

THREE-DIMENSIONAL RECONSTRUCTION AND EDITING FROM SINGLE IMAGES WITH GENERATIVE MODELS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

By
Bahri Batuhan Bilecen
May 2025

Three-dimensional Reconstruction and Editing from Single Images
with Generative Models

By Bahri Batuhan Bilecen

May 2025

We certify that we have read this thesis and that in our opinion it is fully adequate,
in scope and in quality, as a thesis for the degree of Master of Science.

Ayşegül Dünder Boral(Advisor)

Uğur Gündükbay

Ramazan Gökberk Cinbiş

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

THREE-DIMENSIONAL RECONSTRUCTION AND EDITING FROM SINGLE IMAGES WITH GENERATIVE MODELS

Bahri Batuhan Bilecen

M.S. in Computer Engineering

Advisor: Ayşegül Dünder Boral

May 2025

Advancements in generative networks have significantly improved visual synthesis, particularly in three-dimensional (3D) applications. However, key challenges remain in achieving high-fidelity 3D reconstruction, preserving identity in 3D stylization, and enabling reference-based edits with 3D consistency. This thesis attempts to address these gaps through three interconnected studies. First, a framework of high-fidelity 3D head reconstruction from single images is introduced, leveraging dual encoder GAN inversion to reconstruct full 360-degree heads. By integrating an occlusion-aware triplane discriminator, this approach ensures seamless blending of visible and occluded regions, surpassing existing methods in realism and structural accuracy. Next, an identity-preserving 3D head stylization method is developed to balance artistic transformation with facial identity retention. Through multi-view score distillation and likelihood distillation, this technique enhances stylization diversity while maintaining subject-specific features, outperforming prior diffusion-to-GAN adaptation strategies. Finally, a single image reference-based 3D-aware image editing method extends these advancements by enabling precise, high-quality edits using triplane representations. By incorporating automatic feature localization, spatial disentanglement, and fusion learning, this work achieves state-of-the-art performance in 3D-consistent, 2D reference-guided edits across various domains. Together, these contributions attempt to advance the field of 3D-aware generative modeling, providing robust solutions for reconstruction, stylization, and editing with greater fidelity, consistency, and control.

Keywords: 3D reconstruction, 3D editing, 3D stylization, generation from single images.

ÖZET

ÜRETKEN MODELLERLE TEKLİ GÖRSELLERDEN ÜÇ-BOYUTLU YENİDEN YAPILANDIRMA VE DÜZENLEME

Bahri Batuhan Bilecen

Bilgisayar Mühendisliği, Yüksek Lisans

Tez Danışmanı: Ayşegül Dünder Boral

May 2025

Üretken ağlardaki gelişmeler, üç-boyutlu (3B) uygulamalarda görüntü üretimini önemli ölçüde iyileştirmiştir. Ancak yüksek doğrulukta 3B yeniden yapılandırma, üsluplaştırma sırasında kimlik korunumu ve 3B tutarlılığına sahip kaynak tabanlı düzenlemeler sağlama konusunda zorluklar devam etmektedir. Bu çalışma, 3B farkındalıklı alanlardaki üretken modellemeyi geliştiren üç bağlantılı araştırmayla bahsedilen eksiklikleri gidermektedir. İlk olarak, yüksek doğruluklu 3B kafa yeniden yapılandırma modeli sunulmuştur. Tek görüntüden 360 derece kafa üretimi yapmak için çift kodlayıcılı GAN tersine çevirme yöntemi kullanılmaktadır. Görünür ve gizli bölgelerin birleştirilmesini sağlayan örtme farkındalıklı üç-düzlem ayırmacı ile bu yöntem, yapısal doğruluk açısından mevcut yaklaşımları aşmaktadır. Ardından, sanatsal dönüşüm sırasında yüz kimliğinin korunmasını dengeleyen bir 3B kafa üsluplaştırma yöntemi geliştirilmiştir. Çoklu skor damıtma ve olabilirlik damıtma (LD) teknikleri sayesinde bu yöntem, önceki uyarlama stratejilerine göre özneye özgü özellikleri daha iyi korumaktadır. Son olarak, tek görüntüye dayalı kaynak tabanlı 3B farkındalıklı görüntü düzenleme çerçevesi, üç-düzlem temsillerini kullanarak hassas ve yüksek kaliteli 3B düzenlemeler yapılmasını sağlamaktadır. Otomatik özellik konumlandırma, mekansal ayrıştırma ve füzyon öğrenme tekniklerini entegre ederek çeşitli alanlarda 3B tutarlı ve referans odaklı düzenlemelerde en yüksek performansı elde etmektedir. Bu katkılar, 3B üretken modellemeyi ileriye taşıyarak, yeniden yapılandırma, üsluplama ve düzenleme alanlarında daha yüksek doğruluk ve kontrol sağlayan güçlü çözümler sunmaktadır.

Anahtar sözcükler: 3B yeniden yapılandırma, 3B düzenleme, 3B üsluplaştırma, tekli görsellerden 3B üretim.

Acknowledgement

I would like to express my deepest gratitude to my advisor, Asst. Prof. Ayşegül Dündar Boral, for her guidance throughout my academic journey.

I appreciate Prof. Uğur Güdükbay and Assoc. Prof. Ramazan Gökberk Cinbiş for accepting to be my thesis jury, and providing valuable feedback.

I am grateful for my teammates in Bilkent Generative Deep Learning Lab, as well as my close friends Berke Gökmen, Muhammed Tiraki, Yasin Çalışkan, and Yiğit Ekin, for their constant encouragement and support.

A special thanks to my co-workers and collaborators Alparslan Fişne, Alper Güngör, Can Barış Top, Mustafa Ayazoğlu, and Ning Yu for their continuous motivation and belief in my potential.

Finally, I am deeply thankful to my family for their unconditional love and support throughout my life.

Contents

1	Introduction	1
1.1	High-fidelity 3D Head Reconstruction from Single Images	2
1.2	Identity Preserving 3D Head Stylization	2
1.3	Reference-based 3D-aware Image Editing with Triplanes	3
2	Preliminaries and Related Work	5
2.1	Generative Adversarial Networks	5
2.2	Neural Rendering and Neural Radiance Fields	9
2.2.1	3D Representations and Neural Volumetric Rendering . . .	9
2.2.2	Mathematical Formulation of NVR	11
2.2.3	Neural Radiance Fields	14
2.3	Latent Diffusion Models and Score-based Models	15
2.4	Score Distillation Sampling	17
2.5	2D and 3D Style-Based Generators	17

2.5.1	StyleGAN2	17
2.5.2	3D Generators, EG3D and PanoHead	19
2.6	GAN Inversion, Editing, and Stylization	20
2.6.1	GAN Inversion	20
2.6.2	Editing and Stylization in GANs	22
3	High-fidelity 3D Head Reconstruction from Single Images	24
3.1	Method	25
3.1.1	Training an encoder	25
3.1.2	Occlusion-aware triplane discriminator	27
3.1.3	Dual encoder pipeline	30
3.2	Experimental Setup	31
3.2.1	Training	31
3.2.2	Architecture details	32
3.2.3	Baselines	33
3.2.4	Evaluation metrics	33
3.3	Results	34
3.3.1	Comparison with state-of-the-art	34
3.3.2	Ablation study	37
3.3.3	Editing application	39

3.3.4	Broader impacts and limitations	39
4	Identity Preserving 3D Head Stylization	41
4.1	Method	42
4.1.1	Likelihood distillation (LD) objective	43
4.1.2	Rank-weighted score tensors	45
4.1.3	Extending LD with mirror poses and multi-view distillation	46
4.2	Experimental Setup	50
4.2.1	Baselines	50
4.2.2	Training	51
4.2.3	Evaluation metrics	51
4.3	Results	52
4.3.1	Comparison with state-of-the-art	52
4.3.2	Ablation study	55
4.3.3	Limitations	60
5	Reference-based 3D-aware Image Editing with Triplanes	61
5.1	Method	62
5.1.1	Localizing parts in the triplane space	62
5.1.2	Implicit fusion by encoding & decoding	64

5.1.3	Fine-tuning the image encoder	66
5.2	Experimental Setup	68
5.2.1	Evaluation metrics	68
5.2.2	Baselines	68
5.3	Results	69
5.3.1	Comparison with state-of-the-art	69
5.3.2	Ablation study	72
5.3.3	Limitations	74
6	Conclusion and Future Work	77
	Bibliography	79

List of Figures

2.1	Basic structure of a GAN.	6
2.2	Visualization of mass and transport distance for EM.	8
2.3	Different 3D surface representations.	10
2.4	Reparametrized ray, coming out of the camera and passing through the scene.	11
2.5	The case where the ray has hit a particle in the range (t_i, t_{i+1}) . $\mathcal{T}$ is divided into T_i and $1 - \alpha_i$	13
2.6	General pipeline of NeRF. Adapted from [1].	14
2.7	Architecture of LDM. Figure adapted from [2].	16
2.8	Partial architecture of StyleGAN2 and its outputs.	18
2.9	EG3D architecture, without the discriminator.	20
2.10	Various GAN inversion approaches in the literature and encoder- based inversion.	21
2.11	Representative comparison of [3] and [4] finding the editing direc- tions $p_{1,2}$	23

3.1	Reconstruction of 3D representation by inverting images into PanoHead’s latent space.	25
3.2	Our triplane discriminator training involves generating real samples from $\mathcal{Z}^+$ and producing in-domain triplanes using PanoHead.	28
3.3	Visual results of Encoder 1, Encoder 2, and dual encoders for the given input images in Columns 1 and 6.	30
3.4	The inference pipeline with dual encoders for full 3D head reconstruction.	31
3.5	Comparisons of ours and competing methods.	35
3.6	Edited image results.	36
3.7	Inputs (first), reconstructions (second), and 360° mesh renders (rest).	36
3.8	Qualitative results of ablation on occlusion-aware discriminator $\mathcal{D}$	37
3.9	Hair edits from source image (first) to destination image (second) and 360° renders (rest).	39
3.10	Example failure cases. Inputs (first), reconstructions (second), and novel views.	40
4.1	Showcasing our method, effectively balancing stylization and identity preservation.	42
4.2	Our proposed training approach with mirror gradients (a) and grid distillation (b).	43
4.3	Ablation study on rank weighing.	46
4.4	Samples from the utilized dataset.	52

4.5	Qualitative stylization results provided in 360-degree views.	54
4.6	Mesh and rendering visualizations of ours and competing methods for the Pixar prompt.	54
4.7	The percentage of users' preferences is presented for different prompts.	55
4.8	Qualitative Ablation Study I.	56
4.9	Qualitative Ablation Study II.	57
4.10	Additional results with various prompts applied to the input image given in the first column.	58
4.11	Additional comparisons with other 2D-3D editing methods.	59
4.12	Limitations in preserving earrings and hair accessories.	60
5.1	Our approach excels in reference-based edits, utilizing only a single source and reference image.	62
5.2	Triplane part localization stage.	63
5.3	Triplane localization and implicit fusion stages.	65
5.4	Pipeline for the implicit fusion encoder fine-tuning.	67
5.5	Comparisons with the competing editing methods for glasses ad- dition and hair edits.	71
5.6	Qualitative ablation study for glasses and hair edits, showing the effects of all fundamental stages of our pipeline.	73
5.7	Cross-generator edits with stylizing.	74

5.8	Additional editing examples from the CelebA dataset.	75
5.9	Additional editing examples from AFHQ dataset.	76
5.10	Extending our method on try-on edits on [5].	76
5.11	Extending our method on full-head hair edits on [6].	76



List of Tables

3.1	Architecture for the discriminator.	33
3.2	Quantitative scores on various test sets.	34
3.3	Quantitative scores on multi-view MEAD dataset.	35
3.4	Ablation on occlusion-aware discriminator $\mathcal{D}$	37
3.5	Ablation on training data and latent space.	38
3.6	Ablation on dual-encoder.	39
4.1	Quantitative scores with competitive methods.	53
4.2	Quantitative ablation study.	56
5.1	Quantitative scores on CelebA.	70
5.2	The results of our user study, where participants are asked to identify the edited image.	72
5.3	Quantitative ablation study.	72

Chapter 1

Introduction

Advances in three-dimensional (3D) generative modelling have opened exciting new frontiers in visual computing. One of many branches in this research area is synthesising, reconstructing, and editing, even when given only a single input image. Much of this progress is driven by Generative Adversarial Networks (GANs) [7] and Latent Diffusion Models [2]. When these models are combined with Neural Radiance Fields (NeRFs) [1] or triplane representations [8], they deliver high-quality, 3D-aware results that remain consistent across novel viewpoints. Such capabilities pave the way for applications ranging from faithful 3D reconstruction to stylization and intuitive, geometry-aware editing.

This thesis presents three related works on 3D generation and manipulation from single images, mainly shown on human-centric applications: high-fidelity 3D head reconstruction [9] from single images, identity-preserving 3D head stylization [10], and reference-based 3D-aware editing [11] from single images. Each section in this introduction introduces new methods that address fundamental limitations in current approaches by leveraging advanced architectures, innovative training strategies, and carefully designed latent-space operations.

1.1 High-fidelity 3D Head Reconstruction from Single Images

3D GANs, such as EG3D [8], extend 2D generative models with consistent 3D representations by integrating NeRF-like components. Inverting these models—projecting a real image back into the latent space, which is a key task for applications like identity capture and 3D editing. Unlike 2D inversion [12, 13, 14], 3D inversion must synthesize occluded regions realistically and maintain consistency across views.

Early 3D GAN inversion methods rely on optimization [15, 16], which are accurate but computationally heavy. Encoder-based methods [17, 18] are faster but typically use EG3D, which limits generation to near-frontal views.

In this work, we adopt PanoHead [6], which enables 360° synthesis but introduces new challenges such as hallucinating completely invisible regions. Existing EG3D-based inversion methods often fail in this context, since some reconstructions appear accurate only from the original view, while others generate plausible geometry that does not match the input.

To overcome this, we propose a **dual-encoder architecture**. One encoder specializes in reconstructing the visible input view, while the other focuses on predicting occluded regions. Their outputs are fused in the triplane space, with a novel **occlusion-aware discriminator** enforcing coherence across all views. Our approach outperforms state-of-the-art inversion baselines in terms of reconstruction accuracy and quality across novel views.

1.2 Identity Preserving 3D Head Stylization

3D head stylization transforms realistic heads into artistic versions while maintaining their identity across multiple angles. This task has gained interest due to

its relevance in entertainment, AR, education, and virtual avatars. Prior work in 2D stylization using GANs [19, 20] leveraged StyleGAN [21], but recent efforts now target 3D-aware stylization [22, 23].

Most methods are built on EG3D, restricting synthesis to near-frontal views and limiting stylization performance across 360°. We adopt PanoHead to explore full-head stylization; however, we still observe that adapting existing methods often produces inconsistent results at novel angles. Additionally, Score Distillation Sampling (SDS) [24] has shown effectiveness for diffusion-guided stylization [22, 23], but tends to cause identity loss and reduced diversity [23].

To address this, we adapt **Negative Log-likelihood Distillation (LD)** [25] to GANs, which preserves identity and image sharpness better than SDS. We show that utilizing a 3D GAN backbone eliminates extensive regularization to avoid divergence, unlike traditional LD or SDS, which are performed directly on 3D parameters. We further introduce **rank-weighting** of score tensors on VQ-VAE channels and propose **multi-view grid and mirror score gradients** to regularize the distillation process, and better preserve input color and facial identity during distillation. We also avoid score distillation in the super-resolution (SR) layers to retain high-quality stylization. Our multi-view gradient formulation aligns with 3D-GAN design, boosting consistency across all angles and avoiding SR overfitting, outperforming current state-of-the-art methods.

1.3 Reference-based 3D-aware Image Editing with Triplanes

Image editing using GANs has advanced significantly through inversion [26, 12, 27, 17, 28] and latent manipulation [29], but remains largely 2D-focused. Reference-based editing, which transfers attributes from a reference image to a target, is powerful yet underdeveloped for 3D-aware models.

Existing 3D-aware editing methods [17, 23, 30] lack reference-based flexibility,

while 2D methods [31, 32, 33, 29, 20, 34, 35, 36] do not support multi-view or 3D consistency. Our approach bridges this gap by enabling **reference-based 3D-aware image editing via triplanes** [8, 5, 37, 38] for the first time.

Triplanes provide a shared canonical latent 3D space, simplifying cross-view and cross-domain edits. For instance, attributes like eyeglasses and hairstyles can be transferred without requiring perfect 2D alignment. Our framework exploits this property of triplane-based generators, learns to localize and disentangle regions using masked residual gradients, and fuses the features using dedicated encoders.

Our reference-based editing framework is generalized to human faces and heads [6], animal faces, cartoons [22], full-body edits [5], and class-agnostic 3D object renders [39, 38, 37], demonstrating broad applicability.

In the remaining of this thesis, Chapter 2 will present Preliminaries and Related Work, Chapters 3 to 5 will detail three aforementioned works [10, 9, 11], and Chapter 6 will discuss Conclusion and Future Work.

Chapter 2

Preliminaries and Related Work

2.1 Generative Adversarial Networks

Generative adversarial neural networks (GANs) were proposed by Goodfellow et al. [7]. The main idea is making two agents (discriminator $\mathbf{D}$ and generator $\mathbf{G}$) race each other in a minimax game, where one agent's gain is another agent's loss. The game continues oscillating until both agents reach a stable equilibrium. In the process, generator $\mathbf{G}$ produces *fake* data $\mathbf{G}(z)$ from latent vectors $z \sim p(z)$, and discriminator $\mathbf{D}$ tries to identify if $\mathbf{G}(z)$'s are *real* or *fake*. The goal is to create deceptively *real-looking* $\mathbf{G}(z)$'s, with the aid of $\mathbf{D}$'s feedback during the training. A basic GAN architecture is given in Fig. 2.1.

Since training GANs is inherently a classification problem, binary cross entropy (BCE) loss in Eq. (2.1) is utilized:

$$\text{BCE}(x, y) = y \log x + (1 - y) \cdot \log(1 - x) \quad (2.1)$$

Using Eq. (2.1), losses for $\mathbf{D}$ and $\mathbf{G}$ become:

$$\text{loss}_{\mathbf{D}} = \min_{\mathbf{D}} \text{BCE}(\mathbf{D}(x), 1) + \text{BCE}(\mathbf{D}(\mathbf{G}(z)), 0) \quad (2.2)$$

$$\text{loss}_{\mathbf{G}} = \min_{\mathbf{G}} \text{BCE}(\mathbf{D}(\mathbf{G}(z)), 1) \quad (2.3)$$

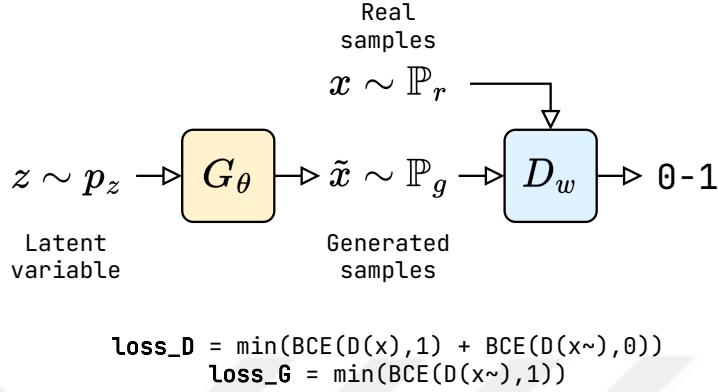


Figure 2.1: Basic structure of a GAN. 0 and 1 denote the labels of fake and real, respectively. BCE is the binary cross-entropy loss.

Upon further simplifications, minimizing `loss_D` and `loss_G` in Eqs. (2.2) and (2.3) becomes equivalent to maximizing Eq. (2.4) and minimizing Eq. (2.5), respectively:

$$\max_{\mathbf{D}} \mathbb{E}_{x \sim \mathbb{P}_r} \log \mathbf{D}(x) + \mathbb{E}_{z \sim p_z} \log(1 - \mathbf{D}(\mathbf{G}(z))) \quad (2.4)$$

$$\min_{\mathbf{G}} \mathbb{E}_{z \sim p_z} \log(1 - \mathbf{D}(\mathbf{G}(z))) \quad (2.5)$$

Combining Eq. (2.4) and Eq. (2.5) yields the overall minimax loss in Eq. (2.6):

$$\boxed{\min_{\mathbf{G}} \max_{\mathbf{D}} \mathbb{E}_{x \sim \mathbb{P}_r} \log \mathbf{D}(x) + \mathbb{E}_{\tilde{x} \sim \mathbb{P}_g} \log(1 - \mathbf{D}(\tilde{x}))}, \quad (2.6)$$

where $\mathbb{P}_r$ denotes the ground truth *real* data distribution, and $\mathbb{P}_g$ denotes the generator model distribution implicitly by $\tilde{x} = \mathbf{G}(z)$, $z \sim p_z$. Most of the time, $p_z \sim \mathcal{N}(0, \mathbb{I})$ (i.e., zero norm, unit variance Gaussian). Algorithm 1 demonstrates how GANs are trained via mini-batch stochastic gradient descent (SGD).

Now, assume $\mathbf{D}$ is trained to optimality while $\mathbf{G}$ is not, which can be achieved through Algorithm 1. In this case, optimizing Eq. (2.6) minimizes the Jensen-Shannon (JS) divergence [40] between $\mathbb{P}_r$ and $\mathbb{P}_g$. However, JS divergence struggles to measure the distance between disjoint distributions. Once $\mathbb{P}_r$ and $\mathbb{P}_g$ become disjoint, their distance remains constant, preventing $\mathbf{G}$ from receiving meaningful feedback to improve [41]. To address this, alternative loss functions that better define the distance between distributions have been proposed.

Algorithm 1 Mini-batch SGD training of GANs. w and θ are the parameters of D and G , respectively. Adapted from [7].

for number of training iterations **do**
 for number of discriminator steps **do**
 Sample m noise (*fake*) samples $\{x^{(1)}, \dots, x^{(m)}\}$ from p_z .
 Sample m *real* examples $\{x^{(1)}, \dots, x^{(m)}\}$ from $\mathbb{P}_r$.
 Update the discriminator by ascending its stochastic gradient:

$$\nabla_w \frac{1}{m} \sum_{i=1}^m [\log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))]$$

end for

for number of generator steps **do**
 Sample m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from p_z .
 Update the generator by descending its stochastic gradient:

$$\nabla_\theta \frac{1}{m} \sum_{i=1}^m \log(1 - D(G(z^{(i)})))$$

end for

end for

One such alternative is the *Earth-Mover (EM) distance*, also known as Wasserstein-1. Informally, the EM distance measures the cost of transporting the mass of one distribution to another, where the cost depends on the mass and transport distance. Formally, the minimum distance $\mathcal{W}$ is defined in Eq. (2.7), where γ is the joint PMF of $\mathbb{P}_r$ and $\mathbb{P}_g$ (mass), and $\|x - \tilde{x}\|$ is the transport distance. A visualization is shown in Fig. 2.2.

$$\mathcal{W}(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \sim \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x, \tilde{x}) \sim \gamma} \|x - \tilde{x}\| = \inf_{\gamma \sim \Pi(\mathbb{P}_r, \mathbb{P}_g)} \sum_{x, \tilde{x}} \gamma(x, \tilde{x}) \|x - \tilde{x}\| \quad (2.7)$$

Using the Kantorovich-Rubinstein duality, Eq. (2.7) can be expressed as Eq. (2.8):

$$\mathcal{W}(\mathbb{P}_r, \mathbb{P}_g) = \sup_{\|\mathcal{D}\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r} \mathcal{D}(x) - \mathbb{E}_{\tilde{x} \sim \mathbb{P}_g} \mathcal{D}(\tilde{x}), \quad (2.8)$$

where the supremum is over all 1-Lipschitz functions $\mathcal{D}$.

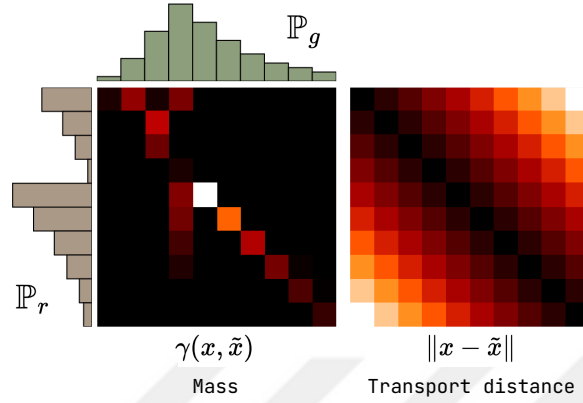


Figure 2.2: Visualization of mass (left) and transport distance (right) for EM. Adapted from [42].

Then, finding the minimum distance $\mathcal{W}$ between $\mathbb{P}_r$ and $\mathbb{P}_g$ is a matter of training neural networks $\mathbf{D}_w$ & $\mathbf{G}_\theta$, assuming that the gradients $\nabla_{\theta,w}\mathcal{W}$ are well-defined and some mild constraints [43, 44] for $\mathbf{D}$ and $\mathbf{G}$ hold. Furthermore, Eq. (2.9) and Eq. (2.10) can also be shown to be true [43, 44], allowing us to write the overall objective in Eq. (2.11):

$$\nabla_\theta \mathcal{W}(\mathbb{P}_r, \mathbb{P}_g) = -\mathbb{E}_{z \sim p_z} \nabla_\theta \mathbf{D}_w(\mathbf{G}_\theta(z)) \quad (2.9)$$

$$\nabla_w \mathcal{W}(\mathbb{P}_r, \mathbb{P}_g) = \mathbb{E}_{x \sim \mathbb{P}_r} \nabla_w \mathbf{D}_w(z) - \mathbb{E}_{z \sim p_z} \nabla_w \mathbf{D}_w(\mathbf{G}_\theta(z)) \quad (2.10)$$

$$\boxed{\min_{\mathbf{G}} \max_{\mathbf{D} \in \mathcal{D}} \mathbb{E}_{x \sim \mathbb{P}_r} \mathbf{D}(x) - \mathbb{E}_{\tilde{x} \sim \mathbb{P}_g} \mathbf{D}(\tilde{x})} \quad (2.11)$$

Eq. (2.11) describes the objective for minimizing $\mathcal{W}$, where $\mathcal{D}$ is the set of 1-Lipschitz functions. Notice that the optimization is very similar to those of traditional GANs but replaces the log with Lipschitz constraints. Eq. (2.11) can be solved via Algorithm 2, which are known as Wasserstein GANs (WGAN) [43, 44] in the literature. Replacing BCE with the Wasserstein-1 distance stabilizes GAN training and improves output quality.

Algorithm 2 Mini-batch training of WGANs. w and θ are the parameters of D and G, respectively. Adapted from [43].

```

for number of training iterations do
  for number of discriminator (critic) steps do
    Sample  $m$  noise (fake) samples  $\{x^{(1)}, \dots, x^{(m)}\}$  from  $p_z$ .
    Sample  $m$  real examples  $\{x^{(1)}, \dots, x^{(m)}\}$  from  $\mathbb{P}_r$ .
    Update the discriminator (critic) by ascending its stochastic gradient:

```

$$\nabla_w \frac{1}{m} \sum_{i=1}^m [\mathbf{D}(x^{(i)}) - \mathbf{D}(\mathbf{G}(z^{(i)}))]$$

```

    Clip gradients or weights of D to heuristically force 1-Lipschitz-ness

```

```

  end for

```

```

  for number of generator steps do

```

```

    Sample  $m$  noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$  from  $p_z$ .

```

```

    Update the generator by descending its stochastic gradient:

```

$$-\nabla_{\theta} \frac{1}{m} \sum_{i=1}^m \mathbf{D}(\mathbf{G}(z^{(i)}))$$

```

  end for

```

```

end for

```

2.2 Neural Rendering and Neural Radiance Fields

2.2.1 3D Representations and Neural Volumetric Rendering

3D representations form the foundation of rendering techniques, providing various ways to model and visualize scenes. These representations are broadly categorized into **explicit**, **hybrid**, and **implicit** (Fig. 2.3), each offering distinct advantages and challenges in rendering.

Explicit representations approximate 3D structures using discrete elements such as voxel grids, meshes, point clouds, 3D Gaussian splats [46], and recent methods like radiant foams [47]. While these methods directly define geometry

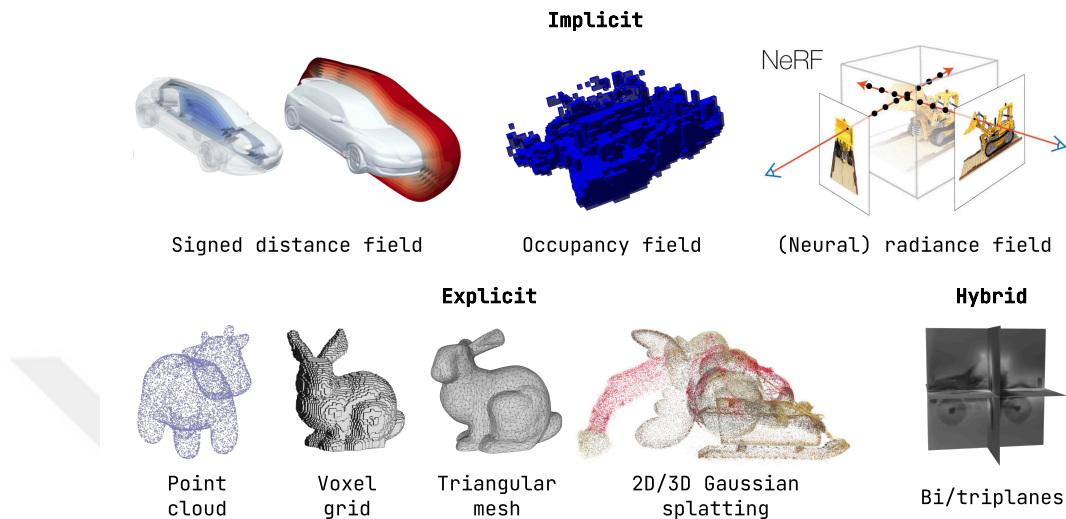


Figure 2.3: Different 3D surface representations. Adapted from [45].

and are compatible with traditional rendering pipelines, they have inherent limitations. The memory usage of voxel grids grows cubically with resolution, meshes struggle to model fine details like hair strands, and ensuring seamless, closed surfaces without artifacts is a significant challenge [48]. These inefficiencies have driven the shift toward more compact and expressive representations.

Implicit representations address these challenges by modeling 3D structures as continuous fields defined by neural networks. Instead of storing discrete geometry, implicit methods parameterize objects or scenes using smooth functions, which can be queried at arbitrary resolutions. Examples include occupancy fields [49], which determine whether a point in space is occupied; signed distance functions (SDFs) [50], which compute the distance from a point to the surface of an object; and radiance fields [1], which encode view-dependent color and density information. Implicit representations are more efficient, expressive, and flexible compared to most explicit ones.

Hybrid representations combine the strengths of both explicit and implicit approaches, offering improved efficiency and interpretability, e.g., triplanes [8].

The development of neural volumetric rendering (NVR) has further revolutionized how 3D scenes are modeled and rendered. Classical volumetric rendering

integrates color and density values along multiple rays shot through the scene. NVR extends this by making the entire process differentiable, allowing neural networks to learn 3D scenes from 2D data via gradient descent. A very popular method Neural Radiance Fields (NeRF) [1] exemplifies this approach.

The topic of neural rendering is very diverse and deserves its own work; therefore, we opted not to give further details on the recent NVR literature in this document. That said, we will briefly mention NVR equations and NeRFs [1] as our work utilizes its fundamental parts in triplanes.

2.2.2 Mathematical Formulation of NVR

For completeness, we will summarize the math behind neural volumetric rendering [51], and clarify how it is tied with NeRF.

We denote a ray in a reparametrized form as $\mathbf{r} = \mathbf{o} + t\mathbf{d}$ running through a 3D medium (Fig. 2.4), where $\mathbf{o}$ is the origin and $\mathbf{d}$ is the unit vector along the ray. Then, $\mathbf{r}$ hitting a particle at point t is probabilistically described as $P(\text{hit at } t) = \sigma(t)dt$, where $\sigma(t)$ is the *volume density*. More informally, the chance that the ray stops in an infinitesimal dt interval around t is $\sigma(t)dt$. On the contrary, $\mathbf{r}$ not hitting a particle traveling over the interval $[0, t)$ is defined by the transmittance function ($\mathcal{T}$), i.e. $P(\text{no hits before } t) = \mathcal{T}(t)$.

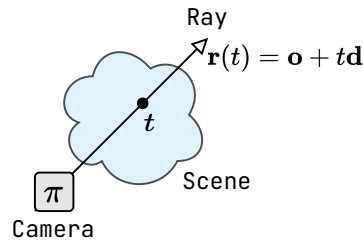


Figure 2.4: Reparametrized ray, coming out of the camera and passing through the scene.

Then, during the travel, the probability of $\mathbf{r}$ not hitting a particle when taking a dt step is the multiplication of not having hit a particle in the interval $[0, t)$,

and not hitting a particle at t (Eq. (2.12))

$$\mathcal{T}(t + dt) = \mathcal{T}(t)(1 - \sigma(t)dt) \quad (2.12)$$

$$\mathcal{T}(t) + \mathcal{T}'(t)dt = \mathcal{T}(t) - \mathcal{T}(t)\sigma(t)dt \quad (2.13)$$

$$-\frac{\mathcal{T}'(t)}{\mathcal{T}(t)}dt = \sigma(t)dt \quad (2.14)$$

$$\log \mathcal{T}(t)|_a^b = - \int_a^b \sigma(s)ds \quad (2.15)$$

$$\mathcal{T}(a \rightarrow b) \triangleq \frac{\mathcal{T}(b)}{\mathcal{T}(a)} = \exp \left(- \int_a^b \sigma(s)ds \right) \quad (2.16)$$

$\mathcal{T}(a \rightarrow b)$ in Eq. (2.16) is defined as the probability that $\mathbf{r}$ travels from a to b without hitting a particle. If $a = 0$ and $b = t$, then $\mathcal{T}(0 \rightarrow t) = \mathcal{T}(t)$ as before. By Eq. (2.16), $\mathcal{T}(t)\sigma(t) = \exp \left(- \int_0^t \sigma(s)ds \right) \sigma(t)$ is the corresponding PDF, which yields the likelihood that $\mathbf{r}$ stops precisely at t without hitting anything before. Then, the expected color value $\mathbf{C}$ returned by the ray is Eq. (2.17):

$$\mathbf{C} = \int_0^D \mathcal{T}(t)\sigma(t)\mathbf{c}(t)dt, \quad (2.17)$$

where $\mathbf{r}$ travels from 0 to D , and $\mathbf{c}$ is the color density at t .

Notice that Eq. (2.17) is a nested integral due to $\mathcal{T}(t)\sigma(t)$ and is intractable. To break up the outer integral, $\mathbf{r}$ is divided into equal segments $\delta_i = t_{i+1} - t_i$, where in each segment we assume a constant σ and $\mathbf{c}$, but non-constant $\mathcal{T}(t)$. Under those constraints, Eq. (2.17) becomes equivalent to Eq. (2.18):

$$\mathbf{C} = \int_0^D \mathcal{T}(t)\sigma(t)\mathbf{c}(t)dt = \sum_{i=1}^n \int_{t_i}^{t_{i+1}} \mathcal{T}(t)\sigma_i\mathbf{c}_i dt \quad (2.18)$$

Using the multiplicative property of transmittance, we can also split up $\mathcal{T}$ into T and $1 - \alpha$, where the first term T denotes how much of the ray is blocked by the previous segments, and the second term $1 - \alpha$ denotes how much is blocked partway through the current segment. Therefore, α itself denotes how much light

has contributed to the current segment i (Fig. 2.5), and is also used synonymously with alpha blending coefficients. Eq. (2.19) demonstrates how to exploit the multiplicative property:

$$\mathcal{T}(t_{i+1}) = \exp\left(-\int_{t_1}^{t_i} \sigma_i ds\right) \exp\left(-\int_{t_i}^{t_{i+1}} \sigma_i ds\right) \quad (2.19)$$

$$= \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right) \exp\left(-\sigma_i \underbrace{(t_{i+1} - t_i)}_{\delta_i}\right) \quad (2.20)$$

$$\triangleq T_i \cdot (1 - \alpha_i) \quad (2.21)$$

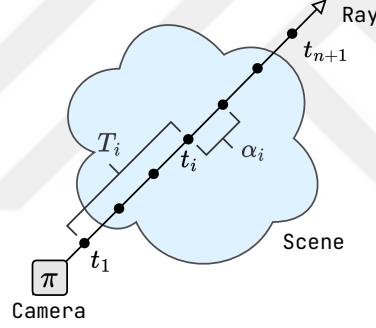


Figure 2.5: The case where the ray has hit a particle in the range (t_i, t_{i+1}) . $\mathcal{T}$ is divided into T_i and $1 - \alpha_i$.

Now, we can simplify Eq. (2.17) as Eq. (2.26):

$$\mathbf{C} = \int_0^D \mathcal{T}(t) \sigma(t) \mathbf{c}(t) dt \quad (2.22)$$

$$= \sum_{i=1}^n \int_{t_i}^{t_{i+1}} \mathcal{T}(t) \sigma_i \mathbf{c}_i dt \quad \text{Eq. (2.18)} \quad (2.23)$$

$$= \sum_{i=1}^n \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right) \sigma_i \mathbf{c}_i \int_{t_i}^{t_{i+1}} \exp(-\sigma_i(t - t_i)) dt \quad \text{Eq. (2.19)} \quad (2.24)$$

$$= \sum_{i=1}^n \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right) \mathbf{c}_i \underbrace{(1 - \exp(-\sigma_i \delta_i))}_{\alpha_i} \quad \text{Eq. (2.21)} \quad (2.25)$$

$$= \boxed{\sum_{i=1}^n T_i \mathbf{c}_i \alpha_i} \quad (2.26)$$

Eq. (2.26) describes the rendered color in a discretized way and establishes the volumetric rendering equations for neural radiance fields. Intuitively, the final color $\mathbf{C}$ is a weighted sum of the colors $\mathbf{c}_i$ of 3D points sampled along this ray, down-weighted by transmittance T_i . Also, notice that $T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$, which establishes a connection to the traditional α -compositing.

It is also worth noting that 3D Gaussian splatting [46] differs from NeRFs by not requiring a neural renderer. However, it does utilize the same concept of volumetric rendering, where the calculations for α are slightly different.

2.2.3 Neural Radiance Fields

Neural radiance fields (NeRFs) estimate color $\mathbf{c}$ and density σ at a 3D location $\mathbf{x}$ and 2D viewing direction $\mathbf{d}$ via a multi-layer perceptron (MLP). Rendering is performed using classical volumetric rendering techniques, as shown in Eq. (2.26), making the 3D scene fully learnable with gradient descent and limited 2D data. The general pipeline of NeRF is visualized in Fig. 2.6.

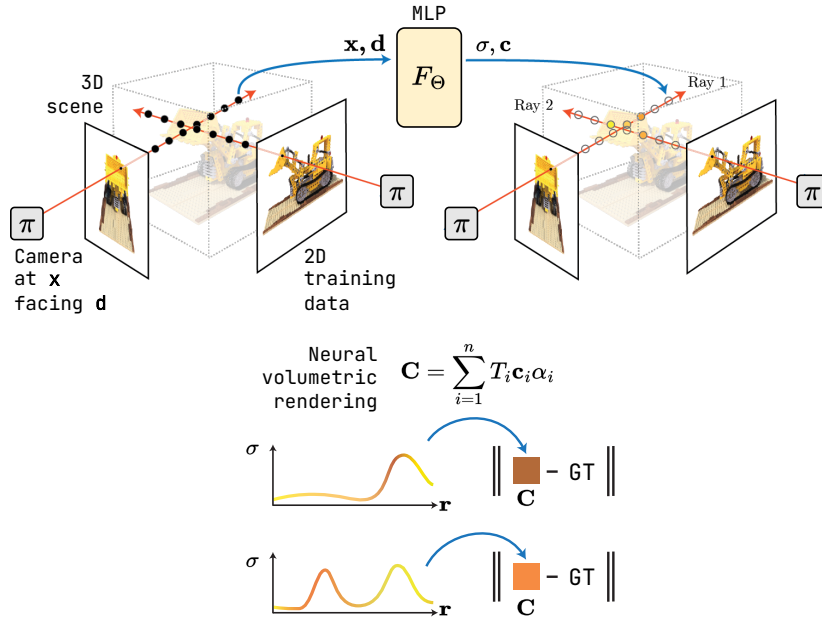


Figure 2.6: General pipeline of NeRF. Adapted from [1].

To improve training and output quality, NeRF employs positional encoding and hierarchical volumetric sampling. Positional encoding captures high-frequency details by processing Fourier-transformed MLP inputs, while hierarchical sampling uses two MLPs (coarse and fine) to focus on the most relevant scene parts.

Since NeRF’s introduction, various extensions have been proposed. Some methods aim to reduce aliasing for higher-quality rendering [52, 53], while others replace the computationally heavy MLP with alternative architectures [54, 55]. Since technically any suitable neural network can replace F_{Θ} in Fig. 2.6, GANs are also utilized for their generative capabilities [56, 8] to learn the radiance and density fields.

2.3 Latent Diffusion Models and Score-based Models

Diffusion models are generative techniques that iteratively remove noise to transform a simple initial distribution into realistic data. A type of diffusion model, Denoising Diffusion Probabilistic Models (DDPMs) [57] achieve this through a forward process q that adds Gaussian noise step-by-step to data $\mathbf{x}_0$ (Eq. (2.27)), and a reverse process p that removes noise to reconstruct the data (Eq. (2.28)):

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1 - \alpha_t)\mathbf{I}), \quad (2.27)$$

$$p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_{\theta}(\mathbf{x}_t, t), \sigma_t^2\mathbf{I}), \quad (2.28)$$

where μ_{θ} is predicted by a neural network $\epsilon_{\theta}(\mathbf{x}_t, t)$ (a denoising convolutional UNet or a diffusion transformer (DiT)). The model is trained to minimize the mean-squared error between the true noise ϵ and the network’s prediction via Eq. (2.29). By learning denoising, DDPMs generate high-quality, realistic samples from pure noise.

$$\mathcal{L} = \mathbb{E}_{t, \mathbf{x}_0, \epsilon} [|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)|^2] \quad (2.29)$$

Latent Diffusion Models (LDMs) [2] improve efficiency of diffusion models by operating in a compressed latent space using a vector-quantized variational autoencoder (VQ-VAE, $\mathcal{E} - \mathcal{D}$ in Fig. 2.7). Eqs. (2.27) and (2.28) occur in this lower-dimensional latent space z , with additional conditioning on external signals τ_θ , significantly reducing computational costs while preserving data quality and enabling controllable generation. LDMs have found widespread applications in image, video, and 3D generation and restoration.

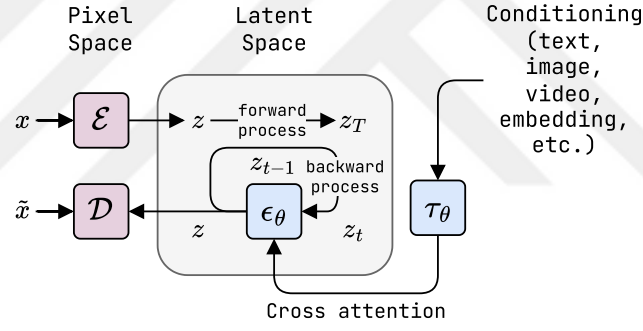


Figure 2.7: Architecture of LDM. Figure adapted from [2].

Score-based models [58] generalize the diffusion process into a continuous-time framework using stochastic differential equations (SDEs). The forward process is represented as Eq. (2.30), where $f(\mathbf{x}, t)$ is a drift term, $g(t)$ is the noise coefficient, and $d\mathbf{w}$ is a Wiener process. The reverse process is represented as Eq. (2.31):

$$d\mathbf{x} = f(\mathbf{x}, t)dt + g(t)d\mathbf{w}, \quad (2.30)$$

$$d\mathbf{x} = [f(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})]dt + g(t)d\mathbf{w}, \quad (2.31)$$

where $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ is the score function, which is learned by training a neural network $s_\theta(\mathbf{x}, t)$ to approximate the gradient of the log data density. Most of the time, $f(\mathbf{x}, t)$ is set to 0.

DDPM can be interpreted as a discretized version of the score-based SDE framework. Specifically, the predicted noise $\epsilon_\theta(\mathbf{x}_t, t)$ in DDPM corresponds to the negative score function (i.e., $\epsilon_\theta(\mathbf{x}_t, t) \propto -\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$).

2.4 Score Distillation Sampling

Score Distillation Sampling (SDS) [24] leverages pre-trained 2D diffusion models to guide the generation of content in other domains, such as 3D. The SDS gradient is defined as:

$$\nabla_\theta L_{\text{SDS}}(\phi, x = g(\theta)) \triangleq \mathbb{E}_{t, \epsilon} [\omega(t)(\hat{\epsilon}_\phi(x_t; y, t) - \epsilon) \frac{\partial x}{\partial \theta}] \quad (2.32)$$

where $x_t = \alpha_t x + \sigma_t \epsilon$ is a noised version of x , $\hat{\epsilon}_\phi(x_t; t)$ is the noise predicted by the diffusion model, $\omega(t)$ weights the timestep contribution, θ is 3D the representation, and g is the differentiable renderer to project 3D to 2D. Intuitively, SDS adds noise ϵ to the rendered output x and updates θ based on the score function of the diffusion model, guiding the target model toward higher-density regions in the diffusion model’s learned distribution. Note that SDS does not require a backward pass through the denoiser $\hat{\epsilon}_\phi$, and θ is directly updated through $\nabla_\theta L_{\text{SDS}}$.

2.5 2D and 3D Style-Based Generators

2.5.1 StyleGAN2

StyleGAN, developed by NVIDIA, is one of the most popular GAN architectures, with three versions [59, 21, 60] improving on each other. These models excel at generating realistic 2D human and animal portraits and allow easy manipulation of results via style vectors. StyleGAN2 [21] is widely preferred for its usability,

pre-trained checkpoints, and interpretable latent space, enabling intuitive edits. Its architecture is shown in Fig. 2.8.

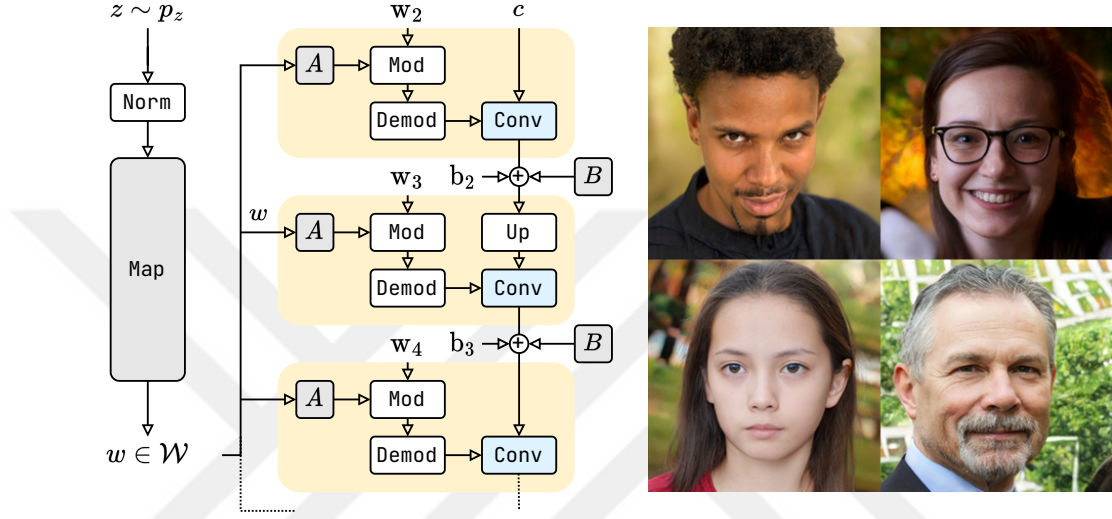


Figure 2.8: Partial architecture of StyleGAN2 and its outputs. Conv block has many sub-layers, but they are omitted for clarity. Adapted from [21].

The generation process works as follows: Latent vectors $\mathcal{Z} \sim p_z$ are normalized and mapped to a 14×512 -dimensional latent space $\mathcal{W}$ via a fully connected mapping network. This network disentangles features and ensures that each layer of the generator $\mathbf{G}$ controls a different style. After mapping, 14 of 1×512 -dimensional style vectors w are injected into different layers of $\mathbf{G}$. These vectors can vary (referred to as $\mathcal{W}^+$ if they are different). The initial input of $\mathbf{G}$ is a learned constant vector c , so the injected w 's ultimately dictate the generated image. Within $\mathbf{G}$, w undergoes a learned affine transformation A and is combined with the convolutional layer weights and biases via modulation layers. Additionally, noise B is injected at various points to enhance detail and provide randomness. Using consciously modified w 's, image editing can be performed in the latent space, and a controlled synthesis can be achieved.

2.5.2 3D Generators, EG3D and PanoHead

GANs have made significant advancements in generating photo-realistic 3D content. In 3D-aware image synthesis, GANs were used in generating 3D explicit representations such as point clouds [61], voxels [62, 63, 64], and meshes [65, 66, 67] from 2D data. The development of neural radiance fields made the implicit representations more prominent.

There are multiple 3D-aware GANs that perform novel image synthesis and utilize volumetric renderers. π -GAN [56] is the first study to use GANs in implicit radiance field representations. StyleNeRF [68] progressively upsamples in 2D feature maps and uses those in the volume rendering via NeRF. StyleSDF [69] utilizes a modified StyleGAN2 backbone with SDF-based volumetric rendering. FeNeRF [70] utilizes decoupled latent codes for semantics and texture with shared geometry for better novel view consistency. EG3D [8] also eliminates the heavy MLP of NeRF and uses a StyleGAN2-based backbone, but they generate orthogonal triplanes for feature extraction and keep the volumetric rendering processes the same with NeRF. LP3D [71] proposes a one-shot method to encode an RGB image into a canonical triplane representation via ViT-based encoders, later to be rendered via a volumetric renderer. PanoHead [6] extends on EG3D with tri-grid triplanes and a more diverse training set. Unlike EG3D, PanoHead is able to synthesize full 360-degree geometry. SphereHead [72] proposes a novel triplane representation in the spherical coordinate system and eliminates mirrored artifacts of PanoHead.

EG3D (Fig. 2.9) and PanoHead generate hybrid 3D representations called *triplanes* with their modified StyleGAN2 backbone (G_θ), which takes 512-dimensional style vectors w and outputs $3 \times N \times 256 \times 256$ resolution triplanes ($N = 32$ for EG3D, $N = 96$ for PanoHead). Rays are shot onto the triplanes, gathered values for each plane are summed and decoded into σ and $\mathbf{c}$ using a lightweight MLP, and rendered from a specified camera pose using a neural volumetric renderer (R_Θ) to synthesize a 2D head image with size 64×64 . Finally, a convolution-based super-resolution (SR) network performs upscaling from 64×64

images to 512×512 final images to refine details. Since the SR network upscales significantly, it plays a critical role in maintaining geometric consistency while prioritizing a rich texture and color.

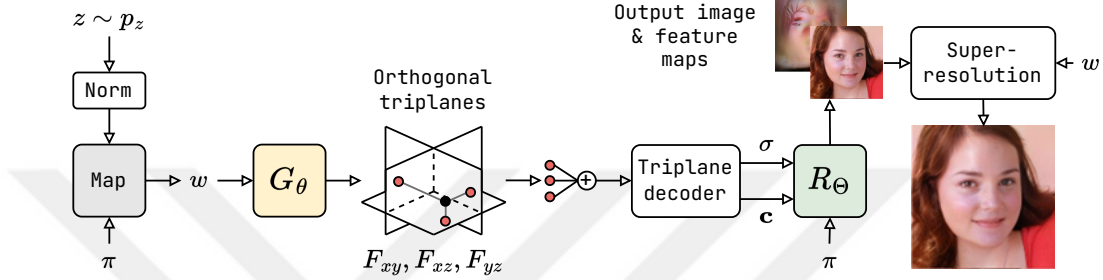


Figure 2.9: EG3D architecture, without the discriminator.

2.6 GAN Inversion, Editing, and Stylization

2.6.1 GAN Inversion

GAN inversion is the task of finding/estimating the latent vectors z by observing the corresponding generator outputs $\mathbf{G}(z)$. It is an ill-posed problem since the generator $\mathbf{G}_\theta$ is not technically invertible (unless $\mathbf{G}$ belongs to normalizing flows [73]) and the dimensionality of the latent space is much smaller than those of $\mathbf{G}$'s outputs, hence it is a challenging task. Nonetheless, if done successfully, the biggest advantage of GAN inversion is *editability*. In other words, one can find the latent vector correspondence of an image and perform editing in the latent domain. Editing in the pixel domain is less desired since achieving satisfying results is harder without human supervision.

Fig. 2.10 demonstrates different approaches to GAN inversion. *Optimization-based* approaches [74, 75] start with a random w and directly optimize w and/or the parameters of $\mathbf{G}_\theta$; hence $\mathbf{G}_\theta$ is aimed to overfit to output a specific image most of the time. On the other hand, the *encoder-based* ones [12, 27, 76, 26]

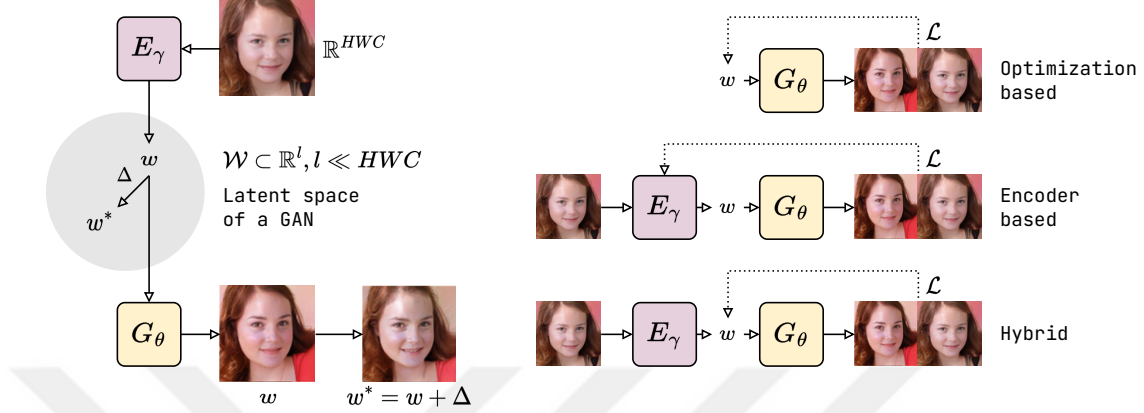


Figure 2.10: Various GAN inversion approaches in the literature (right). Encoder-based inversion (left). Edits such as adding a smile can be done by linear operations (Δ is the smile direction) inside the latent space.

freeze $\mathbf{G}_\theta$ and try to learn the latent space of $\mathbf{G}_\theta$ via an encoder $\mathbf{E}_\gamma$. Encoder-based approaches are more generalizable, and the editability is good, but they lack the superior reconstruction quality of the optimization-based ones. *Hybrid methods* [13] try to take the best of both, where they train a rough encoder $\mathbf{E}_\gamma$ and perform fine-tuning of the w 's with optimization.

One major inversion work is Pivotal Tuning Inversion (PTI) [75], which roughly estimates a pivot w_p by latent code optimization while keeping $\mathbf{G}_\theta$ frozen and then tunes $\mathbf{G}_\theta$ around w_p to obtain the final result. The method uses a locality regularization to ensure that only the latent space around w_p is altered, avoiding possible distortions for other latent vector outputs.

Besides PTI, with the recent development of 3D GANs, new 3D-aware hybrid and optimization-based GAN inversion methods have also emerged. IDE-3D [77] offers a hybrid GAN inversion method. By modifying the EG3D generator, they generate both a textural and a semantic triplane to enable disentangled editing via semantic map manipulations. In addition, in order to eliminate the pose bias for faithful reconstruction and novel portrait synthesis, they employ semantic & canonical encoders combined with PTI [75] based tuning. Pose optimization [15] jointly optimizes latent code and pose inversion for multi-view consistent inversion and editing. A hybrid approach is chosen; e4e [27] is used for latent code

estimation, and their custom pose encoder for the pose estimation. PTI-based optimization is applied for refining the encoder outputs at the end. E3DGE [78] offers an alignment module for high-quality editing and local feature fusion for high-fidelity 3D-GAN inversion. They train their encoder with 3D meshes & 2D multi-view pairs synthesised from the generator, and perform feature fusion of multiple views for consistent novel view generation. HFGI3D [16] utilizes a pseudo-view estimation pipeline for keeping visible textures intact and generating new textures on occluded regions in novel view rendering. More specifically, they optimize a latent code for rough geometry and visible part reconstruction, optimize another latent code for occluded part generation based on a binary difference map calculated using out-of-distribution textures, and perform a final optimization by the previously estimated two pseudo-multi-view supervisions in $\mathcal{W}^+$ space. SPI [79] exploits facial symmetry of human faces, and uses this as a prior in generating novel views. With the prior information, the method first performs a rough inversion with optimization, then a joint optimization with novel views while applying depth-guided 3D warping as a regularization.

Although few, there are also encoder-based methods tailored for 3D GANs. TriPlaneNet [18] first extracts a rough inversion via [12], and fine-tunes the result with estimated triplane offsets. GOAE [17] has a similar approach to [18]; however, it adds an adaptive feature alignment block for better triplane offset addition, and an occlusion-aware mix tri-plane regularizer to perform faithful edits and novel view reconstructions.

2.6.2 Editing and Stylization in GANs

There are abundant editing methods for GAN inversion in the literature [80, 14, 3, 4, 20, 19]. GANSpace [3] finds out that principal components of feature tensor activations on the early layers of generators (or the mapping networks) represent important factors of variation and uses those components for editing. InterfaceGAN [4] assumes a linear hyperplane in the generator’s latent space as a separation boundary for two binary attributes (male-female, old-young, etc.)

and fits a support vector classifier (SVC) to obtain the boundary vectors for editing. For disentangling attributes, they conditionally manipulate the vectors by applying orthogonal projection. StyleCLIP [20] converts the text prompts into CLIP [81] embeddings and embeds them into style vectors with their custom mapping network. StyleGAN-NADA [19] trains another generator in parallel with the feedback from CLIP embedding data without using gradients from any image information and can generate out-of-domain edits. Fig. 2.11 compares two mostly used and straightforward editing methods. Notice that the latent space of GAN is not fully visualizable due to its high dimensionality; therefore, a 2D representation was chosen.

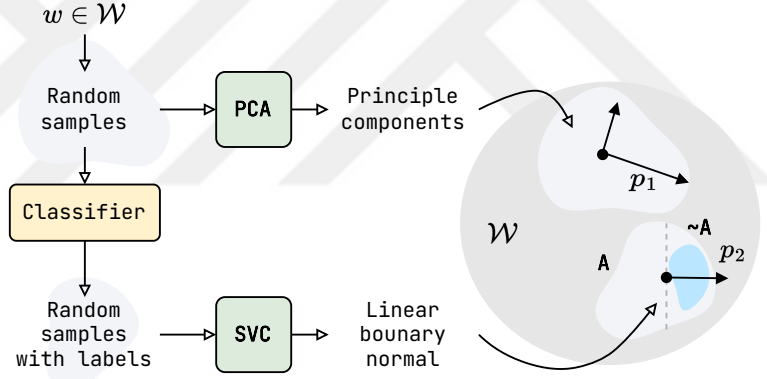


Figure 2.11: Representative comparison of [3] and [4] finding the editing directions $p_{1,2}$. Häarkönen *et al.* [3] finds the most dominant direction p_1 via principle component analysis (PCA), and Shen *et al.* [4] finds the opposite binary attributes' (A and $\sim A$) linear boundary normal p_2 by fitting sampled w 's to a SVC. Attribute label scores of w 's are generated from an external classifier.

Chapter 3

High-fidelity 3D Head Reconstruction from Single Images

In this chapter, we present a new method for high-fidelity 3D head reconstruction from single images, built on recent advances in 3D GANs and GAN inversion [9]. While encoder-based approaches have shown strong results in this area, many rely on EG3D, which is mainly designed for synthesizing near-frontal views. This limitation constrains their performance in capturing comprehensive 3D head geometry. To overcome these shortcomings, we propose a new framework based on PanoHead, which excels in generating 360-degree views and provides a richer representation of the head structure from arbitrary angles.

Our approach introduces a dual encoder system that aims to strike a balance between accurate reconstruction and realistic view generation from multiple angles. To combine the strengths of the two encoders, we propose a stitching framework in the triplane domain that merges their outputs into a unified representation. For this stitching to work effectively, both encoders need to produce consistent features despite being optimized for different tasks. We ensure this

through a carefully designed training strategy that includes a novel occlusion-aware triplane discriminator. This discriminator helps guide the model toward more coherent and realistic results by applying adversarial learning. Experimental results show that our method outperforms existing encoder training techniques, both qualitatively and quantitatively, as illustrated in Fig. 3.1.

The rest of this chapter details the methodology, implementation details, and results of the aforementioned work **B. Bilecen, A. Gökmen, A. DüNDAR**, *Dual Encoder GAN Inversion for High-Fidelity 3D Head Reconstruction from Single Images*, *NeurIPS 2024* [9]. Bold names are joint first authors.

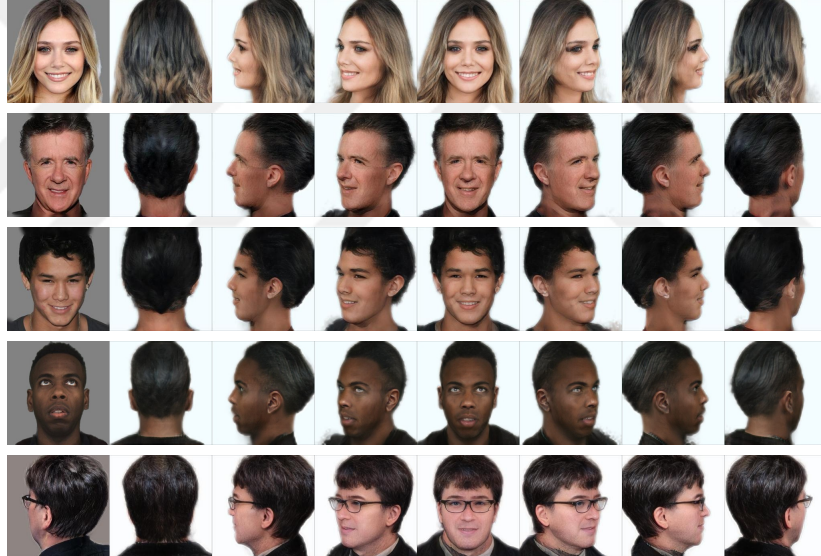


Figure 3.1: Reconstruction of 3D representation by inverting images into PanoHead’s latent space.

3.1 Method

3.1.1 Training an encoder

This section introduces the general pipeline of the encoders employed in our dual-encoder framework. Each encoder takes an input image $\mathbf{I}$ and predicts the latent code w^+ , which is then passed to the generator to produce triplane features.

These synthesized features are then fed into the renderer to generate a 2D image with a specified camera parameter π_{cam} , as described by Eq. (3.1):

$$\begin{aligned} w^+ &= \mathbf{E}_1(\mathbf{I}) \\ \mathbf{I}_{\text{out}}^{\text{sv}} &= \mathcal{R}(\mathbf{G}(w^+), \pi_{cam}) \end{aligned} \quad (3.1)$$

Here, $\mathbf{I}_{\text{out}}^{\text{sv}}$ denotes the output rendering for the same view as the input, and $\mathbf{E}_1$ represents the encoder. While the $\mathcal{W}^+$ space allows for leveraging priors embedded in the generator, its limited expressive power in reconstructing image details has been noted due to the information bottleneck of its 14×512 dimensions. To address this limitation, both 2D inversion techniques [76, 14] and 3D GAN inversion methods [18] permit higher-rate features to pass to the generators, facilitating the capture of fine details. In 3D GAN inversion methods, these higher-rate features are encoded through a smaller second network and transmitted to the triplane features. We adopt a similar approach in our encoders. We refer to the final output as $\mathbf{I}_{\text{final}}^{\text{sv}}$. Further details of the architecture are provided in Section 3.2.2.

The primary challenge in this setting arises from establishing appropriate training objective losses, as our training dataset consists solely of single images, providing ground truth only for the rendered image from the same view as the input. For these output and ground-truth pairs, we set the usual reconstruction losses, namely, LPIPS perceptual loss [82], $\mathcal{L}_2$ reconstruction loss (MSE), and ArcFace [83] based identity loss. The training objective for **Encoder 1** with parameters $\theta_{\mathbf{E}_1}$ is given in Eq. (3.2):

$$\min_{\theta_{\mathbf{E}_1}} \lambda_1 \mathcal{L}_{\text{LPIPS}}(\mathbf{I}_{\text{final}}^{\text{sv}}, \mathbf{I}) + \lambda_2 \mathcal{L}_2(\mathbf{I}_{\text{final}}^{\text{sv}}, \mathbf{I}) + \lambda_3 \mathcal{L}_{\text{identity}}(\mathbf{I}_{\text{final}}^{\text{sv}}, \mathbf{I}), \quad (3.2)$$

where the coefficients are set as $\lambda_1 = 0.8$, $\lambda_2 = 1.0$, $\lambda_3 = 0.5$.

While models trained with the objective given in Eq. (3.2) learn to reconstruct a given view, they often struggle to generalize and produce realistic features from

other camera views. Consequently, we design an adversarial-based loss objective for our second encoder. This second encoder generates realistic predictions for invisible views, as explained in Section 3.1.2.

3.1.2 Occlusion-aware triplane discriminator

To achieve a realistic reconstruction of the 3D model, represented in a triplane structure, it is essential to guide the encoder for visible views and overall coherence. Since we lack a one-to-one ground truth to guide the triplane structure, we experiment with various setups incorporating adversarial losses. A naive approach to utilize adversarial loss would be to render estimated triplanes from other views and assess the realism of these 2D images using a discriminator. However, our experiments observe that this setup hinders the model’s ability to learn high fidelity to the input image, as will be further detailed in Section 3.3.2. Moreover, randomly rendering different views can only guide limited parts of the triplane structure rather than the overall.

To overcome this limitation and avoid the computational burden of rendering unnecessary views, we explore the possibility of training a discriminator in the triplane domain. In our training process for the triplane discriminator, we follow a procedure where we sample latent vectors $\mathcal{Z}^+$ and generate in-domain triplanes using PanoHead, serving as our real samples. Meanwhile, the fake samples are triplanes generated from encoded images, as depicted in Fig. 3.2. Despite the observed improvement in reconstructions facilitated by this adversarial loss, we note a persistent challenge hindering the network’s ability to achieve high fidelity to the input. This discrepancy may stem from the real samples originating from the generator, lacking the detailed feature characteristic of real-world images. Therefore, this may lead the encoder to omit encoding realistic facial details if they are absent in the synthesized samples.

We propose our **occlusion-aware discriminator** to overcome this limitation. This discriminator is exclusively trained with features corresponding to occluded pixels. This approach ensures that triplane features associated with

visible regions, such as a frontal face, are not utilized for discriminator training. Additionally, we introduce a masking mechanism for synthesized triplanes to mitigate any distribution mismatch arising between encoded and synthesized triplanes. This masking process contributes to aligning the distributions of real and fake samples, further enhancing the coherence of the training dynamics.

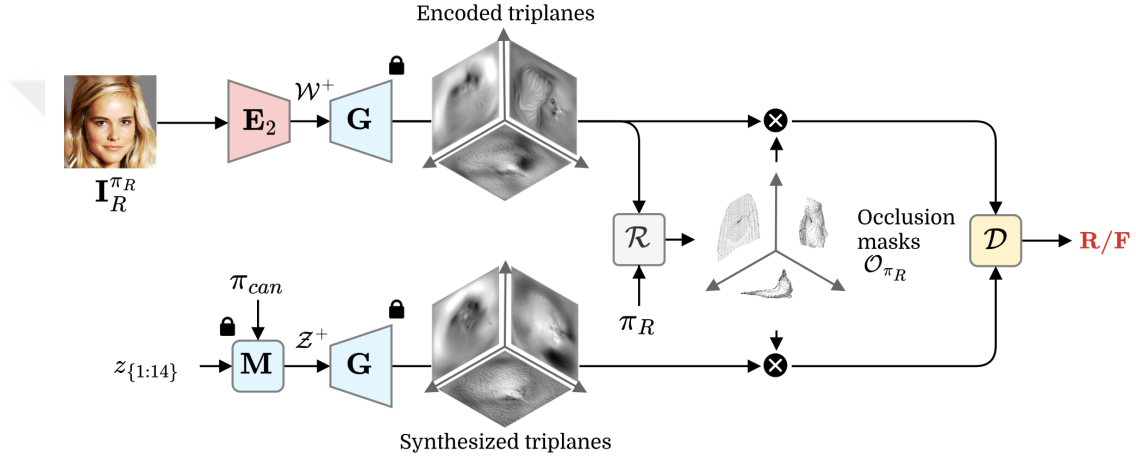


Figure 3.2: Our triplane discriminator training involves generating real samples from $\mathcal{Z}^+$ and producing in-domain triplanes using PanoHead. Fake samples are generated from encoded images. Despite the effectiveness of adversarial loss in enhancing reconstructions, challenges may persist in achieving high fidelity to the input due to the origin of real samples from the generator $\mathbf{G}$. To address this, we use an **occlusion-aware discriminator** $\mathcal{D}$, trained exclusively with features from occluded pixels. This ensures that visible regions, such as frontal views π_R , have reduced influence while training $\mathcal{D}$.

We find the set of visible points based on the depth map of the given view via inverse rendering. Specifically, the occlusion mask $\mathcal{O}_{\pi_R}$ is estimated by Eq. (3.3):

$$\mathcal{O}_{\pi_R} = \mathbb{R}^3 \setminus \{\mathbf{p}[x, y, z] : \pi_R \mathcal{D} \mathcal{K}^{-1} \mathbf{I}[u, v, 1]^T\}, \quad (3.3)$$

where $\mathbf{p}[x, y, z]$ is the triplane coordinates, π_R is the extrinsic camera parameters of the input view, $\mathcal{D}$ is the depth map from the input view, $\mathcal{K}$ is the intrinsic camera parameters, $\mathbf{I}[u, v, 1]$ are the homogeneous coordinates of the input image $\mathbf{I}$ rendered from the input view. More clearly, from the current camera pose, we map back to the depth values to obtain the mask of visible regions $(1 - \mathcal{O}_{\pi_R})$.

Then, we invert the visible region mask to obtain $\mathcal{O}_{\pi_R}$. The utilization of occlusion masks has been previously investigated in 3D methodologies, albeit in different contexts. For instance, they have been used in generating pseudo-ground truth images to facilitate optimization-based 3D reconstruction [79] and integrated into passing high-rate residual features to the triplane [17]. However, it is the first time used in the discriminator. This allows for a selective focus on regions where the encoder may encounter challenges in faithfully replicating realism.

The training objective for **Encoder 2** with parameters $\theta_{\mathbf{E}_2}$ is given in Eq. (3.4).

$$\begin{aligned} \mathbf{T}_{\text{enc}} &= \mathbf{G}(\mathbf{E}_2(\mathbf{I})), \\ \mathbf{I}^{\text{sv}} &= \mathcal{R}(\mathbf{T}_{\text{enc}}, \pi), \\ \min_{\theta_{\mathbf{E}_2}} \quad &\lambda_1 \mathcal{L}_{\text{LPIPS}}(\mathbf{I}^{\text{sv}}, \mathbf{I}) + \lambda_2 \mathcal{L}_2(\mathbf{I}^{\text{sv}}, \mathbf{I}) + \lambda_3 \mathcal{L}_{\text{identity}}(\mathbf{I}^{\text{sv}}, \mathbf{I}) + \lambda_4 \mathcal{L}_{\text{adv}}(\mathcal{D}(\mathcal{O}_{\pi} \mathbf{T}_{\text{enc}})), \end{aligned} \quad (3.4)$$

where $\mathbf{T}_{\text{enc}}$ is the encoded triplane of image $\mathbf{I}$, $\mathcal{D}$ is the discriminator, $\mathcal{O}_{\pi}$ is the occlusion mask from the same view π . $\lambda_{1,2,3}$ are the same as in Eq. (3.2), and $\lambda_4 = 0.001$. $\mathcal{L}_{\text{adv}}$ in Eq. (3.4) is given in Eq. (3.5):

$$\mathcal{L}_{\text{adv}}(x) = \text{softplus}(-x), \quad (3.5)$$

where **softplus** is a smooth and differentiable approximation to ReLU.

The training objective for **discriminator** $\mathcal{D}$ with parameters $\theta_{\mathcal{D}}$ is given in Eq. (3.6).

$$\begin{aligned} \mathbf{T}_{\text{enc}} &= \mathbf{G}(\mathbf{E}_2(\mathbf{I})), \\ \mathbf{T}_{\text{synth}} &= \mathbf{G}(\mathbf{M}(z^+ \sim \mathcal{N}(0, \mathbf{I}_{14}), \pi_{\text{front}})), \\ \min_{\theta_{\mathcal{D}}} \quad &\lambda \mathcal{L}_{\text{adv}}(\mathcal{D}(\mathcal{O}_{\pi} \mathbf{T}_{\text{enc}}), \mathcal{D}(\mathcal{O}_{\pi} \mathbf{T}_{\text{synth}})), \end{aligned} \quad (3.6)$$

where $\mathbf{T}_{\text{synth}}$ is the synthesised triplane from randomly sampled z^+ , $\mathbf{M}$ is the

mapping network, π_{front} is the canonical front pose. $\mathcal{L}_{\text{adv}}$ in Eq. (3.6) is given in Eq. (3.7):

$$\mathcal{L}_{\text{adv}}(x, y) = \text{softplus}(-x) + \text{softplus}(y) \quad (3.7)$$

λ is set as 0.5.

We jointly train Encoder 2 and the discriminator $\mathcal{D}$ in a traditional adversarial fashion. We further employ R1-regularization to encourage L1-Lipschitzness [44] to justify using `softplus`, where its weighting coefficient is 10 and is applied every 16 iterations.

3.1.3 Dual encoder pipeline

In our approach, we train two encoders: the first, as outlined in Section 3.1.1, and the second, augmented with an additional adversarial loss detailed in Section 3.1.2. While the initial encoder excels at reconstructing high-fidelity facial images from the input, it often produces unrealistic results for other viewpoints, as depicted in Fig. 3.3. Conversely, the second encoder yields better overall outcomes, albeit with slightly diminished fidelity to the input face.



Figure 3.3: Visual results of Encoder 1, Encoder 2, and dual encoders for the given input images in Columns 1 and 6.

Our aim is to devise a dual-encoder pipeline that harnesses the strengths of both encoded features. To achieve this, we leverage the occlusion masks derived

in Section 3.1.2. By combining the visible portions from Encoder 1 and the occluded segments from Encoder 2, we generate our final output, as illustrated in Fig. 3.4. Sample outputs are visualized in Row 3 of Fig. 3.3.

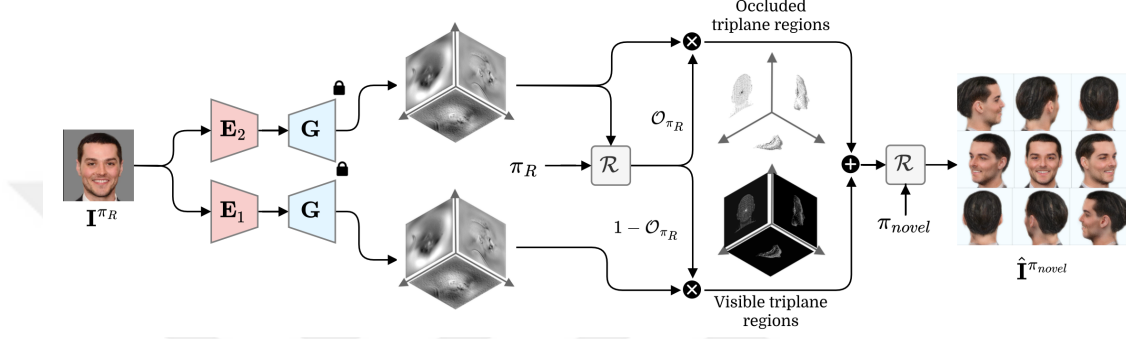


Figure 3.4: The inference pipeline with dual encoders for full 3D head reconstruction. Given a face portrait with pose π_R , we can perform 360-degree rendering from any given pose π_{novel} .

While each encoder contributes partially to the ultimate feature, achieving seamless integration necessitates consistency in the output of both encoders despite their distinct specializations. For instance, if Encoder 1 flawlessly renders a given view of the face but fails to capture the correct geometry, artifacts may arise in the combined result. Thus, it remains imperative to train both encoders comprehensively to ensure an overall high-quality outcome.

3.2 Experimental Setup

3.2.1 Training

We combined images of FFHQ [59] and LPFF [84] and split it for training ($\sim 140k$) and validation ($\sim 14k$). CelebA-HQ [85] and multi-view MEAD [86] are employed for additional evaluation. We removed face portrait backgrounds for training and evaluation datasets, applied camera and image mirroring during training, and performed pose rebalancing proposed in [84] as data augmentation. We utilized the same dataset to train competitive methods for fair evaluation. The models

are trained for 500k iterations with a batch size of 3 on a single RTX 4090 GPU. The learning rate is $1e^{-4}$ for both encoders and the occlusion-aware discriminator. Ranger is utilized as the optimizer, which is a combination of Rectified Adam [87] with Lookahead [88].

3.2.2 Architecture details

Encoder 1 and 2. For our Encoder 1 and 2, we employ 2-stage encoding for $\mathcal{W}^+$ and high-rate $\mathcal{F}$ features, also seen in common with other style-based encoder methods [12, 26, 13, 17, 18]. We opted for the architecture in [12] for $\mathcal{W}^+$ stage (`GradualStyleEncoder`) and in [18] for $\mathcal{F}$ stage (`TriplanenetEncoder`), both for Encoder 1 and 2.

Ablation encoders. `GradualStyleEncoder` is used for the $\mathcal{Z}^+$ encoder, where the resulting 14 latent vectors are later passed through the mapping network with truncation $\psi = 0.85$ and canonical front camera pose. However, for the $\mathcal{Z}$ encoder, a smaller variation of [12] (`BackboneEncoderUsingLastLayerIntoW`) is implemented. This choice is due to $\mathcal{Z}$ being less expressive compared to $\mathcal{Z}^+$ (1 dim vs. 14 dim) and hence a smaller encoder being sufficient. The same mapping network parameters for the $\mathcal{Z}^+$ case are also utilized for $\mathcal{Z}$ encoder outputs.

Occlusion-aware triplane discriminator. We follow a feedforward network approach with a channel bottleneck for our triplane discriminator $\mathcal{D}$ (Table 3.1). Noting that the occluded triplane dimensions are `[batch_size, 3, 96, 256, 256]`, we first add each depth slice to the channel dimension to get the input shape as `[batch_size, 288, 256, 256]`. Output dimensions are `[batch_size, 1]`. We do not utilize saturating functions such as sigmoid at the end since WGAN-based loss is utilized.

Ablation image discriminator. For the back-view image discriminator used in ablations, we change the input channel number of the model in Table 3.1 from 288 to 3.

Table 3.1: Architecture for the discriminator. Conv2D parameters are: (input channels, output channels, kernel size, stride, padding), respectively. Bias terms are disabled.

Conv2D(288,64,3,1,1) LeakyReLU(0.2)
Conv2D(64,128,3,1,1) BatchNorm2D(128) LeakyReLU(0.2)
Conv2D(128,256,3,1,1) BatchNorm2D(256) LeakyReLU(0.2)
Conv2D(256,512,3,1,1) BatchNorm2D(512) LeakyReLU(0.2)
Conv2D(512,1,3,1,0) AvgPool(1)

3.2.3 Baselines

The baseline models are provided in Table 3.2. We note that no encoder pipelines are aimed at full 360-degree head reconstruction. We train the models with the author’s released code to invert images into PanoHead’s latent space. We run experiments with optimization-based [75, 74] and encoder-based methods, e4e and pSp [27, 12], proposed for StyleGAN-based architectures. We also include methods proposed for EG3D, namely, TriplaneNetv2 and GOAE [18, 79].

3.2.4 Evaluation metrics

We report $\mathcal{L}_2$, LPIPS [82], and ID [83] scores for original-view reconstruction, which measure the fidelity to the input image. For the novel-view quality, we measure Fréchet inception distance (FID) [89]. Since our validation datasets have limited angle variance, we measure the distance between 1000 randomly synthesized and 1000 encoded real-life image distributions. The images are rendered from varying yaw angles, covering the 360-degree range to include occluded regions. We also utilize the multi-view image dataset MEAD dataset. Specifically,

we fed front MEAD images (0° yaw) to all methods, rendered them from novel views of MEAD (from 60° to 0° yaw), and compared them with their corresponding ground truths. This allows us to report LPIPS and ID metrics alongside the FID metric for the novel views.

3.3 Results

3.3.1 Comparison with state-of-the-art

Table 3.2 provides quantitative comparisons against state-of-the-art optimization and encoder-based methods. GOAE achieves significantly better same-view reconstruction scores (L2, LPIPS, and ID), however, shows much worse FID scores, indicating their inability to produce realistic views. While TriplaneNetv2 achieves similar same-view reconstruction scores as our method, its FID score is also significantly worse. Overall, the pSp and e4e methods perform worse than ours in all metrics. PTI achieves similar results to our method but takes $250\times$ the computation time of ours and requires a GPU with large memory.

Table 3.2: Quantitative scores on various test sets.

Category	Method	FFHQ + LPFF				CelebAHQ				Time sec ↓
		$\mathcal{L}_2$ ↓	LPIPS ↓	ID ↑	FID ↓	$\mathcal{L}_2$ ↓	LPIPS ↓	ID ↑	FID ↓	
Optimization	$\mathcal{W}^+$ optim. [74]	0.035	0.17	0.57	65.30	0.049	0.19	0.49	63.57	49.19
	PTI [75]	0.019	0.11	0.89	59.40	0.033	0.13	0.86	57.85	86.52
Encoder	pSp[12]	0.028	0.15	0.77	90.52	0.033	0.17	0.70	88.20	0.08
	e4e[27]	0.064	0.24	0.52	82.99	0.080	0.26	0.42	84.70	0.05
	TriplaneNetv2[18]	0.017	0.10	0.86	98.97	0.020	0.11	0.80	99.02	0.17
	GOAE[17]	0.015	0.09	0.87	168.96	0.017	0.10	0.86	173.64	0.15
	Ours	0.017	0.10	0.87	65.44	0.021	0.12	0.84	62.58	0.37

We extend the quantitative analyses to multi-view with the MEAD dataset. Specifically, we feed front MEAD images (0° yaw) to all methods, render them from novel views of MEAD (from 60° to 0° yaw), and compare them with their corresponding ground truths. Table 3.3 reveals that our method significantly improves over compared methods, especially in LPIPS and ID metrics.

Table 3.3: Quantitative scores on multi-view MEAD dataset.

Category	Method	LPIPS ↓		ID ↑		FID ↓	
		$\pm 60^\circ$	$\pm 30^\circ$	$\pm 60^\circ$	$\pm 30^\circ$	$\pm 60^\circ$	$\pm 30^\circ$
Optim.	$\mathcal{W}^+$ opt.	0.249	0.200	0.570	0.558	48.473	43.875
	PTI	0.346	0.262	0.515	0.548	66.399	53.977
Encoder	pSp	0.245	0.189	0.640	0.650	49.364	48.098
	e4e	0.318	0.265	0.544	0.462	67.399	70.854
	Tpn.v2	0.248	0.192	0.658	0.663	48.937	46.493
	GOAE	0.296	0.249	0.654	0.660	87.644	92.758
	Ours	0.223	0.178	0.706	0.726	47.207	43.822

Qualitative results are shown in Figs. 3.5 and 3.6. The competing methods produce unrealistic outputs when viewed from angles other than the input view. Among these, PTI achieves good front and side views but fails to generate realistic hair from the back. Our method achieves the best results overall.

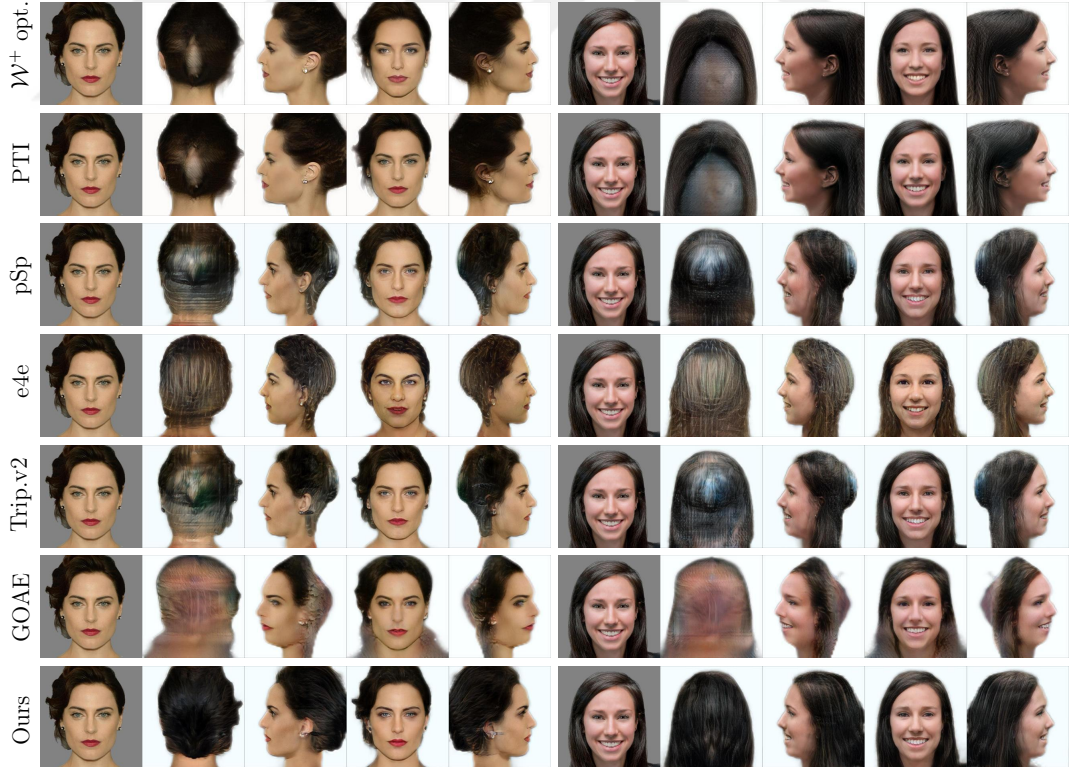


Figure 3.5: Comparisons of ours and competing methods.

We also include mesh comparisons in Fig. 3.7. Ours is better than the most recent encoder-based method [18] and generally performs well compared to PTI. Note that PTI mostly generates smoother meshes (Row 3) but can sometimes

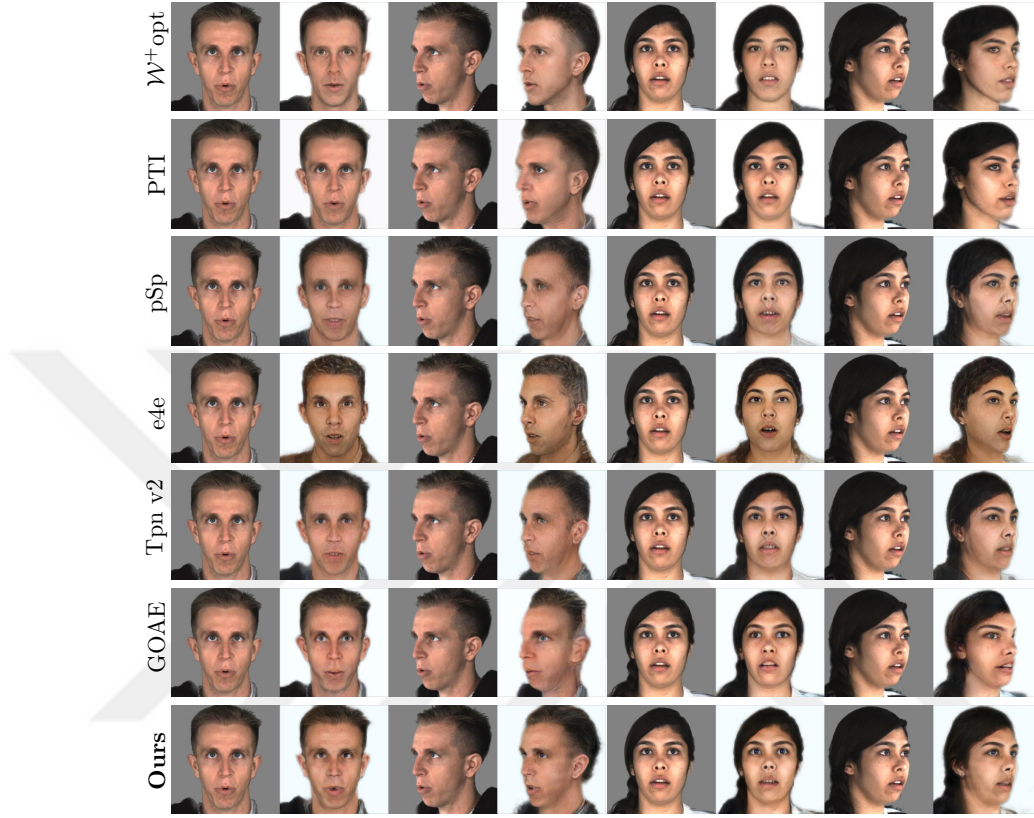


Figure 3.6: Edited image results. From left to right: input (0°), reconstruction (0°), target ($\pm 60^\circ$), and render on $\pm 60^\circ$ using the reconstructed 0° triplanes.



Figure 3.7: Inputs (first), reconstructions (second), and 360° mesh renders (rest).

struggle depending on the input sample (Row 6).

3.3.2 Ablation study

In Table 3.4 and Fig. 3.8, we present an ablation study demonstrating the effectiveness of our occlusion-aware triplane discriminator quantitatively and qualitatively. The first Row of results shows that not using any discriminator achieves good reconstruction of the given view, as indicated by LPIPS and ID scores. This is also visible in the first Row in Fig. 3.8. However, this approach fails to generalize to novel views, as evidenced by the FID score and visual results.

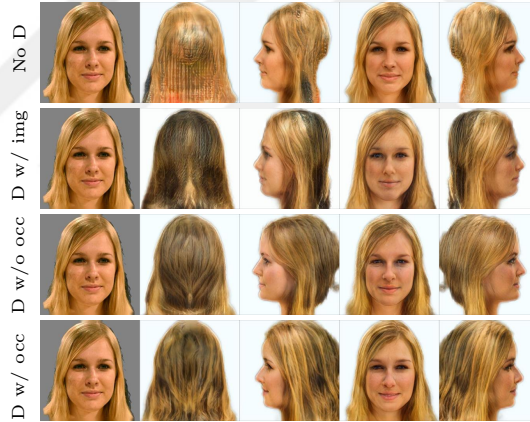


Figure 3.8: Qualitative results of ablation on occlusion-aware discriminator $\mathcal{D}$.

Table 3.4: Ablation on occlusion-aware discriminator $\mathcal{D}$.

	LPIPS ↓	ID ↑	FID ↓
No $\mathcal{D}$	0.10	0.87	89.50
$\mathcal{D}$ w image domain	0.17	0.67	72.86
$\mathcal{D}$ w/o triplane occ.	0.15	0.70	66.24
$\mathcal{D}$ w/ triplane occ.	0.14	0.75	64.02

On the other hand, training the model with an additional adversarial objective that operates on novel images generated using randomly sampled camera parameters improves the FID score but significantly harms the fidelity of the input image. Training a discriminator in the triplane domain and applying adversarial losses from this domain improves overall scores compared to training the discriminator in the 2D image domain. However, as seen in Fig. 3.8 (third Row), the face still

lacks high fidelity to the input, and other views are unrealistic. Lastly, using the occlusion-aware triplane discriminator improves identity fidelity and FID scores. The hair looks more natural, similar to the ones generated by the model when sampled from z .

In our framework, we chose to embed images into the $\mathcal{W}^+$ space. Table 3.5 presents an ablation study that explores utilizing different projection spaces and various combinations of training data. Training an encoder to project images to the $\mathcal{Z}$ or $\mathcal{Z}^+$ space, where $\mathcal{Z}$ is sampled 14 times, results in better FID scores. However, this comes at the cost of high-fidelity reconstruction. Similarly, transitioning to a less constrained $\mathcal{W}^+$ space enhances fidelity to the input but worsens the FID score. Addressing this challenge necessitates additional measures, such as the proposed dual encoder setup with the occlusion-aware discriminator objective. It is important to note that the distinction between the $\mathcal{Z}^+$ and $\mathcal{W}^+$ space arises from the camera parameters incorporated into the mapping network. While $\mathcal{Z}^+$ employs various samples of $\mathcal{Z}$, it adheres to the same set of camera parameters assigned to the mapping network. In contrast, the $\mathcal{W}^+$ space does not impose such constraints during encoding. Given that the real image dataset primarily consists of limited camera poses, typically front-view faces, we investigate training the encoder with synthetically generated images from PanoHead. However, solely utilizing synthetic images generated from samples of $\mathcal{Z}^+$ to introduce more diversity compared to $\mathcal{Z}$ leads to poor performance on real image validation sets regarding reconstruction quality. When combining synthetic and real images, we observe an improvement compared to using them individually.

Table 3.5: Ablation on training data and latent space.

Train data	Proj.	LPIPS ↓	ID ↑	FID ↓
Real img.	$\mathcal{Z}$	0.29	0.31	45.79
Real img.	$\mathcal{Z}^+$	0.22	0.60	76.54
Real img.	$\mathcal{W}^+$	0.10	0.86	98.97
$\mathcal{Z}^+$ gens.	$\mathcal{W}^+$	0.27	0.25	46.93
Real imgs. + $\mathcal{Z}^+$	$\mathcal{W}^+$	0.10	0.87	89.50

Lastly, we show the results of using the dual encoder in Table 3.6. The visual results were previously presented in Fig. 3.3. The dual mechanism leverages the strengths of both encoders, achieving the same LPIPS and ID scores as Encoder

Table 3.6: Ablation on dual-encoder.

Method	LPIPS ↓	ID ↑	FID ↓
E₁	0.10	0.87	89.50
E₂	0.14	0.75	64.02
Dual	0.10	0.87	65.44

1 while also producing FID scores very similar to those of Encoder 2.

3.3.3 Editing application

We follow the reference-based editing in [11] in our pipeline. This method encodes input images, and edits are performed in the triplane space. This approach utilizes the fact that triplanes have a canonical space, allowing for the transfer of local parts from one triplane to another. Fig. 3.9 demonstrates a successful transfer of hairstyle from a reference image to the target human in 3D. Another advantage of encoder-based models over optimization ones is the feasibility of such applications. For example, this would not be possible with PTI since the generator is fine-tuned for each sample, preventing the copying of features from one image to another in the encoded feature space.



Figure 3.9: Hair edits from source image (first) to destination image (second) and 360° renders (rest).

3.3.4 Broader impacts and limitations

Our framework has the potential to revolutionize the movie industry, AR, and VR, enabling applications like animating portraits and creating realistic game environments. However, it raises ethical concerns, particularly the risk of "deep

fakes”. We stress the need for safeguards to ensure the ethical use of this technology.

We acknowledge that there is room for improvements in the fidelity of images, the realism and 3D-consistency of generations (see Fig. 3.10, Row 2), and the smoothness of the meshes (see Fig. 3.7). Since the projection is made onto the latent space of PanoHead, our method may not handle out-of-domain or tail samples well (such as images with high-frequency details or accessories). For instance, our method struggles with hats, as demonstrated in the first Row of Fig. 3.10. We also recognize that, in certain cases, the artifacts are visible in the back middle of the head and are more noticeable in the mesh rendering as shown in Fig. 3.7.



Figure 3.10: Example failure cases. Inputs (first), reconstructions (second), and novel views.

Chapter 4

Identity Preserving 3D Head Stylization

In this chapter, we present a novel approach to 3D-aware head stylization that combines GAN domain adaptation with diffusion-based distillation [10], with a strong emphasis on identity preservation. While 3D stylization aims to convert realistic facial features into artistic representations, existing methods often fall short by focusing primarily on near-frontal views and struggling to retain the subject’s unique identity. To address these limitations, our framework builds on PanoHead, which enables full 360-degree image synthesis, allowing for more expressive and comprehensive stylizations.

Our pipeline introduces a negative log-likelihood distillation (LD) strategy designed to enhance identity consistency while improving stylization quality. We further refine this process by incorporating a multi-view grid score, mirror gradients, and a novel score rank weighting mechanism within the 3D GAN architecture. These contributions collectively lead to significant improvements, both qualitatively and quantitatively. As shown in Fig. 4.1, our results outperform existing methods and offer meaningful insights into the integration of diffusion models and GANs for 3D head stylization with reliable identity preservation.

The rest of this chapter details the methodology, implementation details, and results of the aforementioned work **B. Bilecen, A. Gökmen, F. Güzelant, A. Dündar, Identity Preserving 3D Head Stylization with Multiview Score Distillation, In submission, 2025 [10]**.

4.1 Method



Figure 4.1: Showcasing our method, effectively balancing stylization and identity preservation. Other approaches often struggle to achieve high-quality stylization and accurate identity retention. Notably, other methods generate identical faces for different inputs (e.g., Joker stylization), whereas ours preserves unique features for each input, ensuring distinct and personalized results for every individual face.

Our method involves fine-tuning the pre-trained parameters of PanoHead to adapt it for generating images from different domains. This section will go through the development of the method. Specifically, we will explain the adoption of LD to 3D-aware image generators, mention the differences between SDS, extend LD with cross-pose dependencies and grid denoising, and use rank reduction on score tensors. The general training approach is visualized in Fig. 4.2.

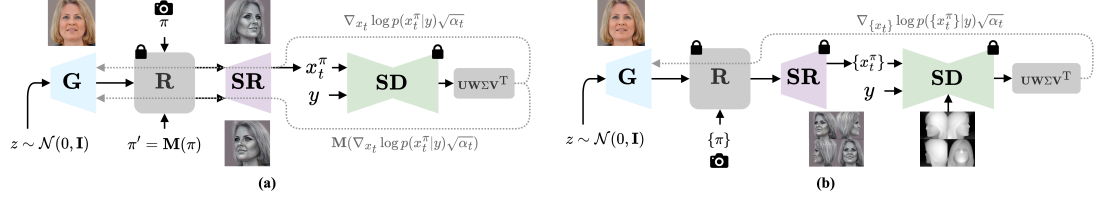


Figure 4.2: Our proposed training approach with mirror gradients (a) and grid distillation (b). Dashed and non-dashed lines show backpropagation and forward paths. **G**, **R**, **SR**, and **SD** denote generator, neural renderer, super-resolver, and denoising UNet, respectively. Output of **SD** is the score, $\nabla_{x_t} \log p(x_t^\pi | y)$. We employ a depth-conditioned ControlNet in (b). For the sake of simplicity, we omit the CFG here.

4.1.1 Likelihood distillation (LD) objective

We partially adapt the distillation procedure in [25] and tailor it to GANs, explained in this section.

Recall the DDPM forward process:

$$\sqrt{\alpha_t}x_0 + \sqrt{1 - \alpha_t}n = x_t, \quad n \sim \mathcal{N}(0, \mathbf{I}) \quad (4.1)$$

Assume that distribution (q) of the 3D representation (θ) conditioned on generation prompt (y) is proportional to the prompt-conditioned distribution (p) of independent 2D renders (x_0^i) on different poses (i) . In our setup, θ is the style-based 3D GAN layers.

$$q(\theta|y) \propto p(x_0^0, x_0^1, \dots, x_0^N | y) = \prod_i^N p(x_0^i | y) \quad (4.2)$$

We optimize negative log-likelihood of Eq. (4.2) to find θ :

$$-\log q(\theta|y) = -\log \prod_i^N p(x_0^i | y) = -\sum_i^N \log p(x_0^i | y) \quad (4.3)$$

Define the loss $\mathcal{L}_{LD}$ as the average of infinitely many N render poses and find gradient ∇_θ to update θ via gradient descent, where π is any given pose:

$$\begin{aligned}\mathcal{L}_{\text{LD}} &= -\frac{1}{N} \lim_{N \rightarrow \infty} \log q(\theta|y) = -\mathbb{E}_{\pi} \{\log p(x_0^{\pi}|y)\} \\ \nabla_{\theta} \mathcal{L}_{\text{LD}} &= -\mathbb{E}_{\pi} \{\nabla_{\theta} \log p(x_0^{\pi}|y)\}\end{aligned}\tag{4.4}$$

Using Eqs. (4.1) and (4.4) and change of variables in probability:

$$p(x_0^{\pi}|y) = p(x_t^{\pi}|y) \left| \frac{\partial x_t}{\partial x_0} \right|^{-1} = \frac{p(x_t^{\pi}|y)}{\sqrt{\bar{\alpha}_t}}\tag{4.5}$$

Take the log of both sides, the partial derivative with respect to x_0^{π} , and decompose the right-hand side with the chain rule using the relation in Eq. (4.1):

$$\begin{aligned}\log p(x_0^{\pi}|y) &= \log p(x_t^{\pi}|y) - \log \sqrt{\bar{\alpha}_t} \\ \frac{\partial \log p(x_0^{\pi}|y)}{\partial x_0^{\pi}} &= \frac{\partial p(x_t^{\pi}|y)}{\partial x_t^{\pi}} \frac{\partial x_t^{\pi}}{\partial x_0^{\pi}} \\ \nabla_{x_0} \log p(x_0^{\pi}|y) &= \nabla_{x_t} \log p(x_t^{\pi}|y) \frac{\partial x_t^{\pi}}{\partial x_0^{\pi}}\end{aligned}\tag{4.6}$$

Extend the partial gradient chain in Eq. (4.6) to θ from x_0^{π} :

$$\nabla_{\theta} \log p(x_0^{\pi}|y) = \nabla_{x_t} \log p(x_t^{\pi}|y) \frac{\partial x_t^{\pi}}{\partial x_0^{\pi}} \frac{\partial x_0^{\pi}}{\partial \theta},\tag{4.7}$$

where $\nabla_{x_t} \log p(x_t^{\pi}|y)$ is the score function estimation. Plugging Eq. (4.7) into Eq. (4.4) yields the update direction:

$$\nabla_{\theta} \mathcal{L}_{\text{LD}} = -\mathbb{E}_{\pi, x_t} \left\{ \nabla_{x_t} \log p(x_t^{\pi}|y) \frac{\partial x_t^{\pi}}{\partial x_0^{\pi}} \frac{\partial x_0^{\pi}}{\partial \theta} \right\},\tag{4.8}$$

where $\frac{\partial x_t^{\pi}}{\partial x_0^{\pi}}$ is $\sqrt{\bar{\alpha}_t}$ from Eq. (4.1). Notice that to update θ , we do not need to back-propagate through the denoising UNet and can acknowledge the UNet output as a part of the gradient. For the domain adaptation task, we assign θ as the PanoHead layers.

Note that LD is different than SDS [24] as it optimizes negative log-likelihood instead of reverse KL-divergence. Compliant with previous observations [90], we

also observe that SDS yields blurry results, as LD is sharp in domain adaptation tasks. We attribute this to the fact that SDS is inherently mode-seeking due to the subtraction of ground truth noise ϵ from the estimated noise $\hat{\epsilon}$ and high classifier-free guidance (CFG) [91] weight to avoid divergence [24]. In contrast, LD is diversity-seeking (by not utilizing ϵ), does not require very high CFG weights, and is more suitable for our task, considering that we utilize a GAN prior. This issue had not been investigated before on GANs, and the focus was diverted to additional losses to solve the underlying issues of SDS, e.g. [22, 23]. We also note that Huang *et al.* [25] proposes multiple-gradient descent on top of Eq. (4.8), which we simply omit since the pretrained GAN backbone is strong enough to avoid divergence.

4.1.2 Rank-weighted score tensors

While LD reduces the smoothness issues, we may come across some artifactual results, especially around the hair, ears, and neck on some prompts (Fig. 4.3, full-rank). Notice that the artifacts are more focused on incorrect color distribution rather than the style itself. Since diffusion latents must also contain color distribution information [92], we decide to investigate the SVD of the score tensors along the VQ-VAE latent channel (4) dimension. *We notice that the first rank contains most of the stylization and surpassing the contributions of the lower three ranks can mitigate undesired tints.* Hence, we re-weigh the diagonal singular value matrix $\mathbf{S}$ with linearly decaying coefficients $\mathbf{W}$ from the largest singular value to the smallest. Specifically,

$$\begin{aligned}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T &= \text{SVD}(\nabla_{\theta} \log p(x_0^{\pi}|y)), \\ \nabla_{\theta} \log \tilde{p}(x_0^{\pi}|y) &= \mathbf{U}\mathbf{W}\mathbf{\Sigma}\mathbf{V}^T,\end{aligned}\tag{4.9}$$

where $\tilde{p}$ is the rank-weighted score distribution, $\mathbf{\Sigma}_{4 \times 4} = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_4)$, $\mathbf{U}_{4 \times 4} = [u_1, u_2, \dots, u_4]$, $\mathbf{V}_{4096 \times 4} = [v_1, v_2, \dots, v_4]$, and $\mathbf{W} = \text{diag}(1, 0.75, 0.5, 0.25)$.

As seen in Fig. 4.3, rank-weighted score tensors eliminate the problematic tints

and set a good baseline for the following improvements. We set the weight scores based on our empirical analysis, and the same weights are used for the training of all prompts.

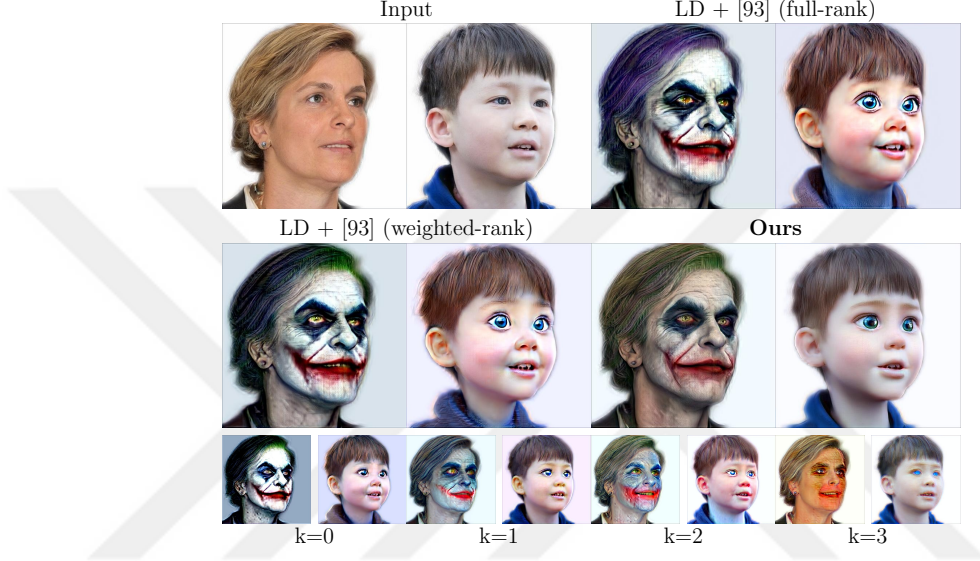


Figure 4.3: Ablation study on rank weighing. After SVD, four rank-1 matrices are obtained, and k^{th} are chosen for reconstruction. Weighted rank improves the results compared to full rank. In this figure, we also present our final results for comparisons with LD + weighted rank and our multi-view and mirror gradient optimizations for completeness.

4.1.3 Extending LD with mirror poses and multi-view distillation

The expression in Eq. (4.8) does not consider the correlation between different views. To add cross-dependencies, we account for another poses where $\pi' \neq \pi$ in Eq. (4.10):

$$\nabla_{\theta} \mathcal{L}_{\text{LD}} = -\mathbb{E}_{\pi, x_t} \left\{ \nabla_{x_t} \log p(x_t^{\pi} | y) \frac{\partial x_t^{\pi}}{\partial x_0^{\pi}} \frac{\partial x_0^{\pi}}{\partial \theta} + \sum_{\pi \neq \pi'} \frac{\partial \log p(x_t^{\pi} | y)}{\partial x_0^{\pi'}} \frac{\partial x_0^{\pi'}}{\partial \theta} \right\} \quad (4.10)$$

Notice that $\frac{\partial \log p(x_t^{\pi} | y)}{\partial x_0^{\pi'}} = \nabla_{x_t} \log p(x_t^{\pi} | y) \frac{\partial x_t^{\pi}}{\partial x_0^{\pi'}}$ in Eq. (4.10):

$$\nabla_{\theta} \mathcal{L}_{\text{LD}} = -\mathbb{E}_{\pi, x_t} \left\{ \nabla_{x_t} \log p(x_t^{\pi} | y) \frac{\partial x_t^{\pi}}{\partial x_0^{\pi}} \frac{\partial x_0^{\pi}}{\partial \theta} + \sum_{\pi \neq \pi'} \nabla_{x_t} \log p(x_t^{\pi'} | y) \frac{\partial x_t^{\pi'}}{\partial x_0^{\pi'}} \frac{\partial x_0^{\pi'}}{\partial \theta} \right\} \quad (4.11)$$

By utilizing the symmetry prior for human heads, we assume that if π and π' are yaw-symmetric camera matrices, $x_t^{\pi} = \mathbf{M}(x_t^{\pi'})$ where $\mathbf{M}$ is the vertical mirror (flip) operator. Then, by Eq. (4.1), $\frac{\partial x_t^{\pi}}{\partial x_0^{\pi}}$ simply becomes $\mathbf{M}\sqrt{\alpha_t}$. Further simplifying the expression Eq. (4.11) yields:

$$\nabla_{\theta} \mathcal{L}_{\text{LD}} = -\mathbb{E}_{\pi, x_t} \left\{ \nabla_{x_t} \log p(x_t^{\pi} | y) \sqrt{\alpha_t} \left(\frac{\partial x_0^{\pi}}{\partial \theta} + \mathbf{M} \frac{\partial x_0^{\pi'}}{\partial \theta} \right) \right\} \quad (4.12)$$

As shown in Fig. 4.2 (a), this intuitively means that we utilize the same score estimation for mirror poses but must also mirror the gradients while back-propagating to the generator. Since 3D-consistent face domain editing with 2D-diffusion models is challenging, keeping the same score estimation for different renders of the same ID under logical constraints avoids deviating from the convergence path. This is because there is no guarantee that $\nabla_{x_t} \log p(x_t^{\pi} | y)$ and $\nabla_{x_t} \log p(x_t^{\pi'} | y)$ will indicate the same direction for the same y .

Notice the mirror approach can be extended to any render pose tuple (π, π') as long as we can find a tractable gradient chain for $\frac{\partial x_t^{\pi}}{\partial x_0^{\pi'}}$. Our experiments show that mirror gradients further improve stylization while accentuating 3D-aware features like glasses (Fig. 4.9).

Mirror gradients only correlate with yaw-symmetric poses. Now, let us consider the joint probability distribution p and not assume explicit independence among poses, yielding Eq. (4.13):

$$q(\theta | y) \propto p(x_0^0, x_0^1, \dots, x_0^N | y) = p(\{x_0^{\pi}\} | y), \quad (4.13)$$

$$\nabla_{\theta} \mathcal{L}_{\text{LD}_g} = -\mathbb{E}_{\pi, \{x_t\}} \left\{ \nabla_{\{x_t\}} \log p(\{x_t^{\pi}\} | y) \frac{\partial \{x_t^{\pi}\}}{\partial \{x_0^{\pi}\}} \frac{\partial \{x_0^{\pi}\}}{\partial \theta} \right\},$$

where $\{x_0^{\pi}\}$ is defined to have all poses $\{0, 1, \dots, N\}$. However, this is not computationally feasible. Instead, as shown in Fig. 4.2 (b), we approximate $\{x_0^{\pi}\}$ with a 2×2 grid, where each element is a x_0^{π} with different render pose π . This way, denoising UNet can correlate between different renders of θ , improving stylization

consistency across views. We further employ a depth-conditioned ControlNet [93] as a regularizer to ensure the estimated score does not collapse to a single fused image.

Our approach does not fine-tune any diffusion model to accommodate multiple inputs [94, 95, 96] for multi-view consistency. Instead, it can use any pre-trained diffusion model and re-purposes grid structures for the distillation task, first proposed for video generation [97] and editing [98] with spatiotemporal consistency.

The PanoHead model generates images at a resolution of 512×512 [6], and the diffusion model similarly processes 512×512 images when using LD-losses [2]. For grid construction, we arrange four images in a 2×2 layout but reduce the individual image size to 256×256 to avoid memory issues during forward passes. This way, the final grid image has the size of 512×512 . Although this maintains a good correlation between poses and consistent stylization, it creates a resolution mismatch when gradients are directly propagated from the PanoHead output. To resolve this, we experiment with feeding the gradients before the SR network, where the renderer outputs images at a lower resolution of 64×64 . This adjustment is only necessary for multi-view distillation, where we backpropagate $\nabla_{\theta} \mathcal{L}_{LD_g}$ before the SR layers. For mirror poses, we do not need to skip the super-resolution layers since they match the correct resolution, and so we backpropagate $\nabla_{\theta} \mathcal{L}_{LD}$ after the SR, as visualized in Fig. 4.2. This strategy improves the stylization quality and reduces unwanted artifacts, such as blur, over-saturation, and color tint.

Algorithm 3 demonstrates the LD algorithm with mirror and grid gradients. `sample_latent` utilizes the mapping network of the generator and maps z to w^+ , later to be fed to the generator. `make_grid` creates a 2×2 grid with 4 inputs. **M** is realized with `torch.flip(x, dims=[-1])`. `with no_grad()` disables gradient calculation. Note that each time x_0 is generated, we implicitly pass it through VQ-VAE to embed it into **SD**’s latent space.

Algorithm 3 LD with mirror and grid grads

Require: Generator $\mathbf{G}_\theta$, neural renderer $\mathbf{R}$, super-resolver $\mathbf{SR}$, depth extractor $\mathbf{D}$, depth and text-conditioned denoising UNet $\mathbf{SD}$, generator mapping truncation parameter ψ , extrinsic triplane render matrix π , mirror pose π' , vertical flip operator $\mathbf{M}$, rank weighing matrix $\mathbf{W}$

```

1: for  $i$  in  $\{0, 1, \dots, N\}$  do
2:    $w^+ \leftarrow \text{sample\_latent}(\psi=0.8)$ 
3:    $\pi, \pi' \leftarrow \text{sample\_pose}()$   $\triangleright \mathbb{E}_\pi$ 
4:    $x_0^\pi, x_0^{\pi'} \leftarrow \mathbf{SR}(\mathbf{R}(\mathbf{G}_\theta(z), \pi, \pi'))$ 
5:    $n, t \leftarrow \text{noise\_scheduler}(0.70, 0.96)$ 
6:    $x_t^\pi \leftarrow \sqrt{\bar{\alpha}_t}x_0^\pi + \sqrt{1 - \bar{\alpha}_t}n$   $\triangleright \mathbb{E}_{x_t}$ 
7:   with  $\text{no\_grad}()$ :
8:      $\nabla_{x_t} \log p(x_t^\pi|y)/\sqrt{1 - \bar{\alpha}_t} \leftarrow \mathbf{SD}(x_t^\pi, y, t, \mathbf{D}(x_0^\pi))$ 
9:      $\text{grad} \leftarrow \nabla_{x_t} \log p(x_t^\pi|y)\sqrt{\bar{\alpha}_t}$ 
10:     $\mathbf{U}\Sigma\mathbf{V}^T \leftarrow \mathbf{SVD}(\text{grad})$ 
11:     $\text{grad} \leftarrow \mathbf{W}\Sigma\mathbf{V}^T$   $\triangleright$  rank weighing
12:     $x_0^\pi.\text{backward}(\text{grad})$   $\triangleright \nabla_\theta \mathcal{L}_{\text{LD}}$ 
13:     $x_0^{\pi'}.\text{backward}(\mathbf{M}(\text{grad}))$   $\triangleright$  mirror gradients
14:     $\text{optimizer.step}()$ 
15: end for

```

```

16: for  $i$  in  $\{0, 1, \dots, N\}$  do
17:    $w^+ \leftarrow \text{sample\_latent}(\psi=0.8)$ 
18:    $\{\pi\} = \pi^0, \pi^1, \pi^2, \pi^3 \leftarrow \text{sample\_pose}()$   $\triangleright \mathbb{E}_{\{\pi\}}$ 
19:    $\{x_0^\pi\}_{\text{LR}} \leftarrow \text{make\_grid}(\mathbf{R}(\mathbf{G}_\theta(w^+), \{\pi\}))$ 
20:    $\{x_0^\pi\} \leftarrow \text{make\_grid}(\mathbf{SR}(\mathbf{R}(\mathbf{G}_\theta(w^+), \{\pi\})))$ 
21:    $n, t \leftarrow \text{noise\_scheduler}(0.30, 0.80)$ 
22:    $\{x_t^\pi\} \leftarrow \sqrt{\bar{\alpha}_t}\{x_0^\pi\} + \sqrt{1 - \bar{\alpha}_t}n$   $\triangleright \mathbb{E}_{\{x_t\}}$ 
23:   with  $\text{no\_grad}()$ :
24:      $\nabla_{\{x_t\}} \log p(\{x_t^\pi\}|y)/\sqrt{1 - \bar{\alpha}_t} \leftarrow \mathbf{SD}(\{x_t^\pi\}, y, t, \mathbf{D}(\{x_0^\pi\}))$ 
25:      $\text{grad} \leftarrow \nabla_{\{x_t\}} \log p(\{x_t^\pi\}|y)\sqrt{\bar{\alpha}_t}$ 
26:      $\mathbf{U}\Sigma\mathbf{V}^T \leftarrow \mathbf{SVD}(\text{grad})$ 
27:      $\text{grad} \leftarrow \mathbf{W}\Sigma\mathbf{V}^T$   $\triangleright$  rank weighing
28:      $\{x_0^\pi\}_{\text{LR}}.\text{backward}(\text{grad})$   $\triangleright$  grid gradients  $\nabla_\theta \mathcal{L}_{\text{LD}_g}$ 
29:      $\text{optimizer.step}()$ 
30: end for

```

31: **return** $\mathbf{G}_\theta$

4.2 Experimental Setup

4.2.1 Baselines

We compare our method against several domain adaptation techniques, all adapted to work with the PanoHead generator. StyleCLIP [20] trains a $\mathcal{W}^+$ mapper network using a CLIP-based loss. StyleGAN-NADA [19] modifies adaptively-selected generator layers using the CLIP objective, instead of altering $\mathcal{W}^+$. StyleGANFusion [22] leverages SDS (score distillation sampling) for domain adaptation, paired with a directional regularizer via a frozen generator for stable distillation. DiffusionGAN3D [23] further improves upon [22] by introducing a relative distance loss for better regularization. We also include 2D-3D editing methods such as InstantID [99], InstructPix2Pix [100] and InstructNerf2Nerf [101] for a broader comparison.

For StyleCLIP [20], we train the latent mapper with PanoHead’s [6] generator. For StyleGAN-NADA [19] and StyleGANFusion [22], we utilize the official repository of [22] and modify the generator backbone to PanoHead, adopting their EG3D configuration for PanoHead. Additionally, we implement adaptive layer selection for [19]. For DiffusionGAN3D [23], where no official codebase is available, we implement the method following the original paper. Our baseline builds on [23], integrating their relative distance loss for domain adaptation while introducing our proposed improvements. After adapting the generators, we invert images by optimizing $\mathcal{W}^+$ to reconstruct the input image using the original PanoHead generator. The optimized $\mathcal{W}^+$ is then fed into each domain-adapted generator to produce renderings from various viewpoints.

For all baselines, we faithfully adhere to their original hyperparameters, including denoiser checkpoints, noise schedulers, learning rates, and optimizers, unless adjustments are necessary due to training divergence.

4.2.2 Training

We train the generator with synthetic $z_{1 \times 512} \sim \mathcal{N}(0, \mathbf{I})$ data for 10k iterations with batch size 1, where the truncation parameter of the generator’s mapping network is $\psi = 0.8$. We use Adam optimizer with a $1e^{-4}$ learning rate. We optimize the `G.backbone.synthesis` and `G.backbone.superresolution` sub-networks of the generator `G` and freeze all convolutional layer biases, using the same configuration as [22]. The classifier-free-guidance (CFG) [91] weight and depth-conditioned ControlNet [93] guidance weight are set to 7.5 and 1.0, respectively. Depth ground truths are extracted from [102] since the neural renderer’s depth estimations are low-resolution and require additional clipping (64×64).

As the conditional denoiser for our method and the ablation study for showing the improvements upon [23], we employ the model Realistic Vision v5.1, similar to Wu *et al.*’s work [103]. For qualitative and quantitative comparison among other methods, we employ the methods’ suggested diffusion checkpoints in their papers and repositories.

For mirror and grid denoising, noise start timestep t is uniformly selected among $(0.70, 0.96)$ and $(0.30, 0.80)$, respectively, where t is from $0 \rightarrow 1$. We use `DDIMScheduler` for the noise scheduler. The number of inference steps in the diffusion pipeline is always 1 since we perform score distillation.

4.2.3 Evaluation metrics

For stylization performance, we measure Fréchet Inception Distance (FID) [89] and CLIP embedding similarity; for ID preservation, we measure ArcFace-based [83] ID similarity and multi-view render depth $\mathcal{L}_2$ difference $\Delta\mathcal{D}$. To construct our test set, we perform $\mathcal{W}^+$ inversion to randomly chosen ~ 100 FFHQ images. These $\mathcal{W}^+$ ’s are input to each generator to stylize a real identity.

In FID and CLIP, we construct two different edited image distributions,

wherein one, the images are edited with a Stable Diffusion pipeline in the 2D image domain (ground-truth distribution), and in the other, the images are edited via the domain-adapted 3D-aware generators. These two distributions are used to compute FID, and individual image pairs between them are used to compute CLIP similarity scores. We generate a third distribution using unedited images to evaluate identity preservation (ID) and $\Delta\mathcal{D}$. Scores for ID and $\Delta\mathcal{D}$ are then calculated between using image pairs from the second and third distributions. Fig. 4.4 visualizes sample images in those three distributions.



Figure 4.4: Samples from the utilized dataset. From left to right: Ours (distribution #2), ground truth unedited image (distribution #3), edited image with full-step diffusion pipeline (distribution #1).

4.3 Results

4.3.1 Comparison with state-of-the-art

We provide quantitative and qualitative results in Table 4.1 and Figs. 4.1 and 4.5, respectively. Table 4.1 demonstrates that our method outperforms others in terms of stylization quality, as measured by FID and CLIP scores, as well as in identity preservation, as evaluated by ArcFace-based ID similarity and multi-view depth difference $\Delta\mathcal{D}$ across nearly all prompts. These numerical findings are consistent with the visual assessment. Figs. 4.1 and 4.5 reveal that StyleCLIP and StyleGAN-NADA produce stylizations of lower quality, with noticeable artifacts in the renderings. While StyleGANFusion and DiffusionGAN3D are capable of producing stylized images, they significantly compromise identity preservation. For instance, in the Joker example in Fig. 4.1, both methods generate identical facial features for two different identities. Similarly, for the sketch prompt, the identities are substantially altered. Competing methods also remove accessories

such as eyeglasses. In contrast, our method effectively preserves the identity of the input image while delivering high-quality stylizations.

		2D Methods		3D Domain Adpt. Methods				
		[100]	[99]	[20]	[19]	[22]	[23]	Ours
Pixar	FID	144.4	160.8	118.2	81.1	168.0	189.3	77.6
	CLIP	0.82	0.68	0.77	0.81	0.76	0.82	0.86
	ID	0.40	0.59	0.52	0.61	0.60	0.46	0.69
	$\Delta\mathcal{D}$	0.024	0.050	0.022	0.013	0.003	0.024	0.014
Joker	FID	116.8	170.3	97.6	116.3	119.3	110.7	67.7
	CLIP	0.90	0.69	0.75	0.81	0.80	0.86	0.89
	ID	0.44	0.47	0.38	0.50	0.41	0.47	0.56
	$\Delta\mathcal{D}$	0.023	0.066	0.031	0.011	0.007	0.009	0.003
Werewolf	FID	178.1	183.3	248.8	212.9	203.9	132.3	99.7
	CLIP	0.82	0.58	0.63	0.75	0.70	0.81	0.85
	ID	0.28	0.53	0.42	0.45	0.41	0.68	0.56
	$\Delta\mathcal{D}$	0.013	0.065	0.039	0.012	0.012	0.002	0.002
Sketch	FID	89.5	162.7	103.7	99.7	99.0	159.4	91.6
	CLIP	0.65	0.65	0.75	0.71	0.74	0.71	0.77
	ID	0.24	0.59	0.54	0.51	0.53	0.48	0.75
	$\Delta\mathcal{D}$	0.002	0.047	0.026	0.034	0.012	0.016	0.005
Statue	FID	60.9	160.4	181.6	154.5	85.5	166.2	144.5
	CLIP	0.91	0.71	0.63	0.76	0.83	0.82	0.82
	ID	0.39	0.56	0.61	0.36	0.47	0.43	0.55
	$\Delta\mathcal{D}$	0.011	0.038	0.012	0.039	0.007	0.009	0.006

Table 4.1: Quantitative scores with competitive methods.

Fig. 4.6 presents the mesh predictions for the Pixar-style stylizations produced by our method and the competing approaches. The first Row shows reconstructions of the input image from different views, both in RGB image space and from the mesh perspective. These results demonstrate that our method achieves significantly better identity preservation both in terms of facial features and distinctive accessories, such as eyeglasses. Unlike other methods, which completely remove or distort the eyeglasses, our approach maintains them with high fidelity.

We then conduct a user study with 25 participants to evaluate the quality of 3D stylization and identity preservation across different methods. Participants are shown images generated by five different models: StyleCLIP, StyleGAN-NADA, StyleGANFusion, DiffusionGAN3D, and our own approach. For each image, they are asked to select which output best balances stylization and identity preservation. The methods are presented in random order for each image to minimize bias. Results of this user study are shown in Fig. 4.7. The data indicate that



Figure 4.5: Qualitative stylization results provided in 360-degree views.

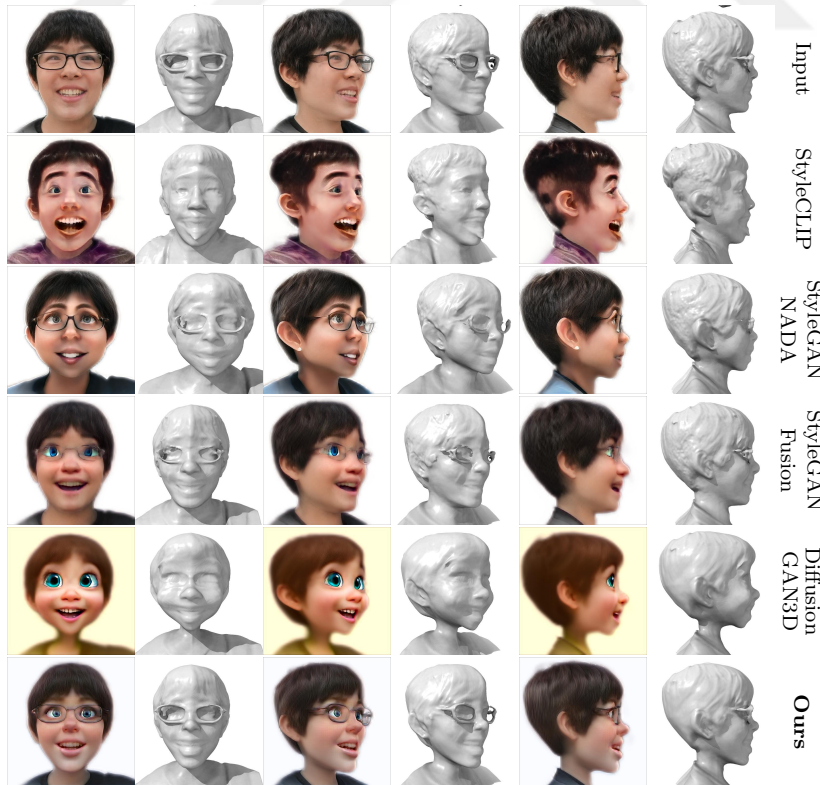


Figure 4.6: Mesh and rendering visualizations of ours and competing methods for the Pixar prompt. The input consists of a single image, and we display the inverted and rendered results from PanoHead from three different viewpoints.

our method is consistently preferred across all the prompts tested, with participants overwhelmingly selecting it as the best for both stylization and identity preservation when compared to the other approaches.

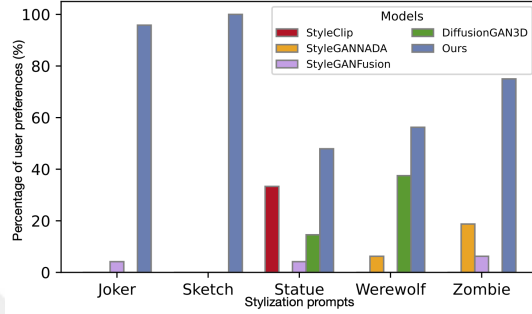


Figure 4.7: The percentage of users’ preferences is presented for different prompts. Users overwhelmingly select our method when compared to the other approaches.

4.3.2 Ablation study

We begin our ablation study with the DiffusionGAN3D [23] baseline and build upon it by incorporating various improvements. [23] uses SDS loss for generator optimization, along with a relative distance loss. This relative distance loss aims to preserve the distance between two images generated by the original generator, which serves as the ground truth. The fine-tuned generator is then optimized to maintain this same distance between the corresponding generated samples.

As shown in Fig. 4.8, while their method successfully achieves stylization, the results fail to preserve the characteristics of the input images. Next, we replace the text-based diffusion model that generates the SDS loss with the depth-based ControlNet [93], which helps retain more characteristics from the input image, though its effectiveness remains limited. In the next setting, we apply negative log-likelihood distillation (LD) alongside the depth-based ControlNet model. This results in less saturation, and we observe improved preservation of the input identity. We also show our final results demonstrating the improved color and geometry alignment with the input images that include the improvements from grid and mirror gradients in the last column. These visuals are consistent with

the quantitative improvements presented in Table 4.2.

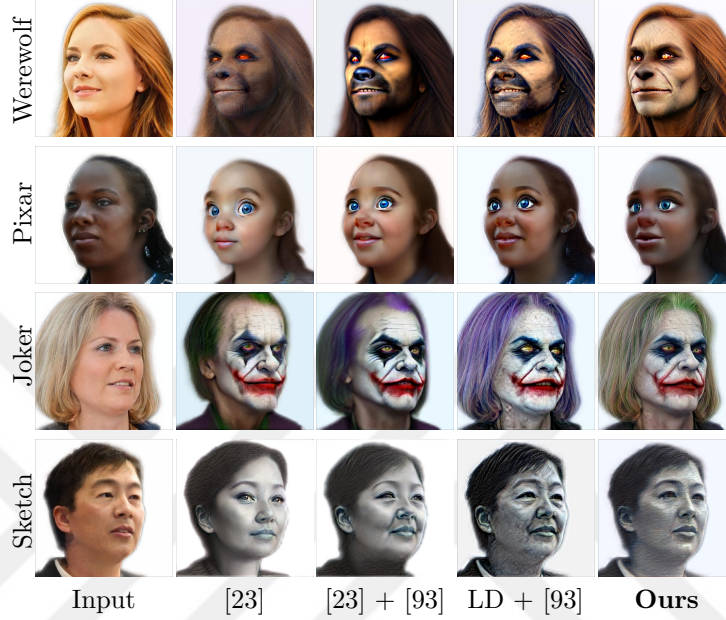


Figure 4.8: Qualitative Ablation Study I. LD and other regularizers like grid and mirror scores help keep the original hair color, skin tone, and geometry while stylizing.

	FID	Δ	CLIP	Δ	ID	Δ
[23]	147.19	-	0.74	-	0.35	-
[23] + [93]	140.20	-6.99	0.75	+0.01	0.34	-0.01
LD + [93]	116.88	-23.32	0.81	+0.06	0.46	+0.12
LD + [93] + W-rank	116.09	+0.79	0.82	+0.01	0.48	+0.02
LD + [93] + W-rank + Grid after SR	99.53	-16.56	0.85	+0.03	0.49	+0.01
LD + [93] + W-rank + Grid before SR	81.11	-18.42	0.85	0.00	0.51	+0.02
LD + [93] + W-rank + Grid before SR + Mirror gradients	84.17	+3.06	0.86	+0.01	0.52	+0.01

Table 4.2: Quantitative ablation study. Δ represents the difference between the results of the current Row and the Row above. W-rank indicates weighted-rank. Our approach improves stylization with ID retention.

We continue our ablation study in Fig. 4.9. We start from the previous study with the results of LD, including rank-weighted score tensors and depth-based ControlNet guidance. The ablation study for rank-weighting is presented in Fig. 4.3 to provide a clearer explanation of the method, so we omit that discussion here. Next, we add the grid-based denoising, which provides multi-view consistent distillation. We observe that the multi-view grid gradients improve the results even when we feed them after the super-resolution layers, but if we

feed the gradients to the generator before the super-resolution network, the results improve further both quantitatively and qualitatively, as shown in Table 4.2 and Fig. 4.9, respectively. Finally, we incorporate cross-dependencies with mirror poses, enhancing intricate details, as evidenced by the zoomed-in views.

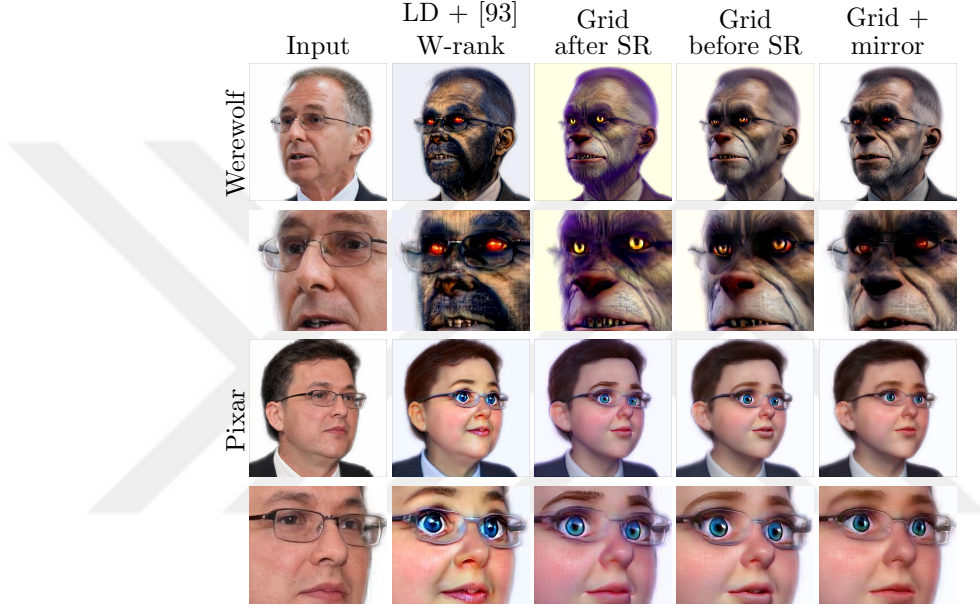


Figure 4.9: Qualitative Ablation Study II. Grid distillation improves stylization while preserving identity, and mirror gradients focus on symmetric, intricate features like glasses.

We present additional results with various prompts in Fig. 4.10. These prompts include local stylization, such as adding a mustache or editing hair to pink. For these experiments, we do not employ attribute-based masking for score gradients (for example, we do not mask the face when editing the hair). Despite this, our method successfully achieves both localized edits and more global transformations, such as modifying the character to resemble an elf under a single framework. In Fig. 4.11, we also compare our method with Instruct-Pix2Pix [100] (IP2P) and Instruct-Nerf2Nerf [101] (IN2N). From the available results, it is clear that our method outperforms others, balancing ID preservation and stylization. Notably, IP2P is not 3D-consistent, while IN2N may generate over-smoothed and unnatural results.

Regarding runtime, our method requires ~ 3 h of generator tuning per prompt



Figure 4.10: Additional results with various prompts applied to the input image given in the first column.

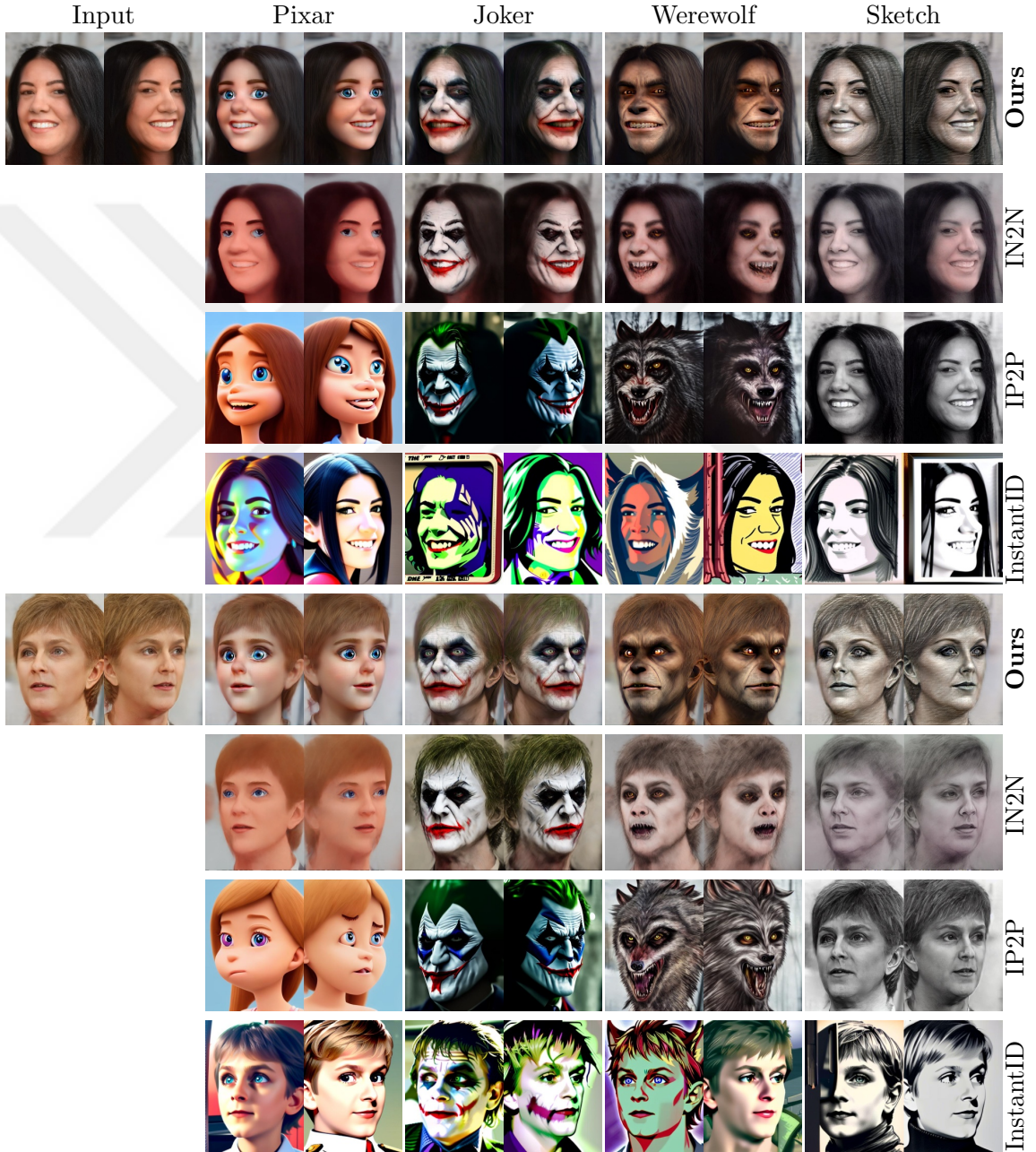


Figure 4.11: Additional comparisons with other 2D-3D editing methods. 2D methods are run with the same seed for different views.

on a single A100 GPU. After tuning, inference on any head image takes only ~ 0.5 s. In comparison, [101] requires ~ 1 h to fit NeRF to a single identity and $+3$ h for stylizing the scene. [99, 100] take 50 iterations through denoiser, taking ~ 10 - 15 s. [23, 22] also tune their generators, and their training and inference times are similar to ours.

4.3.3 Limitations

A limitation, similar to other methods, is that stylization may not fully preserve earrings and hair accessories, as illustrated in Fig. 4.12. Another one is the output resolution, which is bounded by the utilized generative model. However, they can be eliminated by utilizing a heavier model.

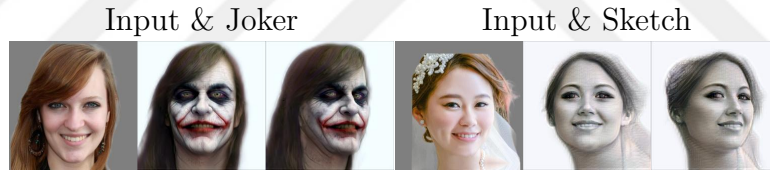


Figure 4.12: Limitations in preserving earrings and hair accessories.

Chapter 5

Reference-based 3D-aware Image Editing with Triplanes

In this chapter, we introduce a reference-based editing framework [11] that leverages the latent space of 3D-aware GANs for high-quality image manipulation. While GANs have proven effective for editing real images through latent space operations, recent models like EG3D extend these capabilities by incorporating 3D structure using triplane representations. Despite this progress, there is limited work on unified frameworks for high-quality, reference-guided editing in 3D-aware settings. This study addresses that gap by demonstrating the effectiveness of the triplane space for performing advanced, spatially consistent edits guided by a reference image.

The proposed method integrates several key components, including a specialized encoder, automatic localization, spatial disentanglement of triplane features, and a fusion learning mechanism that together enable effective and precise editing. We evaluate our approach across a wide range of domains, including human faces, full 360-degree head representations, animal faces, partially stylized transformations such as cartoon edits, full-body clothing changes, and class-agnostic examples. Fig. 5.1 visualizes our framework’s qualitative editing results in various domains.

The rest of this chapter details the methodology, implementation details, and results of the aforementioned work **B. Bilecen, Y. Yalın, N. Yu, A. Dünder, Reference-Based 3D-Aware Image Editing with Triplanes, CVPR 2025 (High-light)** [11].

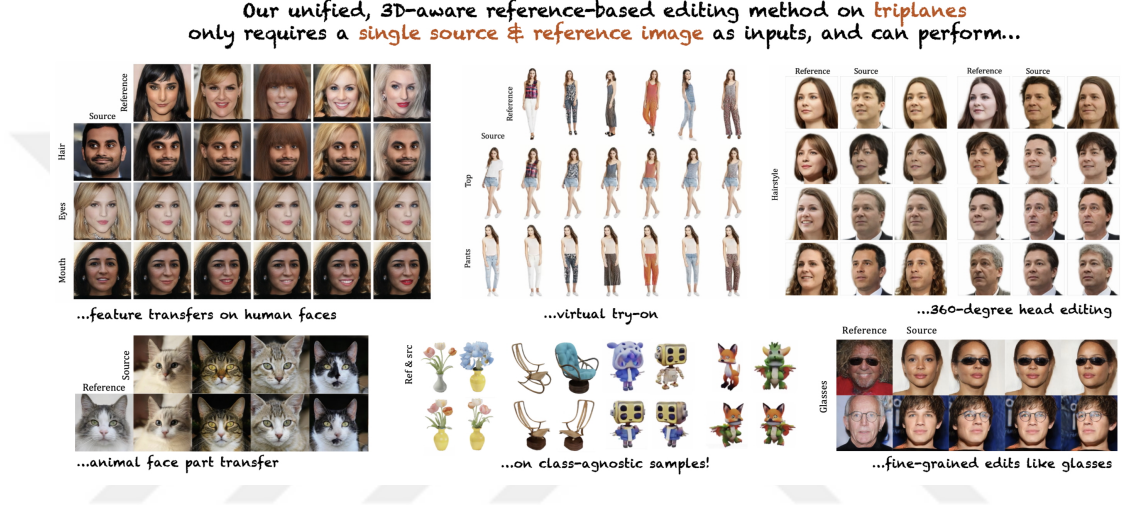


Figure 5.1: Our approach excels in reference-based edits, utilizing only a single source and reference image. Leveraging 3D-aware triplanes, our edits are versatile and 3D consistent, allowing for rendering from various viewpoints. We show results on human faces, heads, bodies, and extend beyond to animal faces and class-agnostic samples.

5.1 Method

Our main motivation lies in the fact that triplanes can be stitched and blended for editing, like in the 2D image domain. However, achieving satisfying results requires important observations and carefully designed steps, which will be detailed in this section.

5.1.1 Localizing parts in the triplane space

To address this, we take advantage of the volumetric renderer used in [8] being fully differentiable, and back-propagate the 2D image domain masks to the 3D

hybrid triplane domain to calculate gradients on triplanes (Fig. 5.2, *triplane localization* in Algorithm 4). First, input images are encoded using a pre-trained model [18, 17] to obtain triplane features. These features are rendered with different camera poses $\pi_{1,2,\dots,N}$ to create multi-view 2D renderings $\mathcal{R}(\mathbf{T}, \pi)$. An off-the-shelf segmentation network $\mathcal{S}_{2D}$ identifies attributes [104] (e.g., hair, eyes, glasses) in each rendering. The segmentation outputs are assigned as output gradients $\nabla_{\mathcal{R}(\mathbf{T}, \pi)}$, and are back-propagated to the triplane domain to accumulate input gradients $\Sigma_{\pi} \nabla_{\mathbf{T}}$, which localizes the triplane mask.

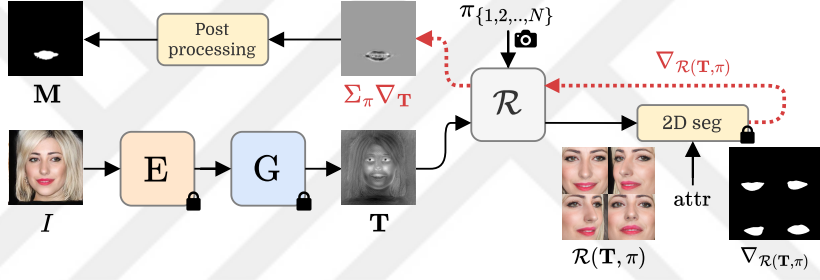


Figure 5.2: Triplane part localization stage. **E**, **G**, and **R** are encoder [17, 18], generator [8], and neural volumetric renderer, respectively. For the 2D segmentation model $\mathcal{S}_{2D}$, we use state-of-the-art off-the-shelf segmentation models [104]. Images other than the input image are cropped to zoom in for visualization purposes.

The most direct way to perform a region transfer from reference to source image would be to mask and copy & paste the region of interest. However, this process becomes intricate in hybrid 3D representations like triplanes, due to the absence of a conventional method for identifying the regions to be masked.

To convert a gradient mask into a binarized one, we perform mean clipping, normalization, and thresholding with parameters (ϵ, γ) (*post-processing* in Algorithm 4). These parameters are set once and used across all experiments for each attribute and domain. This localization is done for source and reference images.

For finer granularity and to avoid copying unwanted attributes, such as copying glasses and not the eyes, $\mathcal{W}^+$ directions [29] can be utilized in addition. By computing the difference between triplanes with and without the attribute,

Algorithm 4 Triplane localization and masking

Require: Generator $\mathbf{G}$, encoder $\mathbf{E}$, renderer $\mathcal{R}$, image I , extrinsic matrices $\pi_{1,2,\dots,N}$, segmentation net $\mathcal{S}_{2D}$, post-processing params (ϵ, γ)
Ensure: Triplane mask $\mathbf{M}$

```
1:  $\mathbf{T}, \nabla_{\mathbf{T}} \leftarrow \mathbf{G}(\mathbf{E}(I)), 0$   $\triangleright$  Initialize triplane and gradients  
2: for each  $\pi_i$  in  $\pi_{1,2,\dots,N}$  do  $\triangleright$  Triplane localization  
3:    $I_{\pi_i} \leftarrow \mathcal{R}(\mathbf{T}, \pi_i)$   
4:    $\nabla_{\mathcal{R}(\mathbf{T}, \pi_i)} \leftarrow \mathcal{S}_{2D}(I_{\pi_i}, \mathbf{attr})$   
5:    $\nabla_{\mathbf{T}} \leftarrow \nabla_{\mathbf{T}} + \mathbf{autograd}(\mathcal{R}, \nabla_{\mathcal{R}(\mathbf{T}, \pi_i)})$   
6: end for
```

```
7: for each  $c$  in  $\nabla_{\mathbf{T}}$  do  $\triangleright$  Post-processing  
8:    $\nabla_{\mathbf{T}_c} \leftarrow \{x \in \nabla_{\mathbf{T}_c} : |\nabla_{\mathbf{T}_c}(x) - \mu_c| \leq \epsilon\}$   
9:    $\nabla_{\mathbf{T}_c} \leftarrow (\nabla_{\mathbf{T}_c} - \min \nabla_{\mathbf{T}_c}) / (\max \nabla_{\mathbf{T}_c} - \min \nabla_{\mathbf{T}_c})$   
10:   $\nabla_{\mathbf{T}_c} \leftarrow \{x \in \nabla_{\mathbf{T}_c} : 1 \text{ if } \nabla_{\mathbf{T}_c}(x) \geq \gamma \text{ else } 0\}$   
11: end for
```

```
12: return  $\mathbf{M} \leftarrow \nabla_{\mathbf{T}}$ 

---


```

$\Delta \mathbf{T}_{\text{attr}} = \mathbf{G}(w) - \mathbf{G}(w - w_{\text{attr}})$ and multiplying with gradient mask $\mathbf{M}$, a more precise mask can be created.

5.1.2 Implicit fusion by encoding & decoding

After finding suitable masks for reference and source triplanes, $\mathbf{M}_{\text{ref}}$ and $\mathbf{M}_{\text{src}}$ respectively, a naive approach would be to follow Eq. (5.1):

$$\mathbf{T}_{\text{tmp}} = \mathbf{M}_{\text{ref}} \cdot \mathbf{T}_{\text{ref}} + \mathbf{M}_{\text{src}} \cdot \mathbf{T}_{\text{src}} \quad (5.1)$$

However, Fig. 5.6 (V1) reveals that only using Eq. (5.1) distorts the geometry and color consistency, and creates stitching seams around the editing borders. In addition, in some cases, the contents in $\mathbf{T}_{\text{ref}}$ and $\mathbf{T}_{\text{src}}$ are not enough to ensure the editing looks natural around the region where two masks meet, necessitating the hallucination of additional content to complement the editing. For example, we may want to remove the long hair from the source image and replace it with the short hair from the reference image. In this case, the pixels that correspond to long hair regions need to be inpainted.

Given the above observations and GAN’s latent spaces embed natural images,

we render the naively fused triplane with the canonical pose π_{can} , re-encode $\mathbf{E}_{\mathcal{W}^+}$, and re-decode via the generator $\mathbf{G}$ to obtain an implicitly fused triplane, shown in Eq. (5.2):

$$\mathbf{T}_{\text{imp}} = \mathbf{G}(\mathbf{E}_{\mathcal{W}^+}(\mathcal{R}(\mathbf{T}_{\text{tmp}}, \pi_{\text{can}}))) \quad (5.2)$$

Note that state-of-the-art image inversion methods for EG3D, such as GOAE [17] and TriPlaneNet [18], employ both low-rate $\mathcal{W}^+$ and high-rate $\mathcal{F}$ codes. The latter is crucial for reconstructing fine image details. However, our objective in this phase is not to achieve perfect image reconstruction. On the contrary, we aim for the encoder to map the image to a latent space with natural reconstructions. To achieve this, we disable the high-frequency restoration branch of $\mathbf{E}_{\mathcal{W}^+}$ to prevent encoding visible seams. This allows us to project the edited image onto its nearest representation on $\mathbf{G}$'s manifold, implicitly fusing the masked triplanes. The output of this step can be seen in Fig. 5.3, showcasing seamless boundaries across the stitches.

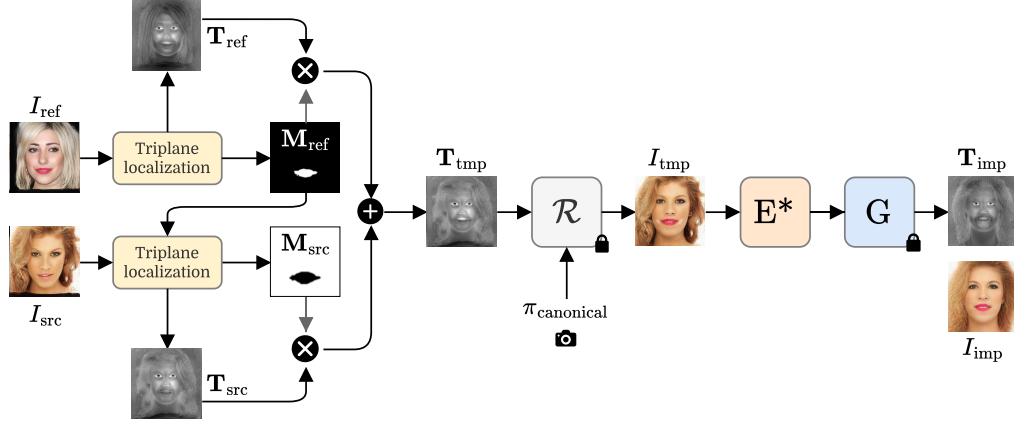


Figure 5.3: Triplane localization and implicit fusion stages, where $\mathbf{E}^*$ denotes the fine-tuned image encoder that is described in Section 5.1.3. Straightforward stitching in the triplane results in color inconsistency across the boundaries, as shown in I_{tmp} (zoom in for details). Leveraging $\mathbf{E}^*$, we aim to attain seamless boundaries and produce outputs with a natural appearance.

Note that image details are compromised at this stage, as we solely depend on the low-rate $\mathcal{W}^+$ space. To bring the details back, we only employ $\mathbf{T}_{\text{imp}}$ at

the transition regions, as depicted in Eq. (5.3). For instance, when we transfer the mouth from reference to source, we aim for the triplane features outside the mouth to originate from $\mathbf{T}_{\text{src}}$, the mouth features from $\mathbf{T}_{\text{ref}}$, and features near mouth from $\mathbf{T}_{\text{imp}}$.

$$\begin{aligned} \mathbf{T}_f = & \mathcal{E}(\mathbf{M}_{\text{ref}}) \cdot \mathbf{T}_{\text{ref}} + \mathcal{E}(\mathbf{M}_{\text{src}}) \cdot \mathbf{T}_{\text{src}} \\ & + (\mathcal{E}(\mathbf{M}_{\text{src}}) - \mathcal{E}(\mathbf{M}_{\text{ref}})) \cdot \mathbf{T}_{\text{imp}} \end{aligned} \quad (5.3)$$

Here, $\mathcal{E}$ denotes morphological erosion. We also apply Gaussian blurring with parameters (μ, σ) onto the masks to avoid sharp edges. This step is not illustrated in Fig. 5.3 for brevity.

Finally, $\mathbf{T}_f$ is rendered from any desired pose π_R , and the final image is obtained, as shown in Eq. (5.4).

$$I_{\text{edited}} = \mathcal{R}(\mathbf{T}_f, \pi_R) \quad (5.4)$$

5.1.3 Fine-tuning the image encoder

Although $\mathbf{T}_{\text{imp}}$ obtained in Section 5.1.2 helps tremendously even though a pre-trained encoder is used to obtain it, we notice in some cases where we have skin color inconsistencies, background leakages, and missing high-frequency details around the editing regions (Fig. 5.6 (V2)). Hence, we fine-tune the implicit fusion encoder $\mathbf{E}_{\mathcal{W}^+}$, jointly with the triplane editing pipeline, to mitigate the aforementioned effects.

During the fine-tuning phase, we generate renders as ground-truths from various viewpoints of the source and reference images corresponding to different attributes. Then, we employ masked losses to guide $\mathbf{E}_{\mathcal{W}^+}$ in encoding only the visible regions, as illustrated in Fig. 5.4. For instance, if our objective is to transfer hair, we mask the reference renderings to exclude pixels that do not represent

hair while doing the opposite for the source ground truth. To ensure that reconstruction losses do not affect the boundaries, we dilate the segmentation masks. The objective function is provided in Eq. (5.5):

$$\begin{aligned} \min_{\mathbf{E}} \lambda_{\Phi} \mathcal{L}_{\Phi}(\mathcal{D}(\tilde{\mathbf{M}}_{\text{ref}}) \cdot \mathcal{R}(\mathbf{T}_{\text{f}}, \pi_i), \mathcal{D}(\tilde{\mathbf{M}}_{\text{ref}}) \cdot \mathcal{R}(\mathbf{T}_{\text{ref}}, \pi_i)) \\ + \lambda_{\text{ID}} \mathcal{L}_{\text{ID}}(\mathcal{D}(\tilde{\mathbf{M}}_{\text{src}}) \cdot \mathcal{R}(\mathbf{T}_{\text{f}}, \pi_i), \mathcal{D}(\tilde{\mathbf{M}}_{\text{src}}) \cdot \mathcal{R}(\mathbf{T}_{\text{src}}, \pi_i)), \end{aligned} \quad (5.5)$$

where $\mathcal{L}_{\Phi}$ is the learned perceptual image patch similarity loss (LPIPS) [82], $\mathcal{L}_{\text{ID}}$ is the identity similarity loss [83], $\mathcal{D}$ is dilation operation, and $\tilde{\mathbf{M}}_{\text{src}}$ and $\tilde{\mathbf{M}}_{\text{ref}}$ are the corresponding 2D segmentation masks for the rendered images $\mathcal{R}(\mathbf{T}_{\text{src}}, \pi_i)$ and $\mathcal{R}(\mathbf{T}_{\text{ref}}, \pi_i)$ with poses π_i , respectively.

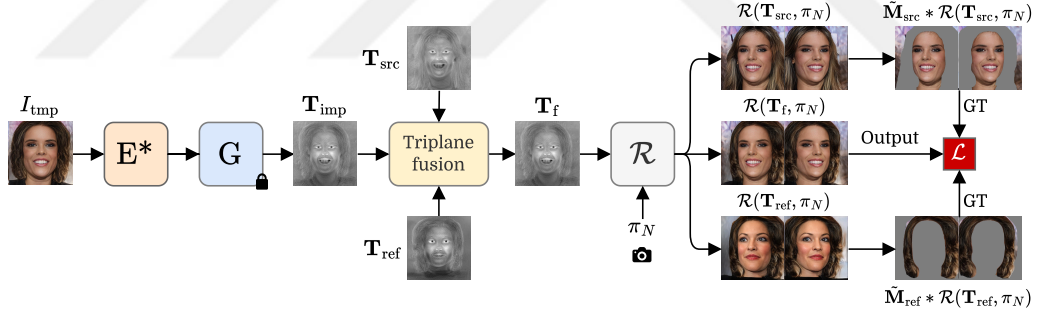


Figure 5.4: Pipeline for the implicit fusion encoder fine-tuning. We generate masked ground truths for our task by utilizing 2D segmentation networks and via renderings from multiple views. We aim to carry the reference parts in great detail to our source image while preserving the source’s identity. The triplane fusion corresponds to Eq. (5.3).

To achieve identity preservation of the source image, we rely on the ID losses, and to copy the attribute with details, we rely on the LPIPS score from Eq. (5.5). We omit pixel-wise losses like $\mathcal{L}_2$ since they are highly dependent on the quality of the off-the-shelf 2D segmentation network. During the fine-tuning, we use each subject tuple $(\mathbf{T}_{\text{src}}, \mathbf{T}_{\text{ref}})$ multiple times, rendered from N randomly chosen π_i ’s.

Compliant with the common encoder training methodology, we use the same training datasets generators are trained with. We employ Ranger optimizer,

which is a combination of Rectified Adam [87] with Lookahead [88]. The learning rate is set to $1e^{-4}$, and fine-tuning is done for 1500 steps with a batch size of 2 on a single RTX 4090.

5.2 Experimental Setup

5.2.1 Evaluation metrics

We present metrics evaluating both the reconstruction and editing qualities of our approach. For editing assessment, we employ the Fréchet Inception Distance (FID) [89], which evaluates realism by comparing the distribution of target images with that of edited images. Specifically, we compute FIDs for adding eyeglasses and hair edits from black to blonde transition using the CelebA [105] dataset. For instance, leveraging ground-truth attribute labels, we add eyeglasses to images without them and compute FIDs between the edited and original images that already have eyeglasses. This procedure is similarly applied to hair edits. For reconstruction evaluation, we mask the edited areas and measure the L2 and Structural Similarity Index (SSIM) between the input and edited images. For instance, in the eyeglasses edit, we mask the eyeglasses and measure the alteration in the unedited regions. This dual assessment framework ensures a robust evaluation of both the fidelity and quality of our editing approach.

5.2.2 Baselines

We conducted comprehensive comparisons by evaluating our method against a range of latent direction, text and image reference-based, 2D and 3D-aware, GAN, and diffusion-based editing methods. Notably, no existing reference-based editing methods achieve 3D consistency within a single framework.

HiSD and VecGAN++ are reference-based 2D image-to-image methods. InterFaceGAN, StyleCLIP ($\mathcal{W}^+$), SFE ($\mathcal{W}^+/\mathcal{F}$), StyleFusion ($\mathcal{W}$), Barbershop, and HairCLIPv2 ($\mathcal{F}/\mathcal{S}$) operate within the latent spaces of StyleGAN and can be adapted to 3D-aware GANs like EG3D. However, only the last three are reference-based, with two focusing on hair edits. For diffusion-based models, LEDITS++ and InfEdit perform text-based editing, while NoiseCLR utilizes pretrained latent edit noise directions. Paint by Example inpaints the masked source image with the reference image but lacks control over which parts of the reference are used.

5.3 Results

5.3.1 Comparison with state-of-the-art

We present quantitative and qualitative comparisons with competing methods in Table 5.1 and Fig. 5.5, respectively. From Table 5.1, it is evident that our method outperforms competing methods significantly in terms of FID and preserves identity better in the non-edited regions.

Observing Fig. 5.5, our method demonstrates superior performance in hair and glasses edits compared to competing methods. HisD and VecGAN++ struggle with maintaining fidelity to the reference, particularly with glasses, due to their reliance on low-rate latent spaces. While InterFaceGAN, StyleCLIP, and SFE can add glasses and perform some hair transfers, they falter with uncommon edits like hat removal (Column 5) and red hair (Column 8) due to the limitations of their $\mathcal{W}^+/\mathcal{F}$ spaces and their non-reference-based approach. LEDITS++ and InfEdit, being text-conditioned, fail to accurately reflect the original reference in their edits. NoiseCLR does not effectively explore hairstyle directions, and its glasses modifications are entangled with makeup changes (Columns 2 and 4). Paint by Example can transfer some features but often produces severe out-of-domain artifacts (Columns 1, 5-8). Barbershop and HairCLIPv2, optimized for hairstyle edits, suffer from geometric inconsistencies (Column 8) and fail in

Table 5.1: Quantitative scores on CelebA. (**X**) indicates the method is not capable of such edits. **First** and second best methods are given in **bold** and underlined, respectively. Time is measured on Tesla T4.

	Eyeglasses			Hair			Time (s)
	FID ↓	$\mathcal{M}_{\text{SSIM}} \uparrow$	$\mathcal{M}_{\mathcal{L}_2} \downarrow$	FID ↓	$\mathcal{M}_{\text{SSIM}} \uparrow$	$\mathcal{M}_{\mathcal{L}_2} \downarrow$	
HiSD [106]	77.56	0.9471	0.0090	94.53	0.9743	0.0036	<u>1.1</u>
VecGAN++ [80]	<u>71.47</u>	0.7483	0.0630	80.47	0.9296	0.0090	2.2
Barbershop [31]	X	X	X	62.80	0.8756	0.0182	125
HairCLIPv2 [32]	X	X	X	85.75	0.8769	0.0173	180
StyleFusion [107]	X	X	X	84.67	0.8435	0.0198	2.4
InterfaceGAN [29]	88.13	0.9398	0.0104	80.93	0.7888	0.0387	0.6
StyleCLIP [20]	80.13	0.8421	0.0476	92.60	0.8716	0.0196	0.6
SFE [34]	106.1	0.9341	0.0099	89.49	0.9355	0.0050	5.1
E3DGE [78]	X	X	X	77.86	0.8083	0.0257	1.2
NoiseCLR [35]	107.1	0.7958	0.0440	X	X	X	17.2
LEDITS++ [36]	115.2	<u>0.9645</u>	<u>0.0047</u>	96.56	0.9717	0.0025	25.9
InfEdit [108]	90.33	0.8338	0.1042	105.4	0.7425	0.0613	9.1
Paint by ex. [33]	74.18	0.8828	0.0252	82.38	0.9229	0.0155	9.6
Ours	66.68	0.9818	0.0021	<u>64.59</u>	<u>0.9720</u>	0.0029	6.0

some hair edit cases (Column 5). Finally, StyleFusion’s feature transfer relies on $\mathcal{W}/\mathcal{W}^+$ directions, resulting in the loss of many high-rate details and outputs that do not fully reflect the original images, especially when the features cannot be well-represented in those domains (Column 8).

Next, we conduct a user study with 25 participants to evaluate our reference-based edits. Participants are shown original and edited images and asked to identify the edited ones. We utilize outputs of EG3D for both original and edited images to neutralize the influence of encoding on the results, and utilize the same angle for the source and edited images with random ordering to minimize bias. Participants could also choose “undecided” if they find it difficult to distinguish. The study focuses on edits to the mouth, eyes, and nose on the FFHQ dataset, and the eyes, nose, and mouth on the AFHQ dataset. Some participants frequently chose “undecided”, while others performed near random chance in identifying edits, in Table 5.2.



Figure 5.5: Comparisons with the competing editing methods for glasses addition and hair edits. Ours, HisD, VecGAN++, Barbershop, StyleFusion, HairCLIPv2, and Paint by Ex. use reference images for editing. InterFaceGAN, StyleCLIP, SFE, and NoiseCLR use previously calculated latent directions. LEDITS++ and InfEdit use text prompts. N/A indicates the model is incapable of such edits.

Table 5.2: The results of our user study, where participants are asked to identify the edited image. Based on this study, we find that our edits are challenging to distinguish. Numbers are percentages.

	FFHQ				AFHQ		
	Eyes	Nose	Mouth	Overall	Eyes	Nose+Mouth	Overall
Original	40	29	34	34	40	42	41
Ours (edited)	49	59	40	49	49	43	46
Undecided	11	13	26	16	11	15	13

5.3.2 Ablation study

We demonstrate the improvements during the development of our pipeline stages, both quantitatively and qualitatively, in Table 5.3 and Fig. 5.6, respectively.

Table 5.3: Quantitative ablation study. V1 is the post-processing triplane gradients, V2 is the implicit fusion, and V3 fine-tunes the implicit fusion encoder.

			Eyeglasses			Hair		
V1	V2	V3	FID ↓	$\mathcal{M}_{\text{SSIM}} \uparrow$	$\mathcal{M}_{\mathcal{L}_2} \downarrow$	FID ↓	$\mathcal{M}_{\text{SSIM}} \uparrow$	$\mathcal{M}_{\mathcal{L}_2} \downarrow$
✗	✗	✗	79.50	0.8451	0.0323	82.42	0.8177	0.0195
✓	✗	✗	74.46	0.9814	0.0022	77.30	0.9674	0.0034
✓	✓	✗	68.19	0.9822	0.0020	67.04	0.9691	0.0033
✓	✓	✓	66.68	0.9818	0.0021	64.59	0.9720	0.0029

In our first ablation study (Fig. 5.6), we apply Eq. (5.1) to merge the triplanes using a mask calculated via an autograd function without any post-processing (No V Column). Due to the intricate volumetric function affecting many pixels for each value in the triplane, the initial mask fails to stitch images effectively, resulting in blurry outputs. Following the introduction of post-processing (+V1 Column), as described in Section 5.1.1, we successfully achieve clear stitching, as depicted in Fig. 5.6. This allows us to transfer the hair of one person to our input while aligning the features using the canonical representation of the triplane. However, the resulting output still lacks realism because it combines features from two different images with varying illuminations, identities, and skin colors. To address the issue of smoothness at stitch boundaries, we follow Section 5.1.2 and perform encoding and decoding via the pretrained encoder and decoder, respectively, on the fused triplane. Since these encoders are trained with real images,

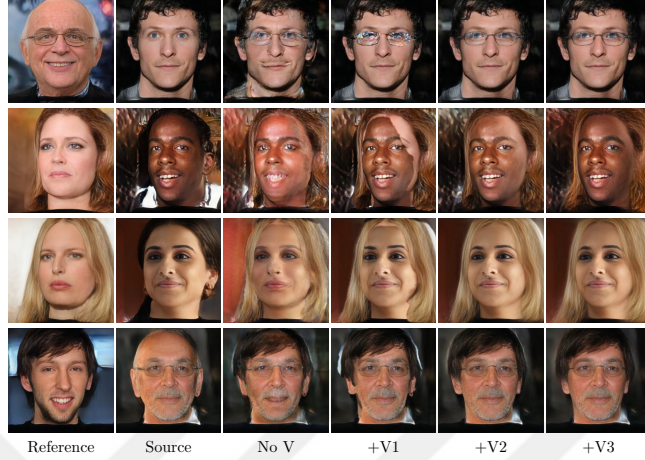


Figure 5.6: Qualitative ablation study for glasses and hair edits, showing the effects of all fundamental stages of our pipeline.

they know about real image priors. Despite the input image not being realistic (+V1 Column), the encoder successfully encodes its latent to the generator’s natural latent space while attempting to preserve the identity in (+V2 Column). However, a pretrained encoder optimized for projecting real images onto the generator’s latent space is not optimal for our specific use case. Consequently, we replace the encoder with one trained specifically for this task, elaborated in Section 5.1.3. This specialized encoder (+V3 Column) ensures color preservation and enhances editing details, such as the coherence of eyeglass frames, as well as reducing background leakage. We note that the qualitative observations comply with the quantitative scores, as displayed in our second ablation study, Table 5.3.

Cross-generator edits. We also provide novel edits in Fig. 5.7, where the reference and source triplanes are gathered from stylized and non-stylized generators, respectively. Specifically, we utilize [22] and fine-tune the EG3D backbones via different text prompts [24]. Then, we synthesize stylized triplanes ($\mathbf{T}_{\text{ref}}$) and perform reference-based editing on the triplanes of default EG3D ($\mathbf{T}_{\text{src}}$). The rest of the steps are the same as before, using the original EG3D and encoder we trained for EG3D. The results presented in Fig. 5.7 demonstrate our method’s independence from backbones, showcasing its capability to achieve part-based attribute stylization. This differs from [22], which offers global stylization.

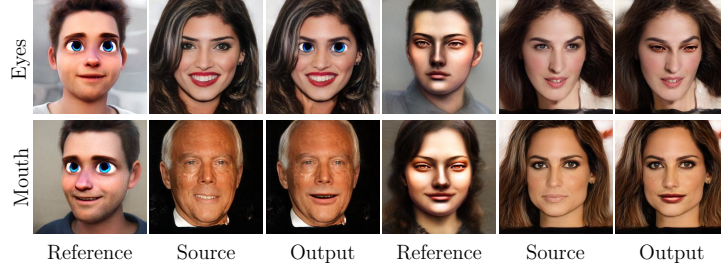


Figure 5.7: Cross-generator edits with stylizing. Our method achieves copying local parts from stylized images, such as cartoon portraits.

Additional face, full body, and 360-degree head edits. Fig. 5.8 demonstrates challenging human face reference-based edits on EG3D [8] like transferring lips, eyeglasses, and nose from one person to another. Fig. 5.9 shows edits on animal face parts for eyes, nose, and mouth. Fig. 5.10 demonstrates fashion edits on AG3D [5] trained with DeepFashion [109] dataset. Fig. 5.11 extends human face part edits to full 360-degree hair edits on PanoHead [6]. It is evident that our approach is generalizable to different triplane generators. While extending our method to different triplane-based generators and datasets, we only changed the 2D segmentation network and the encoder fine-tuning dataset to comply with the generator, when required.

Generalizing to class-agnostic edits. Given the importance of large reconstruction models [39, 38, 37, 110], we extend our method to arbitrary object edits using the triplanes of LN3Diff and InstantMesh [38, 37], proving the potential capabilities of our method. Some results are visualized in Fig. 5.1.

5.3.3 Limitations

Our approach relies on the capabilities of EG3D, AG3D, PanoHead, LN3Diff, and InstantMesh, which may struggle with background generation and high-quality reconstruction. Consequently, in some instances, rich background details may not be fully presented (Column 4 on Fig. 5.5). However, this can be mitigated by not relying on the generator for background generation.

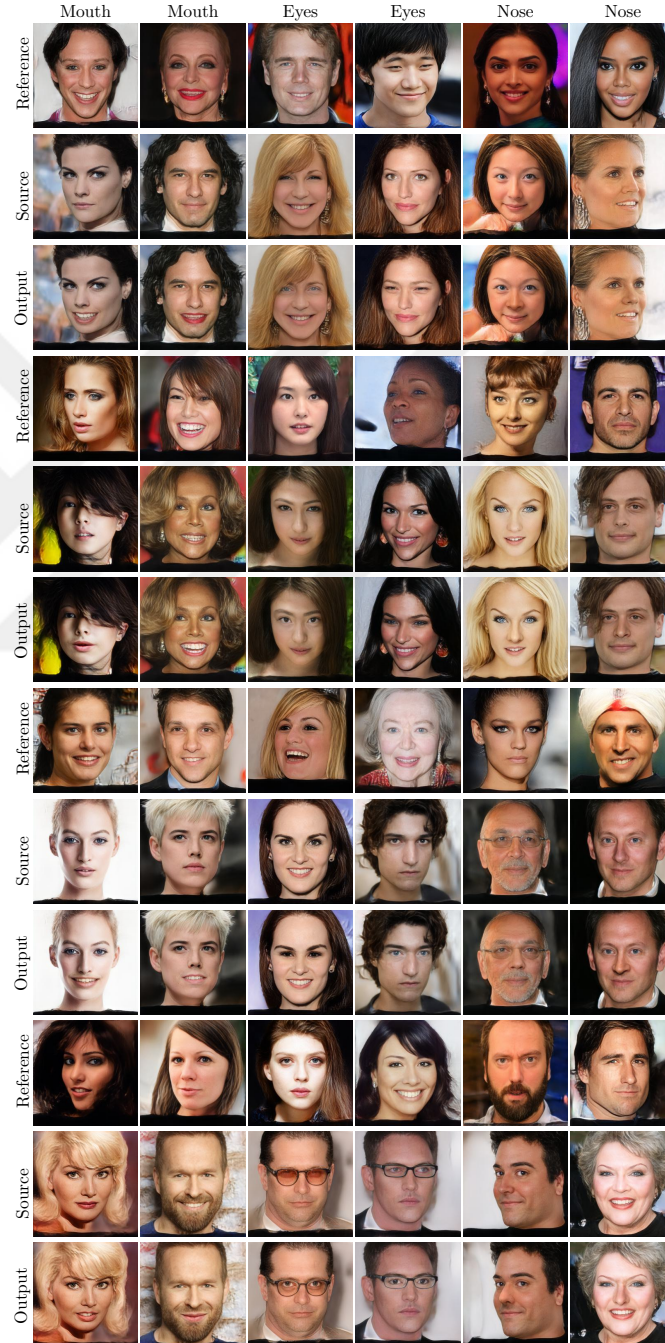


Figure 5.8: Additional editing examples from the CelebA dataset. Our method is able to seamlessly incorporate features such as lips, eyes, and nose from reference to source, despite pose differences and interference like eyeglasses.



Figure 5.9: Additional editing examples from AFHQ dataset. The nose and mouth are handled as a single part.

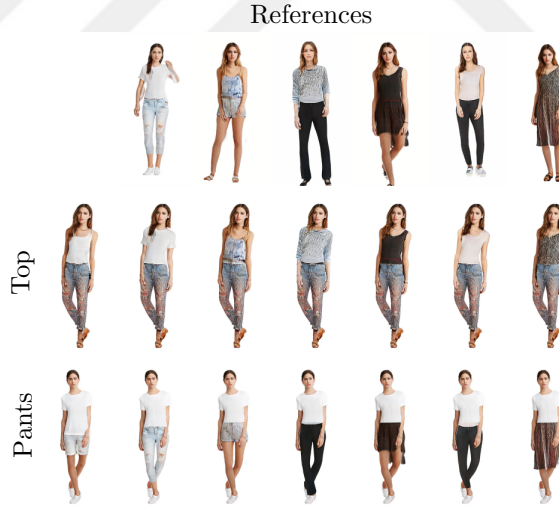


Figure 5.10: Extending our method on try-on edits on [5].



Figure 5.11: Extending our method on full-head hair edits on [6].

Chapter 6

Conclusion and Future Work

This thesis explored single-image generation and editing from three complementary perspectives: faithful reconstruction [9], identity-preserving stylization [10], and reference-guided editing [11]. Leveraging the 360° synthesis ability of PanoHead and recent progress in triplane-based NeRF-GAN hybrids, we introduced new methods outperforming current state-of-the-art and new inspiring perspectives to the research domain.

The first work is a dual-branch GAN-inversion system [9] supported by an occlusion-aware triplane discriminator. Together, they deliver sharper geometry and more reliable novel views than previous single-encoder baselines. The second work [10] distills a latent diffusion model into the GAN domain with negative log-likelihood objectives and score-rank weighting, thereby enabling strong stylization while largely retaining facial identity. The third work [11] presents a spatially disentangled fusion pipeline on the triplane domain that transfers desired features from an example image to another subject, meanwhile maintaining 3D-consistency. Extensive qualitative and quantitative tests confirm improvements over state-of-the-art methods, particularly for challenging poses and attributes.

Future works will seek to perform the proposed methods on larger reconstruction backbones based on diffusion transformers (DiT), extend to class-agnostic

3D models, and experiment with token-level attention fusion to remove the need for canonical alignment spaces.

In closing, the thesis offers a cohesive set of works that reconstructs, stylizes, and edits 3D heads and objects from single images while preserving identity. Although some challenges and limitations are still present, the presented ideas provide a solid baseline for further research and practical applications.



Bibliography

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: Representing scenes as neural radiance fields for view synthesis,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 405 – 421, 2020.
- [2] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 10684–10695, 2022.
- [3] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, “GANSpace: Discovering interpretable GAN controls,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 9841–9850, 2020.
- [4] Y. Shen, C. Yang, X. Tang, and B. Zhou, “InterFaceGAN: Interpreting the disentangled face representation learned by GANs,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 2004–2018, 2022.
- [5] Z. Dong, X. Chen, J. Yang, M. J. Black, O. Hilliges, and A. Geiger, “AG3D: Learning to Generate 3D Avatars from 2D Image Collections,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 14916–14927, 2023.
- [6] S. An, H. Xu, Y. Shi, G. Song, U. Y. Ogras, and L. Luo, “PanoHead: Geometry-aware 3D full-head synthesis in 360 degrees,” in *Proceedings of*

- the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 20950–20959, 2023.
- [7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Proceedings of the International Conference on Neural Information Processing Systems, NeurIPS*, pp. 2672 – 2680, 2014.
 - [8] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. J. Guibas, J. Tremblay, S. Khamis, T. Karras, and G. Wetzstein, “Efficient geometry-aware 3D generative adversarial networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 16123–16133, 2022.
 - [9] B. B. Bilecen, A. B. Gökmen, and A. Dundar, “Dual encoder GAN inversion for high-fidelity 3D head reconstruction from single images,” in *Proceedings of the International Conference on Neural Information Processing Systems, NeurIPS*, pp. 6629 – 6640, 2024.
 - [10] B. B. Bilecen, A. B. Gokmen, F. Guzelant, and A. Dundar, “Identity preserving 3D head stylization with multiview score distillation.” *arXiv preprint, arXiv:2411.13536*, 2024.
 - [11] B. B. Bilecen, Y. Yalin, N. Yu, and A. Dundar, “Reference-based 3D-aware image editing with triplanes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2025.
 - [12] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, “Encoding in Style: A StyleGAN encoder for image-to-image translation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2287–2296, 2021.
 - [13] Y. Alaluf, O. Tov, R. Mokady, R. Gal, and A. Bermano, “HyperStyle: StyleGAN inversion with hypernetworks for real image editing,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 18511–18521, 2022.

- [14] H. Pehlivan, Y. Dalva, and A. Dundar, “StyleRes: Transforming the residuals for real image editing with StyleGAN,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 1828–1837, 2023.
- [15] J. Ko, K. Cho, D. Choi, K. Ryoo, and S. Kim, “3D GAN inversion with pose optimization,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, WACV, pp. 2967–2976, 2023.
- [16] J. Xie, H. Ouyang, J. Piao, C. Lei, and Q. Chen, “High-fidelity 3D GAN inversion by pseudo-multi-view optimization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 321–331, 2023.
- [17] Z. Yuan, Y. Zhu, Y. Li, H. Liu, and C. Yuan, “Make encoder great again in 3D GAN inversion through geometry and occlusion-aware encoding,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 2437–2447, 2023.
- [18] A. R. Bhattarai, M. Nießner, and A. Sevastopolsky, “TriPlaneNet: An encoder for EG3D inversion,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, WACV, pp. 3043–3053, 2024.
- [19] R. Gal, O. Patashnik, H. Maron, A. H. Bermano, G. Chechik, and D. Cohen-Or, “StyleGAN-NADA: CLIP-guided domain adaptation of image generators,” *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 1–13, 2022.
- [20] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, “StyleCLIP: Text-driven manipulation of StyleGAN imagery,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 2065–2074, 2021.
- [21] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of StyleGAN,” in *Proceedings of*

- the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 8107–8116, 2020.
- [22] K. Song, L. Han, B. Liu, D. Metaxas, and A. Elgammal, “StyleGAN-Fusion: Diffusion guided domain adaptation of image generators,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, WACV*, pp. 5441–5451, 2024.
 - [23] B. Lei, K. Yu, M. Feng, M. Cui, and X. Xie, “DiffusionGAN3D: Boosting text-guided 3D generation and domain adaptation by combining 3D GANs and diffusion priors,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 10487–10497, 2024.
 - [24] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, “DreamFusion: Text-to-3D using 2D diffusion,” in *Proceedings of the International Conference on Learning Representations, ICLR*, 2023.
 - [25] S. Huang, S. Sun, Z. Wang, X. Qin, Y. Xiong, Y. Zhang, P. Wan, D. Zhang, and J. Jia, “PlacidDreamer: Advancing harmony in text-to-3D generation,” in *Proceedings of the ACM International Conference on Multimedia*, p. 6880–6889, 2024.
 - [26] Y. Alaluf, O. Patashnik, and D. Cohen-Or, “ReStyle: A residual-based StyleGAN encoder via iterative refinement,” in *Proceedings of the International Conference on Computer Vision, ICCV*, pp. 6711–6720, 2021.
 - [27] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, “Designing an encoder for StyleGAN image manipulation,” *ACM Transactions on Graphics*, vol. 40, pp. 1–14, July 2021.
 - [28] A. B. Yildirim, H. Pehlivan, B. B. Bilecen, and A. Dundar, “Diverse inpainting and editing with GAN inversion,” in *Proceedings of the International Conference on Computer Vision, ICCV*, pp. 23120–23130, October 2023.

- [29] Y. Shen, J. Gu, X. Tang, and B. Zhou, “Interpreting the latent space of GANs for semantic face editing,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 9240–9249, 2020.
- [30] A. Frühstück, N. Sarafianos, Y. Xu, P. Wonka, and T. Tung, “VIVE3D: Viewpoint-independent video editing using 3D-aware GANs,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 4446–4455, 2023.
- [31] P. Zhu, R. Abdal, J. Femiani, and P. Wonka, “Barbershop: GAN-based image compositing using segmentation masks,” *ACM Transactions on Graphics*, vol. 40, no. 6, pp. 1–13, 2021.
- [32] T. Wei, D. Chen, W. Zhou, J. Liao, W. Zhang, G. Hua, and N. Yu, “Hair-CLIPv2: Unifying hair editing via proxy feature blending,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 23589–23599, 2023.
- [33] B. Yang, S. Gu, B. Zhang, T. Zhang, X. Chen, X. Sun, D. Chen, and F. Wen, “Paint by example: Exemplar-based image editing with diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 18381–18391, 2023.
- [34] D. Bobkov, V. Titov, A. Alanov, and D. Vetrov, “The Devil is in the Details: StyleFeatureEditor for detail-rich StyleGAN inversion and high quality image editing,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 9337–9346, 2024.
- [35] Y. Dalva and P. Yanardag, “NoiseCLR: A contrastive learning approach for unsupervised discovery of interpretable directions in diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 24209–24218, 2024.

- [36] M. Brack, F. Friedrich, K. Kornmeier, L. Tsaban, P. Schramowski, K. Kersting, and A. Passos, “LEDITS++: Limitless image editing using text-to-image models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 8861–8870, 2024.
- [37] Y. Lan, F. Hong, S. Yang, S. Zhou, X. Meng, B. Dai, X. Pan, and C. C. Loy, “LN3Diff: Scalable latent neural fields diffusion for speedy 3D generation,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 112 – 130, 2024.
- [38] J. Xu, W. Cheng, Y. Gao, X. Wang, S. Gao, and Y. Shan, “InstantMesh: Efficient 3D mesh generation from a single image with sparse-view large reconstruction models.” *arXiv preprint, arXiv:2404.07191*, 2024.
- [39] Y. Hong, K. Zhang, J. Gu, S. Bi, Y. Zhou, D. Liu, F. Liu, K. Sunkavalli, T. Bui, and H. Tan, “LRM: Large reconstruction model for single image to 3D,” in *Proceedings of the International Conference on Learning Representations*, ICLR, 2024.
- [40] J. Lin, “Divergence measures based on the Shannon entropy,” *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 145–151, 1991.
- [41] L. Weng, “From GAN to WGAN.” Available at <https://lilianweng.github.io/posts/2017-08-20-gan/>.
- [42] V. Herrmann, “Wasserstein GAN and the Kantorovich-Rubinstein duality.” Available at <https://vincentherrmann.github.io/blog/wasserstein/>.
- [43] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proceedings of the International Conference on Machine Learning*, ICML, pp. 214–223, 2017.
- [44] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of Wasserstein GANs,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 5769 – 5779, 2017.

- [45] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, “Implicit geometric regularization for learning shapes,” in *Proceedings of the International Conference on Machine Learning*, ICML, pp. 3789–3799, 2020.
- [46] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3D Gaussian splatting for real-time radiance field rendering,” *ACM Transactions on Graphics*, vol. 42, pp. 1 – 14, July 2023.
- [47] S. Govindarajan, D. Rebain, K. M. Yi, and A. Tagliasacchi, “Radiant Foam: Real-time differentiable ray tracing.” *arXiv preprint, arXiv:1810.04805*, 2025.
- [48] O. Kaduri, “Neural implicit representations for 3D shapes and scenes.” Available at <https://omrikaduri.github.io>.
- [49] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy Networks: Learning 3D reconstruction in function space,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 4455–4465, 2019.
- [50] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning continuous signed distance functions for shape representation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 165–174, 2019.
- [51] A. Tagliasacchi and B. Mildenhall, “Volume rendering digest (for NeRF).” *arXiv preprint, arXiv:2209.02417*, 2022.
- [52] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, “Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 5835–5844, 2021.
- [53] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, “Mip-NeRF 360: Unbounded anti-aliased neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 5460–5469, 2022.

- [54] F. Rivas-Manzanque, J. Sierra-Acosta, A. Penate-Sanchez, F. Moreno-Noguer, and A. Ribeiro, “NeRFLight: Fast and light neural radiance fields using a shared feature grid,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 12417–12427, 2023.
- [55] T. Hu, S. Liu, Y. Chen, T. Shen, and J. Jia, “EfficientNeRF: Efficient neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 12902–12911, 2022.
- [56] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, “Pi-GAN: Periodic implicit generative adversarial networks for 3d-aware image synthesis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 5799–5809, 2021.
- [57] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 6840 – 6851, 2020.
- [58] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-based generative modeling through stochastic differential equations,” in *Proceedings of the International Conference on Learning Representations*, ICLR, 2021.
- [59] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 4396–4405, 2019.
- [60] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, “Alias-free generative adversarial networks,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 852–863, 2021.
- [61] M. Tatarchenko, A. Dosovitskiy, and T. Brox, “Multi-view 3D models from single images with a convolutional network,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 322–337, 2016.

- [62] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik, “Learning shape abstractions by assembling volumetric primitives,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 1466–1474, 2017.
- [63] T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, “Holo-GAN: Unsupervised learning of 3d representations from natural images,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 7587–7596, 2019.
- [64] T. H. Nguyen-Phuoc, C. Richardt, L. Mai, Y. Yang, and N. Mitra, “Block-GAN: Learning 3d object-aware scene representations from unlabelled images,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 6767–6778, 2020.
- [65] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik, “Learning category-specific mesh reconstruction from image collections,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 386–402, 2018.
- [66] S. Goel, A. Kanazawa, and J. Malik, “Shape and viewpoint without keypoints,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 88–104, 2020.
- [67] P. Henderson, V. Tsiminaki, and C. H. Lampert, “Leveraging 2D data to learn textured 3D mesh generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 7495–7504, 2020.
- [68] J. Gu, L. Liu, P. Wang, and C. Theobalt, “StyleNeRF: A style-based 3D-aware generator for high-resolution image synthesis,” in *Proceedings of the International Conference on Learning Representations*, ICLR, 2022.
- [69] R. Or-El, X. Luo, M. Shan, E. Shechtman, J. J. Park, and I. Kemelmacher-Shlizerman, “StyleSDF: High-resolution 3D-consistent image and geometry generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 13503–13513, 2022.

- [70] J. Sun, X. Wang, Y. Zhang, X. Li, Q. Zhang, Y. Liu, and J. Wang, “FENeRF: Face editing in neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 7672–7682, 2022.
- [71] A. Trevithick, M. Chan, M. Stengel, E. Chan, C. Liu, Z. Yu, S. Khamis, M. Chandraker, R. Ramamoorthi, and K. Nagano, “Real-time radiance fields for single-image portrait view synthesis,” *ACM Transactions on Graphics*, vol. 42, pp. 1 – 15, July 2023.
- [72] H. Li, C. Chen, T. Shi, Y. Qiu, S. An, G. Chen, and X. Han, “Sphere-Head: Stable 3D full-head synthesis with spherical tri-plane representation,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 324–341, 2025.
- [73] D. Rezende and S. Mohamed, “Variational inference with normalizing flows,” in *Proceedings of the International Conference on Machine Learning*, vol. 37 of *PMLR*, pp. 1530–1538, 2015.
- [74] R. Abdal, Y. Qin, and P. Wonka, “Image2StyleGAN++: How to edit the embedded images?,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 8293–8302, 2020.
- [75] D. Roich, R. Mokady, A. H. Bermano, and D. Cohen-Or, “Pivotal tuning for latent-based editing of real images,” *ACM Transactions on Graphics*, vol. 42, no. 1, pp. 1 – 13, 2022.
- [76] T. Wang, Y. Zhang, Y. Fan, J. Wang, and Q. Chen, “High-fidelity GAN inversion for image attribute editing,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 11379–11388, 2022.
- [77] J. Sun, X. Wang, Y. Shi, L. Wang, J. Wang, and Y. Liu, “IDE-3D: Interactive disentangled editing for high-resolution 3D-aware portrait synthesis,” *ACM Transactions on Graphics*, vol. 41, no. 6, pp. 1–10, 2022.

- [78] Y. Lan, X. Meng, S. Yang, C. C. Loy, and B. Dai, “Self-supervised geometry-aware encoder for style-based 3D GAN inversion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 20940–20949, 2023.
- [79] F. Yin, Y. Zhang, X. Wang, T. Wang, X. Li, Y. Gong, Y. Fan, X. Cun, Y. Shan, C. Oztireli, and Y. Yang, “3D GAN Inversion with facial symmetry prior,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 342–351, 2023.
- [80] Y. Dalva, S. F. Altındaş, and A. Dundar, “VecGAN: Image-to-image translation with interpretable latent directions,” in *Proceedings of the European Conference on Computer Vision*, ECCV, p. 153–169, 2022.
- [81] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sasstry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, “Learning transferable visual models from natural language supervision.” *arXiv preprint, arXiv:2103.00020*, 2021.
- [82] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 586–595, 2018.
- [83] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “ArcFace: Additive angular margin loss for deep face recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 4685–4694, 2019.
- [84] Y. Wu, J. Zhang, H. Fu, and X. Jin, “LPFF: A portrait dataset for face generators across large poses,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 20327–20337, 2023.
- [85] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” in *Proceedings of the International Conference on Learning Representations*, ICLR, pp. 213–239, 2018.

- [86] K. Wang, Q. Wu, L. Song, Z. Yang, W. Wu, C. Qian, R. He, Y. Qiao, and C. C. Loy, “MEAD: A large-scale audio-visual dataset for emotional talking-face generation,” in *Proceedings of the European Conference on Computer Vision*, ECCV, pp. 700–717, 2020.
- [87] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, “On the variance of the adaptive learning rate and beyond,” in *Proceedings of the International Conference on Learning Representations*, ICLR, pp. 7074–7087, 2020.
- [88] M. Zhang, J. Lucas, J. Ba, and G. E. Hinton, “Lookahead optimizer: k steps forward, 1 step back,” in *Proceedings of the International Conference on Neural Information Processing Systems*, vol. 32 of *NeurIPS*, pp. 9597 – 9608, 2019.
- [89] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs trained by a two time-scale update rule converge to a local nash equilibrium,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 6629 – 6640, 2017.
- [90] Z. Wang, C. Lu, Y. Wang, F. Bao, C. Li, H. Su, and J. Zhu, “Prolific-Dreamer: High-fidelity and diverse text-to-3D generation with variational score distillation,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 8406 – 8441, 2023.
- [91] J. Ho and T. Salimans, “Classifier-free diffusion guidance,” in *Proceedings of the International Conference on Neural Information Processing Systems*, Workshop on Deep Generative Models and Downstream Applications, 2021.
- [92] T. A. Vass, “Explaining the SDXL latent space: A complete demonstration.” Available at <https://huggingface.co/blog/TimothyAlexisVass/explaining-the-sdxl-latent-space>, 2024.
- [93] L. Zhang, A. Rao, and M. Agrawala, “Adding conditional control to text-to-image diffusion models,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 3836–3847, 2023.

- [94] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, “Zero-1-to-3: Zero-shot one image to 3D object,” in *Proceedings of the International Conference on Computer Vision, ICCV*, pp. 9298–9309, 2023.
- [95] Y. Yang, Y. Huang, X. Wu, Y.-C. Guo, S.-H. Zhang, H. Zhao, T. He, and X. Liu, “DreamComposer: Controllable 3D object generation via multi-view conditions,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 8111–8120, 2024.
- [96] Y. Liu, C. Lin, Z. Zeng, X. Long, L. Liu, T. Komura, and W. Wang, “SyncDreamer: Generating multiview-consistent images from a single-view image,” in *Proceedings of the International Conference on Learning Representations, ICLR*, 2024.
- [97] T. Lee, S. Kwon, and T. Kim, “Grid diffusion models for text-to-video generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 8734–8743, 2024.
- [98] O. Kara, B. Kurtkaya, H. Yesiltepe, J. M. Rehg, and P. Yanardag, “RAVE: Randomized noise shuffling for fast and consistent video editing with diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 6507–6516, 2024.
- [99] Q. Wang, X. Bai, H. Wang, Z. Qin, and A. Chen, “InstantID: Zero-shot identity-preserving generation in seconds.” *arXiv preprint, arXiv:2401.07519*, 2024.
- [100] T. Brooks, A. Holynski, and A. A. Efros, “InstructPix2Pix: Learning to follow image editing instructions,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 18392–18402, 2023.
- [101] A. Haque, M. Tancik, A. A. Efros, A. Holynski, and A. Kanazawa, “Instruct-NeRF2NeRF: Editing 3D scenes with instructions,” in *Proceedings of the International Conference on Computer Vision, ICCV*, pp. 19740–19750, 2023.

- [102] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, “Depth Anything V2,” in *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS, pp. 21875–21911, 2024.
- [103] Y. Wu, H. Xu, X. Tang, X. Chen, S. Tang, Z. Zhang, C. Li, and X. Jin, “Portrait3D: Text-guided high-quality 3D portrait generation using pyramid representation and GANs prior,” *ACM Transactions on Graphics*, vol. 43, no. 4, pp. 1–12, 2024.
- [104] zllrunning, “Face parsing in PyTorch.” Available at <https://github.com/zllrunning/face-parsing.PyTorch/>.
- [105] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proceedings of the International Conference on Computer Vision*, ICCV, pp. 3730–3738, 2015.
- [106] X. Li, S. Zhang, J. Hu, L. Cao, X. Hong, X. Mao, F. Huang, Y. Wu, and R. Ji, “Image-to-image translation via hierarchical style disentanglement,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 8639–8648, 2021.
- [107] O. Kafri, O. Patashnik, Y. Alaluf, and D. Cohen-Or, “StyleFusion: Disentangling spatial segments in StyleGAN-generated images,” *ACM Transactions on Graphics*, vol. 41, pp. 1–15, Oct. 2022.
- [108] S. Xu, Y. Huang, J. Pan, Z. Ma, and J. Chai, “Inversion-free image editing with language-guided diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 9452–9461, 2024.
- [109] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, “DeepFashion: Powering robust clothes recognition and retrieval with rich annotations,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR, pp. 1096–1104, 2016.
- [110] L. Zhang, Z. Wang, Q. Zhang, Q. Qiu, A. Pang, H. Jiang, W. Yang, L. Xu, and J. Yu, “CLAY: A controllable large-scale generative model for creating

high-quality 3D assets,” *ACM Transactions on Graphics*, vol. 43, no. 4, pp. 1–20, 2024.

