DOI: 10.1111/cgf.14704
Pacific Graphics 2022
N. Umetani, E. Vouga, and C. Wojtan
(Guest Editors)

Volume 41 (2022), Number 7

Implicit Neural Deformation for Sparse-View Face Reconstruction

Moran Li[1,2], Haibin Huang[†1], Yi Zheng[1], Mengtian Li[1], Nong Sang[2], and Chongyang Ma[1]

[1]Kuaishou Technology, China [2]Huazhong University of Science and Technology, China

Abstract
In this work, we present a new method for 3D face reconstruction from sparse-view RGB images. Unlike previous methods which are built upon 3D morphable models (3DMMs) with limited details, we leverage an implicit representation to encode rich geometric features. Our overall pipeline consists of two major components, including a geometry network, which learns a deformable neural signed distance function (SDF) as the 3D face representation, and a rendering network, which learns to render on-surface points of the neural SDF to match the input images via self-supervised optimization. To handle in-the-wild sparse-view input of the same target with different expressions at test time, we propose residual latent code to effectively expand the shape space of the learned implicit face representation as well as a novel view-switch loss to enforce consistency among different views. Our experimental results on several benchmark datasets demonstrate that our approach outperforms alternative baselines and achieves superior face reconstruction results compared to state-of-the-art methods.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ ***Mesh models; Shape analysis;***

1. Introduction

In this paper, we tackle the problem of 3D face reconstruction given sparse-view input, *i.e.*, to generate a textured face mesh based on a set of RGB images taken from different views. This problem is long-standing in both computer vision and computer graphics with many real-world applications, such as portrait manipulation and augmented/virtual reality.

Compared with reconstruction from a single RGB image or RGBD input, multi-view face reconstruction is a more practical setting with recent development of mobile devices, since it does not require additional depth senor but still provides rich information from different views about the target. Previous methods [BBB*10, BHPS10] propose to reconstruct 3D faces under controlled environments, where the multi-view images are captured from well-calibrated camera arrays with fixed lighting. Although these methods can successfully produce high-fidelity 3D face models, their usage scenarios are quite limited due to the complex hardware setup and their performances downgrade significantly for general input. To address these drawbacks, some recent approaches [BCR*20, BCLT21, SSL*20] exploit 3DMMs [BV03, PKA*09, ZLY*15] together with multi-view algorithms to leverage cross-view geometric consistency and demonstrate promising improvement. However, those methods are built upon 3DMMs or the variants, where the number of vertices is limited and the topology is fixed. Therefore, it remains challenging to generate a faithful 3D face with high-quality details from multi-view input, especially in an uncontrolled setting or when the input views are of different expressions.

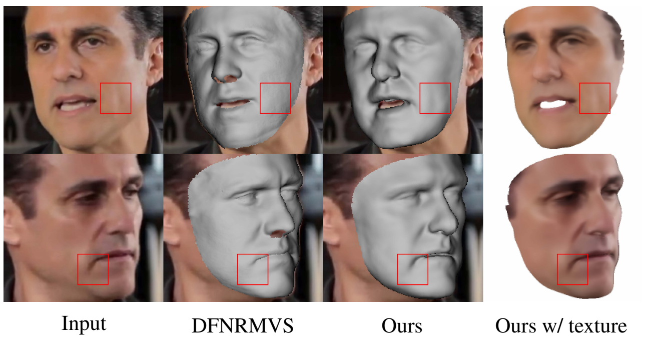


Figure 1: *Face reconstruction results of DFNRMVS [BCR*20] and our method from two-view inputs. The last two columns are our reconstructed geometry and the rendering results. Our method captures more local details (red boxes).*

In this work, our focus is to improve the generalization performance as well as the quality of sparse-view 3D face reconstruction by learning an implicit neural representation. Our key insight is that, unlike 3DMMs that are limited by a pre-defined shape space, implicit functions such as SDFs can represent surfaces with arbitrary resolution and topology [ZYDL21, YKM*20, GYH*20, PFS*19]. To this end, we propose to learn a Geometry Network that serves as

† Corresponding author: jackiehuanghaibin@gmail.com

a neural SDF for reconstruction of the target 3D face. Specifically, the proposed Geometry Network consists of two sub-modules, *i.e.*, a *Reference Network* and a *Deformation Network*. The Reference Network is trained offline to learn the SDF of a mean face given the training set and provides an initiation of SDFs for optimization at the test time. The Deformation Network generates local details and changes topology if necessary by learning to deform the SDFs. Our experiments show that such a decomposition effectively leverages a 3D face prior to enhance the generalization capability of the network and prevents the neural SDFs from collapsing or distorting during optimization with limited views (*e.g.*, 2∼4 views).

Inspired by [YKM*20], we further present a Rendering Network based on self-supervised optimization. This module learns to render on-surface points sampled from the implicit 3D face geometry. The self-supervision is achieved by minimizing the difference between rendered colors and the corresponding input images. Additionally, we use geometry and color latent codes to encode shape and texture information among different instances to enhance the generalization ability of the trained network. To expand the shape space of the learned neural SDF, we introduce *residual latent code* at test time. Furthermore, we design a *view-switch loss* via exchanging the latent code among different views and minimizing the rendering loss to enforce consistency across different views. As a result, our method can reconstruct 3D faces from sparse-view input with high-fidelity details. See Figure 1 for an example of 3D face reconstruction from two in-the-wild images of the same person but with different expressions.

To summarize, our main contributions are as follows:

- We present a pipeline for 3D face reconstruction from sparse-view input, including a Geometry Network to learn a deformable implicit neural representation for 3D shapes and a Rendering Network to model the facial texture.
- We propose a novel view-switch loss as well as a newly designed latent code space of the implicit morphable model. These two terms help expand the underlying shape space and enforce cross-view consistency at the test time.
- We conduct both qualitative and quantitative evaluations on benchmark datasets to demonstrate that our method outperforms baseline approaches and is comparable to state-of-the-art face reconstruction algorithms.

2. Related Work

The literature on 3D face reconstruction is vast and the algorithm input ranges from depth map [KKT*14] and single image [TTHMM17, KZT*18, RSOEK17, DSK17, ZWC*20, CCZ*19, FWS*18, SSL*20, CLL*21, XYC*20, GSL*20] to multi-view images [BCLT21, BCR*20, WBC*19, DYX*19, SBFB19, CLC*22] and videos [GZC*16, TBG*19]. Since our main focus is 3D face reconstruction from sparse-view images using neural SDFs as the geometric representation, in this section, we briefly review 3D morphable models, multi-view 3D face reconstruction methods, and the most relevant implicit neural representations.

Face morphable models. The well-known 3D morphable model [BV03, PKA*09, CWZ*13, ZLY*15] is a bilinear parametric method that decomposes the face geometry/texture into a template and a deformation component with respect to this template based on principal component analysis (PCA). Due to their simplicity and effectiveness, 3DMMs are widely used in faces reconstruction and animation. However, the capability of such models is limited by the basis of PCA. Even though several recent methods (*e.g.*, [CWZ*13, YZW*20, SSD*20]) propose to extend the face basis with more 3D face scans from larger datasets, the geometry or texture space of those methods is still a subspace of real-world face space. For a complete report of 3D morphable face, we refer to [EST*20].

Multi-view face reconstruction. Existing learning-based algorithms for multi-view 3D face reconstruction can be roughly categorized into supervised methods [GZC*16, CHZ14, BCR*20] and self-supervised methods [DYX*19, SBFB19, WBC*19]. [GZC*16] exploits parametric geometry prior information to learn a plausible coarse face mesh and fine-scale details are captured via shading-based refinement from videos. [BCR*20] proposes to expand the basis of 3DMMs via adaptive optimization to improve the representation of such parametric models and enforce multi-view consistency. To alleviate the requirement of large-scale 3D scan datasets, some researchers tackle this problem in a self-supervised manner. [DYX*19] uses aggregated complementary information among different images to achieve multi-view reconstruction. However, those models are built upon 3DMMs, where the mesh topology is fixed and cannot represent high-frequency details easily.

Implicit neural representation. In recent years, methods based on implicit neural representations are emerging for shapes [DYT21, YGKL21, PFS*19, AL20, DNJ20, GCS*20, TLY*21, GYH*20, ZYDL21, TCY*21] and scenes [ERB*18, SZW19, JSM*20, KSW20]. The seminal work DeepSDF [PFS*19] encodes a category of shapes into a neural network, while the specific features of each instance are encoded into a latent code. Based on DeepSDF, [DZW*20] proposes a curriculum architecture to enhance the quality of the reconstructed shape. Those methods are used to obtain the implicit neural representations of shapes, objects, or scenes with 3D data (*e.g.*, point cloud) as the supervision. [ZYDL21, DYT21] further decompose the implicit neural representation for 3D geometry into a deformation and a template implicit representation. [SHN*19] exploits pixel-aligned implicit function to estimate the surface of human subjects and the corresponding texture.

Recently, [SLB*21, GTZN21, MST*20, YYTK21, KJJ*21] propose to synthesize novel views from a set of images by reconstructing the underlying 3D scene/object geometry and the neural radiance field at the same time. [YKM*20, WLL*21] use neural SDF to represent surface geometry and reconstruct the target shape from multi-view images. However, most of those techniques require more than 30 images from different viewpoints for each object/scene, and the reconstructed surface will collapse for sparse-view input due to the lack of prior information about the object/scene. Among those methods, the most relevant one is i3DMM [YTB*21], which builds implicit 3D morphable models for human heads with hair from a dataset of 3D scans and requires calibrated dense-view capture of the subject. Our goal instead is to reconstruct a 3D face from sparse-view 2D input and thus is more challenging than the setting of i3DMM.

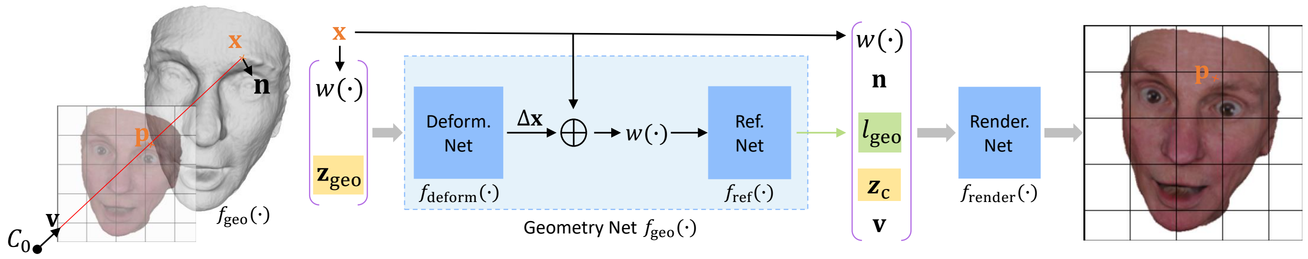


Figure 2: *Overview of the proposed method. Our method consists of the* Geometry Network *and* Rendering Network. *And the* Geometry Network *can be decomposed into a* Deformation Network *and a* Reference Network. **v** *is the view direction from the camera center* C_0 *to the randomly sampled pixel* **p**. **x** *is the corresponding on-surface point, and* **n** *is the normal vector.* $\mathbf{z}_{geo}, \mathbf{z_c}$ *are the geometry and color latent codes, respectively.* l_{geo} *is the feature from Geometry Network (*i.e., $f_{geo}(\cdot)$*).* $w(\cdot)$ *is the positional encoding function.* ( ) *denotes the concatenate operation. For conciseness, we only present one view of the instance, while most of our experiments are sparse-view inputs.*

3. Our Method

3.1. Face Representation and Problem Statement

In an implicit neural representation based on signed distance field (SDF), the geometry of a 3D face can be represented as the zero level set of a scalar valued network f:

$$S_\theta = \{\mathbf{x} \in \mathbb{R}^3 | f_\theta(\mathbf{x}) = 0\} \tag{1}$$

where the network $f_\theta(\mathbf{x})$ gives the signed shortest distance s of a query point $\mathbf{x} \in \mathbb{R}^3$ to the face geometry S_θ and $\theta \in \mathbb{R}^m$ are learnable parameters of the network. To model the facial texture, we extend the network output to include a vector $\mathbf{c} \in \mathbb{R}^3$, which represents the color of the closest point on the face from the query point.

To further model various faces of different identities and expressions, we introduce a latent code $\mathbf{z}$ to represent the face instance in a portrait image. Following i3DMM [YTB*21], we denote the network as $f_\theta(\mathbf{x}, \mathbf{z})$ to take this latent code as additional input.

In both the training and test stages, we jointly optimize the network parameters θ and the latent code $\mathbf{z}$ as described in detail below, in order to obtain the desired morphable model and the corresponding implicit representation of each face instance. To simplify the notation, we omit θ in the subscript and rewrite the network as $\{s, \mathbf{c}\} = f(\mathbf{x}, \mathbf{z})$.

3.2. Network Components

As illustrated in Figure 2, our overall framework consists of two network components, *i.e.*, a Geometry Network f_{geo} and a Rendering Network f_{render}. Accordingly, the latent code $\mathbf{z}$ of each face instance can be decomposed into two parts, *i.e.*, a geometry code $\mathbf{z}_{\text{geo}}$ and a color code $\mathbf{z_c}$, which are used as input of f_{geo} and f_{render}, respectively.

Geometry Network. Our Geometry Network f_{geo} is a scalar valued function to model the implicit 3D face shape. We follow i3DMM [YTB*21] to further decompose f_{geo} into two successive components, *i.e.*, a Reference Network f_{ref} to learn an implicit reference shape, and a Deformation Network f_{deform} to predict a deformation offset $\Delta\mathbf{x}$ conditioned on the reference shape. The Reference Network can be considered as a neural version of the mean face in traditional 3DMMs [BV99], while the Deformation Network models the per-instance variations from the mean face. As a result, the Geometry Network can be formulated as:

$$\begin{aligned} f_{\text{geo}}(\mathbf{x}, \mathbf{z}_{\text{geo}}) &= f_{\text{ref}} \circ f_{\text{deform}}(\mathbf{x}, \mathbf{z}_{\text{geo}}) \\ &= f_{\text{ref}}(\mathbf{x} + \Delta\mathbf{x}). \end{aligned} \tag{2}$$

Rendering Network. Our Rendering Network f_{render} is introduced to model the face texture in a self-supervised manner, where the texture information is encoded as the color latent code $\mathbf{z_c}$ for each face instance. Therefore, for a given surface point $\mathbf{x}$ of a certain instance, the RGB value $\mathbf{c}(\mathbf{x}, \mathbf{z_c})$ can be modeled using our Rendering Network by taking several factors into account together:

$$\mathbf{c}(\mathbf{x}, \mathbf{z_c}) = f_{\text{render}}(\mathbf{x}, \mathbf{z_c}, \mathbf{n}, \mathbf{v}, l_{\text{geo}}), \tag{3}$$

where $\mathbf{n}$ is the surface normal, $\mathbf{v}$ is the view direction, and $l_{\text{geo}} \in \mathbb{R}^{256}$ are geometric features computed as additional output by the Geometry Network f_{geo}. Note that the normalized gradient of the signed distance computed by the Geometry Network at a point $\mathbf{x}$ is the corresponding surface normal, *i.e.*, $\mathbf{n} = \nabla f_{\text{geo}}(\mathbf{x}, \mathbf{z}_{\text{geo}}) / \|\nabla f_{\text{geo}}(\mathbf{x}, \mathbf{z}_{\text{geo}})\|_2$.

3.3. Network Training

Dataset. We use a training partition of the Stirling/ESRC [SE18] dataset to train our network, which contains more than 700 registered 3D face scans of about 95 subjects. To prepare training data for the Geometry Network f_{geo}, the 3D scans are scaled to fit into a unit bounding box and then aligned to the same orientation. We then randomly sample 860K on-surface points from each registered scan in a uniform distribution. We consider these points together with the corresponding normals as the zero level set of each SDF and use them to train the Geometry Network. To prepare training data for the Rendering Network f_{render}, we render about 40 RGB images of each 3D face scan from random view directions. For each rendered image, we also compute a binary mask to represent the face region.

Geometry loss function. Given a face instance i with a geometry latent code $\mathbf{z}^i_{\text{geo}}$ and a set of sample points Ω_I, the overall geometry loss function $\mathcal{L}_{\text{geo}}$ is computed as:

$$\mathcal{L}_{\text{geo}}(\mathbf{z}^i_{\text{geo}}) = \lambda_{\text{I}}\mathcal{L}_{\text{I}} + \lambda_{\text{d}}\mathcal{L}_{\text{d}} + \lambda_{\text{eik}}\mathcal{L}_{\text{eik}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} \tag{4}$$

where λ_{I}, λ_{d}, λ_{eik}, and λ_{reg} are hyperparameters to balance different loss terms. We set $\lambda_I = 1$, $\lambda_d = 0.01$, $\lambda_{eik} = 0.1$, and $\lambda_{reg} =$ 1e-4 in our experiments.

In Eq. (4), $\mathcal{L}_I$ is a reconstruction loss to enforce the signed distance values of sampled on-surface points are close to zero and the normals of those points are close to the ground truth values:

$$\mathcal{L}_I = \frac{1}{\|\Omega_I\|_1} \sum_{\mathbf{x}_j \in \Omega_I} \left(\left\| f_{geo}(\mathbf{x}_j, \mathbf{z}^i_{geo}) \right\|_1 + \lambda_n \left\| \mathbf{n}_j - \hat{\mathbf{n}}_j \right\|_2 \right), \tag{5}$$

where $\|\cdot\|_1$ is the L1 norm, $\|\cdot\|_2$ is the L2 norm, $\Omega_I = \{\mathbf{x}_j\}_{j\in I}$ is a randomly sampled set of the on-surface points, $\mathbf{n}_j = \nabla f_{geo}(\mathbf{x}_j, \mathbf{z}^i_{geo}) / \left\| \nabla f_{geo}(\mathbf{x}_j, \mathbf{z}^i_{geo}) \right\|_2$, and $\hat{\mathbf{n}}_j$ is the ground truth surface normal of the on-surface point $\mathbf{x}_j$. We set $\lambda_n = 1$ in our experiments. $\mathcal{L}_d$ in Eq. (4) is the regularization of the deformation offset:

$$\mathcal{L}_d = \frac{1}{\|\Omega_I\|_1} \sum_{\mathbf{x}_j \in \Omega_I} \left\| \Delta \mathbf{x}^i_j \right\|_2 = \frac{1}{\|\Omega_I\|_1} \sum_{\mathbf{x}_j \in \Omega_I} \left\| f_{deform}(\mathbf{x}_j, \mathbf{z}^i_{geo}) \right\|_2, \tag{6}$$

and $\mathcal{L}_{reg} = \|\mathbf{z}_{geo}\|_2$ is the regularization for the geometry latent code. Finally, $\mathcal{L}_{eik}$ is the Eikonal term to avoid universe zero and ensures that f_{geo} approximates valid SDFs [GYH*20, YKM*20]:

$$\mathcal{L}_{eik} = \frac{1}{\|\Omega_D\|_1} \sum_{\mathbf{x}'_j \in \Omega_D} \left(\left\| \nabla f_{geo}(\mathbf{x}'_j, \mathbf{z}^i_{geo}) \right\|_2 - 1 \right)^2, \tag{7}$$

where $\Omega_D = \{\mathbf{x}'_j\}_{j\in D}$ is a set of points sampled from a uniform distribution within a unit bounding box.

Given N face instances within a mini-batch, we can jointly optimize the parameters θ_{geo} of the Geometry Network f_{geo} and the geometry latent codes $\{\mathbf{z}^i_{geo}, i = 1, \ldots, N\}$ of these N instances by solving the optimization problem below:

$$\underset{\theta_{geo}, \{\mathbf{z}^i_{geo}\}}{\arg\min} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{geo}(\mathbf{z}^i_{geo}) \tag{8}$$

Rendering loss function. Given a pair of a rendered RGB image and the corresponding face mask, we randomly sample a subset of pixels $\mathcal{P}$ in the image plane and use the following rendering loss function $\mathcal{L}_{render}$ to train our Rendering Network f_{render}:

$$\mathcal{L}_{render} = \tau_{rgb}\mathcal{L}_{rgb} + \tau_{mask}\mathcal{L}_{mask} + \tau_d\mathcal{L}_d + \tau_{eik}\mathcal{L}_{eik} + \tau_{reg}\mathcal{L}'_{reg} \tag{9}$$

where τ_{rgb}, τ_{mask}, τ_d, τ_{eik}, and τ_{reg} are set to 1, 100, 1e-4, 0.01, and 1e-4 to balance different loss terms. $\mathcal{L}'_{reg} = \|\mathbf{z}_{geo}\|_2 + \|\mathbf{z}_c\|_2$. In Eq. (9), the loss terms $\mathcal{L}_d$ and $\mathcal{L}_{eik}$ are similar to those in Eq. (4), while $\mathcal{L}_{rgb}$ is the RGB reconstruction loss and $\mathcal{L}_{mask}$ is the mask loss, respectively.

Specifically, the RGB reconstruction loss is computed as:

$$\mathcal{L}_{rgb} = \frac{1}{\|\mathcal{P}\|_1} \sum_{\mathbf{p} \in \mathcal{P}} \|\mathbf{c_p} - \hat{\mathbf{c}}_\mathbf{p}\|_1 \tag{10}$$

where $\mathbf{c_p}$ is the RGB value at the pixel $\mathbf{p}$ predicted by the Rendering Network, and $\hat{\mathbf{c}}_\mathbf{p}$ is the corresponding ground truth RGB value. We use cross-entropy loss to compute the mask loss $\mathcal{L}_{mask}$ as below:

$$\mathcal{L}_{mask} = \frac{1}{\|\mathcal{P}\|_1} \sum_{\mathbf{p} \in \mathcal{P}} \mathrm{CE}(m_\mathbf{p}, \hat{m}_\mathbf{p}) \tag{11}$$

where $m_\mathbf{p}$ and $\hat{m}_\mathbf{p}$ are the predicted and the ground truth mask values at the pixel $\mathbf{p}$, respectively. As in [YKM*20], we use a sigmoid function to achieve differentiable rendering of the mask.

Training strategy. Reconstructing 3D face geometry from a sparse set of RGB input (*i.e.*, $2 \sim 4$ views) with various expressions is an ill-posed problem. Besides, the proposed self-supervision is achieved by enforcing a similarity between the rendered RGB values and the ground truth RGB values. As a result, the Rendering Network tends to overfit the input RGB images and the implicit neural geometry may collapse. To alleviate this problem, we first optimize the Geometry Network with the geometry loss $\mathcal{L}_{geo}$ to obtain a good initialization. Then, we jointly optimize the Rendering Network and the Geometry Network via the rendering loss $\mathcal{L}_{render}$.

3.4. Test-Time Reconstruction

Estimation of camera parameters. To handle in-the-wild images at test time, we estimate camera parameters by optimizing the L1 distance between the projected on-surface point $\mathbf{x}$ and the ground-truth pixel location $\mathbf{p}$:

$$\underset{\mathbf{K},\mathbf{R}}{\arg\min} \left\| \mathbf{K}\mathbf{R}\mathbf{x}_1^\top - \mathbf{p} \right\|_1 \tag{12}$$

where $\mathbf{K} \in \mathbb{R}^{3\times3}$ are camera intrinsic parameters, $\mathbf{R} \in \mathbb{R}^{3\times4}$ is the rotation matrix, $\mathbf{x}_1 = [\mathbf{x}, 1] \in \mathbb{R}^{1\times4}$ are homogeneous coordinates of the point $\mathbf{x}$, and the superscript notation $\top$ is the transpose operator. In our method, the on-surface point is defined as the first intersection point of the ray across the pixel $\mathbf{p}$ and the face geometry S_θ. The intersection point can be represented as a differentiable function of the implicit geometry and camera parameters. We use the differentiable sphere-tracing method [LZP*20] to find the on-surface point.

We use a coarse-to-fine strategy to estimate camera parameters. Specifically, we first use the 68 landmarks of the mean face to obtain a coarse estimation of the camera parameters. Then we refine the camera parameters based on Eq. (12) using the Adam optimizer with a learning rate of 1e-3.

Residual latent code. Given a sparse set of RGB images with various expressions of an instance, it is difficult to learn the latent code directly using the rendering loss described above. Hence, we use principal component analysis (PCA) on the learned latent codes of the training set and infer the weights of those PCA basis at test time. Specifically, the latent code of an instance i can be represented as the weighted sum of the principal components, such as:

$$\begin{aligned} \mathbf{z}^i_{geo} &= \bar{\mathbf{z}}_{geo} + W^i_{geo} B_{geo}, \\ \mathbf{z}^i_c &= \bar{\mathbf{z}}_c + W^i_c B_c \end{aligned} \tag{13}$$

where $W^i_{geo} \in \mathbb{R}^{1\times m_{geo}}, W^i_c \in \mathbb{R}^{1\times m_c}$ are the weights to combine the basis of geometry and color latent codes, respectively. $\bar{\mathbf{z}}_{geo} \in \mathbb{R}^{d_{geo}}$ and $\bar{\mathbf{z}}_c \in \mathbb{R}^{d_c}$ are the mean latent code for the identity and color. $B_{geo} \in \mathbb{R}^{m_{geo}\times d_{geo}}, B_c \in \mathbb{R}^{m_c\times d_c}$ are the PCA basis of geometry and color latent code space. m_{geo} and m_c are the number of the principal components. We set m_{geo} and m_c to be 85 and 65, respectively, such that the explained variance is more than 96% for latent space decomposition. At test time, we can obtain the combination weights

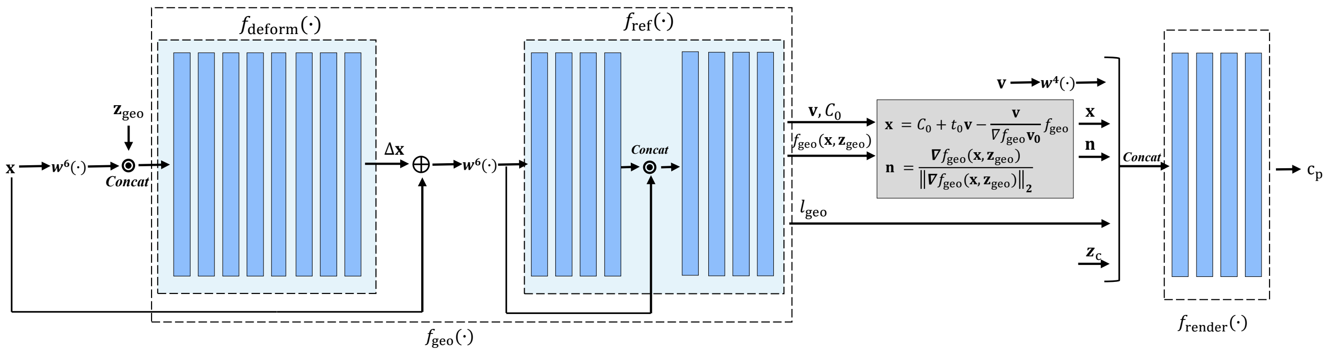

$f_{\text{deform}}(\cdot)$
$f_{\text{ref}}(\cdot)$
$\mathbf{z}_{\text{geo}}$
$\mathbf{x}$
$w^6(\cdot)$
Concat
$\Delta\mathbf{x}$
$w^6(\cdot)$
Concat
$f_{\text{geo}}(\cdot)$
$\mathbf{v}, C_0$
$f_{\text{geo}}(\mathbf{x}, \mathbf{z}_{\text{geo}})$
$\mathbf{x} = C_0 + t_0\mathbf{v} - \frac{\mathbf{v}}{\nabla f_{\text{geo}}\mathbf{v_0}} f_{\text{geo}}$
$\mathbf{n} = \frac{\nabla f_{\text{geo}}(\mathbf{x}, \mathbf{z}_{\text{geo}})}{\|\nabla f_{\text{geo}}(\mathbf{x}, \mathbf{z}_{\text{geo}})\|_2}$
l_{geo}
$\mathbf{v}$
$w^4(\cdot)$
$\mathbf{x}$
$\mathbf{n}$
$\mathbf{z}_c$
Concat
$f_{\text{render}}(\cdot)$
c_p

Figure 3: *Implementation details of our network architecture. The geometry network f_{geo} consists of a deformation network f_{deform} and a reference network f_{ref}. $\mathbf{z}_{geo} \in \mathbb{R}^{256}$ and $\mathbf{z_c} \in \mathbb{R}^{128}$ are the geometry latent code and the color latent code, respectively. $\mathbf{x}$ is the on-surface point of the corresponding image plane pixel $\mathbf{p}$. $\mathbf{c_p} \in \mathbb{R}^3$ is the predicted pixel RGB value. We use $w^6(\cdot)$ and $w^4(\cdot)$ as the positional encoding functions for the on-surface point and the view direction, respectively. $C_0 \in \mathbb{R}^3$ is the camera center, and $\mathbf{v} \in \mathbb{R}^3$ is the view direction that crosses the pixel $\mathbf{p}$. $\odot$ and $\oplus$ denote the* concatenation *and* add *operations, respectively. Each blue box represents a fully connected layer.*

via solving the optimization problem:

$$\underset{W^i_{\text{geo}}, W^i_{\mathbf{c}}}{\arg\min} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{render}}(\mathbf{z}^i_{\text{geo}}, \mathbf{z}^i_{\mathbf{c}}) \tag{14}$$

We further introduce residual latent codes $r^i_{\text{geo}} \in \mathbb{R}^{d_{\text{geo}}}$ and $r^i_{\mathbf{c}} \in \mathbb{R}^{d_{\text{c}}}$ for each instance to expand the underlying representation space. Hence, the latent codes of an instance i can be formulated as:

$$\begin{aligned} \tilde{\mathbf{z}}^i_{\text{geo}} &= \mathbf{z}^i_{\text{geo}} + r^i_{\text{geo}}, \\ \tilde{\mathbf{z}}^i_{\mathbf{c}} &= \mathbf{z}^i_{\mathbf{c}} + r^i_{\mathbf{c}} \end{aligned} \tag{15}$$

The optimization problem at test time can be formulated as:

$$\underset{W^i_{\text{geo}}, W^i_{\mathbf{c}}, r^i_{\text{geo}}, r^i_{\mathbf{c}}}{\arg\min} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{render}}(\tilde{\mathbf{z}}^i_{\text{geo}}, \tilde{\mathbf{z}}^i_{\mathbf{c}}). \tag{16}$$

View-switch loss. A key problem in multi-view reconstruction is how to enforce view consistency to better leverage the multi-view information. In our method, the view consistency is enforced from two aspects. First, we divide the geometry latent code into an identity component and an expression component (*i.e.*, $\mathbf{z}_{\text{geo}} = \{\mathbf{z}_{\text{id}}, \mathbf{z}_{\text{exp}}\}$). Therefore, we can impose cross-view consistency implicitly via enforcing different views of the same instance to share the same identity and color latent code. Second, we propose a view-switch loss $\mathcal{L}_{\text{sw}}$ to further impose an explicit view consistency. The key observation to inspire our view-switch loss is that the rendered images from different expression latent codes of the same identity under the same camera pose should be similar when ignoring local regions that are more likely to be influenced by varying expressions (*e.g.*, the mouth region).

Specifically, given two views i_m and i_n from a sparse set of images for an instance i, we replace the expression latent code of the view i_m with that of the view i_n (*i.e.*, $\mathbf{z}^{i_n}_{\text{exp}}$). Then, we use the switched geometry latent code (*i.e.*, $\mathbf{z}^{i_m}_{\text{geo,sw}} = \{\mathbf{z}^i_{\text{id}}, \mathbf{z}^{i_n}_{\text{exp}}\}$) to calculate the RGB loss as Eq. (10) and the mask loss as Eq. (11). To reduce the impact of expression variations among different views, we use a face parsing network [YWP*18] to omit pixels in the mouth region. We denote the RGB and mask losses after switch views and latent code as the view-switch loss $\mathcal{L}_{\text{sw}}$ to distinguish from previous versions in Eq. (9). Hence, the total loss function $\mathcal{L}'_{\text{render}}$ at test time is:

$$\begin{aligned} \mathcal{L}'_{\text{render}} &= \mathcal{L}_{\text{render}} + \mu\mathcal{L}_{\text{sw}} \\ &= \mathcal{L}_{\text{render}} + \mu(\tau_{\text{rgb}}\mathcal{L}_{\text{rgb, sw}} + \tau_{\text{mask}}\mathcal{L}_{\text{mask, sw}}) \end{aligned} \tag{17}$$

where μ, τ_{rgb}, and τ_{mask} are set to 0.1, 1, and 100 in our experiments.

Mesh recovery. To recover the mesh from our neural SDF, *i.e.*, S^i_θ for a face instance i, we use the Marching Cubes algorithm [LC87] with a resolution of 250, which is a trade-off setting to balance the output quality and the computational cost.

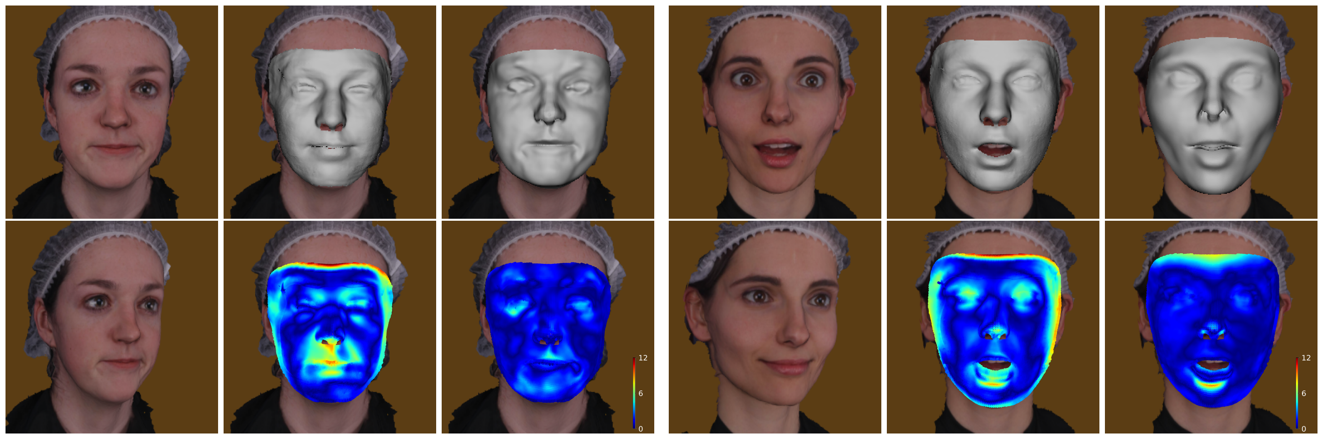
4. Experiments

4.1. Implementation Details

Network optimization. The optimization of the Geometry Network f_{geo} contains two steps. First, we optimize the Reference Network to represent the surface of one scan using the 3D data (*i.e.*, the on-surface points and the corresponding normals of this scan). Then, the 3D training data are used to train the Deformation Network f_{deform} and finetune the Reference Network f_{ref}. In the first step, we use the Adam optimizer with a learning rate of 1e-3. For the second step, the learning rate is 1e-4 with a mini-batch size of 32.

The learning rate for the Rendering Network optimization is set to 1e-4 with a mini-batch size of 32. At the test time, the input images are used as the supervision to find the corresponding latent codes. Since our method does not disentangle the lighting and skin colors, we keep the Geometry Network fixed and finetune the Rendering Network based on Eq. (17) with a small learning rate of 1e-4 to improve the representation capability for in-the-wild RGB images and reduce the domain gap. The hyperparameters of Eq. (17) are similar to that of the Rendering Network optimization with $\mu = 0.1$ which is reduced to zero after 10 iterations.

Network architecture. We use fully connected (FC) layers with a width of 512 for our implicit neural networks. The numbers of FC layers for Reference Network, Deformation Network, and



(a) *Instance 1* **(b)** *Instance 2*

Figure 4: *Qualitative comparison between DFNRMVS [BCR*20] and our method on the Stirling/ESRC test set. For each instance, from left to right, we show the two input views, the results of DFNRMVS [BCR*20], and the ones obtained via our method, respectively. For both methods, we show the reconstructed mesh in the first row and the corresponding error map in the second row. All the results are overlaid with the first input view. The unit of the color bar is millimeter.*

Rendering Network are 8, 8, and 4, respectively. The detailed network architecture is shown in Fig. 3.

Note that, following [YKM*20], we present the intersection of the ray and the surface in a differentiable way as defined in Eq. (18), to compute the on-surface point $\mathbf{x}$ based on a differentiable function of the view direction $\mathbf{v}$ and the implicit face geometry f:

$$\mathbf{x} = C_0 + t_0\mathbf{v} - \frac{\mathbf{v}}{\nabla f_{\text{geo}}(\mathbf{x}_0, \mathbf{z}_{\text{geo}})\mathbf{v}_0} f_{\text{geo}}(C_0 + t_0\mathbf{v}, \mathbf{z}_{\text{geo}}) \tag{18}$$

where $C_0 + t_0\mathbf{v}$ is the first intersection found by the differentiable sphere-tracing algorithm, and t_0 is the initial ray step. Besides, $\mathbf{x}_0 = C_0 + t_0\mathbf{v}$ is the initial intersection position. $\mathbf{v}_0$ is the initial value of the view direction.

Following several previous methods about implicit neural representations [YKM*20, MST*20, YTB*21], we use the Fourier positional encoding [TSM*20] for the input 3D coordinates of the Reference Network and the Deformation Network to reduce the difficulty of learning high-frequency functions. For each of the three coordinates (*i.e.*, x_i) of $\mathbf{x} = \{x_i\}_{i=1}^{3}$, the positional encoding is $w^K(x_i) = \sum_{k=0}^{K}(\cos(2^k\pi x_i) + \sin(2^k\pi x_i))$, where $K = 6$. For the view direction $\mathbf{v}$, we also use such a positional encoding as $w^4(\cdot)$.

Timing statistics. We implement our method using Pytorch with NVIDIA 2080Ti GPUs. The training time is about two days with 8 GPUs. The test-time reconstruction for one instance with 2∼4 input views is about two hours with one GPU.

4.2. Experimental Setup

Datasets. We conduct experiments on two benchmarks for 3D face reconstruction as listed below.

- The Stirling/ESRC dataset [SE18] provides more than 1K high-quality 3D scans and is built upon more than 130 subjects with 8 different expressions. For each scan, a pair of RGB images taken from yaw angles of $\pm 45^\circ$ are used as the texture. We split this dataset into training and testing sets containing 95 and 35 subjects, respectively, in the same way as [BCR*20].
- The Bosphorus dataset [SAD*08] contains 106 subjects with 35 expressions and 13 poses. For each subject, the images of non-neutral expressions are under the frontal view, while those of the neutral expression are under various poses. Following [DYX*19, BCR*20, BCLT21], we adopt this dataset to evaluate the performance of our method under a two-view setting. Specifically, we select a non-neutral frontal view image and another image of a neutral expression under a yaw angle of -30° for each instance. Hence, the overall test set contains 453 pairs of images.

Evaluation protocols. Following previous methods [DYX*19, BCR*20, BCLT21], we use the Euclidean distance between the ground truth 3D face surface points and the aligned output mesh to evaluate the geometric error. The average geometric error (Mean, in mm) and the standard deviation (STD, in mm) among all test samples are computed and reported in our quantitative evaluations. The alignment contains two steps: (i) we first use the ground truth landmarks (*e.g.*, 7 landmarks provided in Bosphorus dataset [SAD*08]) and the landmark points in our reconstruction results to achieve rough alignment [Sch66]; (ii) we use the rigid ICP algorithm [ZPK18] to further improve the alignment between our prediction and the ground truth. Besides, the ground truth is cropped to reduce the noise based on the corresponding 3D landmarks. Note that those strategies are the same as previous methods [DYX*19, BCR*20, BCLT21].

4.3. Qualitative Results

We present qualitative comparisons between DFNRMVS [BCR*20] and our method on both Stirling/ESRC and Bosphorus datasets

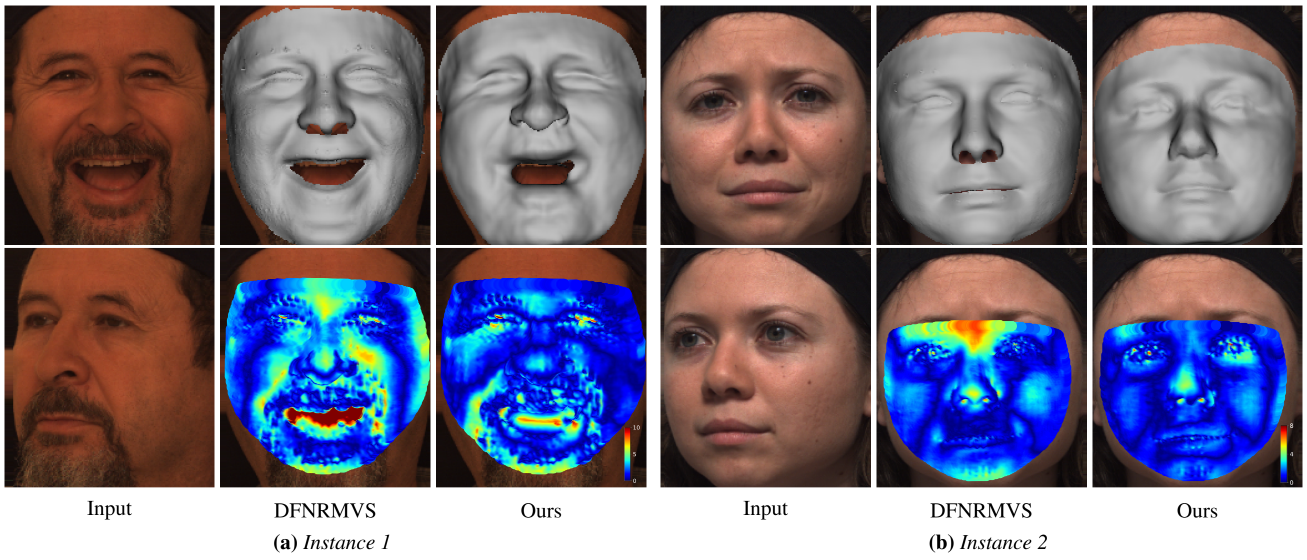


Figure 5: *Qualitative comparison between DFNRMVS [BCR*20] and our method on the Bosphorus dataset [SAD*08]. The images are arranged in the same way as Figure 4.*

Method	Mean (mm)	STD (mm)
Deng et al. [DYX*19]	1.47	0.40
DFNRMVS [BCR*20]	1.44	0.38
Bai et al. [BCLT21]	**1.36**	0.38
Ours	1.39	**0.35**

Table 1: *Quantitative results using two-view input on the Bosphorus dataset [SAD*08].*

Method	Metric	2 views	3 views	4 views
DFNRMVS [BCR*20]	Mean (mm)	1.04	1.03	1.02
	STD (mm)	0.33	0.30	0.29
Ours	Mean (mm)	**0.995**	**0.991**	**0.981**
	STD (mm)	0.177	0.206	0.196

Table 2: *Quantitative results on the Stirling/ESRC dataset [SE18] with different numbers of input views.*

in Figs. 4 and 5, respectively. For each test instance in both figures, we show the two input views on the left, the result by DFNRMVS [BCR*20] in the middle, and our result on the right side. We provide the flat-shaded face mesh in the first row and the corresponding error map in the second row.

From these two figures, we can see that compared to DFNRMVS [BCR*20], the geometry of our method is closer to the ground truth with more local details, such as the forehead and the mouth region. Also, the nose shape of our method is more similar to the input target than that from [BCR*20]. In Figure 5, the region of the error map is determined by the available ground truth data.

In addition, we qualitatively compare our method with DFNRMVS [BCR*20] in a two-view setting using in-the-wild images of several different identities. Since [BCR*20] outperforms other previous methods [CCZ*19, FWS*18, TBG*19], we only present the comparison with [BCR*20] in Fig. 6. In our optimization-based framework, the entire face is treated equally and the amount of pixels in the eye region is relatively small. Therefore, the color of the reconstructed eyes may be slightly influenced by neighboring pixels of skin. It is possible to alleviate this kind of artifact by introducing different penalty weights for different subregions.

4.4. Quantitative Results

To illustrate the effectiveness of the proposed method, we compare with several state-of-the-art methods quantitatively in multi-view 3D face reconstruction, including [DYX*19, BCR*20, BCLT21]. As shown in Table 1, our approach achieves better performance in terms of mean errors compared with previous methods [BCR*20, DYX*19] and is comparable to a more recent method [BCLT21] when evaluate on the Bosphorus dataset.

We also conduct quantitative comparisons on the test set of Stirling/ESRC dataset as shown in Table 2. Since [DYX*19, BCLT21] have not provided results on this test set, we only compare our results with DFNRMVS [BCR*20]. The results demonstrate that our method has consistently lower mean errors with different numbers of input views. Moreover, the performance improvement of our method becomes more significant with increased number of views.

Furthermore, it is worth noting that the number of parameters of our method (4.99M) is about one order of magnitude smaller than those of existing methods, such as 54.69M for [BCLT21], 83.21M for [BCR*20], and 53.35M for [FFBB21]. Overall, we consider our results comparable to [BCLT21] while being superior than other SOTA methods in terms of mean errors.

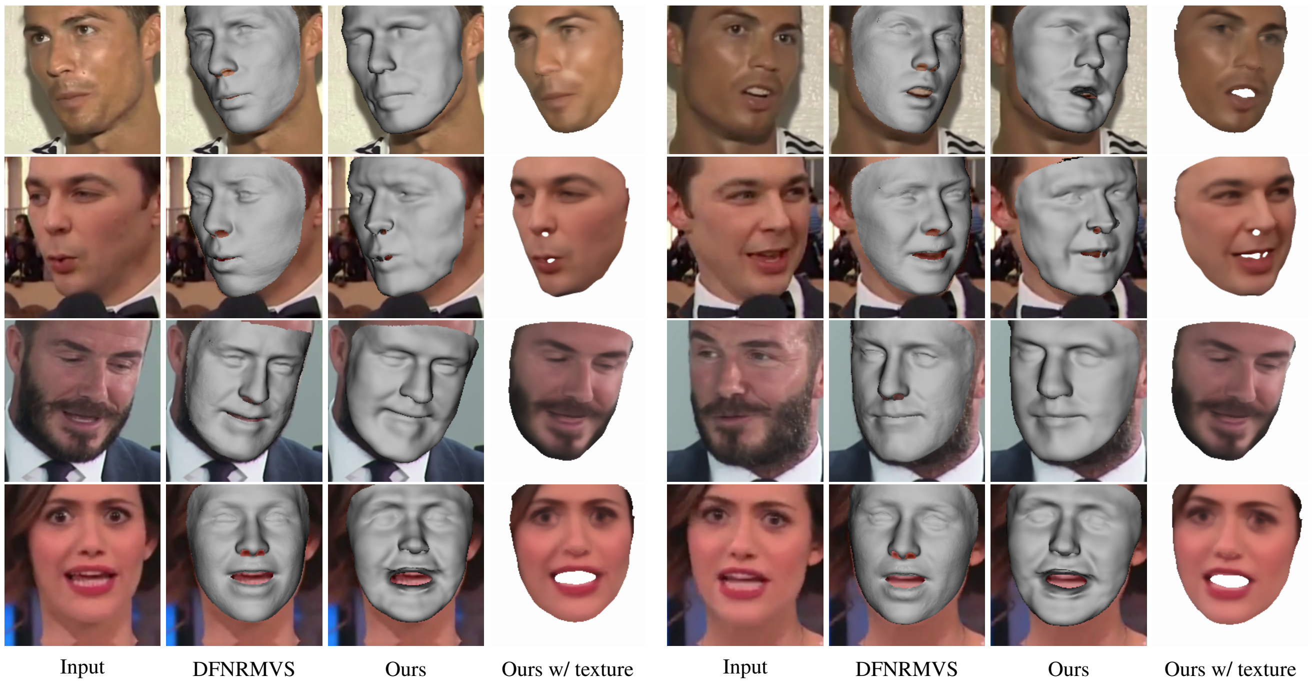


Figure 6: *Qualitative comparison with DFNRMVS [BCR*20] based on in-the-wild input images in a two-view setting. Each row presents the input images (i.e., view1, and view2) and the corresponding results of one identity. In each quadruple, the first column is the one of the two input views, the second column is the reconstruction result of DRNRMVS [BCR*20], the third column is our reconstructed mesh, and the last column is our result rendered with reconstructed texture.*

Method	Mean (mm)	STD (mm)
DFNRMVS	1.040	0.33
DFNRMVS*	1.171	0.35
Baseline	1.152	0.351
Baseline + view-switch loss	1.108	0.163
Our full algorithm	**0.995**	0.177

Table 3: *Ablation study on the Stirling/ESRC dataset [SE18]. The symbol * denotes that the results are obtained by testing the released model of DFNRMVS [BCR*20] with the same samples as ours.*

4.5. Ablation Studies

We perform ablation study on the test set of Stirling/ESRC dataset to evaluate the impact of the proposed view-switch loss and residual latent code. The corresponding two-view reconstruction results are shown in Table 3. Our baseline is the reconstruction performance obtained by solving the optimization problem as Eq. (14), without using our residual latent code or view-switch loss.

As shown in Table 3, our proposed view-switch loss leads to better reconstruction performance. The improvement of our full algorithm with the addition of residual latent code is also noticeable (the last row in Table 3). This fact demonstrates that our residual latent codes effectively extend the shape space of the morphable models. Since the Stirling/ESRC test partition of [BCR*20] is not available, we follow the same selection strategy as in [BCR*20] and present the test results of their released model on our selected test samples for fair comparison in Table 3.

5. Conclusions

In this work, we present a novel method for 3D face reconstruction from multi-view images via implicit neural deformation. By using a neural SDF based representation, we are able to reconstruct faces with high-fidelity details even from sparse-view input of diverse expressions. Different from previous 3DMM based methods, we propose residual latent code to extend the shape space of implicit morphable models. To further enforce consistency among different views of one instance at test time, we introduce a novel view-switch loss for joint optimization of the network and latent code. Besides, we design a training strategy for the implicit neural network to alleviate the collapse issue during self-supervised optimization by introducing prior information of face geometry and colors. Our results on the Stirling/ESRC dataset and the Bosphorus dataset demonstrate that our approach outperforms alternative baselines and state-of-the-art methods.

Our current implementation of test-time reconstruction takes about two hours for a single instance and the major bottleneck is the ray-tracing module. We plan to integrate several recent approaches [RJY*21, GKJ*21, TCY*21, LGZL*20] for accelerated rendering of neural SDFs to speed up the computation.

Acknowledgements. We would like to thank anonymous reviewers for their valuable feedback and suggestions.

References

[AL20] Atzmon M., Lipman Y.: Sal: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 2565–2574. 2

[BBB*10] Beeler T., Bickel B., Beardsley P., Sumner B., Gross M.: High-quality single-shot capture of facial geometry. *ACM Trans. Graph. 29*, 4 (2010). 1

[BCLT21] Bai Z., Cui Z., Liu X., Tan P.: Riggable 3d face reconstruction via in-network optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2021), pp. 6216–6225. 1, 2, 6, 7

[BCR*20] Bai Z., Cui Z., Rahim J. A., Liu X., Tan P.: Deep facial non-rigid multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 5850–5860. 1, 2, 6, 7, 8

[BHPS10] Bradley D., Heidrich W., Popa T., Sheffer A.: High resolution passive facial performance capture. *ACM Trans. Graph. 29*, 4 (2010). 1

[BV99] Blanz V., Vetter T.: A morphable model for the synthesis of 3d faces. In *ACM SIGGRAPH* (1999), pp. 187–194. 3

[BV03] Blanz V., Vetter T.: Face recognition based on fitting a 3d morphable model. *IEEE Transactions on pattern analysis and machine intelligence 25*, 9 (2003), 1063–1074. 1, 2

[CCZ*19] Chen A., Chen Z., Zhang G., Mitchell K., Yu J.: Photo-realistic facial details synthesis from single image. In *Proceedings of the IEEE International Conference on Computer Vision* (2019), pp. 9429–9439. 2, 7

[CHZ14] Cao C., Hou Q., Zhou K.: Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph. 33*, 4 (2014), 1–10. 2

[CLC*22] Chan E. R., Lin C. Z., Chan M. A., Nagano K., Pan B., De Mello S., Gallo O., Guibas L. J., Tremblay J., Khamis S., et al.: Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition* (2022), pp. 16123–16133. 2

[CLL*21] Chen S.-Y., Liu F.-L., Lai Y.-K., Rosin P. L., Li C., Fu H., Gao L.: DeepFaceEditing: Deep face generation and editing with disentangled geometry and appearance control. *ACM Trans. Graph. 40*, 4 (2021), 90:1–90:15. 2

[CWZ*13] Cao C., Weng Y., Zhou S., Tong Y., Zhou K.: Facewarehouse: A 3d facial expression database for visual computing. *IEEE TVCG 20*, 3 (2013), 413–425. 2

[DNJ20] Davies T., Nowrouzezahrai D., Jacobson A.: Overfit neural networks as a compact shape representation. *arXiv preprint arXiv:2009.09808* (2020). 2

[DSK17] Dou P., Shah S. K., Kakadiaris I. A.: End-to-end 3d face reconstruction with deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 5908–5917. 2

[DYT21] Deng Y., Yang J., Tong X.: Deformed implicit field: Modeling 3d shapes with learned dense correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2021), pp. 10286–10296. 2

[DYX*19] Deng Y., Yang J., Xu S., Chen D., Jia Y., Tong X.: Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2019). 2, 6, 7

[DZW*20] Duan Y., Zhu H., Wang H., Yi L., Nevatia R., Guibas L. J.: Curriculum deepsdf. In *European Conference on Computer Vision* (2020), Springer, pp. 51–67. 2

[ERB*18] Eslami S. A., Rezende D. J., Besse F., Viola F., Morcos A. S., Garnelo M., Ruderman A., Rusu A. A., Danihelka I., Gregor K., et al.: Neural scene representation and rendering. *Science 360*, 6394 (2018), 1204–1210. 2

[EST*20] Egger B., Smith W. A., Tewari A., Wuhrer S., Zollhoefer M., Beeler T., Bernard F., Bolkart T., Kortylewski A., Romdhani S., et al.: 3d morphable face models—past, present, and future. *ACM Trans. Graph. 39*, 5 (2020), 1–38. 2

[FFBB21] Feng Y., Feng H., Black M. J., Bolkart T.: Learning an animatable detailed 3d face model from in-the-wild images. *ACM Trans. Graph. 40*, 4 (2021), 1–13. 7

[FWS*18] Feng Y., Wu F., Shao X., Wang Y., Zhou X.: Joint 3d face reconstruction and dense alignment with position map regression network. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 534–551. 2, 7

[GCS*20] Genova K., Cole F., Sud A., Sarna A., Funkhouser T.: Local deep implicit functions for 3d shape. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 4857–4866. 2

[GKJ*21] Garbin S. J., Kowalski M., Johnson M., Shotton J., Valentin J.: Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (2021), pp. 14346–14355. 8

[GSL*20] Gao C., Shih Y., Lai W.-S., Liang C.-K., Huang J.-B.: Portrait neural radiance fields from a single image. *arXiv preprint arXiv:2012.05903* (2020). 2

[GTZN21] Gafni G., Thies J., Zollhofer M., Niessner M.: Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2021), pp. 8649–8658. 2

[GYH*20] Gropp A., Yariv L., Haim N., Atzmon M., Lipman Y.: Implicit geometric regularization for learning shapes. In *Proceedings of the 37th International Conference on Machine Learning* (2020), pp. 3789–3799. 1, 2, 4

[GZC*16] Garrido P., Zollhöfer M., Casas D., Valgaerts L., Varanasi K., Pérez P., Theobalt C.: Reconstruction of personalized 3d face rigs from monocular video. *ACM Trans. Graph. 35*, 3 (2016), 1–15. 2

[JSM*20] Jiang C., Sud A., Makadia A., Huang J., Niessner M., Funkhouser T., et al.: Local implicit grid representations for 3d scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 6001–6010. 2

[KJJ*21] Kellnhofer P., Jebe L. C., Jones A., Spicer R., Pulli K., Wetzstein G.: Neural lumigraph rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 4287–4297. 2

[KKT*14] Kazemi V., Keskin C., Taylor J., Kohli P., Izadi S.: Real-time face reconstruction from a single depth image. In *International Conference on 3D Vision* (2014), pp. 369–376. 2

[KSW20] Kohli A. P. S., Sitzmann V., Wetzstein G.: Semantic implicit neural scene representations with semi-supervised training. In *International Conference on 3D Vision (3DV)* (2020), pp. 423–433. 2

[KZT*18] Kim H., Zollhöfer M., Tewari A., Thies J., Richardt C., Theobalt C.: Inversefacenet: Deep monocular inverse face rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 4625–4634. 2

[LC87] Lorensen W. E., Cline H. E.: Marching cubes: A high resolution 3d surface construction algorithm. In *ACM SIGGRAPH* (1987), pp. 163–169. 5

[LGZL*20] Liu L., Gu J., Zaw Lin K., Chua T.-S., Theobalt C.: Neural sparse voxel fields. In *Advances in Neural Information Processing Systems* (2020), pp. 15651–15663. 8

[LZP*20] LIU S., ZHANG Y., PENG S., SHI B., POLLEFEYS M., CUI Z.: Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 2019–2028. 4

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision* (2020), pp. 405–421. 2, 6

[PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 165–174. 1, 2

[PKA*09] PAYSAN P., KNOTHE R., AMBERG B., ROMDHANI S., VETTER T.: A 3d face model for pose and illumination invariant face recognition. In *IEEE International Conference on Advanced Video and Signal-based Surveillance* (2009), pp. 296–301. 1, 2

[RJY*21] REBAIN D., JIANG W., YAZDANI S., LI K., YI K. M., TAGLIASACCHI A.: Derf: Decomposed radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 14153–14161. 8

[RSOEK17] RICHARDSON E., SELA M., OR-EL R., KIMMEL R.: Learning detailed face reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 1259–1268. 2

[SAD*08] SAVRAN A., ALYÜZ N., DIBEKLIOĞLU H., ÇELIKTUTAN O., GÖKBERK B., SANKUR B., AKARUN L.: Bosphorus database for 3d face analysis. In *European workshop on biometrics and identity management* (2008), Springer, pp. 47–56. 6, 7

[SBFB19] SANYAL S., BOLKART T., FENG H., BLACK M. J.: Learning to regress 3d face shape and expression from an image without 3d supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 7763–7772. 2

[Sch66] SCHÖNEMANN P. H.: A generalized solution of the orthogonal procrustes problem. *Psychometrika 31*, 1 (1966), 1–10. 6

[SE18] STIRLING-ESRC: Stirling/ESRC 3D face database, 2018. http://pics.stir.ac.uk/ESRC/. 3, 6, 7, 8

[SHN*19] SAITO S., HUANG Z., NATSUME R., MORISHIMA S., KANAZAWA A., LI H.: Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE International Conference on Computer Vision* (2019), pp. 2304–2314. 2

[SLB*21] SUN T., LIN K., BI S., XU Z., RAMAMOORTHI R.: Nelf: Neural light-transport field for portrait view synthesis and relighting. In *Eurographics Symposium on Rendering (EGSR)* (2021), pp. 155–166. 2

[SSD*20] SMITH W. A., SECK A., DEE H., TIDDEMAN B., TENENBAUM J. B., EGGER B.: A morphable face albedo model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 5011–5020. 2

[SSL*20] SHANG J., SHEN T., LI S., ZHOU L., ZHEN M., FANG T., QUAN L.: Self-supervised monocular 3d face reconstruction by occlusion-aware multi-view geometry consistency. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2020), pp. 53–70. 1, 2

[SZW19] SITZMANN V., ZOLLHÖFER M., WETZSTEIN G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems* (2019), pp. 1121–1132. 2

[TBG*19] TEWARI A., BERNARD F., GARRIDO P., BHARAJ G., ELGHARIB M., SEIDEL H.-P., PÉREZ P., ZOLLHOFER M., THEOBALT C.: Fml: Face model learning from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 10812–10822. 2, 7

[TCY*21] TANG J.-H., CHEN W., YANG J., WANG B., LIU S., YANG B., GAO L.: Octfield: Hierarchical implicit functions for 3d modeling. In *The Thirty-Fifth Annual Conference on Neural Information Processing Systems (NeurIPS)* (2021). 2, 8

[TLY*21] TAKIKAWA T., LITALIEN J., YIN K., KREIS K., LOOP C., NOWROUZEZAHRAI D., JACOBSON A., MCGUIRE M., FIDLER S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 11358–11367. 2

[TSM*20] TANCIK M., SRINIVASAN P., MILDENHALL B., FRIDOVICH-KEIL S., RAGHAVAN N., SINGHAL U., RAMAMOORTHI R., BARRON J., NG R.: Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems* (2020), pp. 7537–7547. 6

[TTHMM17] TUAN TRAN A., HASSNER T., MASI I., MEDIONI G.: Regressing robust and discriminative 3d morphable models with a very deep neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 5163–5172. 2

[WBC*19] WU F., BAO L., CHEN Y., LING Y., SONG Y., LI S., NGAN K. N., LIU W.: Mvf-net: Multi-view 3d face morphable model regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 959–968. 2

[WLL*21] WANG P., LIU L., LIU Y., THEOBALT C., KOMURA T., WANG W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems* (2021). 2

[XYC*20] XU S., YANG J., CHEN D., WEN F., DENG Y., JIA Y., TONG X.: Deep 3d portrait from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 7710–7720. 2

[YGKL21] YARIV L., GU J., KASTEN Y., LIPMAN Y.: Volume rendering of neural implicit surfaces. In *Advances in Neural Information Processing Systems* (2021). 2

[YKM*20] YARIV L., KASTEN Y., MORAN D., GALUN M., ATZMON M., RONEN B., LIPMAN Y.: Multiview neural surface reconstruction by disentangling geometry and appearance. In *Advances in Neural Information Processing Systems* (2020), pp. 2492–2502. 1, 2, 4, 6

[YTB*21] YENAMANDRA T., TEWARI A., BERNARD F., SEIDEL H.-P., ELGHARIB M., CREMERS D., THEOBALT C.: i3dmm: Deep implicit 3d morphable model of human heads. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 12803–12813. 2, 3, 6

[YWP*18] YU C., WANG J., PENG C., GAO C., YU G., SANG N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 325–341. 5

[YYTK21] YU A., YE V., TANCIK M., KANAZAWA A.: pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 4578–4587. 2

[YZW*20] YANG H., ZHU H., WANG Y., HUANG M., SHEN Q., YANG R., CAO X.: Facescape: a large-scale high quality 3d face dataset and detailed riggable 3d face prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 601–610. 2

[ZLY*15] ZHU X., LEI Z., YAN J., YI D., LI S. Z.: High-fidelity pose and expression normalization for face recognition in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 787–796. 1, 2

[ZPK18] ZHOU Q.-Y., PARK J., KOLTUN V.: Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847* (2018). 6

[ZWC*20] ZHU W., WU H., CHEN Z., VESDAPUNT N., WANG B.: Reda: reinforced differentiable attribute for 3d face reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020), pp. 4958–4967. 2

[ZYDL21] ZHENG Z., YU T., DAI Q., LIU Y.: Deep implicit templates for 3d shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2021), pp. 1429–1439. 1, 2