Training and Tuning Generative Neural Radiance Fields for Attribute-Conditional 3D-Aware Face Generation

Jichao Zhang, Aliaksandr Siarohin, Yahui Liu, Hao Tang, Nicu Sebe Senior Member, IEEE and Wei Wang

Abstract—Generative Neural Radiance Fields (GNeRF)-based 3D-aware GANs have showcased remarkable prowess in crafting high-fidelity images while upholding robust 3D consistency, particularly face generation. However, specific existing models prioritize view consistency over disentanglement, leading to constrained semantic or attribute control during the generation process. While many methods have explored incorporating semantic masks or leveraging 3D Morphable Models (3DMM) priors to imbue models with semantic control, these methods often demand training from scratch, entailing significant computational overhead. In this paper, we propose a novel approach: a conditional GNeRF model that integrates specific attribute labels as input, thus amplifying the controllability and disentanglement capabilities of 3D-aware generative models. Our approach builds upon a pre-trained 3D-aware face model, and we introduce a Training and Tuning (TRIOT) method to train a conditional normalized flow module to enable the facial attribute editing, then optimize the latent vector to improve attribute-editing precision further. Our extensive experiments substantiate the efficacy of our model, showcasing its ability to generate high-quality edits with enhanced view consistency while safeguarding non-target regions. The code for our model is publicly available at [https://github.com/zhangqianhui/TF-GNeRF](https://github.com/zhangqianhui/TF-GNeRF).

Index Terms—Neural Radiance Fields, Generative Model, Generative Adversarial Networks, Image Generation and Editing.

I. INTRODUCTION

High-quality image generation and semantic disentanglement are long-standing goals in computer vision and computer graphics. In recent years, Generative Adversarial Networks (GANs) [1] and their variants have received significant attention for their ability to produce high-quality image generation and editing. These methods have greatly improved visual fidelity, rendering speed, and interactive controls compared to traditional computer graphics pipelines.

Numerous previous works [2]–[5] have focused on realistic face editing. These approaches rely on image-to-image translation models [2], [6] or leverage the disentanglement abilities [3]–[5] of StyleGAN [7], [8]. These methods can be broadly classified into supervised and unsupervised categories. Unsupervised methods typically search for interpretable directions using PCA [9] or introduce soft orthogonality constraints [10], [11] in the latent space. However, these approaches provide only coarse controls. Supervised methods [2], [3], on the other hand, utilize specific attribute labels as conditions. However, these methods lack precise control over 3D factors such as camera pose because they overlook the underlying 3D scene rendering process. To address this issue, some works [4], [12] have integrated 3D Morphable Face Models (3DMM) [13] to enable control over 3D face pose and facial expression. However, these approaches still suffer from significant challenges, such as view inconsistency and unrealistic texture distortion when poses are drastically varied.

Recently, neural radiance fields (NeRF) [14] have attracted much attention due to their impressive results in novel view-rendering tasks. Specifically, NeRF represents a scene using a continuous function parameterized by a multi-layer perceptron (MLP) that maps a 3D position and a viewing direction to density and radiance values. Since then, many works have been proposed to improve NeRF [15], [16] and apply it to various downstream tasks, such as human body modeling [17] and large scene modeling [18].

Some 3D-aware image generation methods [19], [20] combine NeRF with generative models by extending neural radiance fields with latent conditioning, called Generative Neural Radiance Fields (GNeRF). 3D coordinates are sampled from random camera poses and used as input to an implicit function with latent codes. This function predicts the density and the RGB color. However, these methods are computationally intensive and memory inefficient because they sample many rays in the entire 3D-volume space and require a feed-forward process for each point. They are limited to low-resolution and low-quality generation. To improve generation quality and view-consistency, many approaches [21]–[23] borrow ideas from StyleGAN and integrate the ‘Style-modules’ into the implicit function (e.g., SIREN [20]) or neural rendering module. Some novel algorithms and losses have been carefully designed for 3D-aware generation, such as tri-planes [22] or multiple-view warping loss [23]. Although these models create high-quality, view-consistent images, they lack control and disentangling abilities. As explained in VolumeGAN [24], some models are limited to local receptive fields with MLPs, and extracting
global structures from their internal representation is difficult. Thus, VolumeGAN utilizes a 3D feature volume module for querying coordinate descriptors, enabling independent controls on the texture and structure factors. However, VolumeGAN still faces quality and view consistency challenges and does not support attribute controls for face manipulation. Recently, some papers [25]–[29] utilize the 3DMM mesh or semantic masks as a condition to edit or control the 3D face avatar. Mesh-guided methods [26], [27] prioritize expression and head pose transfer but may struggle with appearance editing like “Hair Color” manipulation. Mask-guided methods [25], [28], [29] excel in local attribute editing. However, they may be less effective for global attribute manipulation. An important consideration for both mesh-guided and mask-guided methods is the requirement to train models from scratch, which demands significant computational resources. This high computational cost is an important aspect to keep in mind when considering the feasibility and scalability of these methods.

In this paper, we introduce an attribute-conditional 3D-aware generative model aimed at controlling facial attributes while addressing existing challenges. Our approach offers two key advancements compared to prior methods. Firstly, we leverage attribute labels as conditions for attribute editing tasks, supporting global and local attribute editing. Secondly, we employ a pre-trained 3D-aware GNeRF model as the backbone, eliminating the need for extensive retraining, thus reducing computational resources and time. We propose the “Training-As-Init, Optimizing-for-Tuning” (TRIOT) method (Fig. 2) to achieve latent disentanglement for precise facial attribute editing. TRIOT consists of two main steps. Initially, we adopt a methodology similar to previous 2D techniques [3] to train a conditional normalizing flow, enabling the learning of latent space distributions and facilitating multiple attribute editing via label input. Subsequently, we utilize the edited latent vector corresponding to target attributes as initialization. This initialized latent vector is then optimized using our proposed semantic-guided texture and geometry consistency losses to enhance editing precision and preserve non-target regions in images.

Furthermore, to explore geometry and texture disentanglement, we introduce an unsupervised optimization method to enable reference-based facial geometry transfer tasks (Fig. 5). This method enhances the ability to transfer facial geometry while maintaining texture consistency, thereby broadening the scope of applications for our model. By combining these advances, our model offers a robust framework for precise attribute editing in facial images, with efficient resource utilization and enhanced disentanglement capabilities.

In summary, the main contributions of this work are:
1) We propose a “Training-as-Init, Optimizing-for-Tuning” (TRIOT) method, integrating the module-training and latent-optimization method for disentangling latent space and enabling attribute-editing tasks.
2) Our method is flexible and general, and it can be seamlessly integrated into most 3D-Aware GAN architectures.
3) Compared to the baseline models, our model achieves high-quality editing with improved view consistency while preserving non-target regions.
4) To explore geometry and texture disentanglement, we propose an unsupervised optimization method for the reference-based geometry transfer task.

II. RELATED WORK

Generative Neural Radiance Fields for 3D-Aware Face Generation. Neural radiance fields (NeRF) [14], a continuous neural mapping from a 3D position and a 2D viewing direction to the RGB value and density that allows 3D scene modeling and high-quality novel view synthesis. Recently, several
NeRF-based methods were proposed to improve rendering speed [16, 30, 31] and rendering quality [32–34]. Moreover, NeRF also promotes the development of many computer-graphics applications, such as human body modeling [17], 3D-aware face generation and editing [21, 22, 24, 25, 36–42], large scene modeling [18], and pose estimation [19].

Generative neural radiance fields (GNeRF) are a conditional variant of NeRF, which combines NeRF with GANs to condition the rendering process on a latent code that governs the object’s appearance and shape [19, 20, 36]. For example, GRAF [43] achieves this goal by incorporating shape and appearance codes as input. GRAF [43] achieves better visual fidelity and view consistency than the previous voxel- and feature-based methods [44, 45]. Michael et al. [56] propose the compositional neural feature fields (GIRAFFE) that extend GRAF into 3D-aware multiple-object scene representations. Although GRAF and GIRAFFE can control texture and camera pose, they are limited to low-resolution results and fail to preserve multi-view consistency. Many works [20–23, 37, 46–52] are trying to address these problems, and most of them inherit the “image-as-style” idea from StyleGAN [8]. Yang et al. [49] extend the GIRAFFE to work with high-resolution data. However, this model still suffers from the view-inconsistency problems. Pi-GAN [20] proposes a SIREN module with periodic activation functions. It conditions the style code by feature-wise linear modulation (FILM). The SIREN modules significantly boost image quality and view consistency. To reduce the high computational costs of the volume rendering in Pi-GAN, some models, such as StyleSDF [21], MVCGAN [23] and EG3D [22] propose a hybrid rendering approach. Specifically, they learn a coarse feature field, render it into a low-resolution feature map, and then utilize a style-based 2D network as a “super-resolution” module to refine the features for a final high-resolution image.

In order to improve view consistency, StyleSDF models signed distance fields, while MVCGAN uses explicit multi-view consistency loss. On the other hand, ED3D proposes a hybrid 3D tri-plane representation. Unlike the mentioned works, CIPS-3D [46] keeps the resolution of intermediate feature fields the same as the resolution of the final images. These models can achieve incredible quality generation with strong view consistency, but they cannot edit structures and textures.

Some research has focused on the disentangling abilities of 3D-aware models. VolumeGAN [24] tries to separate shape from texture, while ShadeGAN [48] and NeRFFaceLighting [53] disentangle the light from the albedo. However, they only focus on global factors, such as illumination and textures, and cannot handle more specific attributes, such as hair color and gender.

Recent methods also explore the semantic control of 3D-aware face model and can be categorized into two types: mesh-guided, and mask-guided (or 2D sketches). Mesh-guided Methods [26, 27] prioritize expression and head pose transfer but may struggle with appearance editing like “Hair Color” manipulation. On the other hand, mask (or 2D sketches)-guided methods [25, 28, 29] excel at local attribute editing but may not be as effective for global attributes. It’s worth noting that both mesh-guided and mask-guided methods often necessitate training models from scratch. Our proposed method addresses these limitations by disentangling the latent space of a pre-trained 3D-aware GAN, enabling both global and local attribute editing. In a similar vein, LatentSwap3D [54] also facilitates attribute editing within the latent space of pre-trained 3D-aware GANs. This method employs a random forest classifier to rank features and identify dimensions controlling specific attributes. By swapping the most relevant features, it accomplishes attribute editing tasks. However, it suffers from low editing intensity, particularly in attributes like “Hair Color” and “Gender”. The discussion and comparison

![Diagram](image-url)
with LatentSwap3D can be found at Section 8.

**Image-to-Image Translation Architectures for Face Editing.** Image-to-image translation models, i.e., Pix2Pix [55] and CycleGAN [56], utilize the autoencoder as a generator that has been widely adopted for a variety of different tasks, including face attribute editing [2], [6], [57]–[63]. Specifically, StarGAN [2] is the early work for learning multiple domain face translation, it takes multiple attributes as input, and it transfers one face image from one domain to other domains. During the training, StarGAN exploits a reconstruction and cycle consistency loss to preserve the content of the input face. After that, many works have been improving StarGAN, such as AttGAN [57], STGAN [6], SSCGAN [59] and HifaFace [61]. However, their operation typically involves lower-resolution data (e.g., 256), limiting their capacity to manipulate 3D factors such as camera poses. On the other hand, our approach operates on higher-resolution data (e.g., 1024), enabling for more detailed manipulations and incorporating 3D factors seamlessly.

**Interpreting Latent Space of StyleGAN for Face Editing.** An alternative line of works explores the disentanglement of StyleGAN’s latent space for face editing. These approaches can be roughly classified into two types based on whether they use semantic labels: unsupervised methods and attribute-conditional methods. The former learns to discover interpretable directions in latent space by leveraging techniques such as Principal Component Analysis (PCA) [9], closed-form factorization [64], learnable orthogonal matrices [10], [11], and regularization losses [65], [66]. GANSpace [9] shows that PCA in the latent space of StyleGAN can find important interpretable directions that can be used to control image generation. To compute the PCA of the style codes, GANSpace samples multiple random vectors (i.e., \( z \) space) and computes the corresponding style codes (i.e., \( W \) space). To avoid the extensive data sampling of GANSpace, SeFa [64] directly decomposes the model weights with a closed-form solution. Similarly, recent works [10], [11] propose to obtain a disentangled latent space by learning an orthogonal matrix for editing latent code.

As far as we know, the attribute conditions can be of different types, including global-level (e.g., label vectors) and local-level (e.g., semantic segmentation maps) modalities. The first type [3], [67], [68] usually utilizes off-the-shelf attribute classifier networks to obtain the attribute vectors of training images and then uses these vectors as input. For example, StyleFlow [3] proposes to utilize conditional normalizing flow (CNF) to model the mapping from conditional labels and latent codes (\( z \) space) to intermediate vectors (\( W \) space). StyleFlow trains the flow model (CNF) with triplets consisting of vectors sampled from \( W \) space, corresponding faces, and predicted face attributes. Though StyleFlow can produce facial pose transformation, it suffers from serious view inconsistency as it lacks an understanding of the underlying 3D world. The second type utilizes the coarse masks [69] or predicts face semantics [70] with k-means.

Additionally, some methods [71]–[74] also explore the semantic disentanglement of the model, but they redesign the StyleGAN; thus, they need to retrain the generator. For example, TransEditor [72] presents a transformer-based module for dual space interactions where one latent code is used as the key and value and the other as the query. This dual-space interaction helps disentangle the style and the content representations. Some works focus on local facial controls by integrating face parsing into generation [75] or by adding spatial information for styles code with the conv-based module [74].

Overall, these StyleGAN-based models have demonstrated the ability to produce high-quality images and perform precise editing. However, their limitations become evident when attempting to alter facial pose while maintaining view consistency. This challenge arises from their lack of robust 3D modeling abilities. However, our method overcomes this hurdle by inherently incorporating 3D modeling techniques (GNeRF), allowing for dynamic changes in facial pose while ensuring view consistency.

**3DMM-Guided Face Generation and Editing.** Recently, some works [4], [5], [12], [75], [76] demonstrate high-control over GAN generation via a 3DMM [13]. 3DMM is the 3D Morphable Face Model parameterized by the face shape, expression, and texture. For example, Geng et al. [75] utilizes 3DMM to guide fine-grained face manipulation for arbitrary expression transfer. First, they extract texture and shape coefficients by fitting 3DMM to each real face in the dataset. Then, they utilize the texture generator to create the target textures with the source texture and the target expression and utilize the shape predictor to produce the target shape with the source shape coefficients and the target expression as input. Finally, the global generator utilizes rendered faces and the target expression to produce the final faces. StyleRig [4] and DiscoFaceGAN [12] use 3DMM to manipulate the latent space of StyleGAN. While StyleRig is based on pre-trained StyleGAN models and only tunes a DFR module that learns the mapping from latent code to the coefficients of 3DMM. On the other hand, DiscoFaceGAN re-trains the entire model. It exploits multiple VAE to model the distribution of 3DMM coefficients and introduces self-supervised losses to disentangle different factors. Compared to these models, our model does not require a 3DMM prior and still achieves better multi-view consistency. Additionally, our models can achieve more diverse face editing, such as hair color and age.

Finally, Shi et al. [5] presents a LiftedGAN model, which lifts the pre-trained StyleGAN2 in 3D. This model is free of 3DMM prior. However, this model cannot achieve attribute-conditional control.

**GAN Inversion for Real Face Editing.** GAN inversion aims to find an optimal latent code corresponding to a given real image and has been widely used for real image editing tasks. The previous methods can be divided into two broad categories: optimization-based [77]–[79] and encoder-based [80]–[83]. For example, Roich et al. [79] presents a novel optimization-based method called Pivotal Tuning Inversion (PTI). In PTI, they first obtain the optimized latent code as the pivot by fixing the parameters of the generator, then fix this pivot and finetune the generator parameters to obtain better reconstruction while preserving the editing abilities of the latent code. After the inversion step, they utilize the popular latent-disentanglement method, such as InterfaceGAN or GANSpace, for face editing.
In this paper, we use the PTI for GAN inversion. Concurrent with our work, some methods [25], [28], [84], [85] employ 3D-aware GAN as a basic model instead of StyleGAN to achieve multi-view consistent face editing guided by the segmentation masks. They also apply GAN inversion to project real images into latent space for editing. Different from these methods with segmentation masks, we use attribute labels to guide face editing.

III. Preliminaries

Our method can work with most of GNeRF backbones, and we showcase it with the two most recent ones, StyleSDF [21], EG3D [22]. Since they have similar architecture, we only describe StyleSDF [21].

A. Generative Neural Radiance Fields (GNeRF)

NeRF [14] is a continuous neural mapping $\mathcal{M}$ that maps a 3D position $x$ and a 2D viewing direction $v$ to the rgb color $c$ and density $\sigma$:

$$ (c, \sigma) = \mathcal{M}(\gamma(x), \gamma(v)), $$

where $\gamma$ indicates the positional encoding mapping function.

GNeRF, such as GRAF [43], is a conditional variant of NeRF. Unlike NeRF, which requires multiple views of a single scene with estimated camera poses, GNeRF can be trained with unposed 2D images from different scenes. In PiGAN [20], GNeRF is trained with adversarial learning and conditioned on a latent code $z$:

$$ (c, \sigma) = \mathcal{M}(\gamma(x), \gamma(v), z), $$

where the latent code $z$ with following MLP layers aims to infer frequencies $\alpha$ and shifts $\beta$ of a SIREN layer [20].

StyleSDF. As shown in Fig. 2, StyleSDF also adopts the SIREN layers inside GNeRF. However, it utilizes Signed Distance Fields (SDF) to improve the GNeRF and add a 2D StyleGAN generator as a second-stage rendering. In the first stage, the GNeRF is trained separately. It produces a feature vector $V$, RGB color $c$ and SDF values $d$:

$$ (V, c, d) = \mathcal{M}(\gamma(x), \gamma(v), w), $$

where the learned SDF values define the object surface and thus allow the extract of the mesh via Marching Cubes [86]. Moreover, $d$ will be converted into the density $\sigma$ for volume rendering.

The RGB color $c$ is later rendered into the low-resolution face image using the classical volume rendering. The output image is then used as input for the discriminator.

To achieve high-resolution output, the 3D volume feature $V$, undergoes volume rendering to produce a lower-resolution feature map, represented as $V_r$. Subsequently, $V_r$ is processed through Style-based convolutional modules to generate a high-resolution image. This high-resolution image is subsequently as input of another discriminator.

IV. The Proposed Method

First, we detail the proposed ‘Training-As-Init, Optimizing-for-Tuning’ (TRIOT) method. Next, we introduce a new method for the reference-based geometry transfer task.

A. Training-as-Init, Optimizing-for-Tuning with Attribute-Specific Consistency Losses

As mentioned above, our proposed TRIOT has two stages: the training stage and the optimization stage. Given a pre-trained StyleSDF (or EG3D), we sample training data which includes lots of triplets: latent vector $u$ ($W+$ space), the corresponding generated sample $I$ along with its low-resolution version $I_L$ and attribute labels $A$ predicted by off-the-shelf attribute classifiers. During the training phase, the pre-trained StyleSDF model is augmented with a continuous normalized flows module. This module is trained using both latent vectors and corresponding labels. Subsequently, it facilitates the manipulation of latent codes, thereby enabling 3D-aware attribute editing during the inference phase. More details will be introduced below.

Training Stage with Continuous Normalized Flows. We followed StyleFlow to use continuous normalized flows (CNF) to learn the conditional distribution of latent code $u$. Normalized Flows are a type of generative model to map the prior distribution $z \sim p_z$ to a more complex distribution $x \sim p_x$, where the basic blocks of a normalizing flow is an invertible transformation $x = f(z)$. Let define $f = f_1 \circ f_2 \circ ..., \circ f_n$, we have $x_n = f_n \circ ..., \circ f_1(x_0)$. Then, it can be shown that $f$ is also bijective and invertible. Though a chain of mapping $f_i$, $p(x_n)$ can be represented as using discrete normalized flows:

$$ \ln p_{\theta}(x_n) = \ln p_0(x_0) - \sum_{i=1}^{n} \ln | \det \frac{df_i}{dx_{i-1}} |. $$

where the function $f_i$ can be modeled by a neural network.

StyleFlow uses the continuous normalized flows instead of the discrete normalized flows (DNF) to avoid the costly computation of Jacobian determinants in DNF while ensuring learning a reversible mapping. Specifically, the change in log density is:

$$ \ln p(x_{t_i}) = \ln p(x_{t_{i-1}}) - \int_{t_{i-1}}^{t_i} Tr(dF_{x_i}) dx_i, $$

where $t$ is the time variable in CNF. We use some CNF blocks to represent the function $F$, then train the CNF blocks using the latent code $u$ and the corresponding condition $A$. In the reference stage, we reversely inference the $z_r$ from $u_r$, using the original labels $A_r$, as the condition, then transfer $w_r$ to new latent code $w_r$, given target labels $A_r$, by the forward inference.

Optimization Stage with Attribute-Specific Consistency Losses. Following training, latent codes are manipulated to generate images with altered attributes while maintaining view consistency. However, modifications to certain attributes can inadvertently affect unrelated attributes, particularly those that are local in nature. For instance, transferring a “No-Smile” face into a “Smile” can unintentionally alter the perceived identity of the individual. This phenomenon occurs due to the entanglement of latent codes for various attributes. To alleviate this problem, we propose an optimization stage to search for better latent codes corresponding to the better trade-off between the preservation of non-target regions and target-region editing. We propose an unsupervised method to extract attribute-specific masks to attain the target region, as
shown in Fig. 4. Motivated by previous feature visualization methods [70, 87], we explore applying spherical k-means clustering [88] to the 3D Volume or rendering features of 3D GAN. Specifically, the tensor of 3D VOLUME is defined as \( V \in R^{N \times C \times P \times H \times W} \), where \( N \) is the number of images, \( C \) is the dimension of channels, \( P \) are sampling points of single ray, and \( H, W \) are spatial dimensions. \( V \) are rendered into features \( V_r \in R^{N \times C \times H \times W} \). Fig. 3 visualizes \( S_A \) and \( S_B \). We find that \( S_A \) (4-th column) lacks specific meaningful semantics compared to \( S_B \) (3rd column). We choose \( k = 4 \) (2nd column) for its more accurate semantic decomposition. Specifically, we utilize masks derived from \( S_B \) for attributes like “Hair Color”, “Gender” and “Age”. For other localized attributes, we rely on pre-defined fixed rectangular masks for each facial image, given the well-aligned nature of the training faces. Fig. 4 shows all masks. With these masks, we define the non-target region as \( M \) and target region as \( 1 - M \). As shown in the right of Fig. 2 we obtain: \( I_r, I_r^p, I_e, I_e^p \), and an initial latent code \( w_e \). Given the corresponding optimization objective, we aim to find an optimal latent code \( \hat{w}_e \), corresponding image \( \hat{I}_e \) and normal map \( \hat{I}_e^p \). We use \( \hat{w}_e \) as the target code of the optimization pipeline with corresponding face image and normal pairs (\( \hat{I}_e, \hat{I}_e^p \)). The optimization objective functions are defined as:

\[
\mathcal{L}_{\text{ntloss}} = \| M \odot (\hat{I}_e - I_e) \|_1 + \| (1 - M) \odot (\hat{I}_e - I_e) \|_1,
\]

\[
\mathcal{L}_{\text{ngloss}} = \| M \odot (\hat{I}_e^p - I_e^p) \|_1 + \| (1 - M) \odot (\hat{I}_e^p - I_e^p) \|_1,
\]

\[
\hat{w}_e = \arg \min_{\hat{w}_e} \mathcal{L}_{\text{ntloss}} + \mathcal{L}_{\text{ngloss}},
\]

Fig. 5: Reference-based geometry transfer pipeline. It minimizes the difference between normals (geometry loss) and the differences between texture images (texture loss) in perceptual space to search for a better \( \hat{w}_e \).

This texture loss (\( \mathcal{L}_{\text{ntloss}} \)) reduces differences between the reconstructed texture \( \hat{I}_e \) and the optimized texture \( \hat{I}_e \) in the non-target region while keeping \( \hat{I}_e \) the same as \( I_e \) in the target region for attribute-editing. The geometry consistency loss (\( \mathcal{L}_{\text{ngloss}} \)) has the same objective function as the texture loss.

By iteratively optimizing the latent code \( \hat{w}_e \), we can achieve enhanced editing results while effectively preserving non-target regions. This process typically requires fewer than 1,000 iterations, approximately taking two minutes.

B. Reference-Based Geometry Transfer

Our model can achieve reference-based geometry manipulation by transferring the geometry from a reference face into a target face while preserving the target face’s appearance. Like TRIOT, the optimization objection includes two parts: the geometry consistency loss and the texture consistency loss. Given a reference image and corresponding normal map, we minimize the differences between reference and target normals (geometry loss) and the differences between the input image and target image (texture loss) in perceptual space. Specifically, we use \( L_1 \) loss to compute the normal differences and \( L_{\text{PIPS}} \) metric to compute the texture differences in perceptual space. In addition, we incorporate an ID loss component aimed at preserving the unique identity features present in the target face.

As shown in the Fig. 5, given the original latent code \( w_e \) and reference code \( w_r \), the corresponding face image and normal pairs are \( (I_e, I_e^p) \) (\( I_r, I_r^p \)), respectively. We use \( \hat{w}_e \) as the target code of the optimization pipeline with corresponding face image and normal pairs (\( \hat{I}_e, \hat{I}_e^p \)). The optimization objective functions are defined as:

\[
\mathcal{L}_{\text{loss}} = L_{\text{PIPS}}(I_e, \hat{I}_e) + ID(I_e, \hat{I}_e),
\]

\[
\hat{w}_e = \arg \min_{\hat{w}_e} \mathcal{L}_{\text{ntloss}} + \mathcal{L}_{\text{ngloss}},
\]

where we employ a pre-trained face recognition network, ArcFace [89], to calculate the ID loss for both images. This optimization phase efficiently discovers a latent code for precise reference-based geometry editing within just a few hundred steps.
C. GAN Inversion

We utilize the state-of-the-art GAN Inversion method (PTI) to project real images into the latent space of our 3D-GAN. Then, we perform real image editing using our proposed TRIOT methods. In detail, we follow two steps of PTI: 1) optimizing the latent code to obtain the corresponding projected latent of the real image while fixing the generator parameters (including CNF Blocks); 2) fixing the optimized latent code and the parameters of CNF parameters while tuning the remaining parameters of the generator. Then, given the generator’s tuning parameters and projected latent code $w$ as the initial point, we used the proposed TRIOT method to search for better latent code following the texture and geometry consistency loss. Afterward, we can achieve precise image editing for given real images.

V. EXPERIMENTS

We propose a method named TT-GNeRF (Training and Tuning Generative Neural Radiance Field). Within this framework, TT-GNeRF (S) and TT-GNeRF (E) denote the implementations of our approach using two different backbone networks: StyleSDF [21] and EG3D [22], respectively. Additionally, we apply our method for StyleNeRF [37] and show results in the supplementary materials. We refer to the demo video [1] for more results on multiple-view attribute editing, reference-based geometry editing, and GAN inversion for real image editing.

A. Setting

Datasets. Given that the two backbones (i.e., StyleSDF and EG3D) are trained on the FFHQ dataset [7], we train our model using sampled images and their corresponding latent codes. We create a dataset comprising 40,000 images at a resolution of $1024 \times 1024$ for training StyleSDF and an additional 40,000 images at a resolution of $512 \times 512$ for EG3D. We employ off-the-shelf attribute-classifiers [8] to obtain six attribute labels, including Hair Color, Gender, Bangs, Age, Expression (Smiling), and Beard. We use all generated triplets to train our models. The real images used for GAN inversion in Fig. 14 are gathered from the Internet and subsequently aligned.

Implementation Details. During the training stage, CNF Blocks are trained using the Adam optimizer [90] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a learning rate of $1e-4$ for 10,000 steps.

\begin{table}[h]
\centering
\caption{Quantitative results on the attributes editing results using four metrics: FID, Classification Accuracy (CA), Average Matching Point (aMP), and Face Recognition Similarity (FRS).}
\begin{tabular}{|l|c|c|c|c|c|c|c|c|c|c|c|}
\hline
\textbf{Method} & \textbf{Expression} & \textbf{Gender} & \textbf{Age} & \textbf{FID} \downarrow & \textbf{CA} \uparrow & \textbf{aMP} \uparrow & \textbf{FRS} \uparrow & \textbf{FID} \downarrow & \textbf{CA} \uparrow & \textbf{aMP} \uparrow & \textbf{FRS} \uparrow \\
\hline
DiscoFaceGAN [12] & 77.84 & 55.0 & 1347.4 & 0.587 & - & - & - & 85.61 & 89.9 & 1090.8 & 0.586 \\
LiftedGan [5] & 95.25 & - & 1484.0 & 0.464 & - & - & - & 95.32 & 99.5 & 999.82 & 0.490 \\
StyleFlow [3] & 78.98 & 99.2 & 1089.7 & 0.586 & 82.80 & 79.9 & 1088.4 & 0.588 & 95.61 & 89.9 & 1090.8 & 0.586 \\
TransEditor [72] & 55.97 & 90.5 & 1075.6 & 0.564 & 76.73 & 96.3 & 964.30 & 0.575 & 78.32 & 99.5 & 999.82 & 0.490 \\
TT-GNeRF (E) & 39.52 & 94.5 & 1292.5 & 0.587 & 57.43 & 97.1 & 1527.7 & 0.584 & 54.38 & 98.2 & 1643.5 & 0.867 \\
TT-GNeRF (S) & 58.55 & 95.8 & 1870.4 & 0.829 & 54.78 & 96.2 & 1823.2 & 0.821 & 55.28 & 90.5 & 1974.6 & 0.834 \\
\hline
\end{tabular}
\end{table}

In the optimization stage, we also utilize the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and a learning rate of $5e-4$. Here, the number of steps is set within the range of (100 to 200) for all attributes. For the reference-based geometry-transfer task, we employ the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a learning rate of $5e-3$. The number of steps is fixed at 100. Both $\lambda_1$ and $\lambda_2$ in Eq. 8 are set to 1 and 10, respectively.

Compared Baselines. Since our method is an attribute-conditioned generative model, the most similar supervised method is StyleFlow [3] and LatentSwap3D [54]. StyleFlow is a 2D method for face attribute editing tasks by manipulating the latent space of a 2D generator. To ensure a fair comparison, we extend the original Styleflow to work with 3D-aware generators, i.e., EG3D and StyleSDF. LatentSwap3D is a 3D method that is applied to multiple backbones, such as EG3D.

Thus, we compare our method to LatentSwap3D based on the backbone EG3D. Moreover, we adopt the state-of-the-art generative model, TransEditor [72] as our baseline for comparing face semantic disentanglement with multiple-view generation results. We also compare with the 3DMM-guided model, DiscoFaceGAN [12]. Note that this model can only edit some expression-related attributes, such as “Smile”. We adopt a 3D-Aware LiftedGAN [5] to compare multiple-view generation. However, LiftedGAN cannot control individual attributes. LenovoRF [42] is a very similar model to our TT-GNeRF but does not release the code. DragGAN [91] also optimizes a latent space of GAN to achieve image editing. We do not compare our method with DragGAN as it focuses on the semantic control of 2D GAN. We compare our method with StyleFlow [3] and GOAE [92] for real image editing. GOAE is a recent 3D-aware image inversion and editing method.

Evaluation Metrics. Five metrics are used for evaluation: FID (Fréchet Inception Distance) score [93] to evaluate the quality and diversity of edited results; Classification Accuracy (CA) to evaluate the correctness of edited attributes; average Matching points (aMP) [94] and Face Recognition Similarity (FRS) [67] to quantitatively evaluate the consistency of multiple-view generation results; and Local Preservation (LP) to evaluate the preservation of non-target regions in editing results.

To evaluate the quality and diversity of edited results, we calculate the FID score [93] by using samples from FFHQ as the real distribution and the original image and its edited results as the fake distribution. To calculate FID scores, we sample 5000 real and fake samples from all models for each attribute. A lower FID score indicates a smaller discrepancy between the image quality of the real and generated images.

\footnotesize{https://ttgerf.github.io/TT-GNeRF/}
To evaluate the accuracy of the attribute transfer, we use the off-the-shelf classifiers \cite{8} to classify edited samples and compute the accuracy by comparing the predicted and target labels. We refer to this metric as Classification Accuracy (CA). We calculate the CA using 1000 edited samples from all models for each attribute. Higher CA indicates better accuracy.

Evaluating view consistency without ground truth is a challenging task. We address this challenge by using the average Matched Points metric \cite{94}. This metric involves computing a point-wise matching between two images ($I_1$, $I_2$).
Fig. 8: Qualitative comparisons with StyleFlow [3] and LatentSwap3D [54] based on EG3D.

Fig. 9: Qualitative comparisons with StyleFlow [3] and LatentSwap3D [54] based on EG3D.

generated from the same identity but with different viewpoints using Patch2Pix [95] and counting the number of Matched Points $MP(I_1, I_2)$. We calculate the mean of $MP$ across all pairs of samples for 100 random identities with ten views each to obtain a final average $MP$ (aMP) score. Additionally, we use Face Recognition Similarity (FRS) [67] to evaluate identity preservation across different views. Specifically, we use ArcFace [89], a widely used face recognition method, to estimate feature similarity between two facial images and compute the average score across 1000 samples with ten different views and 100 identities. Higher aMP and FRS scores indicate that synthesized images with different viewpoints have more similar identities to input faces.

We use 1000 input-edited paired samples to evaluate the Local Preservation score (LP) for local attributes, such as “Expression”. For every paired sample, we use the defined masks of Fig. 4 to attain the non-target region. Then, we measure the differences between each paired sample using $L_1$ distance and average them across all pairs.

We have observed limitations in LiftedGAN and DiscoFaceGAN when directly manipulating attributes like “Gender” and “Age” for editing tasks. Consequently, we have opted not to assign scores for these methods in such cases. Instead, we have evaluated LiftedGAN based on FID, aMP, and FRS scores using randomly generated samples rather than attribute-edited ones.
TABLE II: Quantitative evaluation of the editing and preservation trade-off with the baselines, StyleFlow and LatentSwap3D, based on the EG3D. We use two metrics, CA and LP, to evaluate the editing of three attributes: Expression, Gender, and Age. Note that we define the non-target region by the mask of Fig. 4 to compute LP for three attributes.

| Method                  | Expression | Gender | Age |
|-------------------------|------------|--------|-----|
|                         | CA ↑       | LP ↓   | CA ↑| LP ↓| CA ↑  | LP ↓|
| StyleFlow               | 97.1       | 29.62  | 100 | 17.99 | 95.3  | 19.60|
| LatentSwap3D            | 93.2       | 8.070  | 74.5 | 6.243 | 88.2  | 6.002|
| TT-GNeRF (E)            | 94.5       | 17.49  | 97.1 | 9.476 | 96.2  | 9.258|

Input 20 50 100 150 200 300 500

Fig. 10: Ablations for the optimization steps. The attributes “Expression” and “Age” editing results are shown in the 2nd-8th columns.

B. State-of-the-Art Comparison

We compare our method to 2D baselines on facial attribute editing with multiple-view generation in Fig. 6 and Fig. 7. As aforementioned, we use “Expression”, “Gender” and “Age” attributes as examples since most baselines can directly edit the three attributes. Fig. 8 shows the results for “Expression” editing. We can observe that most models can accurately transfer between “Smile” and “No-Smile”. However, TransEditor [72] fails to preserve the non-target region, as shown in the 4-th row of Fig. 8. Moreover, our method outperforms all baselines in the 3D consistency for novel-view generation of edited results. For example, DiscoFaceGAN cannot maintain hair color when changing pose, and expression changes compared to the original view when zooming in on the mouth. StyleFlow and TransEditor struggle with large pose variations and suffer from severe view-inconsistency problems, resulting in significant identity changes such as beard growth on zoomed-in mouths. LiftedGAN demonstrates improved 3D consistency but exhibits limited quality and cannot perform facial attribute editing. Fig. 9 illustrates the superiority of our method in 3D consistency compared to other approaches for “Gender” and “Age” editing. For detailed comparisons, please refer to the zoomed-in hair region.

Table III presents the quantitative evaluation results of “Expression”, “Gender”, and “Age” attributes editing. Our models achieve comparable performance to baselines for all three attributes regarding FID scores. Specifically, our TT-GNeRF (S) achieves the best FID scores for the “Gender” and “Age” attributes and a comparable score to TransEditor for the “Expression” attribute. Our models are also competitive with the baselines in terms of CA. For instance, TT-GNeRF (S) scores 98.2 for the “Gender” attribute, compared to 79.9 for StyleFlow and 96.3 for TransEditor. However, both models perform worse than the previous baselines for the “Expression” attribute. Table I shows that our models outperform previous methods on both aMP and FRS metrics for all three attributes. Specifically, our TT-GNeRF (S) achieves an aMP score of 1870.4 and an FRS score of 0.829 for the “Expression” attribute, outperforming LiftedGAN, which scores 1484.0 in aMP and 0.464 in FRS, as well as DiscoFaceGAN, which scores 1347.4 in aMP and 0.587 in FRS.

Fig. 8 and Fig. 9 compare the attribute editing results between TT-GNeRF and the baselines using the same 3D-aware generator (EG3D). Fig. 8 shows the editing results for local attributes. We use the heatmap to visualize the input and editing results differences for TT-GNeRF and StyleFlow. For the attribute “Bangs”, the 1st column shows that our TT-GNeRF can preserve other non-target regions, such as the mouth region, but StyleFlow significantly changes it (Blue Box). Similar results also can be found in the 3rd column for “Bangs” results. For other results, we can observe that our model can better preserve the identity or clothes than StyleFlow. Fig. 9 shows the editing results for global attributes. We define the background and clothes as the non-target region for global attributes. For attribute “Age”, the clothes of our editing results are more similar to input than StyleFlow (1st column (Blue Box)). For attribute “Gender”, TT-GNeRF can preserve the clothes regions and other attributes (e.g., glasses (3rd column)), which validates the effectiveness of our proposed optimization stage.

For our comparative analysis between LatentSwap3D and our approach, we rigorously ensure a fair comparison by utilizing the same generator and latent code. Despite this, subtle divergences in identities between the two methods become apparent. Thus, to provide a comprehensive evaluation, we present side-by-side visual comparisons of inputs processed by our method and LatentSwap3D. Our findings reveal that LatentSwap3D suffers from very low editing intensity, par-
particularly concerning attributes like “Hair Color” (1st column (Green Box) of Fig. 8) and “Gender” (4th column of Fig. 9). In contrast, our model showcases superior performance, striking a finer balance between effective editing and faithful preservation of the input attributes.

Our method emerges as the preferred choice, demonstrating enhanced editing precision and fidelity compared to LatentSwap3D and StyleFlow. This conclusion is supported by the quantitative results in Table II. In detail, TT-GNeRF has greatly improved LP scores for all attributes while having a CA score similar to StyleFlow. Moreover, our examination of LatentSwap3D reveals a notable trend: it exhibits a notably low CA score alongside a remarkably high LP score for both the “Gender” and “Age” attributes. This observation underscores a significant deficiency in editing intensity.

C. Ablation Study

In the following, we present ablation studies on the effect of the geometry consistency loss, optimization steps in the optimization stage, and ID loss of the reference-based geometry editing.

**Geometry Consistency Loss.** Our proposed TRIOT can further improve identity preservation after face attribute editing. In the optimization stage, the objective function (Eq. 7) includes the mask-based texture consistency loss and the mask-based geometry consistency loss. We compare the full objective function with one ablation baseline, which removes the geometry losses. As shown in Fig. 11, we compare two methods by the attribute editing results. Our method excels in preserving non-target attributes for three attributes, particularly within the designated rectangular region (e.g., eyeglasses for attribute “Gender”). Notably, when examining the normals (2nd row), it becomes evident that the “without” (w/o) method fails to circumvent geometric alterations, whereas the full method adeptly sidesteps this issue. Table III reports the L1 differences between input and editing results in the non-target region, providing a quantitative assessment of the effectiveness of the proposed geometry consistency loss.

**Optimization Steps.** We investigate how the optimization stage steps impact the balance between editing accuracy and identity (or background) preservation. As depicted in Fig. 10, we notice the smiled and aged results have very high transferring intensity in the few steps setting. However, as the optimization steps increase, the aged and smiled results progressively converge towards the input. Therefore, we choose a configuration of (100 and 200) steps, representing the optimal balance between precision in attribute editing and identity preservation.

**Identity Loss (ID Loss).** We investigate the impact of identity loss in reference-based geometry editing tasks. As illustrated in Fig. 13, it is evident that employing identity loss (w/ ID loss) results in superior preservation of the input image’s identity compared to the absence of identity loss (w/o ID loss). For example, in the first row, w/o ID loss fails to maintain the gender of the input image.

D. Applications

**Reference-based Geometry Transfer.** Disentangling geometry and textures has proven challenging in prior methods. We introduce an optimization approach for reference-based geometry transfer. As depicted in Fig. 12, our method yields...
Fig. 14: GAN Inversion for real image editing and corresponding multiple-view generation. The results are obtained using TT-GNeRF (E) with the target attributes of “Bangs”, “Age” and “Expression”. For additional results, please refer to our demo video.

Fig. 15: The comparison of the editing results between StyleFlow [3], GOAE [92] and TT-GNeRF (Ours).

Fig. 16: Multiple-attribute editing results from our TT-GNeRF (E) method. E: Expression, G: Gender, H: Hair Color.

promising results in geometry transfer, accompanied by corresponding multi-view outcomes. Our method effectively integrates textures from input images into the geometry of the reference image, resulting in highly realistic outputs. **GAN Inversion for Real Image Editing.** Fig. 14 shows real image inversion and editing results for “Bangs”, “Age” and “Expression” attributes. The 2-th column shows we can produce almost perfect reconstruction for the real images. After that, our model can achieve accurate attribute editing. Fig. 15 shows the inversion and editing results of StyleFlow, GOAE, and ours. We observe that the non-target region, such as the clothes, is well preserved for our method.

We refer to the demo video for more results about multiple-view attribute editing, reference-based geometry editing, and **GAN inversion for real image editing.**

**Multiple-attribute Editing.** In addition to the previous single-attribute edits, our model can perform sequential editing of multiple attributes. Fig. 16 shows high-quality edits for the sequence “Expression + Gender + Hair color”. Multiple-view results for these edits are presented at the bottom of each row, demonstrating that our model maintains strong 3D-view consistency in these cases.

**VI. Conclusions**

In this work, we propose an attribute-conditional 3D-aware face generation and editing model, which shows the disentangling abilities of the generative neural radiance field with labels as inputs. Moreover, we integrate the training method with normalized flows and the TRIOT optimization method into the 3D-aware face editing models to achieve the best balance between attribute-editing precision and non-target attribute preservation. The qualitative and quantitative results demonstrate the superiority of our method. Additionally, we explore the disentanglement of the geometry and textures by achieving reference-based geometry transfer task with the elegant optimization method while preserving the appearance. However, there still exist some limitations. First, our model fails to edit the facial attribute in some cases. For example, the expression part of Table I shows that our CA score is worse than some methods. Second, our proposed TRIOT still costs some minutes for single attribute editing; thus, it is unacceptable for some real application scenarios. Finally, as shown in Fig. 14, our model can achieve the single image 3D model and perform attribute editing. However, compared to video-based head avatars [96], [97], the identity is not well preserved between real images and projected images. Proposing better GAN inversion techniques adapted for 3D-Aware GAN can further alleviate this problem.

**REFERENCES**

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.
J. Xiang, J. Yang, Y. Deng, and X. Tong, “Gram-hd: 3d-consistent image generation at high resolution with generative radiance manifolds,” arXiv preprint arXiv:2206.07253, 2022.

K. Jiang, S.-Y. Chen, H. Fu, and L. Gao, “Nerffacelighting: Implicit and disentangled face lighting representation leveraging generative prior in neural radius fields,” ACM Transactions on Graphics, vol. 42, no. 3, pp. 1–18, 2023.

E. Simsar, A. Tonioni, E. P. Orner, and F. Tombari, “Latentswap3d: Semantic edits on 3d image gans,” in ICCVW, 2023, pp. 2899–2909.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” CVPR, 2017.

J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in ICCV, 2017.

Z. He, W. Zhao, M. Kan, S. Shan, and X. Chen, “Atgcn: Facial attribute editing by only changing what you want,” IEEE Transactions on Image Processing, vol. 28, no. 11, pp. 5464–5478, 2019.

P.-W. Wu, Y.-J. Lin, C.-H. Chang, E. Y. Chang, and S.-W. Liao, “Relgan: Progressive attention generation at high resolution with generative radiance manifolds,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 2899–2909.

C. Z. Lin, D. B. Lindell, E. R. Chan, and G. Wetzstein, “3d gan inversion for controllable portrait image animation,” arXiv preprint arXiv:2203.13441, 2022.

R. Abdal, Y. Qin, and P. Wonka, “Image2stylegan: How to embed images into the stylegan latent space?” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 4432–4441.

J. Zhu, Y. Shen, D. Zhao, and B. Zhou, “In-domain gan inversion for real image editing,” in European conference on computer vision. Springer, 2020, pp. 592–608.

E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, “Encoding in style: a stylegan encoder for image-to-image translation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 2287–2296.

O. Toy, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, “Designing an encoder for stylegan image manipulation,” ACM Transactions on Graphics (TOG), vol. 40, no. 4, pp. 1–14, 2021.

Y. Alaluf, O. Patashnik, and D. Cohen-Or, “Restyle: A residual-based stylegan encoder via iterative refinement,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 6711–6720.

Y. Chen, Q. Wu, C. Zheng, T.-J. Cham, and J. Cai, “Sem2nerf: Converting single-view semantic masks to neural radius fields,” ECCV, 2022.

K. Jiang, S.-Y. Chen, F.-L. Liu, H. Fu, and L. Gao, “Nerfacingedit: Disentangled face editing in neural radius fields,” in SIGGRAPH Asia 2022 Conference Papers, 2022, pp. 1–9.

W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3d surface construction algorithm,” ACM siggraph computer graphics, vol. 21, no. 4, pp. 163–169, 1987.

E. Collins, R. Achanta, and S. Susstrunk, “Deep feature factorization for concept discovery,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 336–352.

K. Hornik, I. Feinerer, M. Kober, and C. Buchta, “Spherical k-means clustering,” Journal of statistical software, vol. 50, pp. 1–22, 2012.

J. Deng, J. Guo, N. Xue, and S. Zaefiriou, “Arcface: Additive angular margin loss for deep face recognition,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 4690–4699.

D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” ICLR, 2014.

X. Pan, A. Tewari, T. Leimkuhler, L. Liu, A. Meka, and C. Theobalt, “Drag your gan: Interactive point-based manipulation on the generative image manifold,” in ACM SIGGRAPH 2023 Conference Proceedings, 2023.

Z. Yuan, Y. Zhu, Y. Li, H. Liu, and C. Yuan, “Make encoder great again in 3d gan inversion through geometry and occlusion-aware encoding,” ICCV, 2023.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium,” in NeurIPS, 2017.

J. Zhang, E. Sangineto, H. Tang, A. Siarohin, Z. Zhong, N. Sebe, and W. Wang, “3d-aware semantic-guided generative model for human synthesis,” arXiv preprint arXiv:2112.01422, 2021.

Q. Zhou, T. Sattler, and L. Leal-Taixe, “Patch2pix: Epipolar-guided pixel-level correspondences,” in CVPR, 2021.