UNIVERSITÀ DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE
ICT International Doctoral School

Exploring 2D and 3D Human Generation and Editing

Jichao Zhang

Advisor

Prof. Dr. Nicu Sebe

Università degli Studi di Trento

December 2023

Abstract

In modern society, cameras on intelligent devices can generate a huge amount of natural images, including images of the human body and face. Therefore, there is a huge social demand for more efficient editing of images to meet human production and life needs, including entertainment, such as image beauty. In recent years, Generative Models with Deep Learning techniques have attracted lots of attention in the Artificial Intelligence field, and some powerful methods, such as Variational Autoencoder and Generative Adversarial Networks, can generate very high-resolution and realistic images, especially for facial images, human body image. In this thesis, we follow the powerful generative model to achieve image generation and editing tasks, and we focus on human image generation and editing tasks, including local eye and face generation and editing, global human body generation, and editing. We introduce different methods to improve previous baselines based on different human regions. 1) Eye region of human image: Gaze correction and redirection aim to manipulate the eye gaze to a desired direction. Previous common gaze correction methods require annotating training data with precise gaze and head pose information. To address this issue, we proposed the new datasets as training data and formulated the gaze correction task as a generative inpainting problem, addressed using two new modules. 2) Face region of human image: Based on a powerful generative model for face region, many papers have learned to control the latent space to manipulate face attributes. However, they need more pre-

cise controls on 3d factors such as camera pose because they tend to ignore the underlying 3D scene rendering process. Thus, we take the pre-trained 3D-Aware generative model as the backbone and learn to manipulate the latent space using the attribute labels as conditional information to achieve the 3D-Aware face generation and editing task. 3) Human Body region of human image: 3D-Aware generative models have been shown to produce realistic images representing rigid/semi-rigid objects, such as facial regions. However, they usually struggle to generate high-quality images representing non-rigid objects, such as the human body, which greatly interests many computer graphics applications. Thus, we introduce semantic segmentation into the model. We split the entire generation pipeline into two stages and use intermediate segmentation masks to bridge these two stages. Furthermore, our model can control pose, semantic, and appearance codes by using multiple latent codes to achieve human image editing.

Keywords

Generative Model, Generative Adversarial Networks, Generative Neural Radiance Field, Face Generation and Editing, Generative Digital Human.

Contents

List of Tables

List of Figures

Publications

This thesis consists of the following publications:

- Chapter 2:
  - **Jichao Zhang**, Jingjing Chen, Hao Tang, Wei Wang, Yan Yan, Enver Sangineto, Nicu Sebe. Dual In-Painting Model for Unsupervised Gaze Correction and Animation in the Wild. ACM International Conference on Multimedia, 1588-1596, 2020.
  - **Jichao Zhang**, Jingjing Chen, Hao Tang, Enver Sangineto, Peng Wu, Yan Yan, Nicu Sebe, Wei Wang. Unsupervised high-resolution portrait gaze correction and animation. IEEE Transactions on Image Processing 31(7):5272-5286, July 2022.

- Chapter 3:
  - **Jichao Zhang**, Aliaksandr Siarohin, Yahui Liu, Hao Tang, Nicu Sebe, Wei Wang. Training and Tuning Generative Neural Radiance Fields for Attribute-Conditional 3D-Aware Face Generation. IEEE Transactions on Pattern Analysis and Machine Intelligence, in peer-review.

- Chapter 4:
  - **Jichao Zhang**, Enver Sangineto, Hao Tang, Aliaksandr Siarohin, Zhun Zhong, Nicu Sebe, Wei Wang. 3D-Aware Semantic-Guided Generative Model for Human Synthesis. European Conference on Computer Vision, 339-356, 2022.

The following papers are published during the course of the Ph.D but not included in this thesis:

1. Jingjing Chen, **Jichao Zhang**, Enver Sangineto, Tao Chen, Jiayuan Fan, Nicu Sebe. Coarse-to-Fine Gaze Redirection with Numerical and

Pictorial Guidance. IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 3665-3674), 2021.

2. Yue Song*, **Jichao Zhang***, Nicu Sebe, Wei Wang. Householder Projector for Unsupervised Latent Semantics Discovery. International Conference on Computer Vision, 7712-7722, 2023.

Chapter 1

Introduction

1.1 Deep Generative Models

Recently, the critical technology in Artificial Intelligence, Deep Learning, powers many aspects of modern society. In industry, many consumer products, such as smartphones and web services, have widely been used to provide better intelligent services. At the same time, in the academic community, Deep Learning has become the hottest topic and research content in the entire artificial intelligence field and significantly promoted the development of some fields, including computer vision and natural language processing. Deep Learning can be regarded as one type of Machine Learning and one type of data-driven non-linear learning technology. Compared with conventional machine-learning methods with careful feature engineering, Deep Learning with multiple-layers neural networks can directly learn feature representation from raw data and discover intricate structures in high-dimensional data that can better serve downstream applications, such as detection, classification, and recognition. The mainstream neural networks in Deep Learning are Convolutional Neural networks and Recurrent Neural Networks, widely used for computer vision and natural language processing, respectively, and feed on images and text. In this thesis, we focus on image generation and editing, and the prominent architecture in our model is multiple-layers convolutional neural networks.

Deep learning can improve photo generation and semantic editing by introducing models, methods, and techniques that address the challenges. In modern society, intelligent devices, human production, living, and entertainment needs make it possible to obtain a large number of natural and digital pictures through intelligent devices. At the same time, the large-scale data also makes it possible to train the deep model, thus obtaining a powerful image generative model and a more automatic image editing model. Most of the previous deep learning methods [47, 88, 95, 28, 27, 5, 50, 70, 228, 140, 225, 15, 57, 105] for image generation and editing are based on Deep Generative Model which are neural networks with many hidden layers trained to approximate complicated, high-dimensional probability distributions using a large number of samples. We give a detailed introduction to generative models below.

Generative models learn to represent and estimate the distribution of training data p_{data}. The estimated distribution p_{model} is trained to be close to p_{data}. So, why use generative models? There are several reasons for it, including:

1) High-dimension probability distributions are important objects in various applied math and engineering domains. A generative model could provide some methods to represent and sample high-dimension probability distributions.

2) Generative model can be used for learning conditional distribution, which can be applied to multiple image-to-image tasks, such as style transfer, general image editing, face aging, and gaze correction. These tasks have a wide of applications in AI products.

The widely used generative models are Variational AutoEncoder [88], Generative Adversarial Networks [47], Autogressive Models [186], Normalizing Flows [37], and Diffusion Models [39] and lots of variants of these models.

1) Variational AutoEncoder (VAE) [88] belongs to the families of Variational Bayesian methods which are the techniques for approximating intractable integrals arising in Bayesian inference and provide an analytical approximation to the posterior probability of the unobserved variables. In detail, VAE learns stochastic mappings between an observed x space with complicated distribution p_x and a latent-space z with simple distribution, such as Gaussian distribution. Furthermore, it consists of two coupled but independent models: the encoder and the decoder. The encoder $q_\theta(z|x)$ is an inference model, approximates the intractable posterior $p_\theta(z|x)$, and the decoder is a generative model $p_\theta(x|z)$. With the prior distribution $p_\theta(z)$, the final joint distribution is $p_\theta(x, z) = p_\theta(z)p_\theta(x|z)$. Variational Autoencoder is easy to train but tends to produce blurry results and optimizes a lower bound on the log-likelihood of the data.

2) Autoregressive generative model [186] is a type of generative model that learns the joint probability distribution of a sequence of variables by decomposing it into a product of conditional probabilities. Given n variables x_1,...x_n, the joint probability distribution can be expressed as:

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i|x_1, x_2, ..., x_{i-1}). \tag{1.1}$$

The training of autoregressive models is to maximize the likelihood of the training data directly, and their network is used to model the conditional probability by minimizing the negative log-likelihood. Autoregressive generative models are flexible, scalable, and interpretable, making them useful for various applications. They can generate high-quality samples that are often indistinguishable from real data. Compared with other generative models, autoregressive models generate samples sequentially, which can be slow and computationally expensive for long sequences.

3) Normalizing Flows [37]: The basic building block of a normalizing flow is an invertible transformation that maps a simple probability distribution (*e.g.*, standard normal) $z \sim p_\theta(z)$ to a more complex distribution $x \sim p_\theta(x)$. The mapping function f from random variable z to x: $x = f(z)$ and f must be invertible. With the chain rule, we have:

$$p(x) = p(z) \mid det \frac{df^{-1}}{dx} \mid . \tag{1.2}$$

This chain is known as a normalizing flow. Given $f_1, ..., f_n$ be a set of N bijective function, and let define $f = f_1 \circ f_2, ..., \circ f_n$, we have $x_i = f_i \circ, ..., f_2 \circ f_1(x_0)$. Then, it can be shown that f is also bijective and invertible. Though a chain of mapping $f_i, p(x_i)$ can be represented as:

$$\ln p_i(x_i) = \ln p_0(x_0) - \sum_{i=1}^{n} \ln \mid det \frac{df_i}{dx_{i-1}} \mid . \tag{1.3}$$

The main advantage of normalizing flows is interpretability because the probability density of generated samples can be directly calculated, providing an interpretable way to understand the behavior of the generative model.

4) Diffusion Models [63]: Recently, Diffusion models are the most popular generative model, which learns to gradually convert samples from

a simple distribution into a data distribution. Three predominant formulations for diffusion model research are denoising diffusion probabilistic models (DDPM) [63], score-based generative models, and stochastic differential equations, respectively. We take DDPM as an example to introduce details about diffusion models. DDPM consist of two Markov chains.

The forward chain q slowly removes structure from data $x_0 \sim q(x)$ by adding noise z to convert the data distribution into Gaussian distribution using timesteps t. We can generate a sequence of variables $x_1, x_2, ..., x_n$. And the transition $q(x_t|x_{t-1})$ is defined as Gaussian perturbation in general:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}), \tag{1.4}$$

where $\beta_t \in (0, 1)$ is a hyperparameter defined before model training.

The reverse chain p learns to convert the simple distribution into data distribution. The reverse transition $p(x_{t-1}|x_t)$ is learned and parameterized by deep neural networks. Similar to forward transition, the reverse transition takes the form of:

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_\theta(x_t, t), \nu_\theta(x_t, t)), \tag{1.5}$$

where $\mu_\theta(x_t, t), \nu_\theta(x_t, t)$ are parameterized by deep neural networks.

Following [63], the training diffusion model simplifies to a weighted denoising score matching objective for parameters θ:

$$\mathcal{L}(\theta, x_0) = \mathbb{E}_{t\sim U(0,T), \epsilon\sim\mathcal{N}(0,\mathbf{I})}\left[\lambda_t \left\|\epsilon_\theta(x_t, t) - \epsilon\right\|\right], \tag{1.6}$$

where λ_t is a position weighting function. x_t is computed from x_0 and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.

5) Generative Adversarial Networks (GANs) [47] is a powerful implicit generative model to produce a model distribution that mimics a given target distribution. GANs generally consist of two networks, generator G and discriminator D. The generator G learns the mapping from a low-dimension (noise) distribution p_z to a generated high-dimension distribution p_g and meanwhile makes p_g as close to the real sample distribution p_r. The discriminator D is trained to distinguish the real samples from the samples produced by G. Both G and D are parameterized via neural networks, and their training process can be considered a min-max game. By iteratively training G and D using gradient descent techniques, it is expected that GANs can find a Nash equilibrium of this game [154], whereby the generated distribution p_g is equal to p_r, such that the discriminator D fails to differentiate between real and generated samples, *i.e.* $D(x) = 0.5$, $\forall x$. In recent years, many GANs variants have been proposed, such as StyleGAN [80], and these models have achieved high-resolution image modeling and high-quality image generation. *Compared to Variational AutoEncoders and Autoregressive Models, Generative Adversarial Networks (GANs) are capable of generating higher quality samples. Additionally, GANs have the advantage of faster inference speed than Diffusion Models. Normalizing Flows, on the other hand, can model complex distributions and be used for density estimation and generation tasks. However, they are computationally expensive and may not be suitable for large-scale datasets.*

We have reviewed multiple types of generative models. In this thesis, all architectures and networks are based on generative adversarial networks. All training objective functions in our models include adversarial loss, which is usually used to improve the clarity and quality of generated images.

1.2 Contributions and Outlines

In this thesis, we adopt a powerful generative model to achieve image generation and editing tasks, focusing on human image generation and editing, including local eye and face generation and editing, as well as global human body generation and editing. Previous global or local human image editing methods can be categorized into the following types: 1) Image-to-image translation based on Autoencoder; 2) Disentangling the latent space of generative models; and 3) NeRF-based 3D-Aware methods.

The first type [149, 118, 27, 91] is based on the previous image-to-image translation method, which map input human images or parts into target images using variants of the Autoencoder. To achieve human image editing, some methods [118, 121] utilize additional information as guidance, such as segmentation maps or key points. These methods generally regard input images as the source domain and output images as the target domain. The essence of this type of method is to use neural networks to learn domain mapping.

The second type is based on pre-trained generative models, such as variants of StyleGAN [81]. Previous methods [5, 162, 189] learn to disentangle the latent space of GANs using unsupervised or supervised methods. After learning, they can control the latent code to achieve human image editing tasks. The main advantage of this type is that it is free from training the generative model from scratch, thus significantly improving training efficiency. Furthermore, the pre-trained model can generate highly realistic images, ensuring the quality of generated images. This type focuses on disentangling the latent space to achieve image editing.

The third type [73, 209, 64] aims to achieve 3D-Aware human image generation and editing tasks. Most models are based on generative neural radiance fields, a NeRF variant. Previous 2D methods lack precise control over 3D factors such as camera pose. To solve this problem, some methods combine 3D scene representation (*e.g.*, NeRF) with a generative model. However, they focus on the multiple-view generation and lack precise semantic control for the human body or face.

In this thesis, we propose three methods to deal with the problems for generation and editing tasks in eye gaze, face, and human body images. Our three methods, although dealing with different parts of the human body, share commonalities in that they all have a latent space decoupling component and aim to achieve human image editing tasks more ***Efficiently*** or ***Unsupervised***. Our first method, GazeGAN, aims to decouple the latent gaze space without using an accurate gaze label and then control it to achieve unsupervised gaze correction and redirection tasks. Our second method, TT-GNeRF, explores a 3D-Aware face editing task and uses supervised information to decouple the latent space but avoids training from scratch by utilizing a trained generative model, significantly improving training efficiency. The third method, 3D-SGAN, is a 3D-aware human body generation and editing method that also adds multiple latent codes to control different semantics, and the disentangling of these latent codes also utilizes unsupervised ideas.

The contribution of our three models can be summarized into:

1) GazeGAN is the first paper to achieve unsupervised gaze correction and animation. The model's novelty lies in treating this task as a self-supervised image inpainting problem, which results in highly realistic gaze correction outcomes. Furthermore, we have proposed the CelebAGaze and CelebAHQGaze datasets, which have been publicly available for research. GazeGAN will be introduced in Chapter 2.

2) TT-GNeRF is the first paper to explore the disentanglement of latent space of 3D-Aware GAN to achieve view-consistency face editing

task. TTGNeRF needs the attribute labels as supervision. Compared with 2D methods, TT-GNeRF can achieve better view-consistency facial attribute editing. Moreover, compared with some disentangled methods for the 3D-aware model, TT-GNeRF can achieve a better trade-off between attribute editing and non-target region preservation. The main contribution lies in the proposed two-stage learning, which combines training and optimization methods. The first stage aims to achieve attribute editing as the initial result, and the second stage aims to preserve the non-target region of the editing results from the first stage by optimizing the latent code of GAN space. TT-GNeRF will be introduced in Chapter 3.

3) 3D-SGAN is the first paper to apply generative neural radiance fields for 3D-aware human image generation. The main contribution lies in the proposed two-stage architecture and training. We divide the 3D-aware generative model into two stages and use intermediate segmentation masks to bridge these two stages. In the first stage, we train a generative neural radiance field using segmentation maps of human images and learn a 3D-aware semantic generative model for the geometry representation. In the second stage, we pre-train a Variational Autoencoder to learn the mapping from segmentation maps into human images. With two trained models, we construct a 3D-aware controllable generative model for human images guided by segmentation maps. 3D-SGAN will be introduced in Chapter 4.

Chapter 2

Exploring Unsupervised Gaze Correction and Animation

2.1 Introduction

The goal of the gaze correction task is to manipulate the gaze direction of a face image with respect to a specific target direction. The main application of this task is altering the eye appearance so that the person's gaze is directed into the camera. For example, shooting a good portrait is challenging as the subjects may be too nervous to stare at the camera. Another scenario is videoconferencing, where eye contact is very important. The gaze can express attributes such as attentiveness and confidence. Unfortunately, eye contact is frequently lost during a video conference, as the participants look at the monitors and not directly into the camera. Moreover, some works use gaze redirection to improve few-shot gaze estimation task [206, 207].

Early works in gaze correction relied on special hardware, such as stereo cameras [32, 202], Kinect sensors [93] or transparent mirrors [90, 133]. Recently, a few methods based on machine learning showed a good quality synthetic for gaze correction. For instance, Kononenko and Lempitsky [92] propose to solve the problem of monocular gaze correction using decision

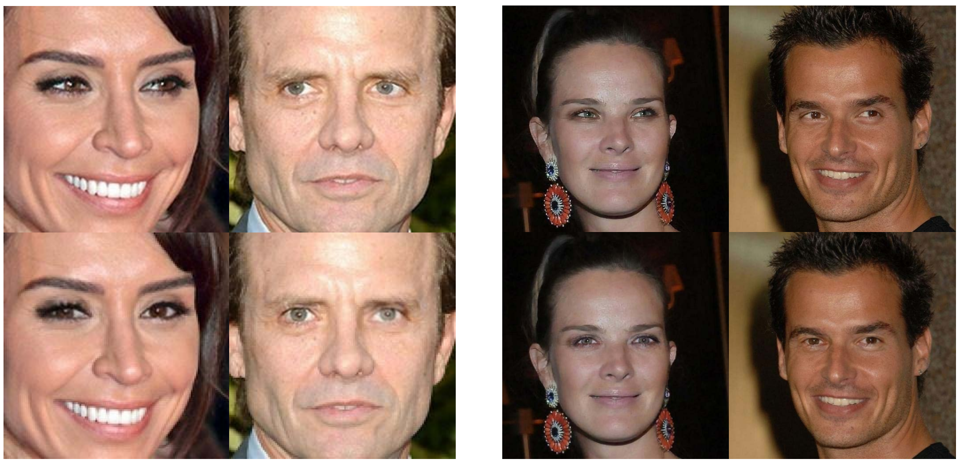

Figure 2.1: Left: 256×256 images and the corresponding gaze-corrected results generated by our method using samples of the CelebGaze dataset. Right: 512×512 high-resolution images and the gaze-corrected results using samples of our dataset CelebHQGaze. The first and second rows show the original images and eye-gaze corrected results, respectively.

forests. DeepWarp [43] uses a deep network to directly predict an image-warping flow field with a coarse-to-fine learning process. However, this method fails in generating photo-realistic images when the gaze redirection involves large angles. Moreover, it produces unnatural eye shapes because of the $L1$ loss, which is used to learn the flow field without any geometric-based regularization. To solve this problem, PRGAN [54] proposes to exploit adversarial learning with a cycle-consistent loss to generate more plausible gaze redirection results. However, these methods [92, 43, 54] fail in obtaining high-quality gaze redirection results in the wild when there are large variations in the head pose and the gaze angles. Recently, Marcel *et al.* [17] proposed a content-consistent model for realistic eye generation. However, their approach is based on semantic segmentation masks, which implies a great annotation effort. Another category of works is based on 3D models without training data, such as GazeDirector [194]. The main idea of GazeDirector is to model the eye region in 3D instead of predicting a flow field directly from an input image. However, modeling in 3D has strong assumptions that do not hold in non-laboratory scenarios.

The unsupervised method can avoid expensive annotations. Moreover, it has essential significance for image representation and semantic disentanglement. Thus, we proposed a novel gaze-correction method, GazeGAN. We collected the CelebGaze dataset, which consists of two image domains: X, with eyes staring at the camera, and Y, with eyes looking somewhere else (see Fig. 2.1, *left*). Note that the CelebGaze images do not annotate the gaze angle or the head pose. Moreover, we propose an unsupervised learning method for gaze correction and animation, which consists of two main modules: the Gaze Correction Module (GCM) and the Gaze Animation Module (GAM). GCM is an inpainting model, trained on a domain X, which learns how to fill in the missing eye regions with a new content representing the gaze-corrected eyes. GAM is another inpainting model used for gaze animation, and it is trained on a domain Y. To generalize the gaze redirection to various angle directions (*i.e.*, in "animations"), we propose a training method (Synthesis-As-Training) that uses synthetic data to train GAM and encourages the encoded features of the eye region to be correlated with the gaze angle. Then, gaze animation can be achieved by interpolating these features in the latent space.

We extend GazeGAN to work also with higher resolution portrait images. Specifically, we first create a new dataset, CelebHQGaze, containing 512×512 high-resolution portrait images, as shown in Fig. 2.1 (*right*). Second, we propose a novel GCM and GAM integrated with a coarse-to-fine module (CFM). In more detail, CFM first allows the inpainting model to be trained using low-resolution images for coarse-grained image generation. Then it uses a global nonparametric model, Laplacian Reconstruction, and a local parametric model, Local-Refinement Autoencoder, to compensate for the high-frequency information loss and to remove possible artifacts for the eye region. Utilizing this new architecture, we can avoid training each

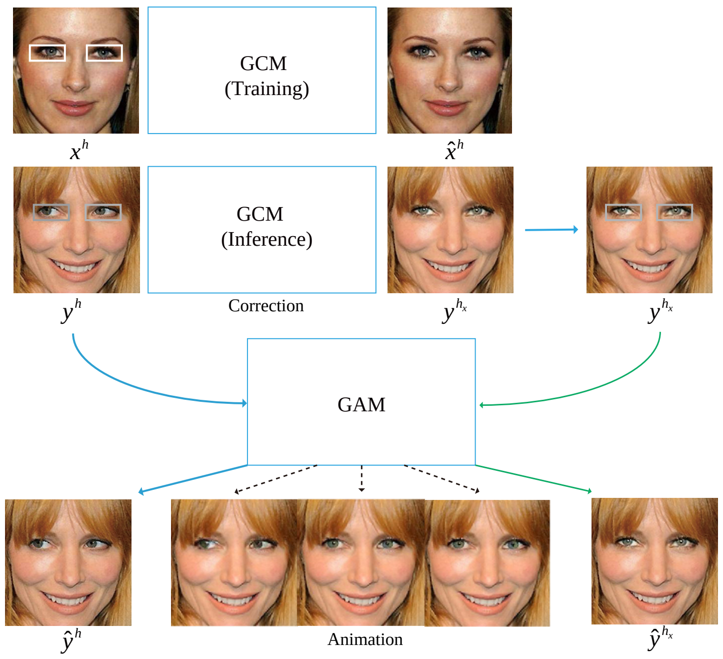

Figure 2.2: Overview of the proposed architecture. We have two main modules: Gaze Correction Module for performing gaze correction (GCM) and Gaze Animation Module for performing gaze animation (GAM). Moreover, we propose to use the gaze-corrected samples from GCM to train GAM (Synthesis-as-Training). The trained GAM can achieve gaze animation by interpolating the latent feature. The white boxes are the eye mask to remove the eye region. The gray boxes represent the cropping of eye region

module using high-resolution images. CFM speeds up both the training and the inference process while obtaining high-quality results, comparable with directly training with high-resolution images.

In our architecture, an autoencoder is pretrained using self-supervised mirror learning (PAM), where the bottleneck features are used as an extra input to the dual inpainting model to preserve the identity of the corrected results. Moreover, global and local discriminators are used to improve the visual quality of the generated samples. Finally, our qualitative and quantitative evaluations show that our method generates higher-quality results with respect to the state-of-the-arts in both the gaze correction and the gaze animation tasks.

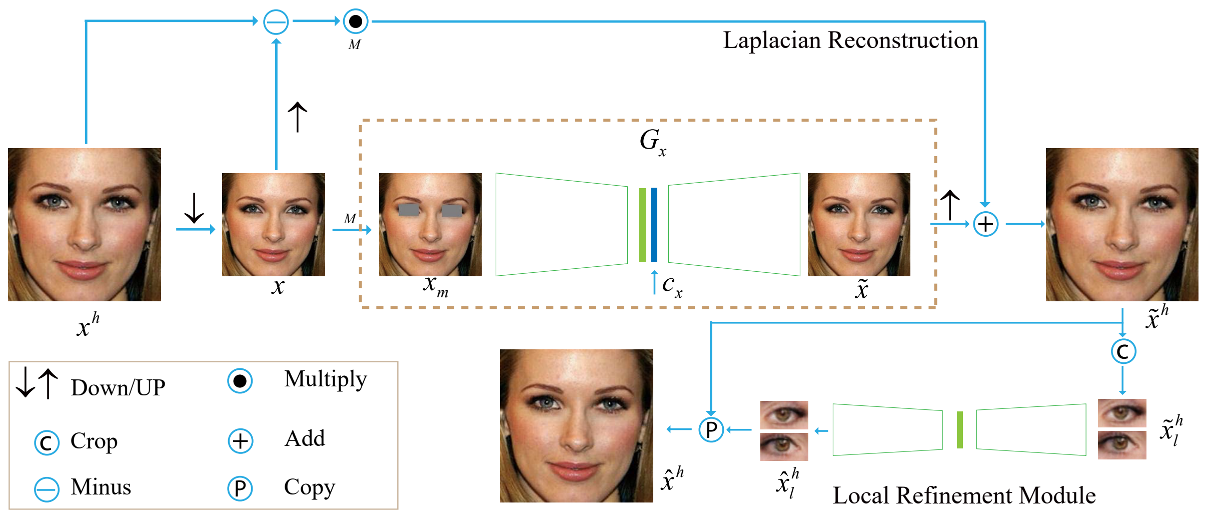

Figure 2.3: Overview of the architecture for Gaze Correction Module (GCM) integrated with Coarse-to-Fine Module (CFM) which consists of one laplacian reconstruction and one local-refinement module. CFM first allows the inpainting network G_x trained using low-resolution images to attain coarse-grained inpainted results, then attains high-resolution results by the global nonparametric Laplacian reconstruction, and finally exploits a parametric local-refinement module (LRM) to compensate for high-frequency information and remove artifacts for the eye region. We use 2× scales for downsampling and upsampling.

We summarize below our main contributions:

1) We introduce an unsupervised inpainting architecture for high-resolution gaze correction and animation.

2) We propose a novel CFM module that can alleviate both the memory and the computational costs in the training and the inference stage while achieving high-quality results comparable with training with high-resolution facial images.

3) We propose a gaze animation module and a Synthesis-As-Training method to generate gaze-correction results with variable angles.

4) We make publicly available the CelebGaze and CelebHQGaze dataset for the research community interested in gaze correction and animation: `https://github.com/zhangqianhui/GazeAnimationV2`.

2.2 Background

Generative Adversarial Networks. As mentioned above, Generative Adversarial Networks (GANs) [47] are powerful generative models which learn a distribution that mimics a given target distribution. They have been applied to many fields, such as low-level image processing tasks (*e.g.*, image inpainting [143, 68], image super-resolution [98, 100, 191]), semantic and style transfer (*e.g.*, image translation [71, 183, 226, 106, 182, 139, 120], image attribute manipulation [211, 57, 56, 105, 22, 29], person image synthesis [180, 181, 167, 210, 212], image manipulation [141]).

Image Inpainting. Image Inpainting is an important task in computer vision and computer graphics, and it aims to fill in the missed/masked pixels of an image utilizing plausible synthesized content. Most of the previous methods can be split into two classes. The first is based on diffusion or patch-based approaches, which rely on handcrafted low-level features. For example, PatchMatch [12] is a fast nearest-neighbor field algorithm, which can perform real-time image inpainting. Generally speaking, this class of methods is based on low-level features. They are usually ineffective in filling in the missing part of an image when the underlying semantic structure is not trivial and cannot generate novel objects that cannot be found in other non-masked parts of the source image. The second class of methods is based on learning approaches. Recently, CNN-based and GAN-based methods have shown promising performance on image inpainting [142, 69, 104, 208]. For instance, inpainting can be used for facial attributes manipulation such as hair, mouth, and eyes [75, 38, 134]. We also adopt an inpainting approach, differently from previous work, our method does not require the data to be labeled with additional information, such as semantic labels, sketches, or reference images.

Gaze Correction. Previous work for gaze correction can be split into three main classes: 1) hardware-driven, 2) rendering and synthesis, 3) learning-based.

The hardware support was indispensable in early research. Kollarits *et al.* [90] use half-silvered mirrors to place the camera on the optical path of the display. Yang *et al.* [202] address the eye contact problem with a view synthesis, and they use a pair of calibrated stereo cameras jointly with a face model to track the head pose in 3D. Generally speaking, these hardware-based methods are expensive.

The second class of approaches typically renders the eye region based on a 3D fitting model, which replaces the original eyes with synthetic eyeballs. Banf *et al.* [11] use an example-based approach for deforming the eyelids and sliding the iris across the model surface with a texture-coordinate interpolation. To fix the limitations caused by the use of a mesh, where the face and eyes are mixed, GazeDirector [194] separately deals with the face and eyeballs, synthesizing more high-quality images, especially for large redirection angles. These methods usually struggle in realistically rendering the corrected eyes. Additionally, modeling methods have strong assumptions that usually do not hold in practice.

Concerning the third class of methods, the core idea for most of the learning-based approaches is to use a large paired training dataset to train a statistical model [92, 91, 206, 138, 21]. Some methods [92, 91] learn to generate the flow field, which is then used to relocate the eye pixels in the original image. For instance, Ganin *et al.* [43] use a CNN to learn the flow field, which warps the input image and redirects the gaze to the target angle. However, [43] fails to generate photo-realistic and natural shapes because it uses only pixel-wise differences between the input and the ground truth as the training loss. To address this problem, He *et al.* [54] use adversarial learning, jointly with a cycle-consistent loss, which can improve the visual quality and the redirection precision. However, these methods can hardly generate plausible results in the wild, *i.e.*, in

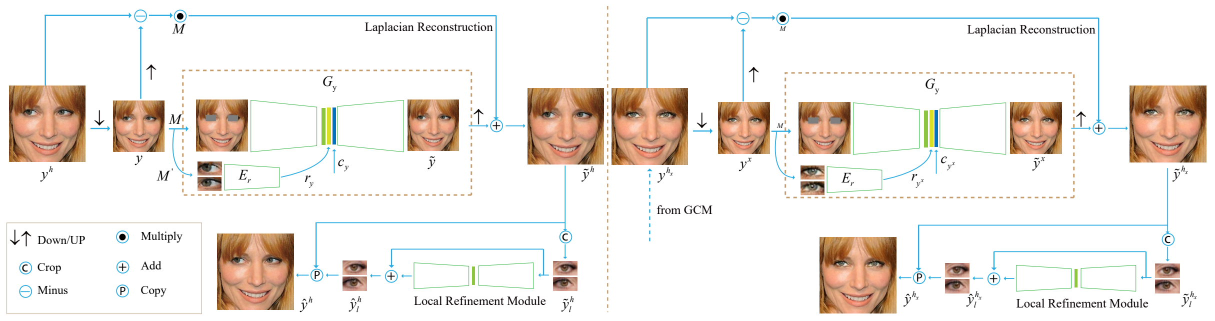

Figure 2.4: An overview of the proposed Gaze Animation Module (GAM) integrated with Coarse-to-Fine Module (CFM). In the left, G_y uses the sample $y \in Y$ for training. Compared to G_x, the decoder of G_y has an extra input r which is provided by the encoder E_r. In the right, we use GCM to generate the gaze-corrected image y^{h_x}, which is then used for training G_y (Synthesis-as-Training). With the paired samples y^h and y^{h_x} for training G_y, the feature from E_r would be correlated with gaze angle, and gaze animation can be achieved by interpolating the feature.

a scenario with large variations in the head pose, the gaze angle, or the illumination conditions. In contrast, we propose to use dual inpainting modules (GCM and GAM) to correct the gaze angle and achieve high-resolution and high-quality gaze redirection in the wild.

2.3 GazeGAN

The overview of our method is shown in Fig. 2.2 and our model consists of two main modules: Gaze Correction Module and Gaze Animation Module. Specifically, Fig. 2.3 illustrates Gaze Correction Module (GCM), integrating with Coarse-to-Fine Module (CFM), which is trained using the sample x from domain X. Fig. 2.4 illustrates Gaze Animation module (GAM), integrating with CFM, which are trained using the sample y from domain Y, and GAM exploits the corrected samples for training to make the eye feature correlate with the gaze angle (Synthesis-as-Training method). Additionally, Fig. 2.5 shows the pretrained autoencoder (PAM), which extracts the angle-invariant content feature as the additional input of GCM and GAM. We here clarify the adopted notations.

• $x \in R^{m \times n \times 3}$ is an image instance, where m and n are the image height and width, and 3 is the number of RGB channels.

• The training set is split into two domains: X, containing images with a gaze staring at the camera, and Y, containing images with a gaze staring somewhere else. X^h and Y^h correspond to the higher-resolution sets of X and Y, respectively.

• $M \in R^{m \times n \times 3}$ denotes a binary mask function of the eye region and $M^{'}$ defines the operation of extracting a rectangular sub-image (the eye region).

• P_x and P_y denote the data distributions in X and Y, respectively. P_m indicates the distribution of the masked data $M(x)$, where the eye region is removed from x. If $x \in X$ and $y \in Y$, both $M(x)$ and $M(y)$ have the same distribution, because the only difference between x and y is in the eye region. Thus, $M(x) \sim P_m$ and $M(y) \sim P_m$.

• $r \in R^{128}$ and $c \in R^{256}$ denote the angle, the content features (being the latter angle invariant), respectively and different encoders extract them.

• F denotes the image horizontal flipping operation (mirroring).

We first introduce the details of our coarse-to-fine module (CFM).

2.3.1 Coarse-to-Fine Module

In order to alleviate the memory costs and reduce the number of overall training parameters while simultaneously being able to generate high-resolution facial images, we propose a CFM for GCM and GAM. This module consists of a global nonparametric Laplacian Reconstruction for the inpainting process and a local parametric Local Refinement Module (LRM) which will be introduced with details, taking GCM as an example.

Global Nonparametric Laplacian Reconstruction

As shown in Fig. 2.3, the high-resolution image x^h is downsampled by a factor of 2, obtaining x, where the latter is as input to the inpainting networks of GCM. The generated image is $\tilde{x}$. Let $u(.)$ be an upsampling operator which smooths and expands x to the original size (*i.e.*, the resolution of x^h). The single-level Laplacian pyramid p can be obtained by:

$$p = x^h - u(x). \tag{2.1}$$

Then, the reconstruction process for the high-resolution image $\tilde{x}^h$ is:

$$\tilde{x}^h = u(\tilde{x}) + M(p), \tag{2.2}$$

where we use M to remove the eye region from p which is replaced by the zero. Then we introduce the local refinement process to improve the visual quality and remove the artifacts of the eye regions.

Local Parametric LRM

We use $M^{'}(\tilde{x}^h)$ to extract the local eye region $\tilde{x}^h_l$. Then, we utilize one autoencoder G_h together with residual image learning, to refine $\tilde{x}^h_l$ and get $\hat{x}^h_l$. Finally, the high-resolution complete image $\hat{x}^h$ can be obtained by replacing the local eye region $\tilde{x}^h_l$ with $\hat{x}^h_l$.

2.3.2 Gaze Correction Module

As shown in Fig. 2.3, we first downsample x_h to attain the low-resolution x, and then take x as the input of inpainting network G_x whose goal is to fill in the masked eye region of $x_m = M(x)$ by generating the missing eyes. This can be formulated as:

$$c_x = E_c(M^{'}(x)), \tilde{x} = G_x(M(x), c_x), \tag{2.3}$$

where c_x are the content (angle-invariant) features encoded using only the eye regions as input $(M^{'}(x))$ of the content encoder E_c. E_c is the encoder of G_{pre} which will be introduced in Sec. 2.3.4.

In principle, G_x can learn the mapping from $M(x) \sim P_m$ to $x \sim P_x$ by training. Given one sample $y \sim P_y$, then, we remove the eye region to get $M(y) \sim P_m$, because x and y have different distributions only in the eye region. Thus G_x can be used to map $M(y)$ into the $G_x(M(y)) \sim P_x$ which is the intuitive basis of our correction module. This can be formulated as:

$$c_y = E_c(M^{'}(y)), y^x = G_x(M(y), c_y), \tag{2.4}$$

where c_y are the content (angle-invariant) features encoded using only the eye regions as input $M^{'}(y)$ of the content encoder E_c.

We train the GCM using high-resolution face images based on the CFM module. At the inference time, for the corrected result y^x, we get a high-resolution result y^{h_x} by compensating for the high-frequency details using the laplacian reconstruction and LRM. Note that the corrected result y^{h_x} is also used for training GAM. More details can be found below.

2.3.3 Gaze Animation Module

Besides correcting the gaze to stare at the camera, a more general task is gaze animation, where the gaze direction should be modified. As shown in Fig. 2.4, another generator G_y is used to in-paint the face image without the eye region by performing the reconstruction learning. Moreover, we extend a new eye encoder E_r for the eye region to extract the angle-specific feature, guiding the gaze redirection generation of the G_y. To achieve the disentanglement of the features, we propose a Synthesis-As-Training method, in which we use the gaze-corrected generated images as training data for training GAM. In detail, our GAM is split into two stages. In the first stage (Left of Fig. 2.4), we downsample y^h to get y, and train the

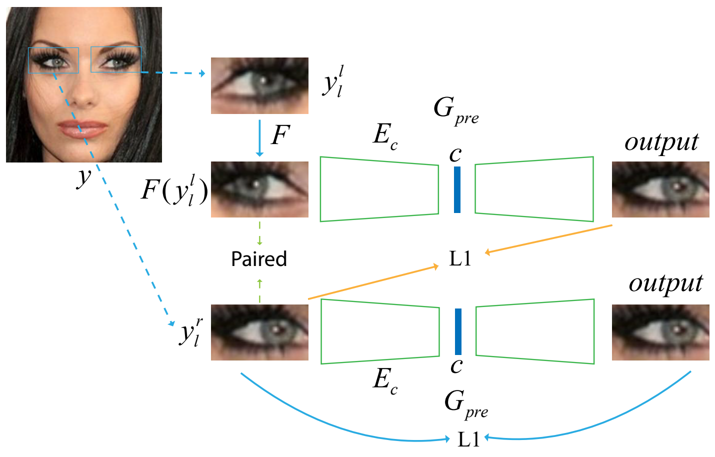

Figure 2.5: The overview of pretrained autoencoder module with generator G_{pre} (PAM). PAM is trained by a self-supervised learning strategy. In detail, we crop y to both left eye y_l^l and right eye y_l^r, then, flip y_l^l by F to attain the pairs $F(y_l^l)$ and y_l^r which have similar identity, but different gaze angle. Then, the pairs would be used to train G_{pre} by reconstructing y_l^r.

generator G_y to fill-in the missing eye regions of an image ($y_m = M(y)$) and produce $\tilde{y}$. G_y encodes the eye region with the latent code $r_y \in R^{128}$ by means of the encoder E_r. Moreover, r_y is used as an extra input for the decoder in G_y. In this way, we can condition G_y using the gaze-dependent feature r_y.

$$\begin{aligned} r_y = E_r(M^{'}(y)), c_y = E_c(M^{'}(y)) \\ \tilde{y} = G_y(M(y), r_y, c_y). \end{aligned} \tag{2.5}$$

In the second stage (Right of Fig. 2.4), we use GCM to correct the gaze of y^h, and it produces the synthetic sample y^{h_x}. Then, y^{h_x} is downsampled to y^x which is used for training G_y, just like y does. With the paired samples (y, y^x), which have the same masked region $M(y)$ but different eye regions, we train GAM to ensure that the encoded feature from E_r has a high correlation with the gaze angle:

$$\begin{aligned} r_{y^x} = E_r(M^{'}(y^x)), c_{y^x} = E_c(M^{'}(y^x)) \\ \tilde{y}^x = G_y(M(y^x), r_{y^x}, c_{y^x}). \end{aligned} \tag{2.6}$$

Then, we can attain high-resolution results $\hat{y}^h$ and $\hat{y}^{h_x}$ by compensating for the high-frequency details using the laplacian reconstruction and LRM.

2.3.4 Pretraining using Self-Supervised Learning

Preserving the consistency of the person's identity (*e.g.*, the iris color, the eye shape) is difficult with the inpainting-based method described. To mitigate this problem, we propose to use the third generator G_{pre}, which is trained (PAM) to learn a latent representation of the content features (c), conditioning both G_x and G_y to preserve the identity information of the generated results consistent with the input.

G_{pre} is pre-trained using a self-supervised learning framework. Although our training dataset is collected from the Internet, most images have a roughly frontal pose. As shown in Fig. 2.5, we can easily collect paired eye-region images: The right eye y_l^r is paired with the mirrored version $F(y_l^l)$ of the left eye y_l^l. Note that y_l^r and $F(y_l^l)$ have different gaze angles, but they have a similar eye shape and iris color. Because they belong to the same person. The same holds for y_l^l and $F(y_l^r)$. Note that the only information we need to collect these pairs is the eye region position (*i.e.*, $M(x)$). At the same time, the mirroring operation ($F(\cdot)$) is a data augmentation technique commonly used in other self-supervised learning approaches. We use these paired samples to pretrain G_{pre} using the following objective function:

$$\begin{aligned} \mathcal{L}_{pre} = \quad & \|y_l^l - G_{pre}(y_l^l)\|_1 + \|y_l^l - G_{pre}(F(y_l^r)\|_1 \\ + \quad & \|y_l^r - G_{pre}(y_l^r)\|_1 + \|y_l^r - G_{pre}(F(y_l^l))\|_1. \end{aligned} \tag{2.7}$$

After training, the bottleneck features c of G_{pre} are almost angle-invariant, representing only the content information (*e.g.*, iris color, eye shape). Thus, we use the encoder network E_c of G_{pre} to provide extra input to G_x and G_y by means of its content features (see Sec. 2.3.2 and 2.3.3). Conditioning the generation process of G_x and G_y using these content features, the inpainted results are more consistent with respect to the input identity information.

2.3.5 Loss Functions

Reconstruction Losses. We use a standard pixel-wise loss ($L1$) for training GCM. It is defined as:

$$\mathcal{L}_{re}^{x} = \|x - \tilde{x}\|_1 + \|x_l^h - \hat{x}_l^h\|_1, \tag{2.8}$$

where x_l^h and $\hat{x}_l^h$ are the eye regions of x^h and $\hat{x}^h$, respectively.

And, the reconstruction loss for GAM is defined as:

$$\mathcal{L}_{re}^{y} = \|y - \tilde{y}\|_1 + \|y_l^h - \hat{y}_l^h\|_1. \tag{2.9}$$

The reconstruction loss of GAM for the synthesis-as-training method is defined as:

$$\mathcal{L}_{re}^{y^x} = \|y^x - \tilde{y}^x\|_1 + \|y_l^{h_x} - \hat{y}_l^{h_x}\|_1. \tag{2.10}$$

We use $\mathcal{L}_{re}^{y} + \mathcal{L}_{re}^{y^x}$ as the objective function to train G_y.

Global and Local Discriminators for Adversarial Learning. Since the $L1$ loss tends to produce blurry results [71], we use three different discriminators D_x, D_y and D_h, adversarially trained together with G_x, G_y and G_h, respectively. Moreover, inspired by [69], our discriminators D_x and D_y are composed of a global part, taking the whole face as input, and a local part with taking only the local eye region as input. The global part is used to coherent the entire image as a whole, while the local part makes the local region more realistic and sharper. We concatenate the final fully-connected feature maps of both parts, which are fed to a sigmoid function to predict the probability of the image being real.

Different from D_x and D_y, D_h consists of only a local discriminator, which uses the eye regions $\hat{x}_l^h$ and $\hat{y}_l^h$ as fake inputs. In practice, we use crops slightly larger than the eye region as the input of the discriminator to alleviate the boundary mismatch problem. The objective function of D_x and G_x is defined as:

$$\begin{aligned}\min_{G_x}\max_{D_x}\mathcal{L}^x_{adv} &= \mathbb{E}_x[logD_x(x, M^{'}(x))] \\ &+ \mathbb{E}_{\tilde{x}}[log(1 - D_x(\tilde{x}, M^{'}(\tilde{x})))] \\ &+ \mathbb{E}_{\tilde{y}^x}[log(1 - D_x(\tilde{y}^x, M^{'}(\tilde{y}^x)))].\end{aligned} \tag{2.11}$$

The objective function of D_y and G_y is defined as:

$$\begin{aligned}\min_{G_y}\max_{D_y}\mathcal{L}^y_{adv} &= \mathbb{E}_y[logD_y(y, M^{'}(y))] \\ &+ \mathbb{E}_{\tilde{y}}[log(1 - D_y(\tilde{y}, M^{'}(\tilde{y})))].\end{aligned} \tag{2.12}$$

Finally, the objective function of D_h and G_h is:

$$\begin{aligned}\min_{G_h}\max_{D_h}\mathcal{L}^h_{adv} &= \mathbb{E}_{x^h}[logD_h(M^{'}(x^h))] \\ &+ \mathbb{E}_{\hat{x}^h_l}[log(1 - D_h(\hat{x}^h_l))] \\ &+ \mathbb{E}_{y^h}[logD_h(M^{'}(y^h))] \\ &+ \mathbb{E}_{\hat{y}^h_l}[log(1 - D_h(\hat{y}^h_l))].\end{aligned} \tag{2.13}$$

2.3.6 Overall Objective Function

Inspired by [67], we use a latent-space reconstruction loss (l_{fp}) for the content features in the latent space to preserve further the identity information between the input image and the gaze-corrected result:

$$\mathcal{L}_{fp} = \|c_y - E_c(M^{'}(\tilde{y}))\|_1 + \|c_{y^x} - E_c(M^{'}(\tilde{y}^x))\|_1. \tag{2.14}$$

We use $-\ell^x_{adv}$, $-\ell^y_{adv}$ to train D_x and D_y, respectively. Concerning G_x, its overall loss is defined as:

$$\mathcal{L}^x_{all} = \mathcal{L}^x_{adv} + \mathcal{L}^h_{adv} + \lambda_1\mathcal{L}^x_{re}. \tag{2.15}$$

For G_y and E_r, the overall loss is defined as:

$$\begin{aligned} \mathcal{L}^{y}_{all} &= \mathcal{L}^{y}_{adv} + \mathcal{L}^{h}_{adv} + \lambda_2 \mathcal{L}^{x}_{adv} \\ &+ \lambda_3 \mathcal{L}^{y}_{re} + \lambda_4 \mathcal{L}^{y^x}_{re} + \lambda_5 \mathcal{L}_{fp}. \end{aligned} \tag{2.16}$$

$\lambda_1, \lambda_2, \lambda_3, \lambda_4$ and λ_5 are hyper-parameters controlling the contribution of each loss term.

2.3.7 Gaze Correction and Animation at Inference Time

At inference time, given an image sample y^h, we first downsample it to attain y, then obtain the gaze-corrected result y^x using G_x, and finally output high-resolution results y^{h_x} by using CFM, which can compensate for high-frequency texture details. In the case of gaze animation, as shown in Fig. 2.4, we modify the angle representation r by interpolating between r_y and r_{y^x}, where both features correspond to the encoded angle features of the eye region y_l and the eye region y^x_l, respectively. The interpolated angle representation can be fed to G_y to obtain an intermediate result. We can produce high-resolution gaze animation results using CFM.

2.4 Experiments

This section introduces the details of our datasets, our network training, and baseline models. Then, we compare the proposed method with the state-of-the-art methods of gaze correction in the wild using both qualitative and quantitative evaluations. Next, we demonstrate the effectiveness of the proposed method on gaze animation with various outputs by interpolating and extrapolating in the latent space. Finally, we present detailed ablation studies to validate the effect of each component of our model, *i.e.*, the Synthesis-As-Training method, the Pretrained Autoencoder (PAM) with self-supervised mirror learning, the Latent Reconstruction Loss, and the

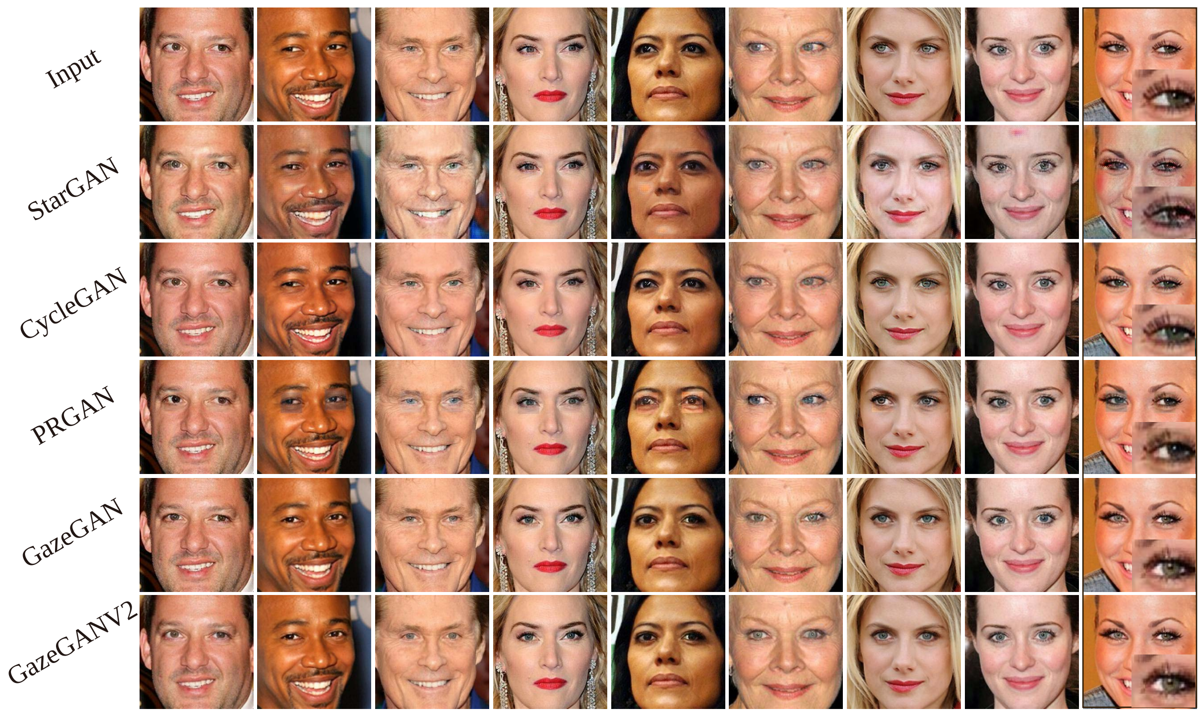

Figure 2.6: Qualitative comparison for the gaze correction task on the CelebGaze dataset. The first row shows the input images, and the following rows show the gaze correction results of StarGAN [27], CycleGAN [226], PRGAN [54], GazeGAN and GazeGANV2. Magnified left eyes are shown in the last column. Zoom in for the best of view.

Coarse-to-Fine Module (CFM). For brevity, we refer to the low-resolution version as **GazeGAN** and the extended high-resolution version as **Gaze-GANV2**. Note that we do not use any post-processing step for GazeGAN and GazeGANV2.

2.4.1 Datasets

Most of the existing benchmarks [40, 171, 218, 216] do not contain enough image variability (*e.g.*, a wide gaze-direction range, various head poses, and different illumination conditions) for our gaze correction task in the wild. Recently, [83] presented a large-scale gaze tracking dataset, called Gaze360, for robust 3D gaze estimation in unconstrained images. Although

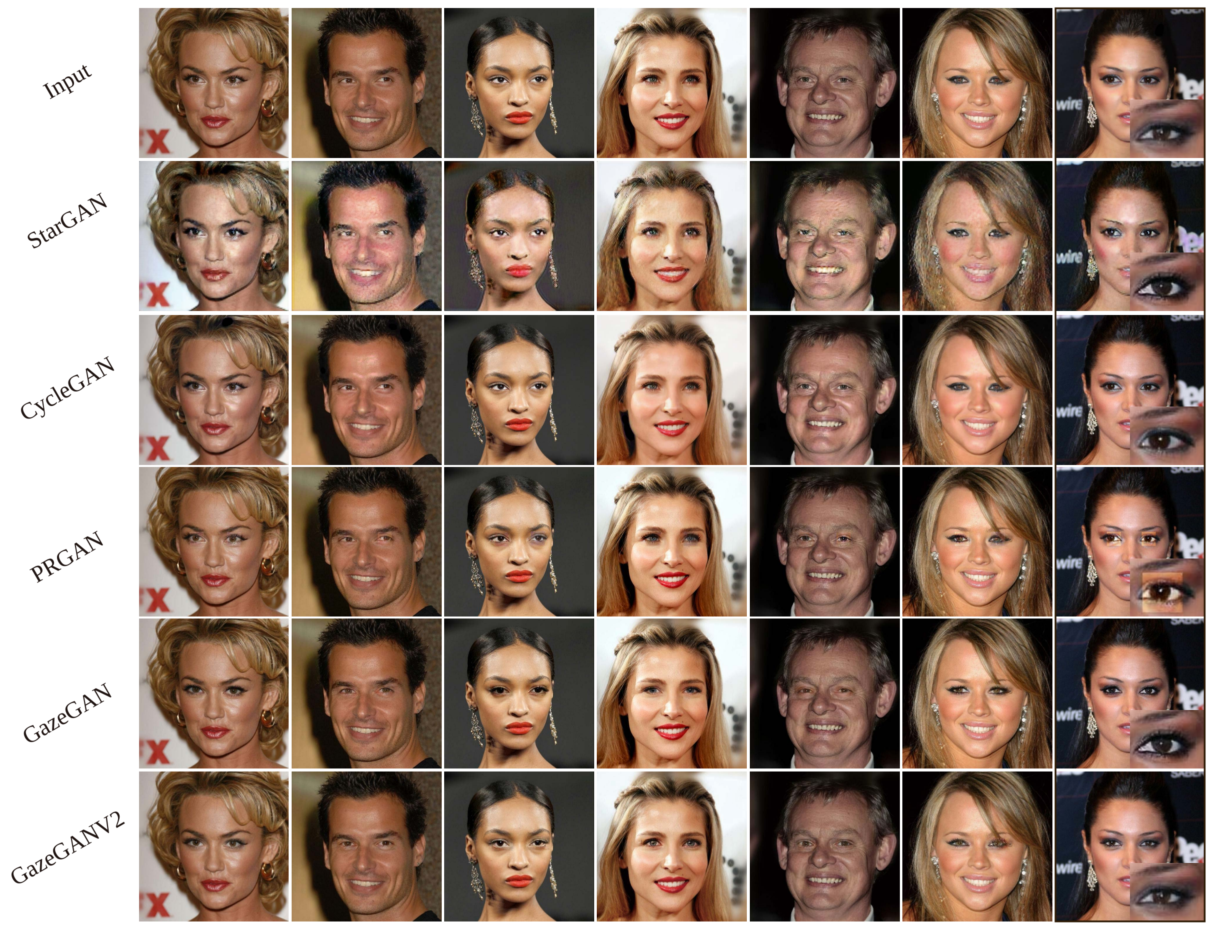

Figure 2.7: Qualitative comparison for the gaze correction task on CelebHQGaze dataset. The first row shows the input images, and the following rows show the gaze correction results of StarGAN [27], CycleGAN [226], PRGAN [54], GazeGAN and GazeGANV2. Magnified left eyes are shown in the last column. Zoom in for the best of view.

this dataset has been labeled with a 3D gaze direction with a wide range of angles and head poses, it still lacks high-resolution images for face and eye regions. Moreover, this dataset does not provide annotations of the eye gaze staring at the camera, which is required in our domain set X. More recently, [217] proposed a large scale (over 1 million samples) of high-resolution images for gaze estimation. However, these images are collected in laboratory conditions and are not suitable for our gaze correction task in the wild. To remedy this problem, we propose collecting new datasets consisting of lots of high-resolution portraits without labelling head poses and gaze information. In detail, five volunteers are asked to divide the row data (face) into two domains according to whether the face eyes are staring at the camera. The gaze and head estimation model can automate 'Staring at the camera' annotation. However, the existing methods [83, 18] cannot achieve accurate gaze estimation for CelebHQGaze, as an overlarge domain shift exists between training data and test data. More details about our datasets can be found below.

CelebGaze. CelebGaze consists of 25,283 celebrity images, most of which have been collected from CelebA [111] and a minority from the Internet. Specifically, there are 21,832 face images with the eyes staring at the camera and 3,451 face images with the eyes looking somewhere else. We crop all the images to 256 $\times$ 256 and compute the eye mask region using Dlib [85]. Specifically, we use Dlib to extract 68 facial landmarks, and we compute the mean of 6 points near the eye region, which is the center point of the mask. The size of the mask is fixed to 30 $\times$ 50. We randomly select 300 samples from domain Y and 100 samples from domain X as their corresponding test sets, and we use the remaining images for the training set. Note that this dataset is unpaired and not labeled with the specific eye angle or the head pose information. We show some samples of the CelebGaze dataset in Fig. 2.1.

CelebHQGaze. CelebHQGaze consists of 29,255 high-resolution celebrity images that are collected from CelebA-HQ [99]. It consists of 21,005 face images with the eyes staring at the camera and 8,250 face images with eyes looking somewhere else. Similarly to CelebGaze, we extract facial landmarks and generate the mask. All images are cropped to 512 $\times$ 512, and the mask size is fixed to 46×80. Similar to the CelebGaze dataset, also for the CelebHQGaze, we randomly select 300 samples from domain Y and 100 samples from domain X for the test set, and we use all the remaining

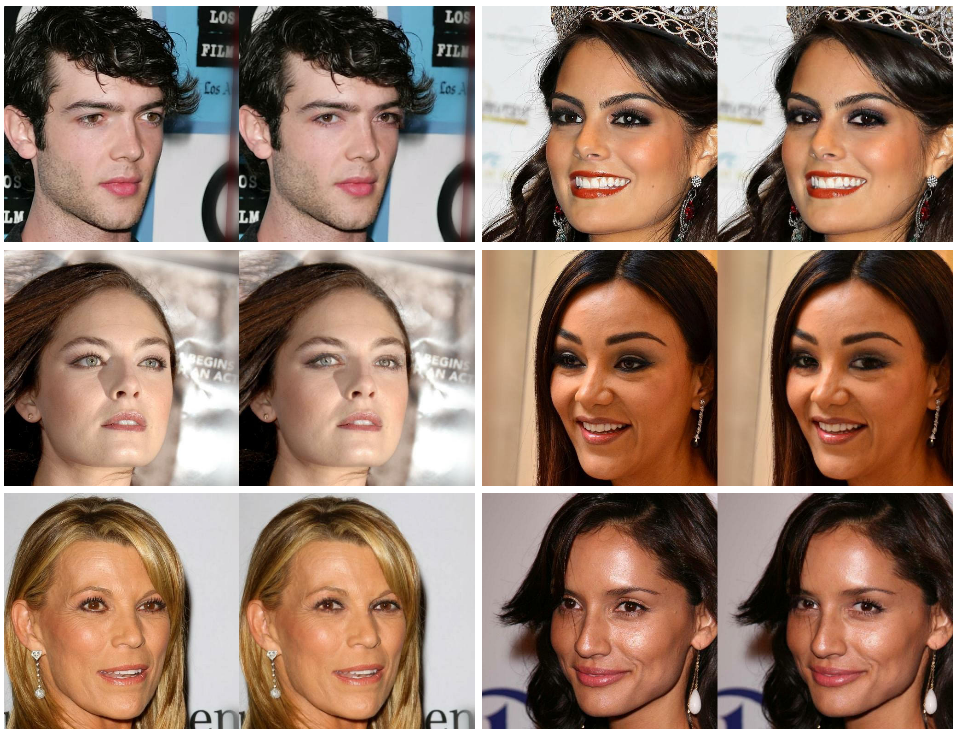

Figure 2.8: More gaze correction results to show that our model can handle diverse head poses.

images for the training set. We show two samples of the CelebHQGaze dataset in Fig. 2.1.

2.4.2 Training Details

We first train the PAM module. Then, the discriminators D_x, D_y and D_h and the generators G_x and G_y and G_h are jointly optimized. We use the Adam optimizer [86] with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The batch size is 16 for CelebGaze and 8 for CelebHQGaze. The initial learning rate is 0.0005 for PAM, 0.0004 for G_h, and 0.0001 for the three discriminators and the two generators in the first 20,000 iterations. The learning rate is linearly decayed to 0 over the remaining iterations. The loss coefficients $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are all set to 1, while λ_5 is 0.1. To stabilize the network training in the adversarial learning, we use spectral normalization [124] for all the conv-

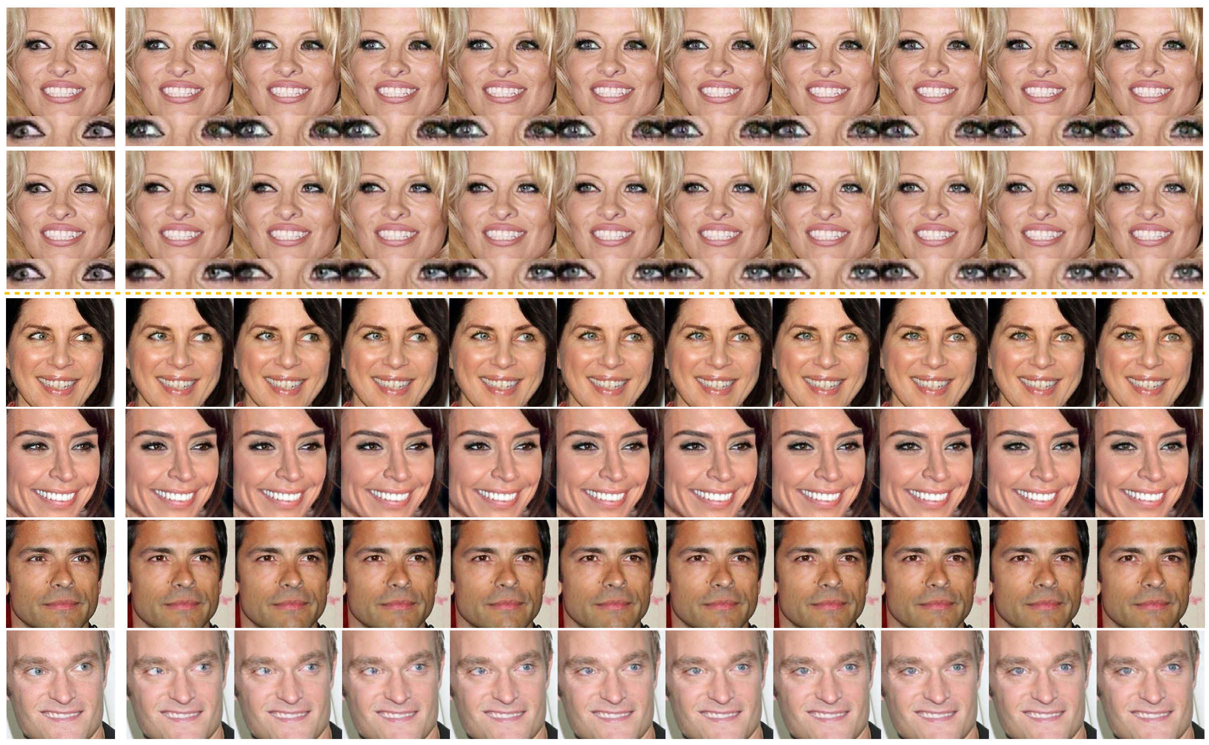

Figure 2.9: Gaze animation results using the interpolation of the latent features r on the CelebGaze dataset. The top two rows show the images generated by GazeGAN and GazeGANV2, respectively, jointly with the eye regions. The other rows show the gaze animation results of GazeGANV2. The first and the last columns show the input images and the gaze-corrected results, respectively. The middle columns show the interpolated images.

layers of the three discriminators, but not for the generators. Our method is implemented in Tensorflow and trained with a single NVIDIA Titan X GPU.

2.4.3 Baseline Models

Gaze Correction. PRGAN [54] achieved state-of-the-art gaze redirection results on the Columbia gaze dataset [171] based on a single encoder-decoder network with adversarial learning, similarly to the StarGAN architecture [27]. The original PRGAN is trained on paired samples with labeled

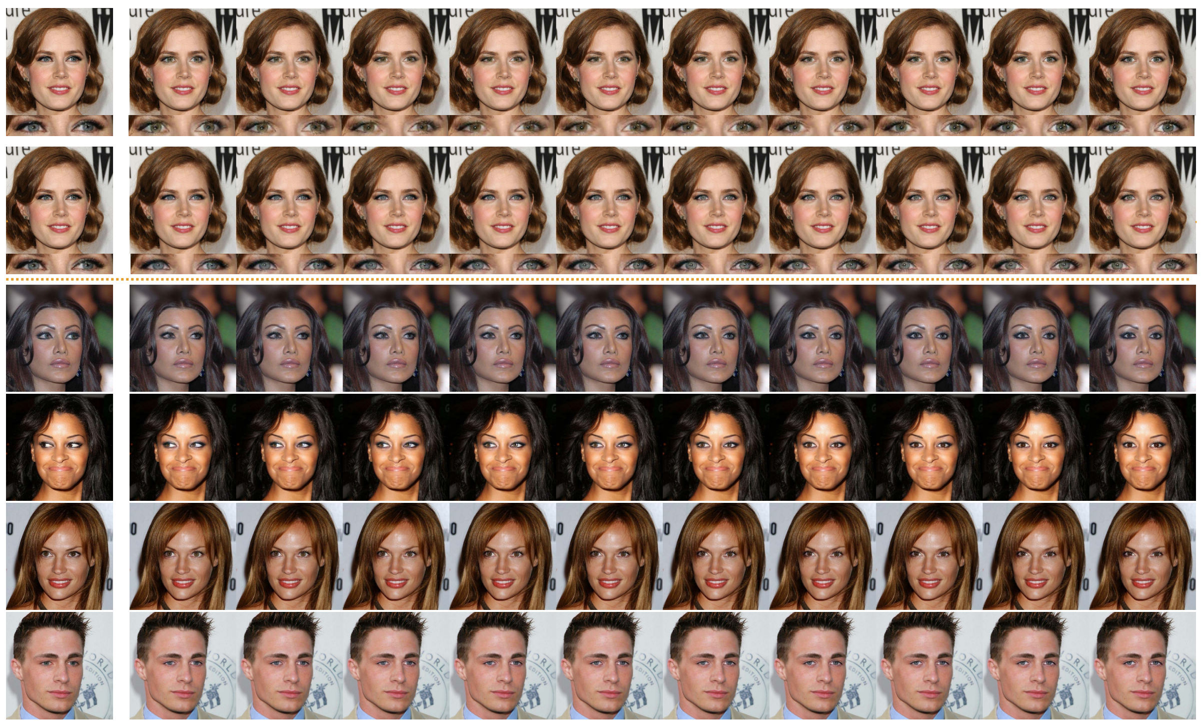

Figure 2.10: Gaze animation results using the interpolation of the latent features r on the CelebHQGaze dataset. The top two rows show the images generated by GazeGAN and GazeGANV2, respectively, jointly with the eye regions. The other rows show the gaze animation results of GazeGANV2. The first and the last columns show the input images and gaze-corrected results, respectively. The middle columns show the interpolated images.

angles. To train PRGAN on the proposed CelebGaze and CelebHQGaze datasets, we remove the VGG perceptual loss of PRGAN, and learn the gaze redirection task between domain X and Y. We train PRGAN only with the local eye region, the same way as the original paper.

Facial Attribute Manipulation. Gaze correction and animation can be regarded as a sub-task of facial attribute manipulation. Recently, StarGAN [27] achieved very high-quality results in facial attribute manipulation. We train StarGAN on the CelebGaze dataset to learn the translation mapping between domain X and domain Y.

Moreover, gaze correction can be considered as an image translation

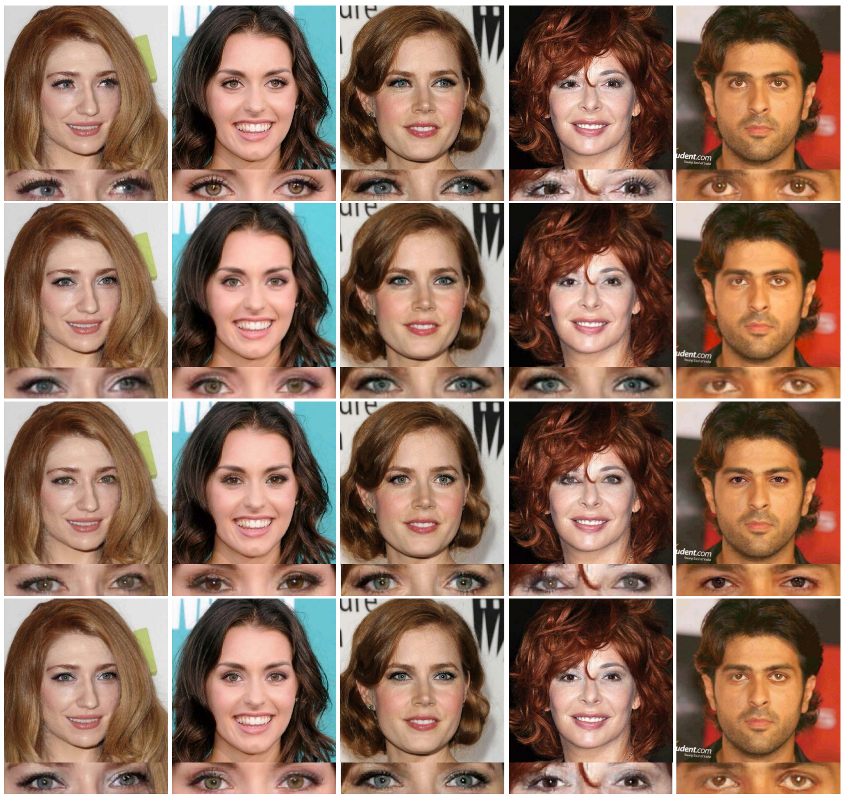

Figure 2.11: A qualitative comparison between GazeGANV2 (4th row), GazeGANV2 W/O A (3rd row), and GazeGANV2 W/O D (2nd row). The first row shows the input images, and the bottom of every row shows the zoom-in eye regions.

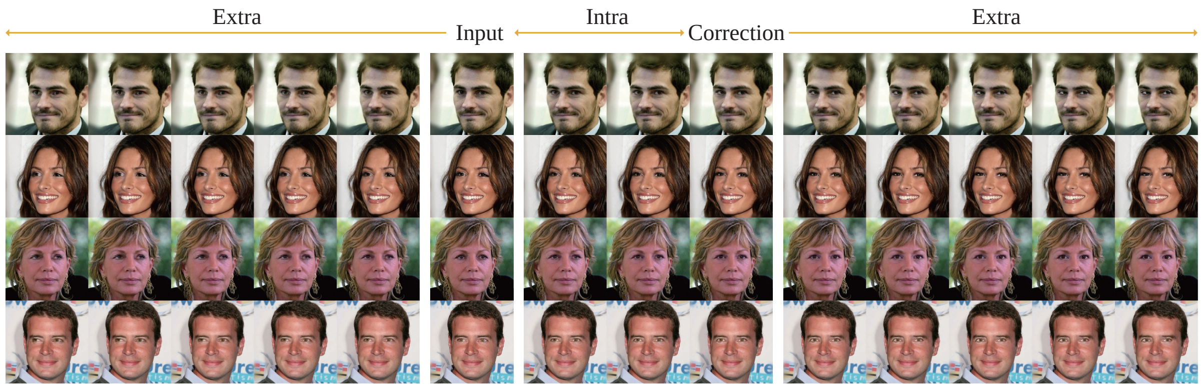

Figure 2.12: Gaze animation examples are obtained by both interpolation and extrapolation of the latent features r. Extra: extrapolation; Intra: interpolation.

task. Thus, we adopt CycleGAN as another baseline for our experiments. Note that we do not compare GazeGAN with AttGAN [57], STGAN [105],

Table 2.1: Quantitative results on both the CelebGaze and the CelebHQGaze dataset. The higher is better for MSSSIM and the user study; the lower is better for LPIPS and FID. The columns Params and FPS report the corresponding network parameters and frame per second at test time, respectively. US: user studies.

Method	CelebGaze						CelebHQGaze					
	MSSSIM ↑	LPIPS ↓	FID ↓	US ↑	Params ↓	FPS ↓	MSSSIM ↑	LPIPS ↓	FID ↓	US ↑	Params ↓	FPS ↓
Other	-	-	-	24.20%	-	-	-	-	-	23.20%	-	-
StarGAN [27]	0.96	0.073	82.49	3.400%	-	-	0.94	0.084	185.47	4.400%	-	-
CycleGAN [226]	0.99	0.026	70.12	15.00%	-	-	0.98	0.028	**53.690**	8.670%	-	-
PRGAN [54]	1.00	0.000	84.61	8.330%	-	-	1.00	0.000	106.79	22.40%	-	-
GazeGAN	**1.00**	**0.000**	62.12	22.40%	73.26M	30.29	**1.00**	**0.000**	60.520	25.50%	183.2M	23.20
GazeGANV2	**1.00**	**0.000**	**56.37**	**32.40%**	**47.20M**	**38.40**	**1.00**	**0.000**	63.590	**27.30%**	**84.18M**	**27.70**
GT	1.00	0.000	-	100%	-	-	1.00	0.000	-	100%	-	-

RelGAN [196], CAFE-GAN [46], SSCGAN [29] as they have a performance very close to StarGAN in the facial attribute manipulation task. We use the public code of StarGAN [1], CycleGAN [2] and PRGAN [3].

2.4.4 Gaze Correction

This section qualitatively and quantitatively compares the proposed method with state-of-the-arts on both CelebGaze and CelebHQGaze datasets for the gaze correction task.

Qualitative Results. As shown in the last row of Fig. 2.6 and Fig. 2.7, GazeGANV2 can correct the eyes to look at the camera while preserving the identity information such as the eye shape and the iris color, validating the effectiveness of the proposed method. The 2nd row of the figure shows the results of StarGAN [27]. We note that StarGAN could not produce precise gazes staring at the camera, and it suffers from a low-quality generation with lots of artifacts (Zoom in for the best of view). The results of CycleGAN are shown in the 3th row. Although the results of CycleGAN are very realistic and with few artifacts in the eye region, this method does not

[1] `https://github.com/yunjey/StarGAN`
[2] `https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix`
[3] `https://github.com/HzDmS/gaze_redirection`

produce a precise correction of the gaze direction (*e.g.*, see the magnified eye regions of Fig. 2.6 and Fig. 2.7). We explain that both StarGAN and CycleGAN use the cycle-consistency loss, which requires that the mapping between X and Y be continuous and invertible. According to the invariance of the Domain Theorem[4], the intrinsic dimensions of the two domains should be the same. However, the intrinsic dimension of Y is much larger than X, as Y has more variations for the gaze angle than X. Moreover, we compare GazeGANV2 with PRGAN [54]. PRGAN is trained using only local eye regions (same as in the original paper), which may help focus on the translation of the eye region. The results of PRGAN are shown in the 4th row of Fig. 2.6. Compared with GazeGANV2, PRGAN does not produce precise and realistic correction results. Additionally, PRGAN suffers from the boundary mismatch problem between the local eye region and the global face (see the last column of Fig. 2.7).

Finally, as shown in the last rows of both Fig. 2.6 and Fig. 2.7, comparing GazeGANV2 with GazeGAN, we observe that both models can produce realistic and faithfully results. Additionally, we show more results of portraits with a diverse head pose. Fig. 2.8 shows that our model can achieve acceptable gaze-correction results for portraits with different head poses.

Quantitative Evaluation Protocol. The qualitative evaluation has validated the effectiveness and the superiority of our proposed GazeGANV2 in the gaze correction task. To further support the previous evaluation with quantitative results, we use the MSSSIM [192] and the LPIPS [214] metrics to measure the preservation ability of the *irrelevant regions*, *i.e.*,, the whole image except the eye region ($M(y^h)$). Specifically, we compute the mean MSSSIM and LPIPS scores between $M(y^h)$ and $M(\hat{y}^h)$ across all the *test* data of Y^h. Moreover, the Fréchet Inception Distance (FID) [62]

[4]`https://en.wikipedia.org/wiki/Invariance_of_domain`

has been shown to correlate well with the human judgment and has become a popular metric for GAN-based methods. We use it to evaluate the quality of the generated eye region for the gaze correction and the gaze animation tasks.

In addition to the aforementioned automatic metrics, we conduct a user study to compare the results of the gaze correction task of different models. In detail, given an input face image of the CelebGaze or CelebHQGaze test dataset (extracted from Y), we show the gaze-corrected results produced by different models to 30 respondents, who were asked to select the best image based on the perceptual realism and the precision of the gaze correction. They also can select "Other", which means that the results of all the models are not satisfactory enough. This study is based on 50 questions (*i.e.*, 50 randomly sampled images) for each respondent.

Quantitative Results. The first two columns of the left part (CelebGaze) and the right part (CelebHQGaze) of Table 2.1 show the MSSSIM and LPIPS scores evaluating the preservation ability of the corrected images using different methods. GazeGANV2 and PRGAN obtain the best results, with 1.0 for MSSSIM and 0.0 for LPIPS. The original irrelevant regions are integrated with the generated eye region in both models using binary masks. StarGAN and CycleGAN get the worse irrelevant region preservation scores. The FID scores of the eye regions are reported in the 3rd column. In the CelebGaze dataset, GazeGANV2 and GazeGAN outperform all the other methods, reaching comparable scores on the CelebHQGaze dataset. Though CycleGAN has the best FID scores, it fails to generate precise gaze correction results. The penultimate column of both parts in Table 2.1 shows the evaluation results of the user study. For the CelebGaze dataset, the average vote for GazeGANV2 is 32.40%, which is higher than all the other methods, *i.e.*, 3.40% for StarGAN, 15.00% for CycleGAN, 8.33% for PRGAN. The same conclusions can be drawn with the

Table 2.2: A comparison on the gaze animation task between GazeGAN and GazeGANV2 with respect to the generation quality.

Method	CelebGaze		CelebHQGaze	
	GazeGAN	GazeGANV2	GazeGAN	GazeGANV2
FID ↓	80.31	**53.32**	**70.56**	71.37

CelebHQGaze dataset. Importantly, GazeGANV2 achieves a performance very close to GazeGAN. However, it has fewer parameters and higher FPS, as shown in the last two columns of Table 2.1.

Overall, the qualitative and quantitative evaluations demonstrate the effectiveness and superiority of the proposed approach.

2.4.5 Gaze Animation

The bottom of Fig. 2.9 and 2.10 show gaze animation results using input images with various gaze directions. The latent-space interpolation results are smooth and plausible in each row. Each column has a different gaze direction angle, but the identity information is overall preserved (*e.g.*, the eye shape, the iris color, etc.).

The top rows of Fig. 2.9 and 2.10 show a gaze animation comparison between GazeGAN and GazeGANV2. GazeGANV2 can produce more realistic images with fewer artifacts than GazeGAN on the CelebGaze dataset, while they have comparable performance on the CelebHQGaze dataset. The quantitative result confirms it in Table 2.2.

Finally, we show gaze animation results obtained by *extrapolating* the features r, in addition to using interpolation methods. With "extrapolation," we mean that we use values of r which lie in the line connecting r_y with r_{y^x}, but they are outside these two points. As shown in Fig. 2.12, our method not only achieves high-quality interpolation results but is also able

Table 2.3: Comparison between GazeGANV2 and GazeGANV2 W/O A, where the latter denotes removing the content representation extracted from E_c. The scores are measured between the input image x and inpainted result $\tilde{x}$ across the test data from X. Note that evaluation samples are from CelebHQGaze.

Metrics	GazeGANV2	GazeGANV2 W/O A
MSSSIM $\uparrow$	**0.6080**	0.5230
LPIPS $\downarrow$	**0.1680**	0.2646

Table 2.4: Comparison with GazeGAN W/O C, which denotes removing the latent reconstruction loss $\mathcal{L}_{fp}$. The scores are measured between the input image y and the reconstruction result $\tilde{y}$ across all the test data of domain Y. The evaluation is based on the CelebHQGaze dataset.

Metrics	GazeGANV2	GazeGANV2 W/O C
MSSSIM $\uparrow$	**0.6290**	0.6100
LPIPS $\downarrow$	**0.2328**	0.2372

to produce plausible gaze animations when the gaze angles are outside the range between the input and the gaze-corrected output.

2.4.6 Ablation Study

In this section, we conduct extensive ablation studies to investigate the contribution of each of four critical components of our proposed GazeGANV2, *i.e.*, the Pretrained Autoencoder for content feature extraction, the Synthesis-As-Training method, the Latent Reconstruction Loss $\mathcal{L}_{fp}$ and the Coarse-to-Fine Module. We refer to these components as A, B, C, and D, respectively.

Pretrained Autoencoder (PAM). Sec. 2.3.4 shows how self-supervised learning is used to pretrain a content encoder. This encoder produces identity-specific features which condition the generation process of G_x and G_y. Here we analyze this aspect of our method.

Fig. 2.11 shows that our full-model (GazeGANV2) can better preserve

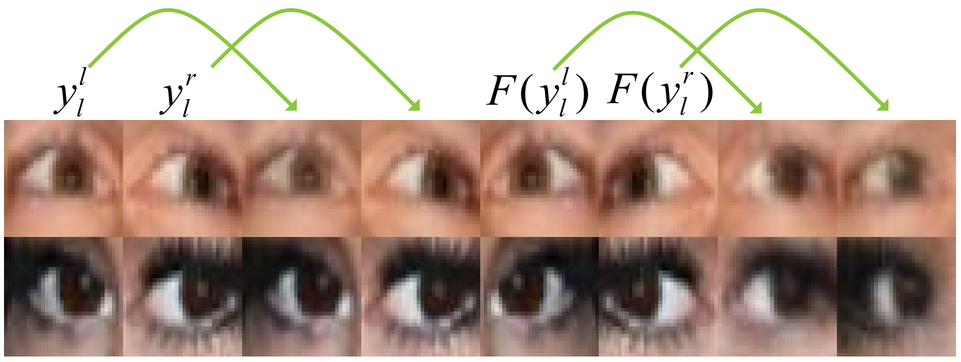

Figure 2.13: The visualization for the results of PAM with taking y_l^l, y_l^r, $F(y_l^l)$ and $F(y_l^r)$ as inputs, respectively. The arrows point to the generated sample.

identity information with respect to GazeGANV2 W/O A. To quantify this, we use G_x to reconstruct test image samples x ($x \in X$), and we measure the difference between the input image and the gaze-corrected result in local eye regions employing both MSSSIM and LPIPS metrics. Table 2.3 shows that GazeGANV2 gets better scores than GazeGANV2 W/O A, confirming our design motivation.

A shown in Fig. 2.13, we visualize the outputs of autoencoder with taking y_l^l, y_l^r, $F(y_l^l)$ and y_l^r as inputs after training. We can observe that the model can attain the similar reconstruction results for y_l^l and $F(y_l^r)$ as inputs, and can also attain the similar reconstruction results for y_l^r and $F(y_l^l)$ as inputs which validates the effectiveness of the objective loss.

Synthesis-As-Training Method. The gaze animation results in Fig. 2.9 and 2.10 show the effectiveness of our method in disentangling the angle representation. Fig. 2.16 (top) shows a t-SNE visualization of points interpolated in the latent space. In more detail, following the procedure explained in Sec. 2.3.7, we uniformly interpolate the line connecting r_y with r_{y^x} in the angle latent space using 5 interpolation points $(I_1, ...I_5)$ for each sample y. Fig. 2.16 (top) shows that for each specific sample y, these five interpolation points are different from each other but strongly

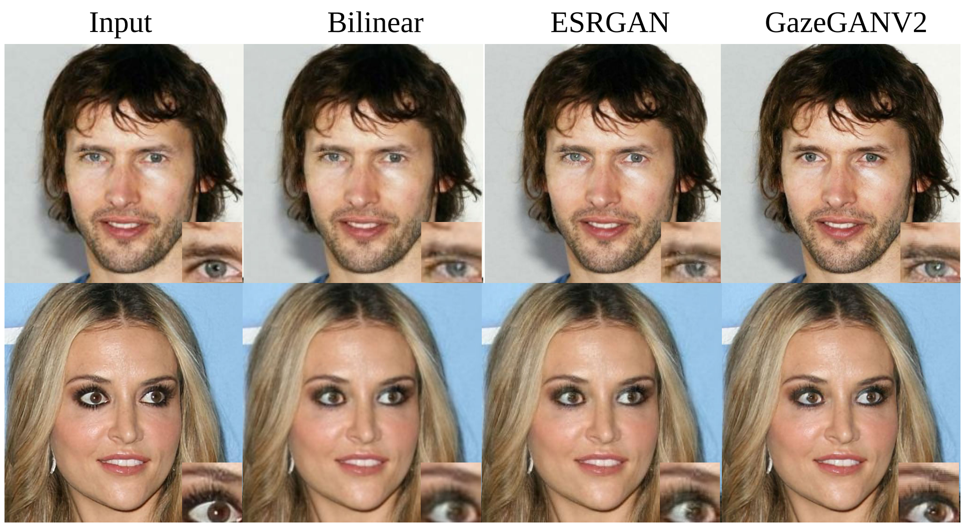

Figure 2.14: Qualitative comparison between GazeGANV2 with Bilinear and super-resolution model ESRGAN [191]. Note that the input image would be downsampled $2\times$ as input of the inpainting model.

Table 2.5: Comparison between GazeGANV2 and GazeGANV2 W/O D with respect to the generation quality in gaze animation.

Method	CelebGaze		CelebHQGaze	
	GazeGANV2	W/O D	GazeGANV2	W/O D
FID $\downarrow$	**53.32**	78.32	**71.37**	74.04

clustered together, which illustrates the disentanglement of the angle latent space.

Latent Reconstruction Loss $\mathcal{L}_{fp}$. We use G_y to fill in the eye region of test images y ($y \in Y$), and we measure the difference between the input images and the generated results employing MSSSIM and LPIPS. In Table 2.4, GazeGANV2 obtains better scores than GazeGANV2 W/O C, which shows that $\mathcal{L}_{fp}$ further improves the ability to preserve identity information. Moreover, we visualize the content features c_y and $c_{\tilde{y}}$ extracted from real samples y and reconstructed samples $\tilde{y}$ across all the Y test data. As shown in Fig. 2.16 (bottom), we observe that using our full model Gaze-

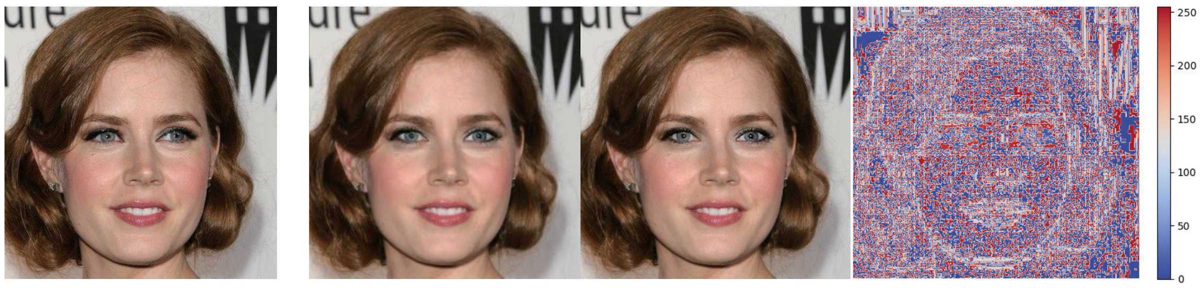

Figure 2.15: A qualitative comparison between the gaze-correction results produced by GazeGANV2 (3rd column) and GazeGANV2 W/O D (2nd column). The 1st column shows the input image, and the final column is a heatmap of the difference between the 3rd column and 2nd column. This residual image clearly shows semantic and texture information.

Table 2.6: Quantitative comparison between CFM of GazeGANV2 with Bilinear and super-resolution model ESRGAN [191].

Metrics	Bilinear	ESRGAN	CFM
MSSSIM ↑	0.9563	0.9595	**0.9827**
LPIPS ↓	0.2393	0.1476	**0.1039**
FPS ↓	30.600	4.3000	**27.700**
Params ↓	**48.88M**	80.88M	49.23M

GANV2, c_y and $c_{\tilde{y}}$ usually lie closer to each other to what happens when using GazeGANV2 W/O C, and it shows that this loss helps to represent content information consistently.

Coarse-to-Fine Module (CFM). The previous experiments validate the effectiveness of CFM. As shown in the 2nd and the 4th row of Fig. 2.11, the gaze correction results of GazeGANV2 are more realistic than those produced by GazeGANV2 W/O D. In Table 2.5, the quantitative comparison between GazeGAN and GazeGAN W/O D confirms the effectiveness of CFM. Then, Fig. 2.15 shows the differences in the gaze correction results obtained with GazeGAN and GazeGAN W/O D. The heatmap of the difference between the two generated images shows that CFM can compensate for the high-frequency information loss of the coarse output. Finally, we compare our CFM with some upsampling methods, such as

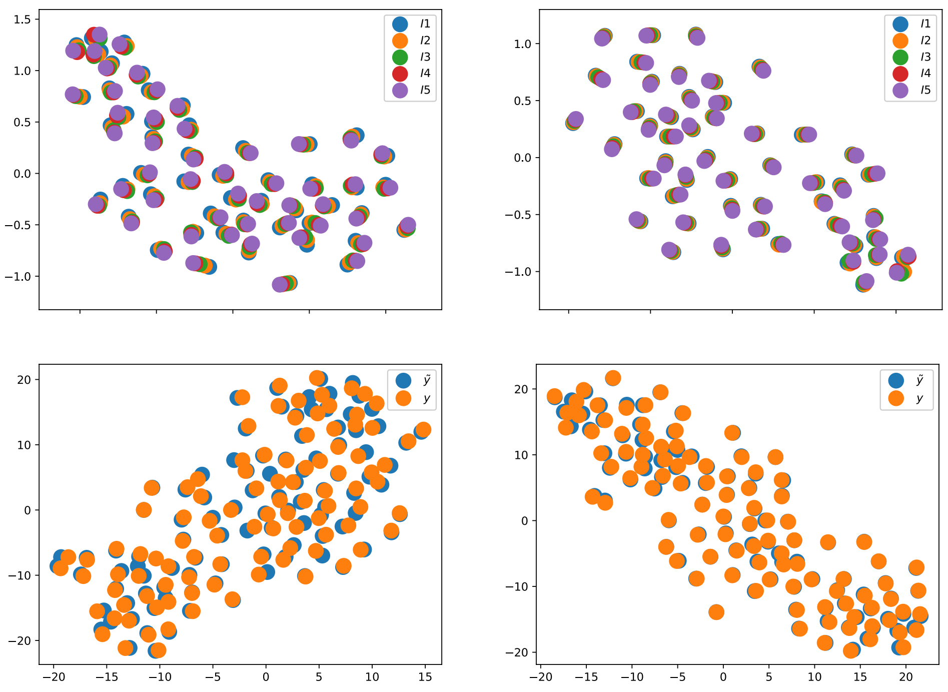

Figure 2.16: Top: a t-SNE based visualization of the latent space r which represents the gaze angle (Sec. 2.3.3).We show the corresponding latent spaces of GazeGAN (top-left) and GazeGANV2 (top-right). We plot 5 interpolated points ($I1 - I5$) for each image and we use 50 images. Bottom: t-SNE visualization of the content features c_y (orange) and $c_{\tilde{y}}$ (blue) extracted from y and $\tilde{y}$, respectively. Bottom-Left: GazeGANV2 W/O C; Bottom-Right: GazeGANV2.

Bilinear and super-resolution method, ESRGAN [191]. By taking all samples y^h from domain Y^h, we attain all low-resolution reconstructed results $\tilde{y}$. Then, three different methods are used for upsampling them to attain high-resolution results. Fig. 2.14 shows our method achieves better reconstruction with fewer artifacts, such as eye regions. Quantitative experiments of Table 2.6 show our CFM achieves better MSSSIM and LPIPS scores and has higher FPS and fewer parameters than ESRGAN.

2.5 Conclusion

In this paper, we introduce two new gaze dataset in the wild, CelebAGaze and CelebHQGaze, which is characterized by a large diversity in head poses and gaze angles. Moreover, we propose a novel unsupervised method, GazeGAN, and extend it into high-resolution version GazeGANV2 for gaze-direction correction and animation. GazeGAN and GazeGANV2 formulate the gaze correction problem as an inpainting task and use a coarse-to-fine learning strategy to generate high-resolution images. Moreover, self-supervised learning and Synthesis-As-Training methods are used to disentangle the content and angle-specific features, which can condition the generation process. The qualitative and quantitative results demonstrate the method's effectiveness and its superiority to the state of the arts. In the next Chapter 3, we will introduce our method, TT-GNeRF for high-quality 3D-Aware face editing.

Chapter 3

Exploring 3D-Aware Face Attribute Editing

3.1 Introduction

High-quality image generation and semantic disentanglement are long-standing goals in the fields of computer vision and computer graphics. In the past few years, Generative Adversarial Networks (GANs) [47] and their variants have garnered 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 prior works [27, 6, 184, 165] have focused on realistic face editing. These approaches either rely on image-to-image translation models [27, 105] or leverage the disentanglement abilities [6, 184, 165] of StyleGAN [81, 82]. These methods can be broadly classified into supervised and unsupervised categories. Unsupervised methods typically search for interpretable directions using PCA [53] or introduce soft orthogonality constraints [55, 189] in the latent space. However, these approaches provide only coarse controls. Supervised methods [27, 6], 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 tend to overlook the underlying 3D scene rendering process. To address this issue, some works [184, 34] have integrated 3D Morphable Face Models (3DMM) [144] to enable control over 3D face pose and facial expression. Nonetheless, 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) [123] have attracted booming attention because of 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 [213, 125] and apply it to various downstream tasks, such as human body modeling [147] and large scene modeling [179].

Some 3D-aware image generation methods [203, 19] 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 density and RGB color. However, these methods are compute-intensive and memory inefficient due to sampling many rays in the entire 3D-Volume space and requiring a feed-forward process for each point. They are limited to low-resolution and low-quality generation. GIRAFFE [132] addresses this by generating low-resolution feature fields and using a convolution-based neural rendering module to map rendered features into high-resolution output. However, it suffers from serious view-inconsistency problems. To improve generation quality and view-consistency, many approaches [135, 20, 215] borrow ideas from StyleGAN and integrate the 'Style-modules' into the implicit function (*e.g.*, SIREN [19]) or neural rendering module. Additionally, some novel algorithms and losses have been carefully designed for 3D-aware generation, such as tri-planes [20] or multiple-view warping loss [215]. While these models create high-quality and view-consistent images, they lack control and disentangling abilities. As explained in VolumeGAN [200], some models are limited to local receptive fields with MLPs, and it is hard to extract global structures from their internal representation. 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 challenges in terms of quality and view-consistency and does not support attribute controls for face manipulation.

We have developed an attribute-conditional 3D-aware generative model that can control facial attributes and address the aforementioned issues. Our model differs from NeRF-based Head-Avatar models [232, 65] in two significant ways. Firstly, our model does not rely on 3DMM priors and does not require modeling of 3DMM coefficients. Secondly, we use a pre-trained 3D-Aware GNeRF as the backbone of our model, eliminating the need for retraining. We have also proposed a Dual-Branches Attribute Editing Module (DAEM) that can edit latent codes in specific and interpretable directions (see Fig. 3.1). To train our DAEM module, we keep our pre-trained GNeRF fixed and only train the DAEM module by sampling the training triplets: the latent codes from the GNeRF, the corresponding generated face images, and the labels obtained by the classifiers. Note that a similar triplet sampling strategy was also utilized in StyleFlow [6] which learns to edit the latent space for 2D-GAN models. To improve our results and better preserve the non-target regions in images, we propose a novel "Training-as-Init, Optimizing-for-Tuning" method (TRIOT) (See the left of Fig. 3.2). In the proposed TRIOT, we use the edited latent vector for target attributes as initialization and then optimize this latent vector with the proposed semantic-guided texture and geometry consistency losses while fixing the rest of the model. Finally, we present an unsupervised optimization method for editing the geometry of the face (See the right of Fig. 4.3). We have released the code at `https://github.com/zhangqianhui/TT-GNeRF`.

In summary, the main contributions of this work are:

1) We propose a Dual Branches Attribute-Editing Module (DAEM) that enhances the controllability and disentanglement of 3D-aware generative models and a novel optimization strategy, 'Training-as-Init, Optimizing-for-Tuning' (TRIOT), combining the module-training and latent-optimization method for the attribute-editing task.

2) Our method is flexible, general and can be easily integrated into most 3D-Aware GAN backbones.

3) Compared to baseline models, our model achieves high-quality editing with improved view consistency while preserving non-target regions.

4) We propose an unsupervised optimization method for reference-based geometry editing.

3.2 Background

Generative Neural Radiance Fields for 3D-Aware Face Generation. Neural radiance fields (NeRF) [123], 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

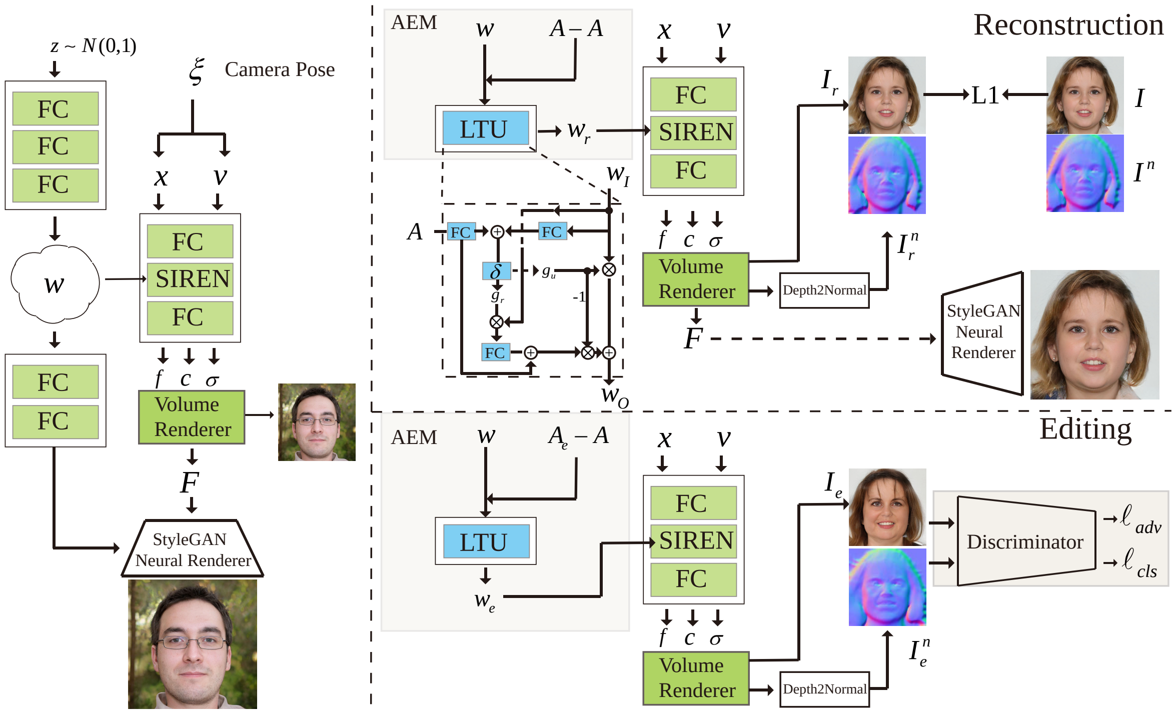

Figure 3.1: The architecture of our backbone StyleSDF (Left) [135]. The overview of the pretrained StyleSDF with double branches attribute-editing module (DAEM) (Right). One branch (AEM) performs the reconstruction using a zero label residual ($A - A = 0$) as input, while the other branch performs the editing using an label residual ($A_e - A$) as input. A_e is a randomly generated target label.

rendering speed [148, 13, 125] and rendering quality [130, 14, 188]. Moreover, NeRF also promotes the development of many computer-graphics applications, such as human body modeling [147, 146], 3D-aware face generation [132, 135, 50, 200, 20, 117, 1], large scene modeling [179], and pose estimation [203].

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 [203, 19, 132]. For example, GRAF [160] achieves this goal by incorporating shape and appearance codes as input. GRAF [160] achieves better visual fidelity and view consistency than the previous voxel- and feature-based methods [59, 127]. Michael *et al.* [132] 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 [19, 220, 35, 137, 50, 135, 20, 215, 201, 161, 170, 197] are trying to address these problems, and most of them inherit the "image-as-style" idea from StyleGAN [82]. Yang *et al.* [201] extend the GIRAFFE to work with high-resolution data. However, this model still suffers from the view-inconsistency problems. Pi-GAN [19] 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 [135], MVCGAN [215] and EG3D [20] 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 [220] keeps the resolution of intermediate feature fields same as resolution of the final images. Though these models can achieve incredible quality generation with strong view consistency, they cannot edit structures and textures.

Recently, some research has focused on the disentangling abilities of the 3D-aware models, VolumeGAN [200] tries to separate shape from texture, while ShadeGAN [137] disentangles 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. Some

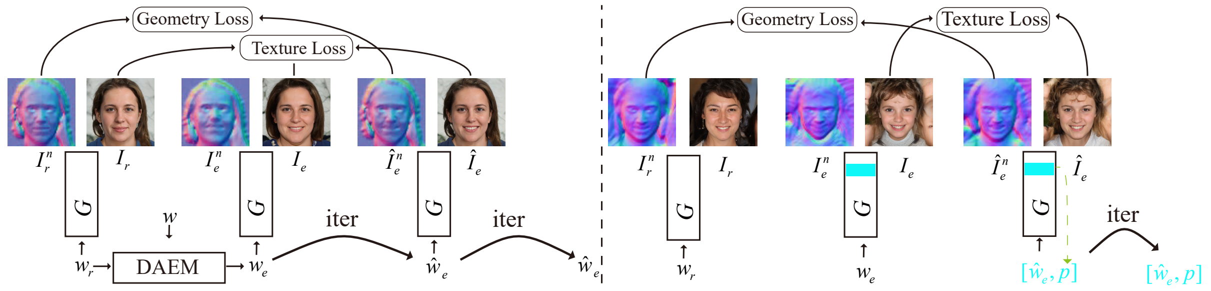

Figure 3.2: Left: The pipeline of TRIOT with geometry consistency loss and texture consistency loss for the expression editing. Right: the reference-based geometry editing pipeline. It minimizes difference between normals (geometry loss) and the differences between texture images (texture loss) in perceptual space to search for a better w_e. Meanwhile, partial parameters p of generator G will be tuned.

of the concurrent methods [174, 177] provide high-quality expression control but are limited in their ability to handle diverse attributes, and require the model to be trained from scratch. To address this problem, we propose an attribute-conditional 3D-aware GAN model that inputs specific attribute labels. Compared to the methods exploiting 3DMM prior [232, 65], this model leverages a pretrained GNeRF and integrates a double-branches attribute-editing module (DAEM) that takes specific attributes as input. Moreover, we propose a novel "Training-as-init, Optimized-for-Turning" method (TRIOT) to train and optimize attribute-editing modules (AEM), that helps to achieve better 3D-aware face generation and editing while preserving non-target regions.

Image-to-Image Translation Architectures for Face Editing. Image-to-Image translation models, *i.e.*, Pix2Pix [70] and CycleGAN [228], utilize the autoencoder as a generator that has been widely adopted for a variety of different tasks, including face attribute editing [27, 57, 105, 196, 29, 56, 44, 23, 109]. Specifically, StarGAN [27] is the early work for learning multiple-domain face translation, it takes multiple attributes as input, and transfer 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 [105], SSCGAN [29] and HifaFace [44]. However, these models operate on a low-resolution data, they cannot manipulate 3D factors, such as camera poses.

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) [53], closed-form factorization [163], learnable orthogonal matrices [55, 189], and regularization losses [145, 193]. GANSpace [53] 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.*, $\mathcal{W}$ space). To avoid the extensive data sampling of GANSpace, SeFa [163] directly decomposes the model weights with a closed-form solution. Similarly, recent works [55, 189] 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 [6, 108, 101] 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 [6] proposes to utilize conditional normalizing flow (CNF) to model the mapping from conditional labels and latent codes (z space) to intermediate vectors ($\mathcal{W}$ space). StyleFlow trains the flow model (CNF) with triplets consisting of vectors sampled from $\mathcal{W}$ 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 underlining 3D world. The second type utilizes the coarse masks [223] or predicts face semantics [31] with k-means.

Additionally, some methods [94, 199, 166, 84] also explore the semantic disentanglement of the model, but they redesign the StyleGAN; thus, they need to retrain the generator. For example, TransEditor [199] 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 [166] or by adding spatial information for styles code with the conv-based module [84].

Overall, these StyleGAN-based models have demonstrated the ability to produce high-quality images and perform precise editing. However, they fail to change the facial pose and preserve view consistency due to a lack of 3D modeling abilities.

3DMM-Guided Face Generation and Editing. Recently, some works [45, 34, 184, 165, 103] demonstrate high-quality control over GAN generation via a 3DMM [144]. 3DMM is the 3D Morphable Face Model parameterized by the face shape, expression, and texture. For example, Geng *et al.* [45] 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 [184] and DiscoFaceGAN [34] use 3DMM to manipulate the latent space of StyleGAN. While StyleRig is based on pretrained 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 variable face editing, such as hair color and age.

Finally, Shi *et al.* [165] presents a LiftedGAN model, which lifts the pretrained 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 [2, 4, 152] and encoder-based [224, 151, 185, 7]. For example, Roich *et al.* [152] 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 [176, 173, 26] employ 3D-aware GAN as 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.

3.3 TT-GNeRF

We start with the introduction of the 3D-aware GAN with generative neural radiance fields (GNeRF) (Section 3.3.1). Our method can potentially work with most of GNeRF backbones and we showcase it with the two most recent ones StyleSDF [135] and EG3D [20]. Since they have similar architecture, we only describe StyleSDF [135]. Then, we detail the proposed double branches attribute editing module (DAEM) which consists of two branches, i.e., reconstruction branch and editing branch (Section 3.3.2). Each contains an Attribute Editing Module (AEM). Next, we introduce 'Training-As-Init, Optimizing-for-Tuning' (TRIOT) method with attribute-specific consistency loss to search for better latent codes that preserve non-target regions and provide more meaningful target-region editing (Section 3.3.3). Finally, we introduce a new method for the reference-based geometry editing. (Section 3.3.4)

3.3.1 Generative Neural Radiance Fields (GNeRF)

NeRF [123] is a continuous neural mapping M that maps a 3D position $\boldsymbol{x}$ and a 2D viewing direction $\boldsymbol{v}$ to the rgb color $\boldsymbol{c}$ and density σ:

$$(\boldsymbol{c}, \sigma) = \mathcal{M}(\gamma(\boldsymbol{x}), \gamma(\boldsymbol{v})), \tag{3.1}$$

where γ indicates the positional encoding mapping function.

GNeRF, such as GRAF [160] is a conditional variant of NeRF. Unlike NeRF, it requires multiple views of a single scene with estimated camera poses. Notably, GNeRF can be trained with unposed 2D images from different scenes. In Pi-GAN [19], GNeRF is trained with adversarial learning and conditioned on a latent code z:

$$(\boldsymbol{c}, \sigma) = \mathcal{M}(\gamma(\boldsymbol{x}), \gamma(\boldsymbol{v}), \boldsymbol{z}), \tag{3.2}$$

where the latent code z with following MLP layers aims to infer frequencies α and shifts β of a SIREN layer [19].

StyleSDF. As shown in Fig. 3.1, 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 $\boldsymbol{f}$, RGB color $\boldsymbol{c}$ and SDF values $\boldsymbol{d}$:

$$(\boldsymbol{f}, \boldsymbol{c}, \boldsymbol{d}) = \mathcal{M}(\gamma(\boldsymbol{x}), \gamma(\boldsymbol{v}), \boldsymbol{w}), \tag{3.3}$$

where the learned SDF values define the object surface and thus allow to extract of the mesh via Marching Cubes [114]. Moreover, $\boldsymbol{d}$ will be converted into the density σ for volume rendering.

The RGB color $\boldsymbol{c}$ 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.

In the second stage, all GNeRF parameters are fixed. The feature vector $\boldsymbol{f}$ is volume-rendered into low-resolution feature map $\boldsymbol{F}$ which is then mapped into a high-resolution result using Style-based convolutional modules. This high-resolution image is passed to another discriminator.

3.3.2 Double Branches Attribute Editing Module (DAEM)

The pre-trained StyleSDF (or EG3D) is used as the backbone of our model. Training triplets are sampled from the pretained StyleSDF model: latent vector $\boldsymbol{w}$, the corresponding generated sample $\boldsymbol{I}$ along with its low-resolution $\boldsymbol{I}_L$ version and attribute labels $\boldsymbol{A}$ predicted by the *off-the-shelf* attribute classifiers. The pretrained StyleSDF is then extended with the proposed Double-Branch Attribute Editing Module (DAEM), which manipulates the latent code for 3D-aware attribute editing. The DAEM consists of a reconstruction branch and an editing branch. Both branches share same networks, but take different labels as input to perform different optimization. We use two branches to perform both the reconstruction and editing operation to achieve a better trade-off of the editing-preservation. In detail, the network for the branches contains an Attribute Editing Module (AEM). The AEM uses Latent Transfer Unit (LTU) blocks, a variant of the Gated Recurrent Unit (GRU) [30], in order to improve the quality of the transfer.

Latent Transfer Unit (LTU). Facial editing involves balancing trade-off between attribute editing and preservation. To accurately edit the target region while preserving non-target regions, it is necessary to manipulate some components of the latent code while keeping others unchanged. Following the idea of the Gated Recurrent Unit (GRU), which has reset- and update-gates to control how much information is forgotten and updated, the Latent Transfer Unit (LTU) is specifically designed to model editing and preservation of the different latent code components.

As shown in the right of Fig. 3.1, we take the label $\boldsymbol{A}$ as a condition for our LTU and one latent code $\boldsymbol{w}_I$ as the latent input. First, we obtain the values of the gate:

$$\begin{aligned} \boldsymbol{G}_r &= \sigma(FC(FC(\boldsymbol{A}) + FC(\boldsymbol{w}_I))) \\ \boldsymbol{G}_u &= \sigma(FC(FC(\boldsymbol{A}) + FC(\boldsymbol{w}_I))), \end{aligned} \tag{3.4}$$

where $\boldsymbol{G}_r$ and $\boldsymbol{G}_u$ are reset gate and update gate, σ is the Sigmoid function, and FC is some fully-connected layers. Then, the candidate latent with new information can be defined as:

$$\tilde{\boldsymbol{w}}_I = FC(FC(\boldsymbol{A}) + FC(\boldsymbol{G}_r \odot \boldsymbol{w}_I)). \tag{3.5}$$

Table 3.1: Quantitative results on the attributes editing results using four metrics: FID, Classification Accuracy (CA), Average Matching Point (aMP), Face Recognition Similarity (FRS), and Local Preservation (LP).

Method	Expression					Gender				Age			
	FID ↓	CA ↑	aMP ↑	FRS ↑	LP ↓	FID ↓	CA ↑	aMP ↑	FRS ↑	FID ↓	CA ↑	aMP ↑	FRS ↑
DiscoFaceGAN [34]	77.84	55.00	1347.4	0.587	7.280	-	-	-	-	-	-	-	-
LiftedGAN [165]	95.25	-	1484.0	0.464	-	-	-	-	-	-	-	-	-
StyleFlow [6]	78.98	**99.24**	1089.7	0.586	19.36	82.80	79.90	1088.4	0.588	95.61	89.9	1090.8	0.586
TransEditor [199]	**55.97**	90.23	1075.6	0.564	40.23	56.60	76.73	964.30	0.575	78.32	**99.5**	959.82	0.490
TT-GNeRF (E)	64.83	93.20	1527.5	**0.852**	18.20	65.84	**86.00**	1528.6	**0.825**	63.09	79.2	1554.5	**0.864**
TT-GNeRF (S)	56.37	88.70	**1899.6**	0.812	**5.870**	**55.74**	73.40	**1825.1**	0.822	**55.43**	81.7	**1982.8**	0.850

The updated latent code can be obtained using the following equation:

$$\boldsymbol{w}_O = \boldsymbol{G}_u \odot \tilde{\boldsymbol{w}}_I + (1 - \boldsymbol{G}_u) \odot \boldsymbol{w}_I, \tag{3.6}$$

where we do not use Tanh function for $\tilde{\boldsymbol{w}}_I$ to preserve the same range of value between $\tilde{\boldsymbol{w}}_I$ and $\boldsymbol{w}_I$, compared to the typical GRU.

In the following part, we introduce the details of our reconstruction and editing branches.

Reconstruction branch. As shown in the right of Fig. 3.1, the single Attribute-Editing Module (AEM) AEM_r consists of a one-layer LTU that takes the latent code $\boldsymbol{w}$ and the label residual $\boldsymbol{A} - \boldsymbol{A} = 0$ as input, and outputs new latent vector $\boldsymbol{w}_r$:

$$\boldsymbol{w}_r = AEM_r(\boldsymbol{w}, \boldsymbol{A} - \boldsymbol{A}). \tag{3.7}$$

The learned $\boldsymbol{w}_r$ is mapped into frequencies α and shifts β by several fully-connected layers and used as input of the SIREN module. This produces a reconstructed face image $\boldsymbol{I}_r$ and normal map $\boldsymbol{I}_r^n$ using the pretrained StyleSDF. In order to estimate the normal map, we calculate the weights by the integration along the ray, then, multiply these weights with z values to obtain the depth map which is then converted into a normal map using the cross product of neighboring pixels.

Editing branch. Similar to the reconstruction branch, our editing branch AEM_e is:

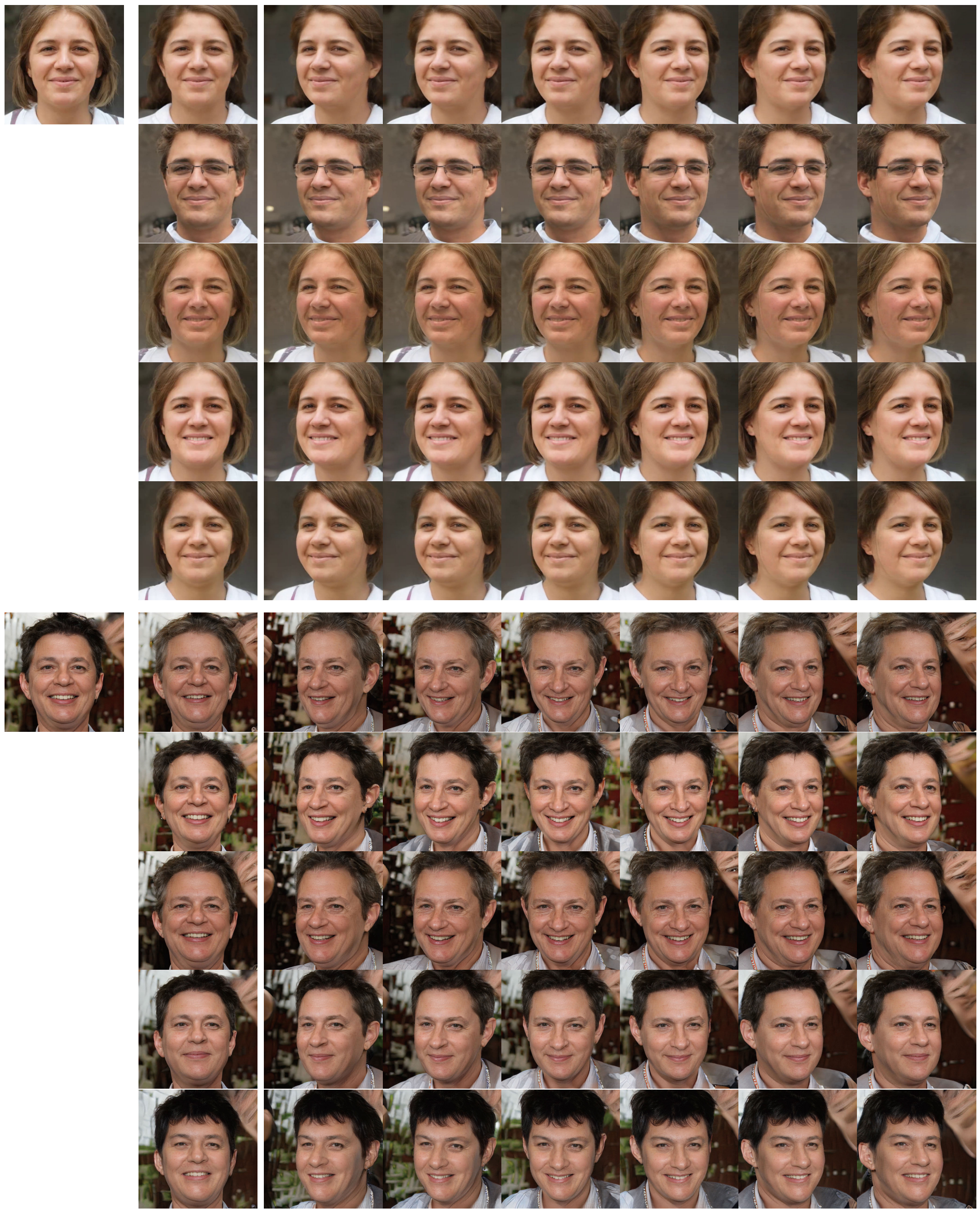

Figure 3.3: Visual results of face attribute editing and multi-view renderings from our TT-GNeRF (E) and TT-GNeRF (S). We use attributes "Hair Color", "Gender", "Age", "Expresssion", "Bangs" as example (Zoom in for best view.). TT-GNeRF (S) and TT-GNeRF (E) mean our method with different backbones: StyleSDF and EG3D, respectively.

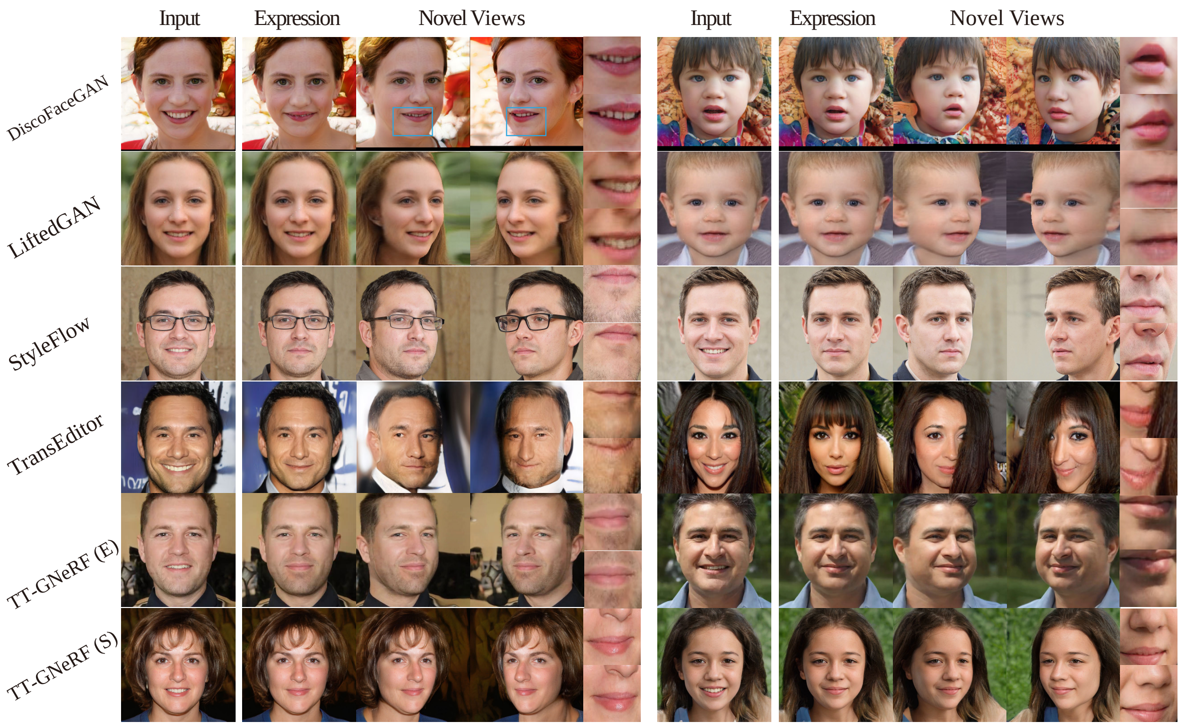

Figure 3.4: Qualitative comparisons between our method and the baselines, *i.e.*, DiscoFaceGAN [34], LiftedGAN [165], StyeFlow [6], and TransEditor [199] on "Expression" attribute and the corresponding multiple-view renderings.

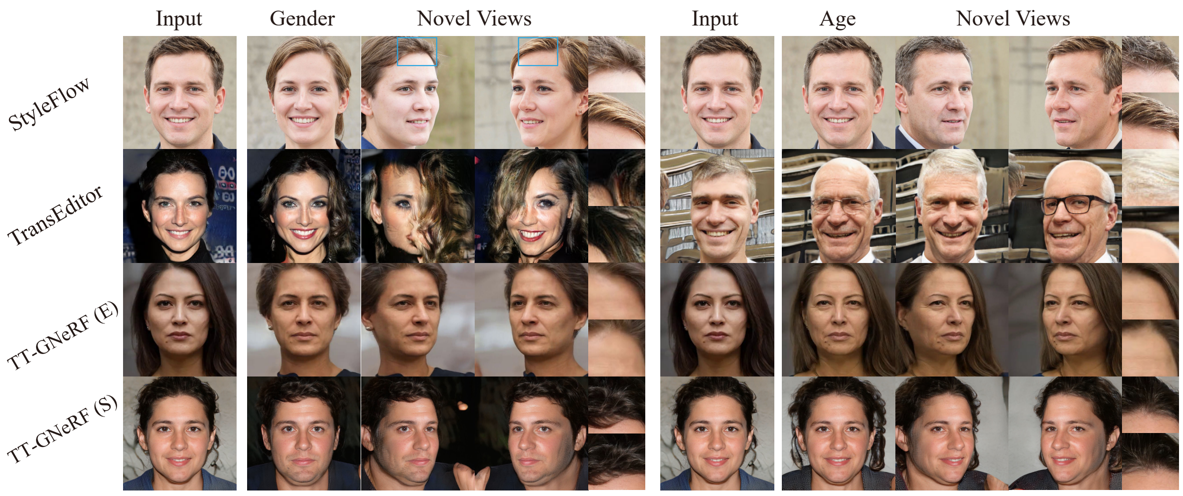

Figure 3.5: Qualitative comparisons between our method and the baselines, *i.e.*, StyeFlow [6], and TransEditor [199] on "Gender" and "Age" attributes and the corresponding multiple-view renderings. Note that DiscoFaceGAN and LiftedGAN cannot deal with these two attributes.

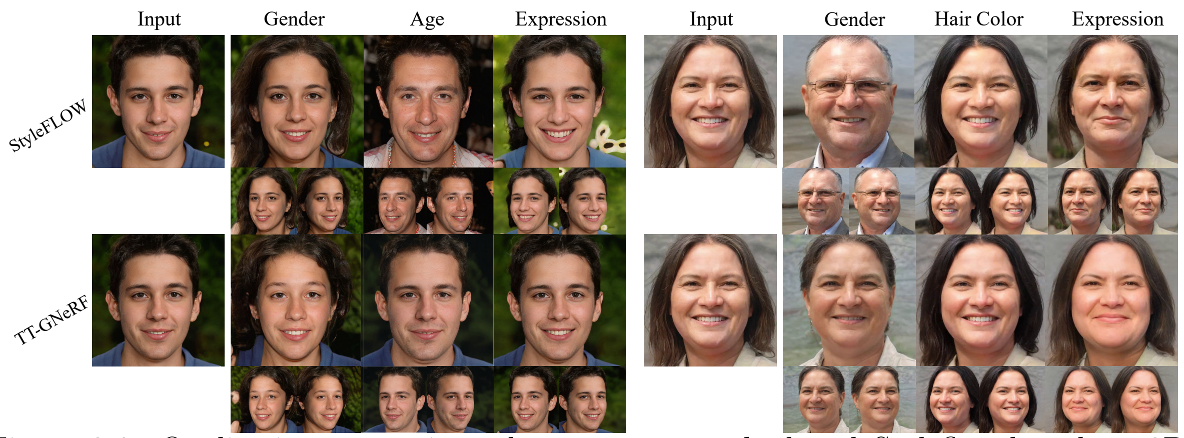

Figure 3.6: Qualitative comparisons between our method and Styleflow based on 3D-Aware model StyleSDF (Left) and EG3D (Right).

$$\boldsymbol{w}_e = AEM_e(\boldsymbol{w}, \boldsymbol{A}_e - \boldsymbol{A}), \tag{3.8}$$

where $\boldsymbol{A}_e$ is a random target label. With the pretrained StyleSDF, we can obtain an editing face image $\boldsymbol{I}_e$ and a normal map $\boldsymbol{I}_e^n$.

Loss Functions. The overall objective function for training DAEM consists of three components: adversarial loss $\mathcal{L}_{adv}$, classification loss $\mathcal{L}_{cls}$ and reconstruction loss $\mathcal{L}_{recon}$, and it is defined as follows:

$$\mathcal{L}_{DAEM} = \mathcal{L}_{adv} + \lambda_1 \mathcal{L}_{cls} + \lambda_2 \mathcal{L}_{recon}. \tag{3.9}$$

where λ_1 and λ_2 are hyper-parameters that control the contribution of the corresponding loss terms.

Unlike other 3D-aware models [220, 135], our discriminator D takes the image $\boldsymbol{I}$ and normal map $\boldsymbol{I}^n$ as input to learn consistent texture and geometry. The adversarial loss $\mathcal{L}_{adv}$ is defined as:

$$\begin{aligned} \underset{DAEM}{min}\ \underset{D}{max}\, \mathcal{L}_{adv} &= \mathbb{E}_{\boldsymbol{I},\boldsymbol{I}^n}[log D_{adv}(\boldsymbol{I}, \boldsymbol{I}^n)] \\ &+ \mathbb{E}_{\boldsymbol{I}_e,\boldsymbol{I}_e^n}[log(1 - D_{adv}(\boldsymbol{I}_e, \boldsymbol{I}_e^n))]. \end{aligned} \tag{3.10}$$

The classification loss $\mathcal{L}_{cls}$ is defined as:

$$\begin{aligned} \mathcal{L}_{cls} &= \mathbb{E}_{\boldsymbol{I},\boldsymbol{A}}[-log D_{cls}(\boldsymbol{A}|\boldsymbol{I})] \\ &+ \mathbb{E}_{\boldsymbol{I}_e,\boldsymbol{A}_e}[-log(D_{cls}(\boldsymbol{A}_e|\boldsymbol{I}_e))]. \end{aligned} \tag{3.11}$$

Finally, the reconstruction loss $\mathcal{L}_{recon}$ is defined as:

$$\mathcal{L}_{recon} = \|\boldsymbol{I}_r - \boldsymbol{I}\|_1 + \|\boldsymbol{w}_r - \boldsymbol{w}\|_1. \tag{3.12}$$

Note that D_{adv} and D_{cls} mean the adversarial branch and the classification branch, respectively. Reconstruction loss only guides the reconstructed image $\boldsymbol{I}_r$, while the adversarial and classification loss only guides the edited image $\boldsymbol{I}_e$.

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

After training, our DAEM can edit attributes to generate view consistent images with modified attributes. However, we observe that editing of some attributes can affect unrelated attributes, especially the local ones. For example, converting a "No-Smile" face into a "Smile" can modify the identity. We believe this is due to the entanglement of latent codes for some of the attributes. To alleviate this problem, we propose a novel method called 'Training-as-Init, Optimizing-for-Tuning' (TRIOT), which search for better latent codes. These latent codes preserve non-target regions and provide more meaningful target-region editing. We use an *off-the-shelf* face parsing model [205] to obtain an attribute-specific mask $\boldsymbol{M}$. Given this mask, we select the non-target region $\boldsymbol{M}$ and target region $1 - \boldsymbol{M}$. As shown in the left of Fig. 3.2, we obtain: $\boldsymbol{I}_r$, $\boldsymbol{I}_r^n$, $\boldsymbol{I}_e$ $\boldsymbol{I}_e^n$, and an initial latent code $\boldsymbol{w}_e$. Given the corresponding optimization objective function, we aim to find an optimal latent code $\hat{\boldsymbol{w}}_e$, and corresponding image $\hat{\boldsymbol{I}}_e$and normal map $\hat{\boldsymbol{I}}_e^n$.

Specifically, our objective functions are attribute-specific and consist of two parts: geometry consistency loss and texture consistency loss. As shown in the left of Fig. 3.2, our texture and geometry consistency optimization objective function is defined as follows:

$$\begin{aligned}\hat{\boldsymbol{w}}_e &= \arg\min_{\hat{\boldsymbol{w}}_e} \|\boldsymbol{M} \odot (\hat{\boldsymbol{I}}_e - \boldsymbol{I}_r)\|_1 + \|(1-\boldsymbol{M}) \odot (\hat{\boldsymbol{I}}_e - \boldsymbol{I}_e)\|_1, \\ &+ \|\boldsymbol{M} \odot (\hat{\boldsymbol{I}}^n_e - \boldsymbol{I}^n_r)\|_1 + \|(1-\boldsymbol{M}) \odot (\hat{\boldsymbol{I}}^n_e - \boldsymbol{I}^n_e)\|_1. \end{aligned} \tag{3.13}$$

This texture loss reduces differences between the reconstructed texture $\boldsymbol{I}_r$ and the optimized texture $\hat{\boldsymbol{I}}_e$ in the non-target region while keeping $\hat{\boldsymbol{I}}_e$ the same as $\boldsymbol{I}_e$ in the target region for attribute-editing. The geometry consistency loss has the same objective function as the texture loss.

By optimizing consistency losses as the objective function, we can achieve better editing results and preserve non-target regions in less than 1,000 iterations. However, we only apply this method to certain attributes where the first step does not achieve acceptable results. This allow us to balance training speed and editing quality which is not allowed for the training or optimization-only methods.

3.3.4 Reference-Based Geometry Editing

Our model can achieve reference-based geometry editing by transferring the geometry from a reference face into a target face while preserving the target face's appearance. Similar to TRIOT, the optimization objective function 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 normals and target normals (geometry loss) as well as the differences between input image and target image (texture loss) in perceptual space. To fully utilize NeRF networks' disentanglement of geometry and textures, we also optimize NeRF's RGB-branches parameters p in addition to the latent code w_e. We observe that tuning of partial networks can achieve more accurate transfer of geometry.

As shown in the right of Fig. 3.2, given the original latent code $\boldsymbol{w_e}$ and reference code $\boldsymbol{w_r}$, the corresponding face image and normal pairs are

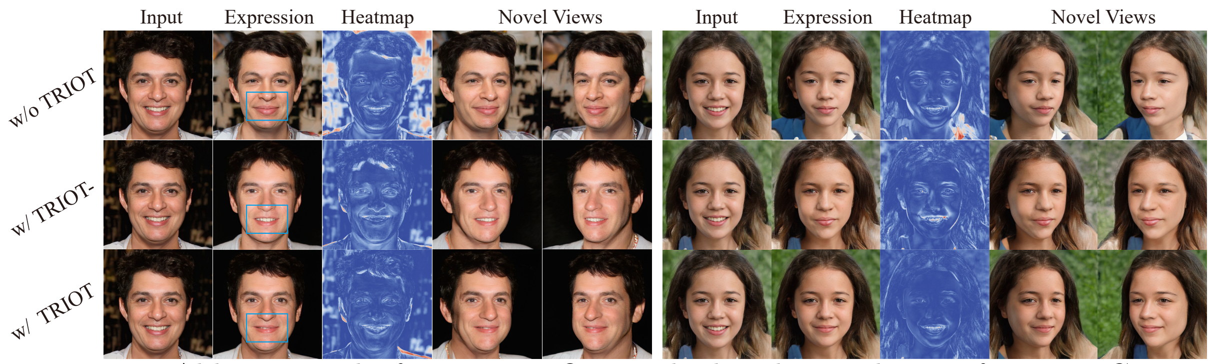

Figure 3.7: Ablation study for our TRIOT method. The results are from TT-GNeRF (S) with "Expression" as the target attribute. We visualize the differences between input and edited images using a coolworm heatmap. TRIOT- means the proposed optimization objective excludes geometry consistency loss.

($\boldsymbol{I}_e$,$\boldsymbol{I}^n_e$) ($\boldsymbol{I}_r$,$\boldsymbol{I}^n_r$), respectively. We use $\hat{\boldsymbol{w}}_{\boldsymbol{e}}$ as the target code of the optimization pipeline with corresponding face image and normal pairs ($\hat{\boldsymbol{I}}_e$,$\hat{\boldsymbol{I}}^n_e$). The optimization objective is defined as:

$$\hat{\boldsymbol{w}}_{\boldsymbol{e}}, p = \underset{(\hat{\boldsymbol{w}}_{\boldsymbol{e}},p)}{\arg\min}\, LPIPS(\boldsymbol{I}_{\boldsymbol{e}}, \hat{\boldsymbol{I}}_{\boldsymbol{e}}) + \|(\boldsymbol{I}^n_r - \hat{\boldsymbol{I}}^n_e)\|_1. \tag{3.14}$$

This optimization stage can efficiently find a latent code for accurate reference-based geometry editing within hundreds of steps.

3.4 Experiments

We name our method TT-GNeRF (**T**raining and **T**uning **G**enerative **N**eural **R**adiance **F**ield) and use TT-GNeRF (S) and TT-GNeRF (E) to refer to our method with two different backbones: StyleSDF [135] and EG3D [20], respectively.

3.4.1 Setting

Dataset. Given that the two backbones (*i.e.*, StyleSDF and EG3D) are pretrained on the FFHQ dataset [81], we train our model using sampled images and their corresponding latent codes. We use 100,000 images for

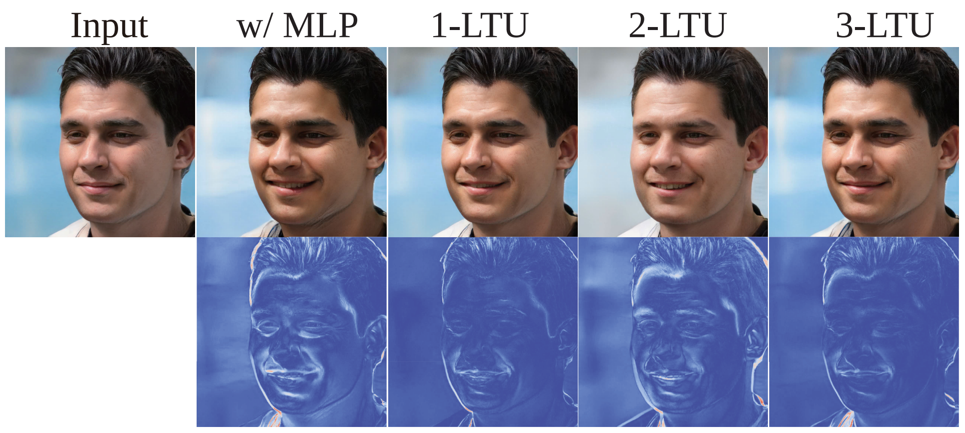

Figure 3.8: Ablation study for Attribute editing module (AEM). We visualize the "Expression" editing results corresponding to the difference heatmaps between each edited image and the input image.

training StyleSDF and 40,000 for EG3D. We employ *off-the-shelf* attribute-classifiers [82] to obtain several attribute labels, including Hair Color, Gender, Bangs, Age, Expression (Smile), and Beard. We use all generated triplets to train our models.

Implementation Details. In order to speed up the training and reduce memory consumption, we compute the reconstruction and adversarial losses using the low-resolution images rendered from RGB values $\boldsymbol{c}$. We set $\lambda_1 = 10$, $\lambda_2 = 1.0$ for the classification and reconstruction losses. DAEM is trained with the Adam optimizer [86] β_1=0.0, β_2=0.99, and a learning rate=$1e-4$ for 50,000 steps. We also use the Adam optimizer with β_1=0.9, β_2=0.99, and learning rate=$5e-4$. In this stage, we train for 1000 optimization steps. Our discriminator has an architecture similar to StyleGAN2 [82], but with an additional classification branch. To improve adversarial training stability, we adopt the non-saturating logistic loss [47] and R1 regularization [122].

Compared Baselines. Since our method is an attribute-conditioned generative model, the most similar supervised method is StyleFlow [6], which can be used for face attribute editing tasks and multiple-view generation. To ensure a fair comparison, we extend the original Styleflow, which manipulates the latent space of a 2D generator, to work with 3D-Aware generators such as EG3D and StyleSDF. Moreover, we adopt the state-of-the-art generative model, TransEditor [199] as our baseline for comparing face semantic disentanglement with multiple-view generation results. We also compare with the 3DMM-guided model, DiscoFaceGAN [34]. Note that this model can only edit some expression-related attributes, such as "Smile". Finally, we also adopt a 3D-aware LiftedGAN [165] to compare multiple-view generation. However, LiftedGAN cannot control individual attributes.

Evaluation Metrics. Five metrics are used for evaluation: FID (Fréchet Inception Distance) score [61] to evaluate the quality and diversity of edited images; Classification Accuracy (CA) to evaluate the correctness of edited attributes; average Matching points (aMP) [210] and Face Recognition Similarity (FRS) [108] 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 the edited results, we calculate the FID score [61] by using samples from FFHQ as the real distribution and the original image and its edited results as the fake distribution. We sample 5000 real and fake samples from all models for each attribute to calculate FID scores. A lower FID score indicates a lower discrepancy between the image quality of the real and generated images.

To evaluate the accuracy of the attribute transfer, we use the *off-the-shelf* classifiers [82] to classify the edited samples and compute the accuracy by comparing the predicted and the 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 indicate better accuracy.

Evaluating view consistency without ground truth is a challenging task. To address this challenge, we use proxy metrics such as the average Matched Points metric [210]. This metric involves computing a point-wise matching between two images (I_1, I_2) generated from the same identity but with different viewpoints using Patch2Pix [221], 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) [108] to evaluate identity preservation across different views. Specifically, we use ArcFace [33], a state-of-the-art 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 an *off-the-shelf* face parsing network [205] to collect the corresponding mask. We then measure the differences between each paired sample using $L1$ distance and average them across all pairs.

We found that LiftedGAN and DiscoFaceGAN are unable to directly perform editing tasks for attributes, such as “Gender” and “Age”. As a result, we do not provide scores for these methods. Instead, we provide the FID, aMP, and FRS scores for LiftedGAN using randomly generated samples rather than the attribute-edited samples.

Table 3.2: Quantitative evaluation of the editing and preservation trade-off between our method and StyleFlow, based on the backbone StyleSDF. We use two metrics: CA and LP to evaluate the editing of three attributes: Expression, Gender and Age. Note that we define the background as non-target region using semantic masks to compute LP for attribute "Gender" and "Age".

Method	Expression		Gender		Age	
	CA ↑	LP ↓	CA ↑	LP ↓	CA ↑	LP ↓
StyleFlow [6]	86.30	12.45	**78.29**	17.49	**87.60**	19.42
TT-GNeRF (S)	**88.70**	**5.870**	73.40	**6.270**	81.70	**5.820**

Table 3.3: Ablation study for our TRIOT.

Method	CA ↑	aMP ↑	FRS ↑	LP ↓
w/o TRIOT	**85.0**	1645.1	**0.869**	11.3
w/ TRIOT-	83.4	**1766.9**	0.832	10.8
w/ TRIOT	82.0	1706.3	0.854	**7.56**

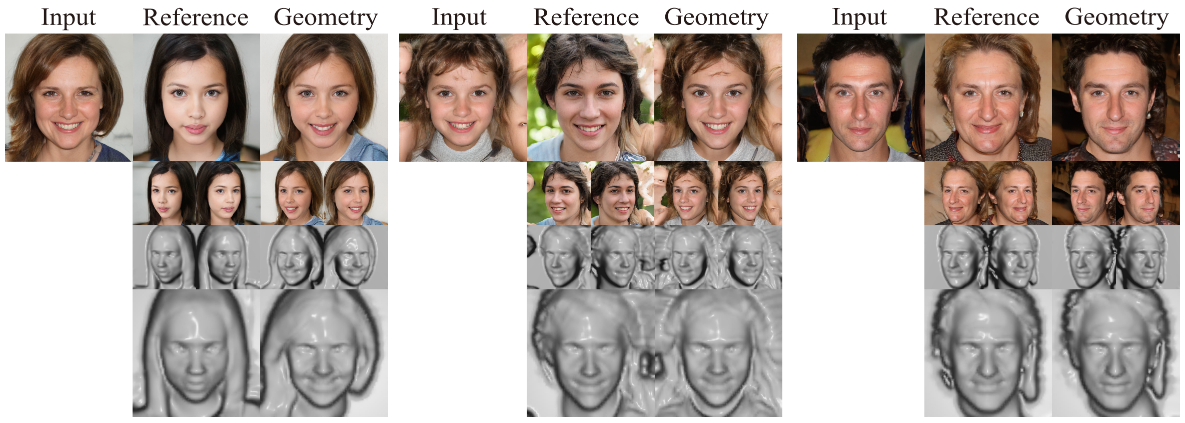

Figure 3.9: Visual results of reference-based geometry editing from TT-GNeRF (S).

3.4.2 State-of-the-Art Comparison

Fig. 3.3 shows that our method can achieve high-quality face editing results with accurate attribute transfer and preservation of non-target region . For example, the first row of Fig. 3.3 demonstrates hair-color transfer from brown to black while maintaining facial identity and expression. Ad-

Table 3.4: Ablation study for Attribute editing module (AEM). We use the attribute "Expression" for this comparison.

Method	FID ↓	CA ↑	LP ↓
w/ MLP	57.99	86.80	6.62
1-LTU	**56.37**	88.70	**5.87**
2-LTU	58.63	**93.20**	6.87
3-LTU	57.71	88.70	13.33

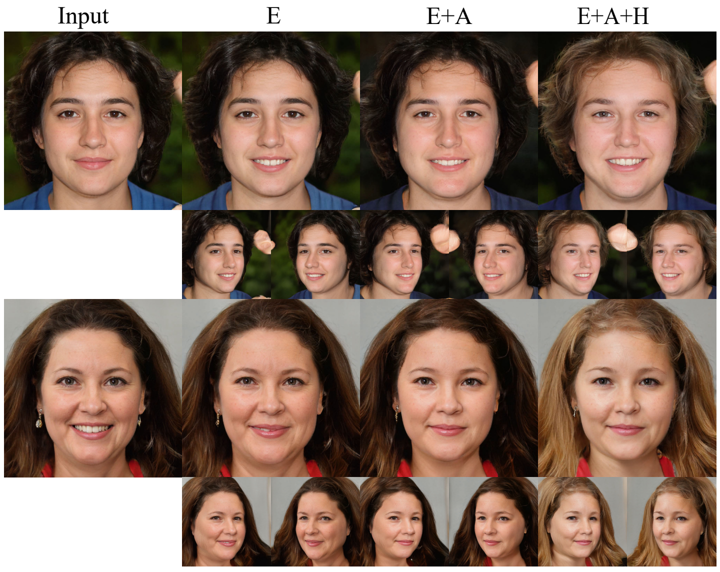

Figure 3.10: Multiple-attribute editing results from our TT-GNeRF (S) method. E: expression, A: Age, H: Hair Color.

ditionally, 3-8 columns show multiple-view generation of edited results, indicating strong 3D consistency in our results. When comparing TT-GNeRF (S) to TT-GNeRF (E), the latter tends to learn a wider variety in appearance for global attributes, such as "Gender".

We compare our method to baselines on facial attribute editing with multiple-view generation in Fig. 3.4 and Fig. 3.5. As aforementioned, we use "Expression", "Gender" and "Age" attributes as examples since most baselines can directly edit these three attributes. Fig. 3.4 shows the results for "Expression" editing. We can observe that most models can achieve accurate transfer from "Smile" to "No-Smile" while preserving non-target

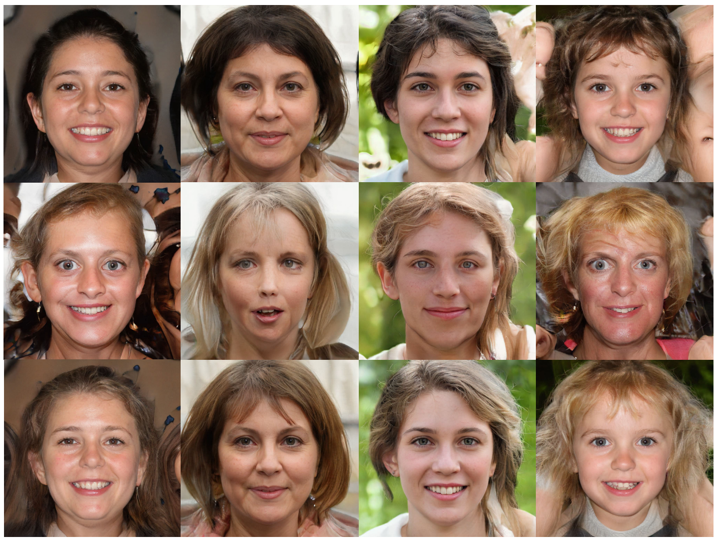

Figure 3.11: Ablation study for double-branches attribute editing branches (DAEM). We changed the attribute 'Hair Color' using four inputs (shown in the first row). The 2nd and 3rd rows show the results from DAEM *w/o* reconstruction branch and DAEM, respectively.

region well. However, TransEditor [199] fails to preserve non-target region, as shown in the 4-th rows of Fig. 3.4. 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 has improved 3D consistency but has limited quality and cannot perform facial attribute editing. Fig. 3.5 shows our method's superiority in 3D consistency compared to other methods for "Gender" and "Age" editing results; please see the zoomed-in hair region for detailed comparisons.

Table 3.1 shows the quantitative evaluation results of "Expression",

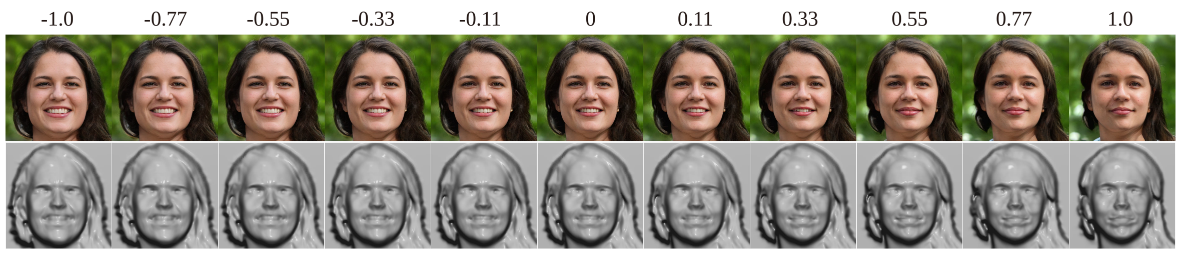

Figure 3.12: Label interpolation from -1 to 1 for face attribute editing. The results are obtained using TT-GNeRF (S) with "Expression" as the target attribute.

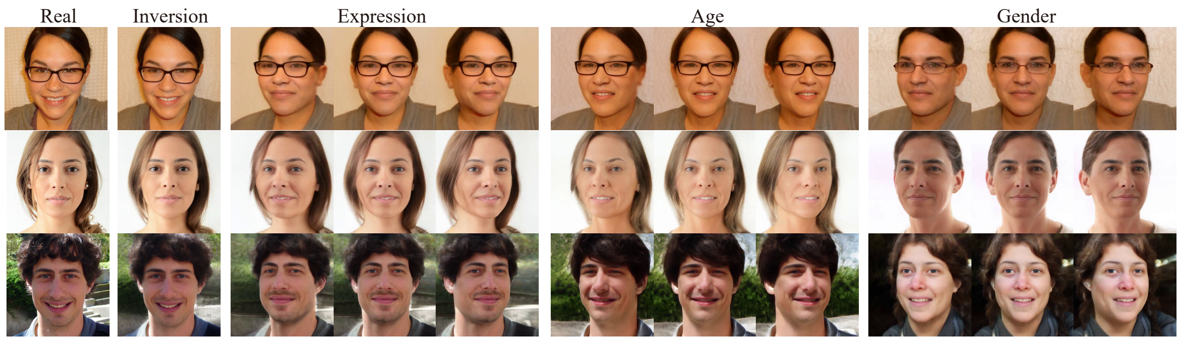

Figure 3.13: GAN Inversion for real image editing and corresponding multiple-view generation. The results are obtained using TT-GNeRF (E) with "Expression', "Age" and "Gender" as the target attributes.

"Gender", and "Age" attributes editing. Our models achieve comparable performance to baselines for all three attributes in terms of FID scores. Specifically, our TT-GNeRF (S) achieves the best FID scores for both 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 example, TT-GNeRF (E) achieves a score of 86.00 compared to 79.90 of StyleFlow and 76.73 of TransEditor for the "Gender" attribute. However, both models performs worse than the previous baselines for the "Age" attribute. We suspect this is due to 3D-Aware GAN having worse disentanglement than 2D methods, particularly for the "Age" attribute. Table 3.1 shows that our models outperform previous methods on both aMP and FRS metrics for all three attributes; specifically, our

TT-GNeRF (S) model achieves an aMP score of 1899.6 and an FRS score of 0.812 for the "Expression" attribute, better than LiftedGAN's scores of 1484.0 aMP and 0.464 FRS and DiscoFaceGAN's scores of 1347.4 aMP and 0.587 FRS.

Fig. 3.6 compares the attribute editing results between TT-GNeRF and StyleFlow using the same 3D-Aware generator. Our TT-GNeRF achieves a better trade-off between editing and preservation compared to StyleFlow. For example, in expression transfer, both models accurately edit the expression, but StyleFlow results in significant changes to the identity. This conclusion is supported by the quantitative results in Table 3.2.

3.4.3 Ablation Study

Latent Transfer Unit (LTU). In our Attribute Editing Module (AEM), we use LTU for more accurate editing of the target region. Alternatively one can use simple MLP layers for this task. To better understand the effect of LTU on performance, we compare four variants of the model: w/ MLP, 1-LTU, 2-LTU, 3-LTU. The w/ MLP variants uses a two-layer MLP instead of LTU while the 1-LTU, 2-LTU, 3-LTU variants have one, two or three LTU layers respectively. As shown in Fig. 3.8, LTU variants edit the expression of the input while better preserving non-target region than the w/ MLP variant. Table 3.4 shows that the 1-LTU variant achieves the best scores in terms of FID and LP metrics and performs better than w/ MLP in CA metric. However, using multiple LTU layers can harm model performance, especially in preserving non-target regions.

Training-As-Init, Optimizing for Tuning (TRIOT). Our proposed TRIOT can further improve the identity preservation after face attribute editing, especially for local attributes. We compare the TRIOT with two ablation baselines: w/o TRIOT, and w/ TRIOT-. The w/o TRIOT eliminates the optimization stage, and w/TRIOT- excludes the geometry consistency loss from the optimization objective. As shown in Fig. 3.7, our model can change the attribute “Expression” from “Smile” to “No-Smile” and the variant with TRIOT shows better preservation of non-target regions than the variant w/o TRIOT. This can be seen from the styles of bangs and corresponding heatmaps. Compared to w/ TRIOT-, w/TRIOT achieves a better balance between editing and preservation while w/TRIOT- suffers from inaccurate editing in geometry. Quantitative results from 100 test samples are presented in Table 3.3. The variant with TRIOT shows the best LP score, consistent with visual results in Fig. 3.7. All three variants achieve similar scores in other metrics (CA, aMP and FRS), demonstrating that our TRIOT does not harm editing ability or view consistency of the model.

Double branches Attribute Editing Module (DAEM). We discard the reconstruction branches with the corresponding loss to compare our original double branches attribute editing module (DAEM). Fig. 3.11 shows our DAEM is better at preserving the non-target attribute and regions than it without reconstruction branches.

3.4.4 Applications

Multiple-attribute Editing. In addition to the previous single-attribute edits shown in Fig. 3.3, our model can also perform sequential editing of multiple attributes. Fig. 3.10 shows high-quality edits for the sequence “Expression + Age + 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.

Reference-based Geometry Editing. Disentangling geometry and textures is not easy to achieve in previous methods. We propose a simple optimization method for reference-based geometry editing. Fig. 3.9 shows our geometry editing results, corresponding multiple-view results and meshes.

In the middle case, we can see that the face has been enlarged in size to match the geometry of the reference image while preserving appearance (e.g., identity and hair color).

Label Interpolation. We show the attribute transfer results by continuously interpolating the attribute label. We take the attribute "Expression" as an example. Fig. 3.12 shows the interpolation results corresponding to the labels ranging from -1 to 1. We can observe that the facial expression has been changed gradually from "Smile" to "No-Smile", while the identity (including the geometry) has minor changes. As mentioned above, the proposed TRIOT method can be used to alleviate this problem.

GAN Inversion for Real Image Editing. 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 with our proposed DAEM and TRIOT methods. We show the results of the PTI method for TT-GeNRF (E). PTI for TT-GNeRF (S) suffers from low-quality generation and editing. We observe that tuning step of PTI can harm the geometry of StyleSDF. In detail for TT-GNeRF (E), 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 DAEM); 2) fixing the optimized latent code and the parameters of DAEM while finetuning the remaining parameters of the generator. Afterward, we perform image editing using DAEM for the projected latent code.

Fig. 3.13 shows real image inversion and editing results for "Expression", "Gender" and "Age" attributes. The 2nd column shows that we can produce almost perfect reconstruction for the real images. After that, our model can achieve attribute editing. Moreover, the non-target region is well preserved. Finally, our model's multiple-view results also show strong 3D consistency.

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.

3.5 Conclusion

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 attributes as inputs. Moreover, we integrate the training method for the proposed DAEM and the optimization method (TRIOT) into the 3D-aware face editing task to balance the best trade-off between quality and efficiency. Our model can achieve higher-quality 3D-aware face attribute editing compared to previous methods while better preserving the 3D consistency for different view generations. The qualitative and quantitative results demonstrate the superiority of our method. Additionally, our model achieves geometry editing with the simple optimization method while preserving the appearance.

However, there still exist some limitations. First, our model fails in editing the facial attribute in some cases. For example, the age column of Table 3.1 shows that our CA score is worse than some methods. Future work could alleviate this by learning a better classifier in the discriminator. 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. 3.13, our model can achieve the single image 3D model and perform attribute editing. However, compared to video-based head avatars [42, 219], 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. In the Chapter 4, we introduce our method, 3DSGAN for 3D-aware human image editing.

Chapter 4

Exploring 3D-Aware Human Image Editing

4.1 Introduction

Recent deep generative models can generate and manipulate high-quality images. For instance, Generative Adversarial Networks (GANs) [47], have been applied to different tasks, such as image-to-image translation [227, 27, 67], portrait editing [5, 162, 185, 164], and semantic image synthesis [140], to mention a few. However, most state-of-the-art GAN models [52, 76, 80, 82, 77, 55, 78] are trained using 2D images only, operate in the 2D domain, and ignore the 3D nature of the world. Thus, they often struggle to disentangle the underlying 3D factors of the represented objects.

Recently, different 3D-aware generative models [128, 129, 195] have been proposed to solve this problem. Since most of these methods do not need 3D annotations, they can create 3D content while reducing the hardware costs of common computer graphics alternatives. Differently from generating 3D untextured shapes [195, 41], some of these methods [229, 24, 128, 102, 129] focus on 3D-aware realistic image generation and controllability. Generally speaking, these models mimic the traditional computer graphics rendering pipeline: they first model the 3D structure, then they use a (differentiable) projection module to project the 3D structure into 2D images. The latter may be a depth map [24], a sketch [229] or a feature map [128] which is finally mapped into the real image by a rendering module. During training, some methods require 3D data [229, 24], and some [128, 102, 129] can learn a 3D representation directly from raw images.

An important class of *implicit* 3D representations are the Neural Radiance Fields (NeRFs), which can generate high-quality unseen views of complex scenes [123, 72, 36, 147, 146, 148, 19]. Generative NeRFs (GNeRFs) combine NeRFs with GANs in order to condition the generation process with a latent code governing the object's appearance or shape [160, 19, 132]. However, these methods [160, 19, 132, 137, 220] focus on relatively simple and "rigid" objects, such as cars and faces, and they usually struggle to generate highly non-rigid objects such as the human body (*e.g.*, see Fig. 4.1). This is likely due to the fact that the human body appearance is highly variable because of both its articulated poses and the variability of the clothes texture, being these two factors entangled with each other. Thus, adversarially learning the data distribution modeling all these factors, is a hard task, especially when the training set is relatively small.

To mitigate this problem, we propose to *split* the human generation process in two separate steps and use intermediate segmentation masks as the bridge of these two stages. Specifically, our 3D-aware Semantic-Guided Generative model (3D-SGAN) is composed of two generators: a GNeRF model and a texture generator. The GNeRF model learns the 3D structure of the human body and generates a semantic segmentation of the main body components, which is largely invariant to the surface texture. The texture generator translates the previous segmentation output into a photo-realistic image. To control the texture style, a Variational AutoEncoder (VAE [89]) approach with a StyleGAN-like [82] decoder is used to modulate the final decoding process. The similar idea has been used in [97], but their semantic generator is 2D, and it cannot perform 3D manipulations. We empirically show that splitting the human generation process into these two stages brings the following three advantages. First, the GNeRF model is able to learn the intrinsic 3D geometry of the human body, even when trained with a small dataset. Second, the texture generator can successfully translate semantic information into a textured object. Third, both generators can be controlled by explicitly varying their respective conditioning latent codes. Moreover, we propose two consistency losses to further disentangle the latent codes representing the garment type (which we call the "semantic" code) and the human pose. Finally, since there is no general metric which can be used to evaluate the 3D consistency of image generation with multiple viewpoints, we propose a point matching-based metric which we name *average Matched Points (aMP)*. Experiments conducted on the DeepFashion dataset [110] show that 3D-SGAN can generate high-quality person images significantly outperforming state-of-the-art approaches. In summary, the main contributions of this work are:

1) We propose 3D-SGAN, which combines a GNeRF with a VAE-conditioned texture generator for human-image synthesis. And our code has been released at `https://github.com/zhangqianhui/3DSGAN`

2) We propose two consistency losses to increase the disentanglement between semantic information (*e.g.*, garment type) and the human pose.

3) We show that 3D-SGAN generates high-quality human images, significantly outperforming the previous controllable state-of-the-art methods.

4) We propose a new metric (aMP) to evaluate the 3D-view consistency.

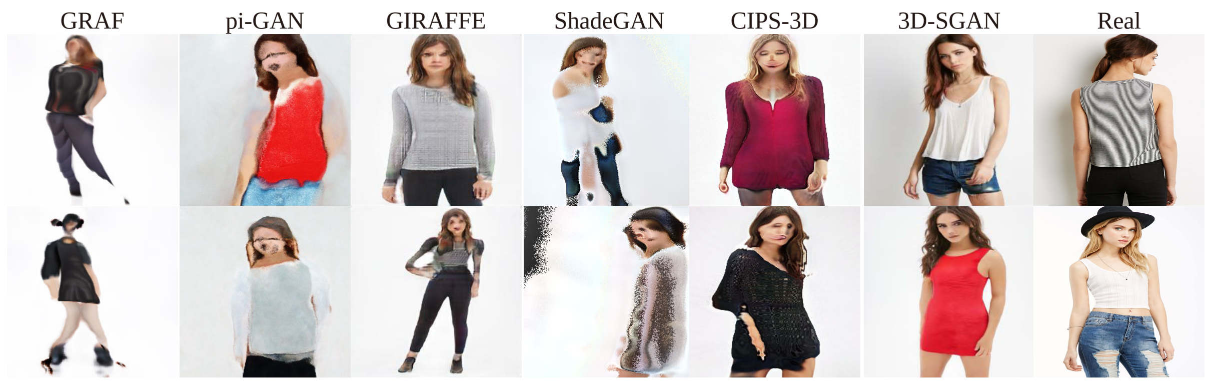

Figure 4.1: A qualitative comparison between different generation methods: GRAF [160], pi-GAN [19], GIRAFFE [132], ShadeGAN [137] CIPS-3D [220], and 3D-SGAN (Ours).

4.2 Background

3D-aware image synthesis is based on generative models which incorporate a 3D scene representation. This allows rendering photo-realistic images from different viewpoints. Early methods use GAN-based architectures for building 3D voxel [195, 41, 115, 60] or mesh [150, 58] representations. However, they mostly focus on learning untextured 3D structures. More recently, different methods learn textured representations directly from 2D images [190, 229, 128, 160, 36, 131, 132]. The resulting controllable 3D scene representation can be used for image synthesis. Some of these methods [229, 24] require extra 3D data for disentangling shape from texture. The main idea is to generate an internal 3D shape and then project this shape into 2D sketches [229] or depth maps [24], which are finally rendered in a realistic image. Other methods are directly trained on 2D images without using 3D data [128, 160, 129, 132, 198, 175]. For instance, inspired by StyleGAN2 [82], Thu et al. [128] propose HoloGAN, which predicts 3D abstract features using 3D convolutions, and then projects these features into a 2D representation which is finally decoded into an image. However, the learnable projection function, *e.g.*, the decoder, results in an entangled representation, thus the view-consistency of the generated images is degraded. Katja et al. [160] use a NeRF to represent the 3D scene and a volume rendering technique to render the final image. However, this model works at relatively low image resolutions and it is restricted to single-object scenes. To tackle these issues, some works propose object-aware scene representations. For example, Liao et al. [102] combine a 2D generator and a projection module with a 3D generator which outputs multiple abstract 3D primitives. Every stage in this model outputs multiple terms to separately represent each object. Instead of abstract 3D primitives, Phuoc et al. [129] use a voxel feature grid as the 3D representation, but their method fails to generate consistent images at high-resolution. Michael et al. [132] recently introduced GIRAFFE, a multiple-object scene representation based on NeRFs, jointly with an object composition operator. GIRAFFE is the state-of-the-art 3D-aware approach for both single and multiple object generation tasks. A few very recent papers [137, 178] propose to learn an accurate object geometry by introducing a relighting module into the rendering process. Xu et al. [200], explicitly learn a structural and a textural representation (a feature volume), which is used jointly with the implicit NeRF mechanism. Chan et al. [20], propose to replace the 3D volume with three projection feature planes. Finally, 3D-consistency is addressed in [50, 220, 135], where, *e.g.*, StyleGAN-based networks are used for neural rendering. However, most of these works fail to disentangle the semantic attributes.

While previous methods can achieve impressive rigid-object generation and manipulation results, they usually struggle to deal with non-rigid objects with complex pose and texture variations. For instance, the human body is a non-rigid object which is very important in many generative applications.

GANs for human generation. GANs [47] have been widely used for different object categories, for instance, faces [79, 76, 80, 16], cars [79, 80,

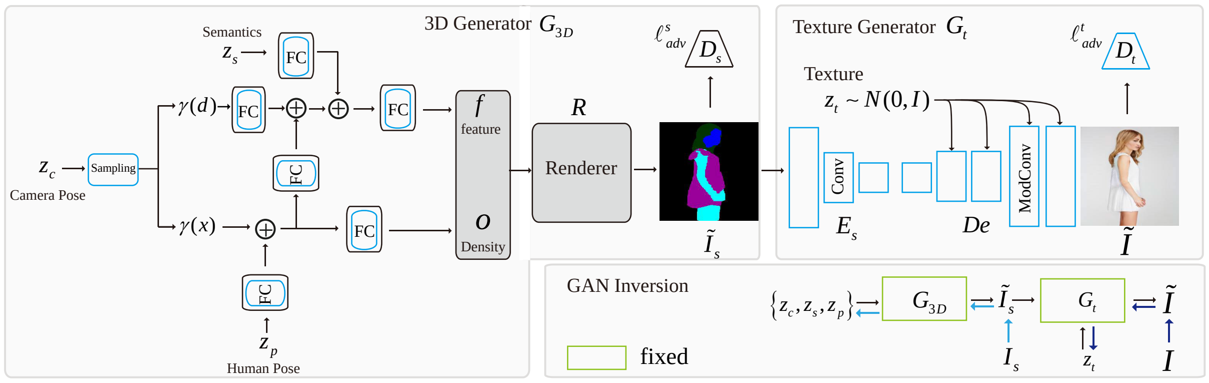

Figure 4.2: An overview of the proposed 3D-SGAN architecture, composed of two main generators. G_{3D} (on the left) follows a GNeRF structure, with a NeRF kernel used to represent implicit 3D information, latent codes governing different appearance variations and a discriminator (D_s) which is used for adversarial training. The output of G_{3D} is the semantic masks $\tilde{I}_s$ (middle). The second generator (G_t, right) translates the semantic masks into a photo-realistic image $\tilde{I}$. Also G_t is trained adversarially (see top right, the second discriminator D_t). The human generation process can be controlled by interpolating different latent codes: the semantics $\boldsymbol{z}_s$, the pose $\boldsymbol{z}_p$, the camera $\boldsymbol{z}_c$, and the texture code $\boldsymbol{z}_t$. The bottom of the figure shows the GAN inversion scheme.

160, 129], and churches [132]. However, GANs still struggle to produce high-quality full-human body images, because of the complex pose variations. Very recently, Sarkar et al. [158] proposed a VAE-GAN model for the pose transfer and the part sampling tasks. In more detail, this model extracts an UV texture map from the input image using DensePose [8], and then encodes the texture into a Gaussian distribution. Then, it samples from this distribution and warps the sample into the target pose space. Finally, the warped latent code is used as input to the decoder. Compared to Sarkar et al. [158], our method does not use an SMPL nor DensePose to extract the point correspondences as additional supervised information. Despite that, 3D-SGAN can learn 3D representations of the human body and control the generation process (*e.g.*, by changing the input camera parameters).

StylePeople [49] is based on a full-body human avatar, which combines

StyleGANv2 [82] with *neural dressing*. The StyleGANv2 module samples neural textures, and these textures are superimposed on the meshes of an SMPL. The textured meshes are finally rendered into an image. In contrast, our 3D-SGAN can perform semantic disentanglement and manipulation using semantic codes.

Pose transfer aims to synthesize person images in a novel view or in a new pose. This is a very challenging task, since it requires very complicated spatial transformations to account for different poses. Most works in this field can be categorized by the way in which the human pose is represented. Early works are based on keypoints [118, 168, 231, 149, 180, 119, 121, 66, 10, 116, 74, 169, 222, 155, 172, 204, 156]. More recent methods [48, 126, 159, 157, 107] use correspondences between pixel location in 2D images and points in SMPL [112] (usually estimated using DensePose [51]). However, these approaches usually struggle to simultaneously provide a realistic and a 3D controllable person generation.

4.3 Preliminaries

NeRF [123] is an implicit model which represents a 3D scene using the weights of a multilayer perceptron (MLP). This MLP (h) takes as input a 3D coordinate $\boldsymbol{x} \in \mathbb{R}^3$ and a view direction $\boldsymbol{d} \in \mathbb{R}^2$, and outputs the density (or "opacity", o) and the view-dependent RGB color value $\boldsymbol{c}$:

$$(\boldsymbol{c}, o) = h(\gamma(\boldsymbol{x}), \gamma(\boldsymbol{d})), \tag{4.1}$$

where γ is a positional encoding function [187]. On the other hand, Generative NeRF (GNeRF) [160] is a NeRF conditioned on the latent codes $\boldsymbol{z}_g$ and $\boldsymbol{z}_a$, respectively representing the geometric shape and the object appearance, and drawn from a priori distributions. GNeRFs [160, 132] are trained using an adversarial approach. In GIRAFFE [132], the color value

($\boldsymbol{c}$ in Eq. 4.1) is replaced by an intermediate feature vector $\boldsymbol{f}$:

$$(\boldsymbol{f}, o) = h(\gamma(\boldsymbol{x}), \gamma(\boldsymbol{d}), \boldsymbol{z}_g, \boldsymbol{z}_a). \tag{4.2}$$

$\boldsymbol{f}$ is mapped into a photo-realistic image using a volume and neural rendering module R and fed to a discriminator (more details in [160, 132]).

Our 3D generator (see Sec. 4.4) is inspired by GIRAFFE [132]. However, it learns to produce a segmentation image, a simpler task with respect to directly generating a photo-realistic image.

4.4 3D-SGAN

Fig. 4.2 shows the proposed 3D-SGAN architecture, composed of two main modules: a 3D-based segmentation mask generator and a texture generator. The former (G_{3D}) generates semantic segmentation masks of the human body which correspond to the main body parts and depend on the type of clothes, the camera viewpoint and the human pose. On the other hand, the texture generator (G_t) takes as input these segmentation masks and translates them into a photo-realistic image, adding a texture style randomly drawn from a pre-learned marginal distribution. The two modules are trained separately.

4.4.1 3D Generator for Semantic Mask Rendering

Given a set of 2D human image samples $\{I^i\}_{i=1}^N$, we first use an off-the-shelf human parsing tool [9] to obtain the corresponding ground-truth semantic segmentation masks $\{I_s^i\}_{i=1}^N$. Using $T = \{(I^i, I_s^i)\}_{i=1}^N$ as our training set, the goal is to train a two-step generative model:

$$\tilde{I} = G(\boldsymbol{z}_c, \boldsymbol{z}_s, \boldsymbol{z}_p, \boldsymbol{z}_t) = G_t(G_{3D}(\boldsymbol{z}_c, \boldsymbol{z}_s, \boldsymbol{z}_p), \boldsymbol{z}_t), \tag{4.3}$$

where $\tilde{I}$ is the final generated image. The latent codes $\boldsymbol{z}_c \sim P_c$ (see Sec.4.5), $\boldsymbol{z}_s \sim \mathcal{N}(0, \boldsymbol{I})$, $\boldsymbol{z}_p \sim \mathcal{N}(0, \boldsymbol{I})$, and $\boldsymbol{z}_t \sim \mathcal{N}(0, \boldsymbol{I})$ represent, respectively: the camera viewpoint, the semantics (*i.e.*, the garment type), the body pose and the human texture.

The structure of our 3D Generator G_{3D} is inspired by GIRAFFE [132] (Sec. 4.3). However, differently from [160, 132], which learn to generate a textured object, in our case, h learns to generate a semantically segmented image. Specifically, we use a latent semantic code ($\boldsymbol{z}_s$) to condition the final segmentation output on the type of garment. As shown in Fig. 4.2, $\boldsymbol{z}_s$ does not influence the opacity generation branch, and it is injected into the direction-dependent branch, which finally outputs a feature vector $\boldsymbol{f}$, representing a point-wise semantic content. Formally, we have:

$$(\boldsymbol{f}, o) = h(\gamma(\boldsymbol{x}), \gamma(\boldsymbol{d}), \boldsymbol{z}_c, \boldsymbol{z}_s, \boldsymbol{z}_p). \tag{4.4}$$

Following [160, 132], we generate a set of pairs $\{(\boldsymbol{f}, o)\}$ which are finally projected into the 2D plane using a rendering module R [123, 132] (see section 4.3), and represented by the segmentation masks $\tilde{I}_s$. Specifically, $\tilde{I}_s$ is a tensor composed of n_s channels, where each channel represents a segmentation mask of the same spatial resolution of the real images in T (Fig. 4.2).

G_{3D} is trained jointly with a discriminator D_s, which learns to discriminate between real (I_s) and fake ($\tilde{I}_s$) segmentation masks (more details in Sec. 4.4.4).

4.4.2 VAE-Conditioned Texture Generator

The goal of our texture generator G_t is twofold: (1) mapping the segmentation masks $\tilde{I}_s$ generated by G_{3D} into a textured human image and (2) learning a marginal distribution of the human texture using the dataset T. The latter is obtained using a Variational AutoEncoder (VAE [89]) framework, which we use to learn how to *modulate* the texture style of the decoder. Specifically, as shown in Fig. 4.3, G_t is composed of a semantic encoder E_s, a texture encoder E_t, and a decoder De. De is based on a StyleGANv2 architecture [82], in which a style code is used to "demodulate" the weights of each convolutional layer. We modify this architecture using a variational approach, in which the style code, *at inference time*, is extracted from a learned marginal distribution. In more detail, given a segmentation tensor I_s, we use E_s to extract the semantic content which is decoded into the final image using De and a texture code $\boldsymbol{z}_t$. The latter is sampled using the VAE encoder E_t, which converts a real input image I into a latent-space normal distribution ($\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$), from which $\boldsymbol{z}_t$ is randomly chosen:

$$(\boldsymbol{\mu}, \boldsymbol{\sigma}) = E_t(I), \boldsymbol{z}_t \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}), \tilde{I} = De(E_s(I_s), \boldsymbol{z}_t). \tag{4.5}$$

G_t is trained using the pairs in T. Specifically, given a pair of samples (I^i, I^i_s), we use an adversarial loss $\mathcal{L}^t_{adv}$ (and a dedicated discriminator D_t), jointly with a reconstruction loss $\mathcal{L}_r$, and a standard Kullback-Leibler divergence ($\mathcal{D}_{kl}$) loss ($\mathcal{L}_{kl}$) [89]:

$$\mathcal{L}_r = ||G_t(I^i_s, \boldsymbol{z}_t) - I^i||_1, \ell_{kl} = \mathcal{D}_{kl}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})||\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})). \tag{4.6}$$

Note that, in the reconstruction process, G_t cannot ignore the segmentation tensor (I^i_s) and its corresponding encoder E_s. In fact, the information extracted from the real image I, and encoded using E_t, is not enough for the decoder to represent the image content, since $\boldsymbol{z}_t$ is used only as a style modulator in De.

4.4.3 Consistency Losses for Semantics and Pose Disentanglement

In G_{3D}, the opacity value (o), computed by h, does not depend on the latent code $\boldsymbol{z}_s$. Despite that, we have empirically observed that the semantics ($\boldsymbol{z}_s$) and the pose ($\boldsymbol{z}_p$) representations are highly entangled. We presume this

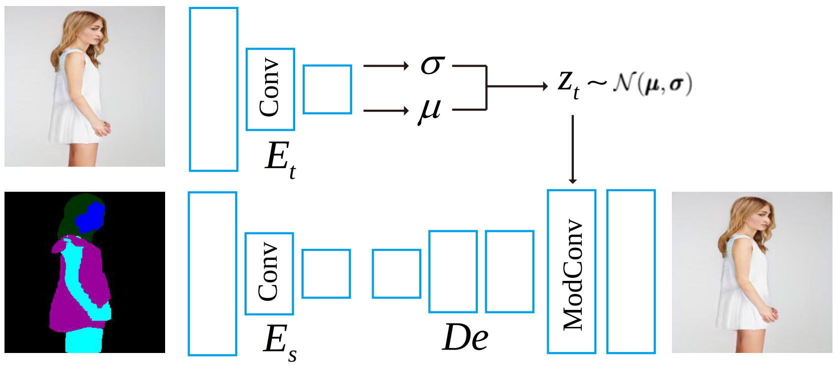

Figure 4.3: The proposed VAE-conditioned texture generator. ModConv stands for “Modulated Convolution” [82].

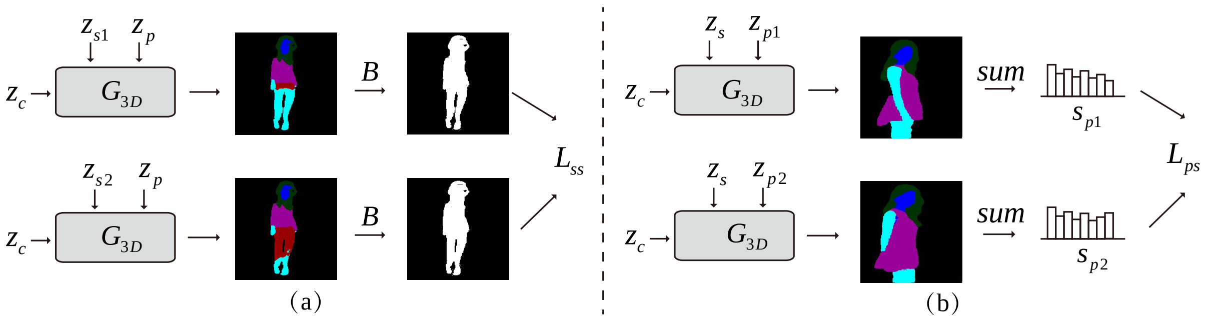

Figure 4.4: A schematic representation of our consistency losses ℓ_{ss} (a) and ℓ_{ps} (b).

is due to the convolutional filters in R (section 4.3), where the two latent factors are implicitly merged. In order to increase the disentanglement of these factors, we propose two self-supervised consistency losses.

Silhouette-based geometric consistency. This loss is based on the idea that two different body segmentations (*e.g.*, long-sleeve vs. short-sleeve, etc.), produced using two different semantic codes $\boldsymbol{z}_{s1}$ and $\boldsymbol{z}_{s2}$, *but keeping fixed the pose and the camera codes*, once they are binarized, should correspond to roughly the same silhouette (see Fig. 4.4 (a)). Formally, the proposed geometric consistency loss $\mathcal{L}_{ss}$ is defined as:

$$\mathcal{L}_{ss} = \|B(G_{3D}(\boldsymbol{z}_c, \boldsymbol{z}_{s1}, \boldsymbol{z}_p)) - B(G_{3D}(\boldsymbol{z}_c, \boldsymbol{z}_{s2}, \boldsymbol{z}_p))\|_1, \quad (4.7)$$

where $B(\tilde{I}_s)$ maps the segmentation masks $\tilde{I}_s$ into a binary silhouette image.

Pose-based semantic consistency. Analogously to $\mathcal{L}_{ss}$, the proposed pose-based semantic consistency loss is based on the idea that two different pose codes should produce a similar body segmentation. However, as shown in Fig. 4.4 (b), despite the body being partitioned in similar semantic segments (*e.g.*, because the clothes have not changed), when the human pose changes, the overall spatial layout of these segments can also change (*e.g.*, see the two different arm positions in Fig. 4.4 (b)). For this reason, we formulate a semantic consistency loss ($\mathcal{L}_{ps}$) which is spatial-invariant, and it is based on the channel-by-channel comparison of two segmentation masks. In more detail, given two different pose codes $\boldsymbol{z}_{p1}$ and $\boldsymbol{z}_{p2}$, and fixing the semantics and the camera code, we first produce two corresponding segmentation tensors $\tilde{I}_s^1$ and $\tilde{I}_s^2$. Then, for each tensor and each channel, we sum all the channel-specific mask values over the spatial dimension and we get two spatial invariant vectors $\boldsymbol{s}_{p1}, \boldsymbol{s}_{p2} \in \mathbb{R}^{n_s}$. Finally, $\mathcal{L}_{ps}$ is given by:

$$\mathcal{L}_{ps} = \sum\nolimits_{i=1}^{n_s} [\max(\frac{\mid \boldsymbol{s}_{p1}[i] - \boldsymbol{s}_{p2}[i] \mid}{\boldsymbol{s}_{p1}[i] + \epsilon}, \rho) - \rho], \tag{4.8}$$

where $\boldsymbol{s}[i]$ is the i-th channel value of vector $\boldsymbol{s}$, ϵ is a small value used for numerical stability, and ρ is a margin representing the tolerable channel-wise difference.

4.4.4 Training and inference

G_{3D} is trained using an adversarial loss ($\mathcal{L}^s_{adv}$) jointly with $\mathcal{L}_{ss}$ and $\mathcal{L}_{ps}$ (Sec. 4.4.3):

$$\mathcal{L}_{3D} = \mathcal{L}^s_{adv} + \lambda_1 \mathcal{L}_{ss} + \lambda_2 \mathcal{L}_{ps}, \tag{4.9}$$

where λ_1 and λ_2 are hyper-parameters controlling the contribution of each loss term.

G_t is trained using a variational-adversarial approach (VAE-GAN [96]):

$$\mathcal{L}_{tr} = \mathcal{L}^t_{adv} + \lambda_3 \mathcal{L}_r + \lambda_4 \mathcal{L}_{kl}, \tag{4.10}$$

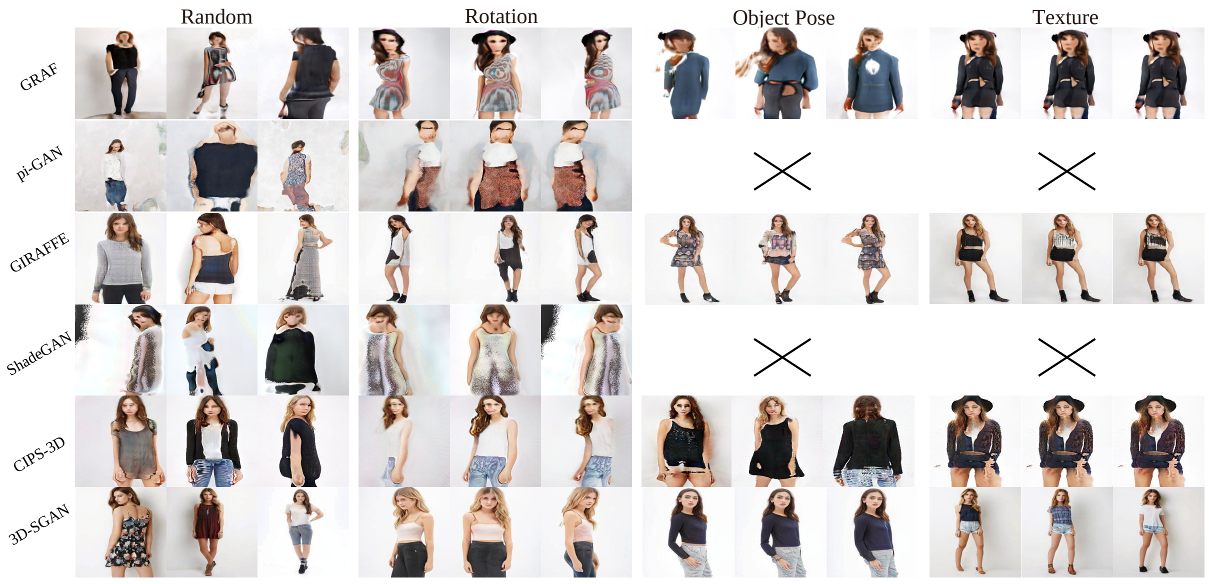

Figure 4.5: A qualitative comparison. 'Random' means that the results are generated by randomly sampling the latent codes from the corresponding learned marginal distributions. The other 3 columns show controllable person generations with respect to the rotation, the human pose, and the texture attribute. The lack of the 'Object Pose' and the 'Texture' results for both pi-GAN and ShadeGAN is due to the fact that both methods use a single latent code to model both the texture and the geometry.

where λ_3 and λ_4 are hyper-parameters, and $\mathcal{L}_{tr}$ is the overall objective function of G_t.

G_{3D} and G_t are trained separately. However, at inference time, the tensor $\tilde{I}_s$, generated by G_{3D}, is fed to G_t, along with a texture code $\boldsymbol{z}_t$, randomly drawn from a standard normal distribution:

$$\boldsymbol{z}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tilde{I} = De(E_s(\tilde{I}_s), \boldsymbol{z}_t). \qquad (4.11)$$

4.4.5 Real Image Editing using GAN Inversion

The variational method proposed in section 4.4.2 cannot completely reconstruct the input image. For real image editing, we use a GAN inversion technique [3] to optimize the values of the latent codes corresponding to a real input image I. Since we have two separate generators (G_{3D}

Table 4.1: A quantitative comparison using the FID (↓) and the aMP (↑) scores.

Method	FID ↓				aMP ↑
	Random	Rotation	Object-Pose	Texture	Rotation
GRAF [160]	52.68	176.9	57.76	220.9	64.0
pi-GAN [19]	137.6	213.7	-	-	58.0
GIRAFFE [132]	42.73	123.4	82.61	98.41	51.0
ShadeGAN [137]	134.7	232.4	-	-	**89.0**
CIPS-3D [220]	69.45	156.9	233.6	**36.16**	60.0
3D-SGAN	**8.240**	**117.3**	**54.00**	60.63	81.0

and G_t), the optimization process is based on two steps (see Fig. 4.2, bottom). Specifically, given a pair of real image and its corresponding segmentation masks (extracted using [9], see Sec. 4.4.1) (I, I_s), we first generate $\tilde{I}_s = G_{3D}(\boldsymbol{z}_c, \boldsymbol{z}_s, \boldsymbol{z}_p)$ and we optimize $||\tilde{I}_s - I_s||_1$ with respect to $\boldsymbol{z}_c, \boldsymbol{z}_s$ and $\boldsymbol{z}_p$. Let $\boldsymbol{z}_c^*, \boldsymbol{z}_s^*$ and $\boldsymbol{z}_p^*$ be the optimal values so found, and let $\tilde{I}_s^* = G_{3D}(\boldsymbol{z}_c^*, \boldsymbol{z}_s^*, \boldsymbol{z}_p^*)$. Then, we use $\tilde{I} = G_t(\tilde{I}_s^*, \boldsymbol{z}_t)$ and we optimize $||\tilde{I} - I||_1 + \tau LPIPS(\tilde{I}, I)$ with respect to $\boldsymbol{z}_t$, where $LPIPS(I^1, I^2)$ is the $LPIPS$ distance between two images [214] and we use $\tau = 10$.

Once obtained the latent codes $(\boldsymbol{z}_c^*, \boldsymbol{z}_s^*, \boldsymbol{z}_p^*, \boldsymbol{z}_t^*)$ corresponding to a real image, editing can be easily done by changing these codes.

4.5 Experiments

Datasets. We use the DeepFashion In-shop Clothes Retrieval benchmark [110], which consists of 52,712 high-resolution (1101×750 pixels) person images with various appearances and poses. This dataset has been widely used in pose transfer tasks. We use the following preprocessing. First, we remove overly cropped images, such as incomplete images of humans. Then, the remaining 42,977 images are resized into a 256×256 resolution, and are divided into 41,001 training and 1,976 testing images.

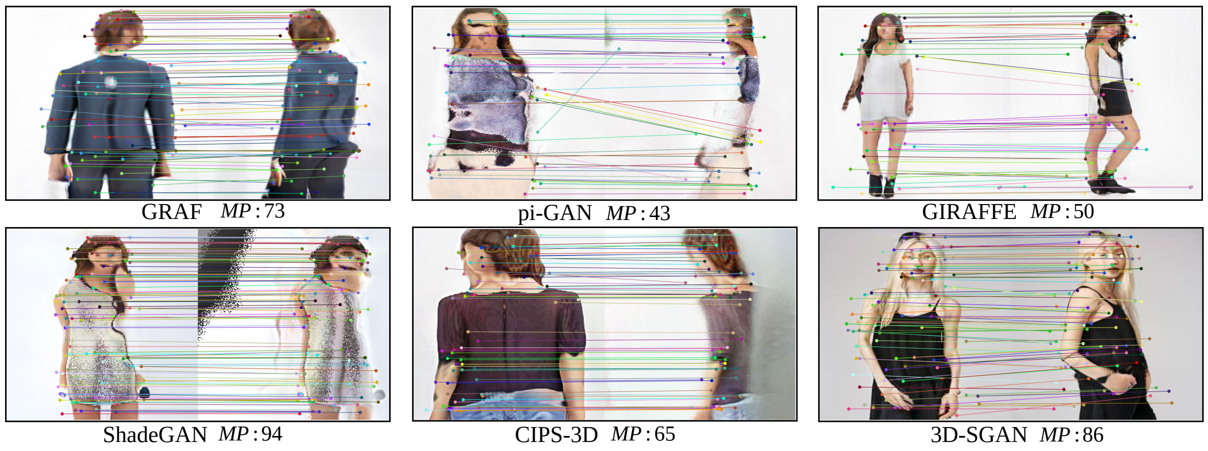

Figure 4.6: Computing MP between pairs of generated images with 2 different viewpoints.

Training details. Following GIRAFFE [132], the camera distribution P_c can be implemented by first sampling the camera code from a uniform distribution over the dataset-dependent camera elevation angles, and then applying an object affine transformation to sample 3D points and rays. Both G_t and G_{3D} are trained using the RMSprop optimizer [88]. The learning rate for both the discriminator and the generator is set to 10^{-4}. For the loss weights, we use: λ_1=0.01, λ_2=0.01, λ_3=1, and λ_4=1. For GAN inversion, we use the Adam optimizer [87] with a learning rate of 10^{-2}.

Baselines. We compare 3D-SGAN with five state-of-the-art 3D-aware generative approaches, *i.e.*, GRAF [160], pi-GAN [19], GIRAFFE [132], ShadeGAN [137] and CIPS-3D [220]. For each baseline, we use the corresponding publicly available code with a few minor adaptations for the DeepFashion dataset. Note that some concurrent methods, such as StyleNeRF [50], GRAM [35], Tri-plane [20], StyleSDF [136] achieve a performance very similar to CIPS-3D. Moreover, for some of them there is no released code yet, thus, a direct comparison is not possible.

Metrics. We adopt the widely used FID [61] scores to evaluate the quality of the generated human images, following common protocols (*e.g.*, using 5,000 fake samples, etc.). And we propose the average Matched Points

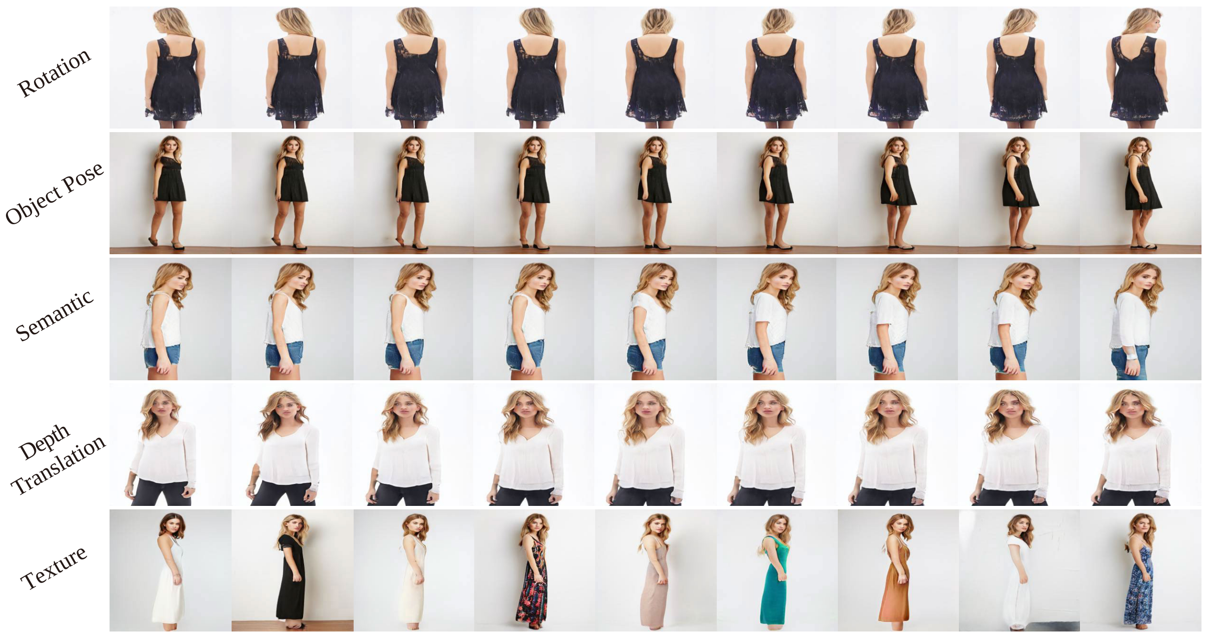

Figure 4.7: Controllable person generation by interpolating latent codes (Rows 1-4). The fifth row shows texture generation results obtained randomly sampling $\boldsymbol{z}_t$.

(aMP) to evaluate the 3D-view consistency of the generated images.

4.5.1 Comparisons with State-Of-the-Art Methods

Unconditioned human generation. Fig. 4.5 ("Random" column) shows a qualitative comparison between image samples generated by all the models. GRAF [160], pi-GAN [19] and ShadeGAN [137] fail to generate realistic human images. GIRAFFE [132] and CIPS-3D [220] generate reasonable human images, but they suffer from visual artifacts and texture blurs. In contrast, 3D-SGAN synthesizes much better and more photo-realistic images. This qualitative analysis is confirmed in Tab. 4.1, where the corresponding FID scores show that 3D-SGAN significantly outperforms all the other baselines.

Controllable human generation. We analyse the representation controllability of all the models, which reflects the ability to disentangle different attributes from each other. We do this by manipulating a single latent code while fixing the others. Fig. 4.5 (columns “Rotation”, “Object Pose” and “Texture”) shows a qualitative comparison by varying only a single latent code. We observe that all the models can rotate the camera viewpoint. However, GRAF and CIPS-3D fail to disentangle the object pose and the texture. Moreover, pi-GAN and ShadeGAN also suffer from the same problem, since they use one single latent code to model both texture and geometry. On the other hand, both GIRAFFE [132] and 3D-SGAN can effectively disentangle the different variation factors, but GIRAFFE [132] suffers from multi-view inconsistencies and mode collapse for the texture generation. In Table 4.1, we use FID scores to evaluate the realistic degree of each attribute (*e.g.*, “Rotation”, etc.). This is done computing FIDs using only the manipulated (*e.g.*, rotated) fake images, which are compared with all the real images in the dataset. Note that this protocol cannot measure the attribute-based consistency. In most cases, 3D-SGAN has better FID scores than the other baselines.

In order to evaluate the 3D-view consistency, we use our proposed aMP metric (section 4.5). Table 4.1 shows that 3D-SGAN gets the best aMP scores with respect to all the other methods except from ShadeGAN, which however generates much less realistic images, as testified by the very high FIDs (134.7 vs. our 8.24, Table 4.1, first column) and qualitatively shown in Fig. 4.6.

Fig. 4.7 shows additional controllable human image generation results obtained with 3D-SGAN. The generated images are realistic and, in most cases, the attributes are effectively disentangled. Specifically, Fig. 4.7 (1-st row) shows camera rotation results. The images generated by interpolating the camera pose parameter are consistent, and the transition from one image to the next is smooth, while simultaneously preserving the other attributes such as the texture and the pose. On the other hand, the second row shows images generated by interpolating the pose code. We again

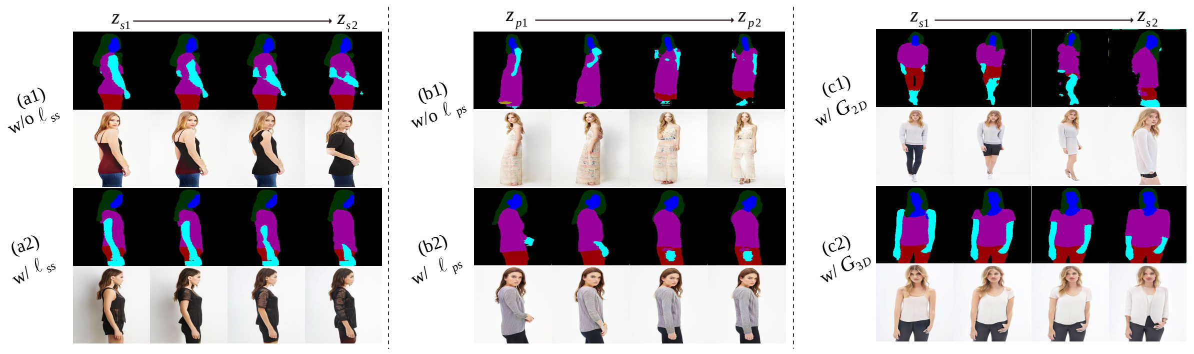

Figure 4.8: A qualitative analysis of ℓ_{ss} (a), ℓ_{ps} (b), and G_{3D} (c). (a) and (c) show interpolation results between semantics codes $\boldsymbol{z}_{s1}$ and $\boldsymbol{z}_{s2}$. (b) shows interpolation results between pose codes $\boldsymbol{z}_{p1}$ and $\boldsymbol{z}_{p2}$.

Table 4.2: A quantitative analysis of ℓ_{ss} (left) and ℓ_{ps} (right). In the latter case, we use LPIPS to measure the diversity of sample pairs generated by interpolating $\boldsymbol{z}_p$.

Metrics	w/o ℓ_{ss}	w/ ℓ_{ss}	Metrics	w/o ℓ_{ps}	w/ ℓ_{ps}
L1 ↓	5.2489	**3.9614**	LPIPS ↓	0.1132	**0.0393**

observe that human identity has been well preserved. Similarly, the other rows show that the non-target attributes have been well preserved. Finally, the third row shows that the head poses from left to right undergo only minor changes ("face frontalization"). This is likely due to both the limited training data and the data bias of the typical fashion images, where people have a frontal face.

4.5.2 Ablation study

The consistency losses. Fig. 4.8 (a) shows a comparison between the results generated by 3D-SGAN with and without ℓ_{ss}. The effectiveness of ℓ_{ss} is shown by observing that, when removed, the generation process suffers from serious geometric inconsistencies. Specifically, the segmentation masks in Fig. 4.8 (a1) have undesirable pose variations, while Fig. 4.8 (a2) shows that ℓ_{ss} can largely alleviate this problem. To quantitatively

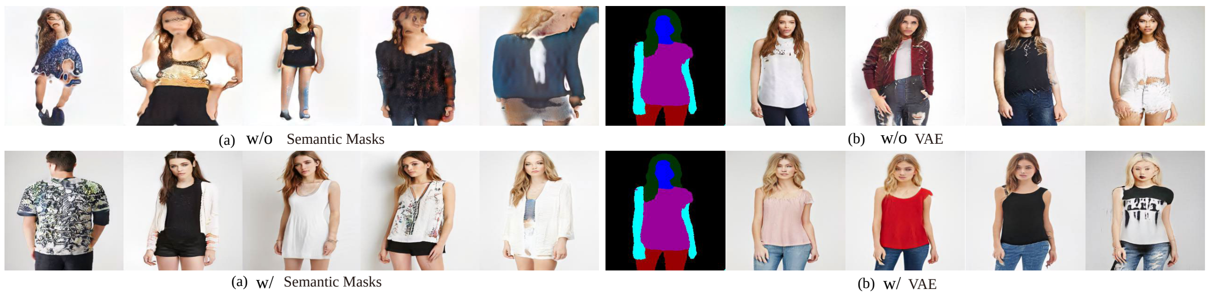

Figure 4.9: An analysis of the impact of the semantic masks (a) and the VAE-conditioned texture generator (b).

Table 4.3: A quantitative analysis of the 3D generator G_{3D}, the semantic masks (SMs) and the VAE in our 3D-SGAN.

Metrics	w/ G_{2D}	w/o SMs	w/o VAE	full
FID ↓	13.24	66.79	14.35	**8.240**

evaluate this effect, we randomly sample two different semantic codes and we compute the $L1$ distance between the silhouettes of the corresponding generated segmentations. We average the scores over 500 different samples. The results reported in Table 4.2 (left) validate the effectiveness of this loss for improving the geometric consistency.

Analogously, Fig. 4.8 (b) qualitatively evaluates the impact of ℓ_{ps} with respect to the semantic consistency over different pose codes. For instance, in Fig. 4.8 (b1) there is no “red” region in the segmentation masks in the first and in the second column. However, this region is present in columns 3 and 4. Conversely, Fig. 4.8 (a2) shows that ℓ_{ps} can alleviate this phenomenon. To quantitatively evaluate l_{ps}, we use LPIPS [214], and we measure the average *pairwise* diversity of the sample pairs generated by interpolating z_p (the lower the diversity, the higher the intra-pair consistency). Tab. 4.2 (right) shows that the full model achieves a lower diversity than the variant without ℓ_{ps}.

The 3D generator. To evaluate the benefit of using a GNeRF based generator, we replace it with a vanilla GAN (G_{2D}), which takes the pose and the semantics code as inputs. In this experiment, we keep all the other modules fixed. Note that G_{2D} cannot manipulate the camera parameters and, thus, it cannot generate images from multiple viewpoints. Moreover, G_{3D} can better disentangle the semantics and the pose factors with respect to G_{2D}, as demonstrated by Fig. 4.8 (c), where we show interpolation results between two different semantic codes. Tab. 4.3 shows the G_{3D} (the full model) achieves significantly better FID scores than G_{2D}.

The semantic masks and the texture generator. Existing methods such as GRAF [160] and GIRAFFE [132] do not use an additional texture generator which translates semantic masks into textured images. In contrast, the effectiveness of our semantic-based approach is shown in Fig. 4.5 and Table 4.1. However, to provide an apple-to-apple comparison and further verify the effectiveness of the semantic masks, we use an additional baseline. Specifically, in this baseline, we render the 3D representation of G_{3D} into features rather than semantic masks and we use the texture generator to map these features into the final image. Fig. 4.9 (a) shows the comparison of our full model with this baseline. We observe that the baseline (w/o semantic masks) fails to generate high-quality human images. Tab. 4.3 shows that the full model quantitatively outperforms this baseline in terms of FID scores.

The Variational Autoencoder. We evaluate the effect of conditioning G_t using a VAE (section 4.4.2). This is done by removing the texture encoder E_t jointly with ℓ_{kl} and ℓ_r from eq. (4.10). Fig. 4.9 (b) shows the comparison between the VAE-based approach and this variant. Both models generate human images with a high texture variability. However, the variant w/o VAE fails to preserve semantic information, *i.e.*, the coherence between the semantic masks, describing the clothes layout, and the final generated clothes. This shows that our VAE-based G_t learns to effectively map the semantic tensors to human images while modeling the texture

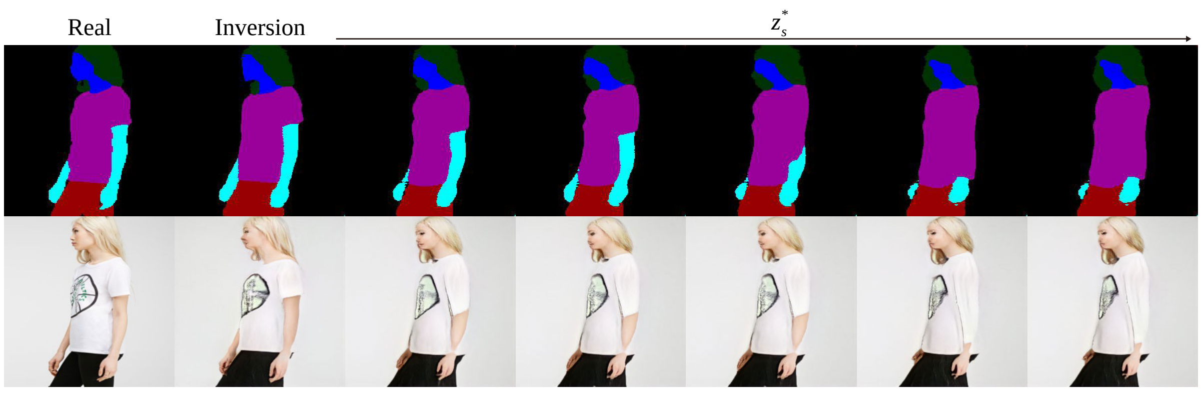

Figure 4.10: Real data semantic editing results using GAN inversion.

distribution with the latent code $\boldsymbol{z}_t$. Tab. 4.3 shows that this variant is significantly outperformed by the proposed VAE-based encoder.

4.5.3 Real Human Image Editing

In this section, we use GAN inversion for real data editing tasks. The second column of Fig. 4.10 shows that the optimal code values $(\boldsymbol{z}_c^*, \boldsymbol{z}_s^*, \boldsymbol{z}_p^*, \boldsymbol{z}_t^*)$, obtained using the procedure described in Sec. 4.4.5, lead to an effective reconstruction of the real input data (first column). In the other columns, we linearly manipulate the semantic code $\boldsymbol{z}_s^*$ while keeping fixed the other codes. Specifically, the second row of Fig. 4.10 shows the generated images corresponding to the semantic masks in the first row. These results demonstrates the effectiveness of the GAN inversion mechanism and the possibility to apply our model to a wide range of human image editing tasks. We computed the average LPIPS and MS-SSIM scores between real and inversion images, respectively obtaining 0.0301 and 0.912, which confirms the high reconstruction quality of our inversion.

4.6 Conclusion

We proposed a 3D-aware Semantic-Guided Generative model (3D-SGAN) for human synthesis. We use a generative NeRF to implicitly represent the 3D human body and we render the 3D representation into 2D segmentation masks. Then, these masks are mapped into the final images using a VAE-conditioned texture generator. Moreover, we propose two consistency losses further disentangle the pose and the semantics factors. Our experiments show that the proposed approach generates human images which are significantly more realistic and more controllable than state-of-the-art methods.

Chapter 5

Conclusion and Future Work

In this thesis, we have presented three methods focusing on global or local human image generation and editing tasks: local gaze editing, local face attribute editing, and global human image generation and editing.

In Chapter 2, we propose a novel unsupervised method, GazeGAN, and extend it to a high-resolution version, GazeGANV2, for gaze-direction correction and animation tasks. GazeGAN and GazeGANV2 formulate the gaze correction problem as an inpainting task and employ a coarse-to-fine learning strategy to generate high-resolution images. Additionally, we introduce a gaze animation module and a Synthesis-As-Training method to generate gaze-correction results with variable angles. Compared to previous methods, GazeGAN is free of gaze labels and can achieve high-quality gaze correction and animation in an unsupervised manner.

In Chapter 3, we have developed an attribute-conditional 3D-aware generative model, TT-GNeRF, which enables the control of multiple facial attributes. Our approach builds upon a pre-trained 3D-aware model and integrates a Dual-Branches Attribute-Editing Module to enable attribute-based generation control. To achieve a better trade-off between attribute editing and non-target region preservation, we fixed the training model, used attribute editing results as initial results, and optimized the latent vector to search for better results following the proposed objective function. Compared to previous 2D methods, our approach achieves 3D-aware facial attribute editing with better view consistency and non-target region preservation than previous latent-space disentangled methods.

In Chapter 4, we propose a 3D-Aware generative model for semantic-guided human image generation and editing (3D-SGAN). Based on the generative neural radiance field, we split the previous pipeline into two stages using segmentation maps as a bridge. Firstly, we pre-train a Variational Autoencoder to learn the mapping from semantic segmentation to human images. Then, we use semantic segmentation to train a 3D-Aware generative model that can control semantics using 3D factors, such as camera pose. Based on these two modules, we can attain a 3D-aware controllable human generative model that supports view-consistent human attributes and semantic editing tasks. Compared to previous 2D methods, our model can generate more realistic human images and disentangle more factors for controllable generation, such as camera pose.

We have conducted an in-depth exploration and development of human generation and editing tasks using the proposed three models. However, these models still have several limitations that need to be solved in future research and development.

1) All of the models are based on generative adversarial networks (GANs), which suffer from training instability and low-quality generation. Recently, Diffusion model [63] has outperformed GANs in most image generation tasks, and it can produce more realistic results, especially for complex content such as scenes and human images [230, 153]. Therefore, we can consider using the diffusion model instead of adversarial loss to improve the quality of generation results.

2) GazeGAN and GazeGANV2 are categorized as generative inpainting

models, which provide a novel approach to achieving gaze correction and redirection unsupervised. However, they are not general gaze redirection models that can redirect eye gaze to a target angle. In comparison, the latter has more outstanding research and application value. Furthermore, our model focuses on 2D gaze editing and cannot perform 3D-aware gaze redirection task. Therefore, we plan to integrate the generative radiance field into the GazeGAN model to extend GazeGAN into a 3D-aware gaze model capable of gaze editing with multi-view generation.

3) TT-GNeRF incorporates a double-branch attribute editing module trained with labels to disentangle the latent space of the generative neural radiance field. To further enhance the performance of this module, we propose utilizing the diffusion model to model the label-conditional distribution of the latent space. We anticipate achieving more precise attribute editing by employing a conditional diffusion model for disentangling the latent code.

4) 3D-SGAN is a 3D-aware controllable human generative model with several limitations. Firstly, it suffers from view inconsistency due to the neural rendering module that renders the radiance feature field to the semantics, which is implemented using convolution networks as the renderer. To address this issue, we propose removing or replacing the neural rendering module with a super-resolution module, as demonstrated in EG3D [20].

   Secondly, 3D-SGAN fails to disentangle the pose, shape, and texture factors. To overcome this limitation, we suggest integrating the human mesh from the SMPL model [113] to learn a controllable neural radiance field. This approach can help disentangle the pose and shape factors.

Lastly, to disentangle the texture from geometry, we can follow the UV-VOLUME [25] and utilize UV to train a generative UV-Volume and a generative SMPL-Texture. By incorporating the SMPL model and UV as guidances, we can disentangle the pose, shape, and texture factors, enabling us to control them independently.

Acknowledgement

I would like to express my heartfelt gratitude to the many individuals who have supported and assisted me throughout my Ph.D. journey. Without their help, my thesis would not have been completed. First and foremost, I am immensely grateful to my supervisor, Prof. Nicu Sebe, for being the perfect mentor I have encountered in my academic career. He has not only been exceptional teachers but also close friend. His unwavering support, care, and guidance have been invaluable to me, both in my research and personal life. Whenever I faced challenges, he went above and beyond to assist me. I would also like to extend my appreciation to Prof. Enver Sangineto and Wei Wang, who have been my close collaborators during my Ph.D. studies. I have been consistently impressed by their meticulous problem-solving approach and their ability to explain complex concepts with utmost clarity. During my internship at Tencent AI Lab in Shenzhen, China, I had the privilege of working under the mentorship of Dr. Xiaoyu Li. They provided me with abundant resources, including computational power and data, to explore cutting-edge industrial problems, specifically in the field of 3D Digital Human Reconstruction. Their constructive feedback and guidance greatly contributed to the success of my research projects. I am also grateful to my colleagues and friends at University of Trento, including Aliaksandr Siarohin, Hao Tang, Yahui Liu, Yue Song, Zhun Zhong, Subhankar Roy, and Weijie Wang, who have brought me immense joy and have always been friendly and supportive. Lastly, I would like to express my deepest gratitude to my family for their unwavering support throughout my academic pursuits. Their unconditional love and encouragement have been instrumental in my success. I wish them all the happiness and good health. I am truly thankful to all the individuals mentioned above and I extend my best wishes to everyone.

Bibliography

[1] Rameen Abdal, Hsin-Ying Lee, Peihao Zhu, Menglei Chai, Aliaksandr Siarohin, Peter Wonka, and Sergey Tulyakov. 3davatargan: Bridging domains for personalized editable avatars. *CVPR*, 2023.

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, 2019.

[3] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, 2019.

[4] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *CVPR*, 2020.

[5] Rameen Abdal, Peihao Zhu, Niloy Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *TOG*, 2020.

[6] Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (TOG)*, 2021.

[7] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *CVPR*, pages 6711–6720, 2021.

[8] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *CVPR*, 2018.

[9] Vijay Badrinarayanan, Alex Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *TPAMI*, 39, 2017.

[10] G. Balakrishnan, Amy Zhao, Adrian V. Dalca, F. Durand, and J. Guttag. Synthesizing images of humans in unseen poses. In *CVPR*, 2018.

[11] Michael Banf and Volker Blanz. Example-based rendering of eye movements. In *CGF*, 2009.

[12] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *TOG*, 2009.

[13] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *CVPR*, 2021.

[14] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022.

[15] David Bau, Hendrik Strobelt, William Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. Semantic photo manipulation with a generative image prior. *TOG*, 2020.

[16] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

[17] Marcel C. Buehler, Seonwook Park, Shalini De Mello, Xucong Zhang, and Otmar Hilliges. Content-consistent generation of realistic eyes with style. In *ICCVW*, 2019.

[18] Xin Cai, Boyu Chen, Jiabei Zeng, Jiajun Zhang, Yunjia Sun, Xiao Wang, Zhilong Ji, Xiao Liu, Xilin Chen, and Shiguang Shan. Gaze estimation with an ensemble of four architectures. *arXiv preprint arXiv:2107.01980*, 2021.

[19] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021.

[20] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. *CVPR*, 2022.

[21] Jingjing Chen, Jichao Zhang, Enver Sangineto, Tao Chen, Jiayuan Fan, and Nicu Sebe. Coarse-to-fine gaze redirection with numerical and pictorial guidance. In *WACV*, 2021.

[22] Xuanhong Chen, Bingbing Ni, Naiyuan Liu, Ziang Liu, Yiliu Jiang, Loc Truong, and Qi Tian. Coogan: A memory-efficient framework for high-resolution facial attribute editing. 2020.

[23] Xuanhong Chen, Bingbing Ni, Naiyuan Liu, Ziang Liu, Yiliu Jiang, Loc Truong, and Qi Tian. Coogan: A memory-efficient framework for high-resolution facial attribute editing. In *ECCV*, 2020.

[24] Xuelin Chen, Daniel Cohen-Or, Baoquan Chen, and Niloy Jyoti Mitra. Towards a neural graphics pipeline for controllable image generation. *CGF*, 2021.

[25] Yue Chen, Xuan Wang, Xingyu Chen, Qi Zhang, Xiaoyu Li, Yu Guo, Jue Wang, and Fei Wang. Uv volumes for real-time rendering of editable free-view human performance. In *CVPR*, 2023.

[26] Yuedong Chen, Qianyi Wu, Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Sem2nerf: Converting single-view semantic masks to neural radiance fields. *ECCV*, 2022.

[27] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018.

[28] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR*, 2020.

[29] Wenqing Chu, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Rongrong Ji. Sscgan: Facial attribute editing via style skip connections. 2020.

[30] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[31] Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *CVPR*, 2020.

[32] Antonio Criminisi, Jamie Shotton, Andrew Blake, and Philip HS Torr. Gaze manipulation for one-to-one teleconferencing. In *ICCV*, 2003.

[33] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019.

[34] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and controllable face image generation via 3d imitative-contrastive learning. In *CVPR*, 2020.

[35] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation, 2022.

[36] Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W. Taylor, and Joshua M. Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *ICCV*, 2021.

[37] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[38] Brian Dolhansky and Cristian Canton Ferrer. Eye in-painting with exemplar generative adversarial networks. In *CVPR*, 2018.

[39] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

[40] Kenneth Alberto Funes Mora, Florent Monay, and Jean-Marc Odobez. Eyediap: A database for the development and evaluation of gaze estimation algorithms from rgb and rgb-d cameras. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, 2014.

[41] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *3DV*, 2017.

[42] Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *CVPR*, 2021.

[43] Yaroslav Ganin, Daniil Kononenko, Diana Sungatullina, and Victor Lempitsky. Deepwarp: Photorealistic image resynthesis for gaze manipulation. In *ECCV*. Springer, 2016.

[44] Yue Gao, Fangyun Wei, Jianmin Bao, Shuyang Gu, Dong Chen, Fang Wen, and Zhouhui Lian. High-fidelity and arbitrary face editing. In *CVPR*, 2021.

[45] Zhenglin Geng, Chen Cao, and Sergey Tulyakov. 3d guided fine-grained face manipulation. In *CVPR*, 2019.

[46] Jeong gi Kwak, David K Han, and Hanseok Ko. Cafe-gan: Arbitrary face attribute editing with complementary attention feature. 2020.

[47] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[48] A. Grigorev, A. Sevastopolsky, Alexander Vakhitov, and V. Lempitsky. Coordinate-based texture inpainting for pose-guided image generation. In *CVPR*, 2019.

[49] Artur Grigorev, Karim Iskakov, Anastasia Ianina, Renat Bashirov, Ilya Zakharkin, Alexander Vakhitov, and Victor Lempitsky. Stylepeople: A generative model of fullbody human avatars. In *CVPR*, 2021.

[50] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *ICLR 2022*, 2022.

[51] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *CVPR*, 2018.

[52] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NeurIPS*, 2017.

[53] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *NeurIPS*, 2020.

[54] Zhe He, Adrian Spurr, Xucong Zhang, and Otmar Hilliges. Photo-realistic monocular gaze redirection using generative adversarial networks. In *ICCV*, 2019.

[55] Zhenliang He, Meina Kan, and Shiguang Shan. Eigengan: Layer-wise eigen-learning for gans. In *CVPR*, 2021.

[56] Zhenliang He, Meina Kan, Jichao Zhang, and Shiguang Shan. Pa-gan: Progressive attention generative adversarial network for facial attribute editing. *arXiv preprint arXiv:2007.05892*, 2020.

[57] Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute editing by only changing what you want. *TIP*, 2019.

[58] Paul Henderson and Vittorio Ferrari. Learning single-image 3d reconstruction by generative modelling of shape, pose and shading. *IJCV*, 2019.

[59] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *ICCV*, 2019.

[60] Philipp Henzler, Niloy J. Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *ICCV*, 2019.

[61] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[62] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. In *NeurIPS*, 2017.

[63] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeuriPS*, 2020.

[64] Fangzhou Hong, Zhaoxi Chen, Yushi Lan, Liang Pan, and Ziwei Liu. Eva3d: Compositional 3d human generation from 2d image collections. *ICLR*, 2022.

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

[66] Siyu Huang, Haoyi Xiong, Zhi-Qi Cheng, Qingzhong Wang, Xingran Zhou, Bihan Wen, Jun Huan, and Dejing Dou. Generating person images with appearance-aware pose stylizer. In *IJCAI*, 2020.

[67] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

[68] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and Locally Consistent Image Completion. *TOG*, 2017.

[69] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *TOG*, 2017.

[70] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[71] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[72] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *ICCV*, 2021.

[73] Suyi Jiang, Haoran Jiang, Ziyu Wang, Haimin Luo, Wenzheng Chen, and Lan Xu. Humangen: Generating human radiance fields with explicit priors. In *CVPR*, pages 12543–12554, 2023.

[74] Zhang Jinsong, Li Kun, Lai Yu-Kun, and Yang Jingyu. PISE: Person image synthesis and editing with decoupled gan. In *CVPR*, 2021.

[75] Youngjoo Jo and Jongyoul Park. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *ICCV*, 2019.

[76] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018.

[77] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *NeurIPS*, 2020.

[78] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *NeurIPS*, 2021.

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

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

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

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

[83] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *CVPR*, 2019.

[84] Hyunsu Kim, Yunjey Choi, Junho Kim, Sungjoo Yoo, and Youngjung Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *CVPR*, 2021.

[85] Davis E King. Dlib-ml: A machine learning toolkit. *JMLR*, 2009.

[86] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2014.

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

[88] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2013.

[89] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[90] R Kollarits, C Woodworth, J Ribera, and R Gitlin. 34.4: An eye contact camera/display system for videophone applications using a conventional direct-view lcd. In *Society for Information Display, International Symposium*, 1996.

[91] Daniil Kononenko, Yaroslav Ganin, Diana Sungatullina, and Victor Lempitsky. Photorealistic monocular gaze redirection using machine learning. *TPAMI*, 2017.

[92] Daniil Kononenko and Victor Lempitsky. Learning to look up: Realtime monocular gaze correction using machine learning. In *CVPR*, 2015.

[93] Claudia Kuster, Tiberiu Popa, Jean-Charles Bazin, Craig Gotsman, and Markus Gross. Gaze correction for home video conferencing. *TOG*, 2012.

[94] Gihyun Kwon and Jong Chul Ye. Diagonal attention and style-based gan for content-style disentanglement in image generation and translation. In *CVPR*, 2021.

[95] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

[96] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

[97] Christoph Lassner, Gerard Pons-Moll, and Peter V Gehler. A generative model of people in clothing. In *ICCV*, 2017.

[98] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani,

Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017.

[99] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *CVPR*, 2020.

[100] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Kumar Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018.

[101] Hanbang Liang, Xianxu Hou, and Linlin Shen. Ssflow: Style-guided neural spline flows for face image manipulation. In *ACM MM*, 2021.

[102] Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *CVPR*, 2020.

[103] Connor Z Lin, David B Lindell, Eric R Chan, and Gordon Wetzstein. 3d gan inversion for controllable portrait image animation. *arXiv preprint arXiv:2203.13441*, 2022.

[104] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting. In *CVPR*, 2019.

[105] Ming Liu, Yukang Ding, Min Xia, Xiao Liu, Errui Ding, Wangmeng Zuo, and Shilei Wen. Stgan: A unified selective transfer network for arbitrary image attribute editing. In *CVPR*, 2019.

[106] Si Liu, Yao Sun, Defa Zhu, Renda Bao, Wei Wang, Xiangbo Shu, and Shuicheng Yan. Face aging with contextual generative adversarial nets. In *ACM MM*, 2017.

[107] Wen Liu, Zhixin Piao, Zhi Tu, Wenhan Luo, Lin Ma, and Shenghua Gao. Liquid warping gan with attention: A unified framework for human image synthesis. *TPAMI*, 2021.

[108] Yahui Liu, Yajing Chen, Linchao Bao, Nicu Sebe, Bruno Lepri, and Marco De Nadai. Isf-gan: An implicit style function for high-resolution image-to-image translation. *TMM*, 2021.

[109] Yahui Liu, Enver Sangineto, Yajing Chen, Linchao Bao, Haoxian Zhang, Nicu Sebe, Bruno Lepri, Wei Wang, and Marco De Nadai. Smoothing the disentangled latent style space for unsupervised image-to-image translation. In *CVPR*, 2021.

[110] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, 2016.

[111] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *CVPR*, 2015.

[112] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *TOG*, 2015.

[113] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *TOG*, 2015.

[114] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

[115] Sebastian Lunz, Yingzhen Li, Andrew W. Fitzgibbon, and Nate Kushman. Inverse graphics GAN: learning to generate 3d shapes from unstructured 2d data. *CoRR*, abs/2002.12674, 2020.

[116] Zhengyao Lv, Xiaoming Li, Xin Li, Fu Li, Tianwei Lin, Dongliang He, and Wangmeng Zuo. Learning semantic person image generation by region-adaptive normalization. In *CVPR*, 2021.

[117] Li Ma, Xiaoyu Li, Jing Liao, Xuan Wang, Qi Zhang, Jue Wang, and Pedro Sander. Neural parameterization for dynamic human head editing. *TOG*, 2022.

[118] Liqian Ma, Xu Jia, Qianru Sun, Bernt Schiele, Tinne Tuytelaars, and Luc Van Gool. Pose guided person image generation. In *NeurIPS*, 2017.

[119] Liqian Ma, Qianru Sun, Stamatios Georgoulis, Luc Van Gool, Bernt Schiele, and Mario Fritz. Disentangled person image generation. In *CVPR*, 2018.

[120] Arun Mallya, Ting-Chun Wang, Karan Sapra, and Ming-Yu Liu. World-consistent video-to-video synthesis. *arXiv preprint arXiv:2007.08509*, 2020.

[121] Yifang Men, Yiming Mao, Yuning Jiang, Wei-Ying Ma, and Zhouhui Lian. Controllable person image synthesis with attribute-decomposed gan. In *CVPR*, 2020.

[122] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *ICML*, 2018.

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

[124] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *ICLR*, 2018.

[125] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *TOG*, 2022.

[126] Natalia Neverova, Riza Alp Guler, and Iasonas Kokkinos. Dense pose transfer. In *ECCV*, 2018.

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

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

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

[130] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022.

[131] Michael Niemeyer and Andreas Geiger. CAMPARI: camera-aware decomposed generative neural radiance fields. In *3DV*, 2021.

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

[133] Ken-Ichi Okada, Fumihiko Maeda, Yusuke Ichikawaa, and Yutaka Matsushita. Multiparty videoconferencing at virtual social distance: Majic design. In *ACM conference on Computer supported cooperative work*, 1994.

[134] Kyle Olszewski, Duygu Ceylan, Jun Xing, Jose Echevarria, Zhili Chen, Weikai Chen, and Hao Li. Intuitive, interactive beard and hair synthesis with generative models. *CVPR*, 2020.

[135] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. StyleSDF: High-Resolution 3D-Consistent Image and Geometry Generation. *CVPR2022*, 2022.

[136] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. 2022.

[137] Xingang Pan, Xudong Xu, Chen Change Loy, Christian Theobalt, and Bo Dai. A shading-guided generative implicit model for shape-accurate 3d-aware image synthesis. *NeurIPS*, 2021.

[138] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. Few-shot adaptive gaze estimation. In *CVPR*, 2019.

[139] Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. *arXiv preprint arXiv:2007.15651*, 2020.

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

[141] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. *NeurIPS*, 2020.

[142] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[143] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[144] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *2009 sixth IEEE international conference on advanced video and signal based surveillance*, pages 296–301. Ieee, 2009.

[145] William Peebles, John Peebles, Jun-Yan Zhu, Alexei Efros, and Antonio Torralba. The hessian penalty: A weak prior for unsupervised disentanglement. In *ECCV*, 2020.

[146] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Animatable neural radiance fields for human body modeling. In *ICCV*, 2021.

[147] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*, 2021.

[148] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, 2021.

[149] Yurui Ren, Xiaoming Yu, Junming Chen, Thomas H Li, and Ge Li. Deep image spatial transformation for person image generation. In *CVPR*, 2020.

[150] Danilo Jimenez Rezende, S. M. Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *NeurIPS*, 2016.

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

[152] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *arXiv preprint arXiv:2106.05744*, 2021.

[153] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[154] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, 2016.

[155] Soubhik Sanyal, Alex Vorobiov, Timo Bolkart, Matthew Loper, Betty Mohler, Larry S. Davis, Javier Romero, and Michael J. Black. Learning realistic human reposing using cyclic self-supervision with 3d shape, pose, and appearance consistency. In *ICCV*, 2021.

[156] Soubhik Sanyal, Alex Vorobiov, Timo Bolkart, Matthew Loper, Betty Mohler, Larry S Davis, Javier Romero, and Michael J Black. Learning realistic human reposing using cyclic self-supervision with 3d shape, pose, and appearance consistency. In *CVPR*, 2021.

[157] Kripasindhu Sarkar, Vladislav Golyanik, Lingjie Liu, and Christian Theobalt. Style and pose control for image synthesis of humans from a single monocular view. *arXiv preprint arXiv:2102.11263*, 2021.

[158] Kripasindhu Sarkar, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. Humangan: A generative model of humans images. 2021.

[159] Kripasindhu Sarkar, Dushyant Mehta, Weipeng Xu, Vladislav Golyanik, and Christian Theobalt. Neural re-rendering of humans from a single image. In *ECCV*, 2020.

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

[161] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. *NeurIPS*, 2022.

[162] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020.

[163] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *CVPR*, 2021.

[164] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *CVPR*, 2021.

[165] Yichun Shi, Divyansh Aggarwal, and Anil K Jain. Lifting 2d stylegan for 3d-aware face generation. In *CVPR*, 2021.

[166] Yichun Shi, Xiao Yang, Yangyue Wan, and Xiaohui Shen. Semanticstylegan: Learning compositional generative priors for controllable image synthesis and editing. *CVPR*, 2022.

[167] Aliaksandr Siarohin, Stéphane Lathuilière, Enver Sangineto, and Nicu Sebe. Appearance and Pose-Conditioned Human Image Generation using Deformable GANs. *TPAMI*, 2020.

[168] Aliaksandr Siarohin, Stéphane Lathuilière, Enver Sangineto, and Nicu Sebe. Appearance and Pose-Conditioned Human Image Generation using Deformable GANs. *TPAMI*, 2020.

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

[170] Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. Epigraf: Rethinking training of 3d gans. *arXiv preprint arXiv:2206.10535*, 2022.

[171] Brian A Smith, Qi Yin, Steven K Feiner, and Shree K Nayar. Gaze locking: passive eye contact detection for human-object interaction. In *ACM symposium on User interface software and technology*, 2013.

[172] Sijie Song, Wei Zhang, Jiaying Liu, and Tao Mei. Unsupervised person image generation with semantic parsing transformation. In *CVPR*, 2019.

[173] Jingxiang Sun, Xuan Wang, Yichun Shi, Lizhen Wang, Jue Wang, and Yebin Liu. Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis. *TOG*, 2022.

[174] Jingxiang Sun, Xuan Wang, Lizhen Wang, Xiaoyu Li, Yong Zhang, Hongwen Zhang, and Yebin Liu. Next3d: Generative neural texture rasterization for 3d-aware head avatars. *CVPR*, 2023.

[175] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. Fenerf: Face editing in neural radiance fields. *arXiv preprint arXiv:2111.15490*, 2021.

[176] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. Fenerf: Face editing in neural radiance fields. In *CVPR*, 2022.

[177] Keqiang Sun, Shangzhe Wu, Zhaoyang Huang, Ning Zhang, Quan Wang, and HongSheng Li. Controllable 3d face synthesis with conditional generative occupancy fields. *NeurIPS*, 2022.

[178] Feitong Tan, Sean Fanello, Abhimitra Meka, Sergio Orts-Escolano, Danhang Tang, Rohit Pandey, Jonathan Taylor, Ping Tan, and Yinda Zhang. Volux-gan: A generative model for 3d face synthesis with hdri relighting. *arXiv preprint arXiv:2201.04873*, 2022.

[179] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. *arXiv preprint arXiv:2202.05263*, 2022.

[180] Hao Tang, Song Bai, Li Zhang, Philip HS Torr, and Nicu Sebe. Xinggan for person image generation. In *ECCV*, 2020.

[181] Hao Tang, Dan Xu, Gaowen Liu, Wei Wang, Nicu Sebe, and Yan Yan. Cycle in cycle generative adversarial networks for keypoint-guided image generation. In *ACM MM*, 2019.

[182] Hao Tang, Dan Xu, Nicu Sebe, Yanzhi Wang, Jason J. Corso, and Yan Yan. Multi-channel attention selection gan with cascaded semantic guidance for cross-view image translation. In *CVPR*, 2019.

[183] Hao Tang, Dan Xu, Yan Yan, Philip HS Torr, and Nicu Sebe. Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation. In *CVPR*, 2020.

[184] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. In *CVPR*, 2020.

[185] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *TOG*, 2021.

[186] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, 2016.

[187] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[188] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[189] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *ICML*, 2020.

[190] X. Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. In *ECCV*, 2016.

[191] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCVW*, 2018.

[192] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, 2003.

[193] Yuxiang Wei, Yupeng Shi, Xiao Liu, Zhilong Ji, Yuan Gao, Zhongqin Wu, and Wangmeng Zuo. Orthogonal jacobian regularization for unsupervised disentanglement in image generation. In *CVPR*, 2021.

[194] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. Gazedirector: Fully articulated eye gaze redirection in video. In *CGF*, 2018.

[195] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NeurIPS*, 2016.

[196] Po-Wei Wu, Yu-Jing Lin, Che-Han Chang, Edward Y Chang, and Shih-Wei Liao. Relgan: Multi-domain image-to-image translation via relative attributes. In *CVPR*, 2019.

[197] Jianfeng Xiang, Jiaolong Yang, Yu Deng, and Xin Tong. Gram-hd: 3d-consistent image generation at high resolution with generative radiance manifolds. *arXiv preprint arXiv:2206.07255*, 2022.

[198] Xudong Xu, Xingang Pan, Dahua Lin, and Bo Dai. Generative occupancy fields for 3d surface-aware image synthesis. In *NeurIPS*, 2021.

[199] Yanbo Xu, Yueqin Yin, Liming Jiang, Qianyi Wu, Chengyao Zheng, Chen Change Loy, Bo Dai, and Wayne Wu. Transeditor:

Transformer-based dual-space gan for highly controllable facial editing. *CVPR*, 2022.

[200] Yinghao Xu, Sida Peng, Ceyuan Yang, Yujun Shen, and Bolei Zhou. 3d-aware image synthesis via learning structural and textural representations. *CVPR*, 2022.

[201] Yang Xue, Yuheng Li, Krishna Kumar Singh, and Yong Jae Lee. Giraffe hd: A high-resolution 3d-aware generative model. *CVPR*, 2022.

[202] Ruigang Yang and Zhengyou Zhang. Eye gaze correction with stereovision for video-teleconferencing. In *ECCV*, 2002.

[203] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iNeRF: Inverting neural radiance fields for pose estimation. In *IROS*, 2021.

[204] Gokhan Yildirim, Nikolay Jetchev, Roland Vollgraf, and Urs Bergmann. Generating high-resolution fashion model images wearing custom outfits. In *ICCVW*, pages 0–0, 2019.

[205] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, 2018.

[206] Yu Yu, Gang Liu, and Jean-Marc Odobez. Improving few-shot user-specific gaze adaptation via gaze redirection synthesis. In *CVPR*, 2019.

[207] Yu Yu and Jean-Marc Odobez. Unsupervised representation learning for gaze estimation. In *CVPR*, 2020.

[208] Haoran Zhang, Zhenzhen Hu, Changzhi Luo, Wangmeng Zuo, and Meng Wang. Semantic image inpainting with progressive generative networks. In *ACM MM*, 2018.

[209] Jianfeng Zhang, Zihang Jiang, Dingdong Yang, Hongyi Xu, Yichun Shi, Guoxian Song, Zhongcong Xu, Xinchao Wang, and Jiashi Feng. Avatargen: a 3d generative model for animatable human avatars. *arXiv preprint arXiv:2211.14589*, 2022.

[210] Jichao Zhang, Enver Sangineto, Hao Tang, Aliaksandr Siarohin, Zhun Zhong, Nicu Sebe, and Wei Wang. 3d-aware semantic-guided generative model for human synthesis. *arXiv preprint arXiv:2112.01422*, 2021.

[211] Jichao Zhang, Yezhi Shu, Songhua Xu, Gongze Cao, Fan Zhong, Meng Liu, and Xueying Qin. Sparsely grouped multi-task generative adversarial networks for facial attribute manipulation. In *ACM MM*, 2018.

[212] Jichao Zhang, Aliaksandr Siarohin, Hao Tang, Jingjing Chen, Enver Sangineto, Wei Wang, and Nicu Sebe. Controllable person image synthesis with spatially-adaptive warped normalization. *arXiv preprint arXiv:2105.14739*, 2021.

[213] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

[214] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[215] Xuanmeng Zhang, Zhedong Zheng, Daiheng Gao, Bang Zhang, Pan Pan, and Yi Yang. Multi-view consistent generative adversarial networks for 3d-aware image synthesis. *CVPR*, 2022.

[216] Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang, and Otmar Hilliges. Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation. *ECCV*, 2020.

[217] Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang, and Otmar Hilliges. Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation. In *ECCV*, 2020.

[218] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Mpiigaze: Real-world dataset and deep appearance-based gaze estimation. *TPAMI*, 2017.

[219] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C Bühler, Xu Chen, Michael J Black, and Otmar Hilliges. Im avatar: Implicit morphable head avatars from videos. In *CVPR*, 2022.

[220] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. CIPS-3D: A 3D-Aware Generator of GANs Based on Conditionally-Independent Pixel Synthesis. 2021.

[221] Qunjie Zhou, Torsten Sattler, and Laura Leal-Taixe. Patch2pix: Epipolar-guided pixel-level correspondences. In *CVPR*, 2021.

[222] Xingran Zhou, Bo Zhang, Ting Zhang, Pan Zhang, Jianmin Bao, Dong Chen, Zhongfei Zhang, and Fang Wen. Cocosnet v2: Full-resolution correspondence learning for image translation. In *CVPR*, 2021.

[223] Jiapeng Zhu, Yujun Shen, Yinghao Xu, Deli Zhao, and Qifeng Chen. Region-based semantic factorization in gans. *CVPR*, 2022.

[224] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *ECCV*, 2020.

[225] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2016.

[226] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[227] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[228] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *ICCV*, 2017.

[229] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Joshua B. Tenenbaum, and William T. Freeman. Visual object networks: Image generation with disentangled 3D representations. In *NeurIPS*, 2018.

[230] Luyang Zhu, Dawei Yang, Tyler Zhu, Fitsum Reda, William Chan, Chitwan Saharia, Mohammad Norouzi, and Ira Kemelmacher-Shlizerman. Tryondiffusion: A tale of two unets. In *CVPR*, 2023.

[231] Zhen Zhu, Tengteng Huang, Baoguang Shi, Miao Yu, Bofei Wang, and Xiang Bai. Progressive pose attention transfer for person image generation. In *CVPR*, 2019.

[232] Yiyu Zhuang, Hao Zhu, Xusen Sun, and Xun Cao. Mofanerf: Morphable facial neural radiance field. *arXiv preprint arXiv:2112.02308*, 2021.