FNeVR: Neural Volume Rendering for Face Animation

Anonymous Author(s)
Affiliation
Address
email

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 novel 3D Face Volume Rendering (FVR) module to enhance the facial details for image rendering. To the best of our knowledge, this is the first work to formulate the neural volume rendering of face animation with a new architecture design and improved learning strategy. Additionally, we 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.

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 [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 [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 [55, 45, 46, 52, 7, 51] realize compelling reenactment by modeling the relative motion field from the source to the driving faces. Notably, First Order Motion Model (FOMM) [46] significantly improves the performance of image animation by using self-supervised keypoints with 2D local affine transformations. However, it is insufficient to model the deformation when the motion changes dramatically (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]. However, these 3D structure-based methods [30, 12, 51] may fail to transform the expression fully or have limited ability to produce ultra-realistic images such as Fig. 1(d).

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

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

(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.

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 capture 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 CNN 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. Additionally, we design a perceptual loss to guarantee the quality of rendering feature maps. As a result, our FNeVR can generate identity preserving and photo-realistic images (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 proposed 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. To the best of our knowledge, this is the first work of neural volume rendering for face animation.
- A simple yet effective Lightweight Pose Editing (LPE) module is presented only based on the Euler angle provided by the 3D face reconstruction model. A new loss is further designed to guide the LPE module to generate photo-realistic images according to a given pose.
- 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

2.1 Face Animation

Model-free models [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

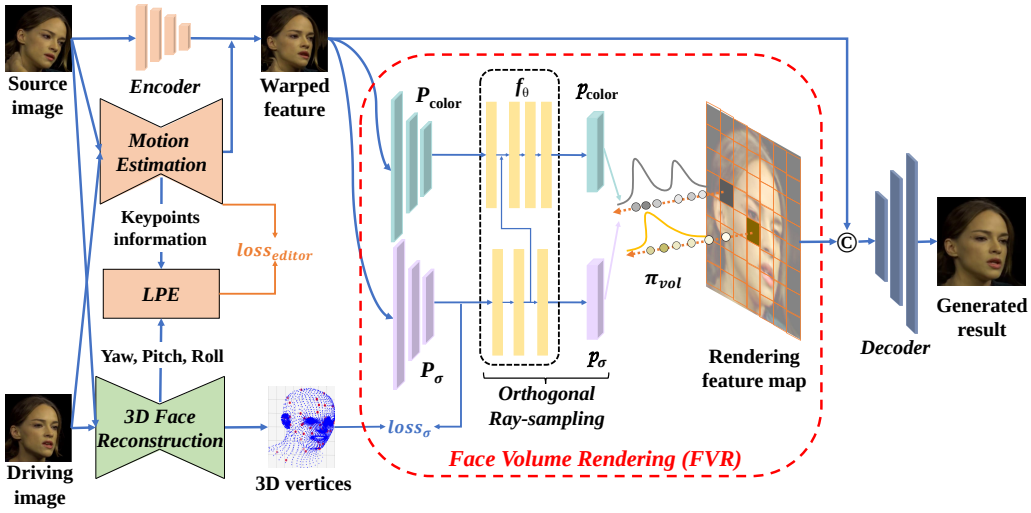

Figure 2: Overview of the proposed FNeVR, where © indicates the channel-wise concatenation. We first extract the warped feature and 3D face reconstruction representation, on which FVR is then performed to obtain the 3D shape and color information for face rendering. The 3D face reconstruction result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

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 [47] and GIF [17] respectively employ pre-trained StyleGAN [25] and StyleGAN2 [26] to warp face images. PIRenderer [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.

2.2 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 [40, 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.

2.3 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 2D features achieves a consistent motion transfer of facial details. Based on this observation, we utilize the 2D motion estimation module proposed in FOMM [46] to complete the warping process. As shown in Fig. 3(a), we use an Hourglass Network [35] 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 [46]:

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

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 [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 $\boldsymbol{\beta} \in \mathbb{R}^{|\boldsymbol{\beta}|}$, 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.

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

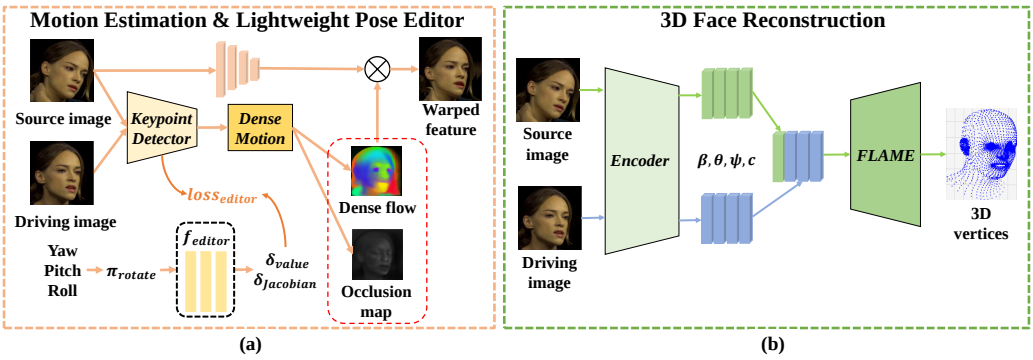

Figure 3: Illustration of Motion Estimation and LPE modules, and 3D Face Reconstruction module.

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 [33].

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. 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 [18, 43, 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, unlike a hierarchical (two-stage) volume sampling with an enormous computational burden as in [33], we design an orthogonal ray-sampling module to adaptively estimate the sampled color field p_{color} and the voxel probability p_σ, which is the integral of the volume density with an adaptive step (orthogonal to the image plane) for D intervals, based on a single MLP-based one-stage process:

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

where f_θ denotes the parameterized MLP, and M_{color} is the number of channels representing the color information of p_{color}. Note that our interval sampling number D is consistent with the previous spatial channel number D. 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.

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. Let π_{vol} denote the volume rendering formula, modified from [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 formula π_{vol} 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:

$$F_{r,i} = \pi_{vol}(p_{\sigma,i,j}, p_{color,i,j}) \in \mathbb{R}^{M_{color}}, \tag{9}$$

which is similar to [33] and can be further solved as:

$$F_{r,i} = \sum_{j=1}^{D} \tau_j \left(1 - \exp(-p_{\sigma,i,j})\right) p_{color,i,j}, \tag{10}$$

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 [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 $\mathcal{L}_R$ between I_m and the driving frame D_i as:

$$\mathcal{L}_R = VGG_{perceptual}(D_i, I_m). \tag{11}$$

Afterward, we concatenate the volume rendering feature map F_r and the warped feature F_w, and feed them into the rendering decoder which we design based on 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 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 $r_{rotate} \in \mathbb{R}^{3\times 3}$, and then introduce an MLP as f_{editor} to compute δ_{value} and $\delta_{Jacobian}$ under a specific rotation angle θ_{rotate}:

$$\delta_{value}, \delta_{Jacobian} = f_{editor}(\theta_{rotate}, p_S, J_S) \in \mathbb{R}^{K\times 2} \times \mathbb{R}^{K\times 2\times 2}. \tag{12}$$

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:

$$\mathcal{L}_{editor} = \lambda_1 L_1(p_D, \delta_{value}) + \lambda_2 L_1(J_D, \delta_{Jacobian}), \tag{13}$$

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 $\lambda_2 = 0.5$.

Table 1: Quantitative comparison of same-identity reconstruction on VoxCeleb [34].

Method	$\mathcal{L}_1 \downarrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	AKD $\downarrow$	AED $\downarrow$	FID $\downarrow$
Bilayer [56]	0.1197	0.4247	15.219	0.3968	12.60	0.0546	219.8
FOMM [46]	0.0450	0.1099	23.210	0.7475	1.383	0.0244	11.56
Face vid2vid [51]	0.0485	0.1051	22.642	0.7268	1.616	0.0395	9.142
Face vid2vid-S [51]	0.0445	0.0901	23.357	0.7473	1.421	0.0243	9.151
DaGAN [21]	0.0462	0.0981	23.263	0.7536	1.441	0.0247	9.660
FNeVR (ours)	**0.0404**	**0.0804**	**24.292**	**0.7773**	**1.254**	**0.0231**	**8.443**

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 $\mathcal{L}_{total}$:

$$\begin{aligned}\mathcal{L}_{total} = \mathcal{L}_P(D_i, x_{generated}) + \mathcal{L}_G(D_i, x_{generated}) + \mathcal{L}_E(p_D, J_D) + \\ \mathcal{L}_R(D_i, I_m) + \mathcal{L}_\sigma(F_\sigma, F_m) + \mathcal{L}_{editor}(p_D, J_D, \delta_{value}, \delta_{Jacobian}),\end{aligned} \tag{14}$$

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 [46], 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 [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 256×256 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 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].

Figure 4: Qualitative comparison with SOTA baselines for (a) same-identity reconstruction and (b) 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 [34] and VoxCeleb2 [9].

Method	VoxCeleb		VoxCeleb2	
	FID↓	CSIM ↑	FID↓	CSIM ↑
FOMM [46]	106.9	0.5491	138.1	0.5228
Face vid2vid-S [51]	106.6	**0.6447**	148.6	**0.6290**
DaGAN [21]	110.3	0.5305	139.6	0.4932
FNeVR (ours)	**98.23**	0.5505	**133.9**	0.5282

Table 3: Quantitative comparison of Flops and memory consumption.

Method	Flops(G)↓	Params(M)↓
Face vid2vid [51]	231.038	125.216
Face vid2vid-S [51]	636.941	173.109
FNeVR (ours)	**130.109**	**61.378**

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], and DaGAN [21]. We use the official pre-trained models for Bilayer, FOMM, and DaGAN, 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 (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 Params) than FNeVR, as indicated in Table 3, demonstrating FNeVR's outstanding efficiency in compact computation. Moreover, Fig. 4(b) visualizes some synthesis results

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

Table 4: Quantitative results of 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	23.21	0.7475	1.383	0.0244	11.56
FOMM + SPADE	0.0426	0.0889	24.006	0.771	1.316	0.0246	11.129
FOMM + FVR	**0.0404**	0.0875	**24.374**	**0.7783**	1.290	0.0233	10.755
FNeVR (ours)	**0.0404**	**0.0804**	24.292	0.7773	**1.254**	**0.0231**	**8.443**

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 the proposed FVR module and SPADE [39] layer-based decoder. Specially, we employ FOMM [46] as the baseline, which uses 2D motion warping and a common rendering decoder, to compare our full model FNeVR, which consist of 2D warping, FVR and the SPADE layer-based decoder. In addition, we respectively substitute FOMM's decoder with the SPADE layer-based decoder and insert FVR into FOMM, as another two comparison models. The results are shown in Table 4. Apparently, both the proposed modules have significant effectiveness to generate high-quality faces. Particularly, FOMM with the SPADE layer-based decoder obtains better performance than the baseline, demonstrating its capability to better preserve semantic information and generate reasonable and realistic results with the significant decreases of AKD and FID. However, the SPADE decoder has little effect on several metrics, even being slightly counterproductive in terms of AED. Moreover, FOMM with FVR achieves better performance on the accurate motion transfer (AKD), authenticity (FID) and image quality (PSNR), which illustrates its remarkable properties to retain more abundant and realistic facial details. Note that our full model produces the best overall results and surpasses other models a lot in several metrics, fully demonstrating 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.

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] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[4] Timo Bolkart and Stefanie Wuhrer. A groupwise multilinear correspondence optimization for 3d faces. In *ICCV*, 2015.

[5] Timo Bolkart and Stefanie Wuhrer. A robust multilinear model learning framework for 3d faces. In *CVPR*, 2016.

[6] James Booth, Anastasios Roussos, Stefanos Zafeiriou, Allan Ponniah, and David Dunaway. A 3d morphable model learnt from 10,000 faces. In *CVPR*, 2016.

[7] Egor Burkov, Igor Pasechnik, Artur Grigorev, and Victor Lempitsky. Neural head reenactment with latent pose descriptors. In *CVPR*, 2020.

[8] 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.

[9] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. *INTERSPEECH*, 2018.

[10] Zhigang Deng and Junyong Noh. Computer facial animation: A survey. In *Data-Driven 3D Facial Animation*. Springer, 2008.

[11] Pengfei Dou, Shishir K Shah, and Ioannis A Kakadiaris. End-to-end 3d face reconstruction with deep neural networks. In *CVPR*, 2017.

[12] Michail Christos Doukas, Stefanos Zafeiriou, and Viktoriia Sharmanska. Headgan: One-shot neural head synthesis and editing. In *ICCV*, 2021.

[13] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *TOG*, 2021.

[14] 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.

[15] Jiahao Geng, Tianjia Shao, Youyi Zheng, Yanlin Weng, and Kun Zhou. Warp-guided gans for single-photo facial animation. *TOG*, 2018.

[16] 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.

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

[18] 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.

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

[20] 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.

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

[22] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. *arXiv:2112.05637*, 2021.

[23] 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.

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

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

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

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

[28] 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.

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

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

[31] 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.

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

[33] 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.

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

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

[36] 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.

[37] 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.

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

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

[40] 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.

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

[42] 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.

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

[44] 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.

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

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

[47] 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.

[48] Qiulin Wang, Lu Zhang, and Bo Li. Safa: Structure aware face animation. In *3DV*, 2021.

[49] Ting-Chun Wang, Ming-Yu Liu, Andrew Tao, Guilin Liu, Bryan Catanzaro, and Jan Kautz. Few-shot video-to-video synthesis. In *NeurIPS*, 2019.

[50] 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.

[51] Ting-Chun Wang, Arun Mallya, and Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *CVPR*, 2021.

[52] Yaohui Wang, Di Yang, Francois Bremond, and Antitza Dantcheva. Latent image animator: Learning to animate images via latent space navigation. *arXiv:2203.09043*, 2022.

[53] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004.

[54] Huawei Wei, Shuang Liang, and Yichen Wei. 3d dense face alignment via graph convolution networks. *arXiv:1904.05562*, 2019.

[55] Olivia Wiles, A Koepke, and Andrew Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *ECCV*, 2018.

[56] Egor Zakharov, Aleksei Ivakhnenko, Aliaksandra Shysheya, and Victor Lempitsky. Fast bi-layer neural synthesis of one-shot realistic head avatars. In *ECCV*, 2020.

[57] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *ICCV*, 2019.

[58] 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.

[59] 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]