative motion transfer, we can maximize the preservation of identity information. Therefore, we conduct additional reenactment experiments based on this approach on VoxCeleb [40], and the results are shown in Fig. 13. It can be seen that Bilayer [64] has a greater advantage in maintaining identity information but cannot handle the image background. FOMM [52] has a desirable performance in expression transfer, but the generated results are distorted in some areas. Face vid2vid [59] works best on identity information due to the use of 3D representation. However, the facial details are not well captured, such as the inconsistency of the irises between the driving images and the source images. DaGAN [25] has a significant deficiency in maintaining identity information, and the contours of the faces in the generated images are completely deformed. Both Fig. 12 and Fig. 13 show that our FNeVR achieves the best results by maximizing the transfer of facial expressions without losing the facial features.

E Social Impact

This work does not have a direct negative social impact. However, we should be aware of the power of realistic face animation and prevent it from being abused for malicious purposes.