Expanding the Generality of Neural Fields

Yushi Lan

College of Computing and Data Science

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2025

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

01/08/2024

Date

lan yushi

Yushi Lan

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

01/08/2024

Date

Prof. Loy Chen Change

Authorship Attribution Statement

This thesis includes material from one accepted journal paper and three accepted conference papers, in which I am listed as an author.

Chapter 3 is published as **Yushi Lan**, Chen Change Loy and Bo Dai. DDF: Correspondence Distillation from NeRF-based GAN. In *IJCV*, 2022.

The contributions of the co-authors are outlined as follows:

- Prof. Loy and Dr. Dai provided the initial project direction and revised the manuscript drafts.
- All authors participated in the regular discussion.
- I prepared the manuscript draft. The manuscript was revised by other co-authors.
- I designed the network and performed all the laboratory work.

Chapter 4 is published as **Yushi Lan**, Xuyi Meng, Shuai Yang, Chen Change Loy and Dai, Bo. E3DGE: Self-Supervised Geometry-Aware Encoder for Style-based 3D GAN Inversion. In *CVPR*, 2023.

The contributions of the co-authors are outlined as follows:

- Prof. Loy and Dr. Yang provided the initial project direction and revised the manuscript drafts.
- All authors participated in the regular discussion.
- I prepared the manuscript draft. The manuscript was revised by other co-authors.
- I designed the network and performed all the laboratory work.

Chapter 5 is published as **Yushi Lan**, Feitong Tan, Qiangeng Xu, Di Qiu, Kyle Genova, Zeng Huang, Sean Fanello, Rohit Pandey, Thomas Funkhouser, Chen Change Loy and Yinda Zhang. Loc3Diff: Local Diffusion for 3D Human Head Synthesis and Editing. In *ECCV*, 2024.

The contributions of the co-authors are outlined as follows:

- All authors discussed and came up with the initial project direction.
- All authors participated in the regular discussion.
- I prepared the manuscript draft. The manuscript was revised by other co-authors.
- I collected the dataset, designed the networks and performed the laboratory work.

Chapter 6 is published as **Yushi Lan**, Fangzhou Hong, Shuai Yang, Shangchen Zhou, Xuyi Meng, Bo Dai, Xingang Pan and Chen Change Loy. LN3Diff: Scalable Latent Neural Fields Diffusion for Speedy 3D Generation. In *ECCV*, 2024.

The contributions of the co-authors are outlined as follows:

- Prof. Loy and Prof. Pan provided the initial project direction and revised the manuscript drafts.
- All authors participated in the regular discussion.
- I prepared the manuscript draft. The manuscript was revised by other co-authors.
- I designed the networks and performed the laboratory work.

20/08/2024

Date

Yushi Lan

Acknowledgements

Completing this Ph.D. journey has been a profoundly rewarding experience, and it would not have been possible without the invaluable support, guidance, and encouragement from my beloved ones whom I am deeply grateful to acknowledge.

First and foremost, I would like to express my deepest gratitude to my advisor, Prof. Chen Change Loy, for granting me the opportunity to pursue my PhD study in the MMLab@NTU. Over the years, he has perpetually offered me support and guidance. When my project stalls, Prof. Loy always encouraged and believed in me, reminding me that "*No worries, good work always takes time.*" It is Prof. Loy who makes me successfully complete my PhD journey. His exceptional research insight, rigorous attitude, and amiable character have profoundly influenced and shaped me a lot.

Special thanks go to Prof. Xingang Pan for his selfless guidance and assistance. His support and unwavering belief in me lit up my path during a perplexing period and encouraged me to embrace the challenge of pursuing a PhD.

Heartfelt gratitude extends to my other amazing collaborators, Prof. Shuai Yang, Prof. Bo Dai, Dr. Shangchen Zhou, Mr. Fangzhou Hong, Dr. Kelvin C.K. Chan, Dr. Yuming Jiang, Mr. Ruicheng Feng, Dr. Yinda Zhang, Mr. Zhaoxi Chen, Dr. Feitong Tan, Dr. Qiangeng Xu, Dr. Di Qiu, Dr. Zhaoyang Lyv, Mr. Tianxing Wu, Mr. Jiawei Ren, Mr. Jianyi Wang, Mr. Yongwei Chen, Dr. Honghua Chen, Ms. Zhexin Liang, Ms. Ziqi Huang, and amongst others. The moments we spent discussing together and our enjoyable collaborations hold precious memories for me. A special acknowledgement is also dedicated to Prof. Ziwei Liu for being my internal TAC members throughout my PhD study, and to my thesis committee for their invaluable feedback and comments.

I am immensely grateful for my cherished labmates and friends their friendship and support. My PhD life would have been pretty dull without them. They added so

much meaning to my studies and brought immense joy into my life. The fun times we shared will always hold a special place in my heart.

Last but not least, I am indebted beyond words to my family in particular my mother Hongmei Zhang and my father Keqin Lan for their enduring love, support and understanding. They have always had my back, supporting and believing in me no matter what. Without them, I would never have finished my studies. Their love has been my strongest support all along.

Abstract

Neural fields have emerged as a groundbreaking approach to representing 3D shapes, garnering significant attention due to their compatibility with modern deep-learning techniques. Neural fields, which parameterize physical properties of scenes or objects across space and time, have achieved remarkable success in tasks such as 3D shape and image synthesis, animation, 3D reconstruction, and pose estimation. However, these promising results are predominantly achieved by overfitting on individual scenes or objects. That is, the research on *generalizable neural fields* has been largely overlooked. This limitation hinders neural fields' potential applications in downstream tasks, e.g., single-view 3D reconstruction, 3D object generation, and editing. To address this issue, this thesis advances generalizable neural field methods, including *generalizable neural field algorithms*, which improve training methodologies, and *generalizable neural field representations*, which propose novel 3D representations for specific applications. These methods demonstrate significant potential for developing solutions that are effective, robust, and practical.

For *generalizable neural field algorithms*, correspondence-level generalization is first explored. Unlike explicit shape representations, e.g., meshes, it remains an open problem to establish dense correspondences across Neural Radiance Fields (NeRFs) of the same category. This problem is particularly challenging due to the implicit nature of NeRFs and the lack of ground-truth correspondence annotations. This thesis shows that these challenges can be addressed by leveraging the rich semantics and structural priors encapsulated in pre-trained NeRF-based GANs. Specifically, three key innovations are introduced: 1) a dual deformation field guided by latent codes as global structural indicators, 2) a learning objective that uses generator features as geometry-aware local descriptors, and 3) a method for generating infinite object-specific NeRF samples. Experiments demonstrate that these innovations enable accurate, smooth, and robust 3D dense correspondences, facilitating downstream applications such as texture transfer.

To bridge the gap to real-world scenarios, this thesis further explores object-level generalization for neural fields. Specifically, this thesis proposes E3DGE, a framework addressing the challenge of 3D GAN inversion—predicting a latent code from a single 2D image to faithfully recover 3D shapes and textures. The inherent ill-posed nature of the problem, coupled with the limited capacity of global latent codes, presents significant challenges. To overcome these challenges, this thesis introduces an efficient self-training scheme that does not rely on real-world 2D-3D pairs but instead utilizes proxy samples generated from a 3D GAN. Additionally, the proposed approach enhances the generation network with a local branch that incorporates pixel-aligned features to accurately reconstruct texture details. Furthermore, a novel pipeline for 3D view-consistent editing is introduced. The efficacy of the proposed method is validated on two representative 3D GANs, namely StyleSDF and EG3D. Extensive experiments demonstrate that the proposed approach consistently outperforms state-of-the-art inversion methods, delivering superior quality in both shape and texture reconstruction.

For *generalizable neural field representations*, this thesis first investigates neural field representations for generalizable 3D avatar heads. A novel framework is presented for generating photorealistic 3D human heads and subsequently manipulating and reposing them with remarkable flexibility. The proposed approach constructs an implicit representation of 3D human heads, anchored on a parametric face model. To enhance representational capabilities and encode spatial information, the semantically consistent head region is represented by a local tri-plane modulated by a 3D Gaussian. Additionally, these tri-planes are further parameterized in a 2D UV space via a 3DMM, enabling effective utilization of the diffusion model for 3D head avatar generation. The proposed method facilitates the creation of diverse and realistic 3D human heads with flexible global and fine-grained region-based editing over facial structures, appearance, and expressions. Extensive experiments demonstrate the effectiveness of the proposed method.

Finally, this thesis focuses on designing neural field representations for general 3D objects and unifying the 3D diffusion pipeline within the latent diffusion paradigm. Specifically, a novel framework called LN3Diff is proposed to enable fast, high-quality, and generic conditional 3D generation. The proposed method employs a 3D-aware architecture and variational autoencoder (VAE) to encode the input image into a structured, compact, 3D latent space. The latent is decoded by a

transformer-based decoder into a high-capacity 3D neural field. By training a diffusion model on this 3D-aware latent space, the proposed method achieves state-of-the-art performance on ShapeNet for 3D generation. It also demonstrates superior performance in monocular 3D reconstruction and conditional 3D generation across various datasets. Moreover, it surpasses existing 3D diffusion methods in inference speed, requiring no per-instance optimization. The proposed LN3Diff represents a significant advancement in 3D generative modeling and holds promise for diverse applications in 3D vision and graphics tasks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Novel view synthesis has received significant attention over the past decades and has many practical applications in computer vision and the graphics community. The objective of novel view synthesis is to recover the underlying 3D scene given 2D observations to enable free viewpoint scene rendering. However, with only partial 2D observations available, the inevitable color-shape ambiguity inherent in this formulation makes itself an ill-posed problem, and the ill-posedness of this problem increases with the decrease of visible views.

Researchers have proposed numerous methods to tackle this task. With the rise of deep learning, researchers propose to combine neural networks with 3D representations from the graphics domain. These attempts have led to recent success of neural fields [9], which has shown remarkable progress in 3D modeling [10], view synthesis [2], and 3D content generation [11]. Unlike explicit 3D representations, e.g., mesh, neural fields represent each 3D object as a continuous parametric function. By approximating the parametric function with modern neural networks, neural fields have shown great potential in capturing geometric scene details, rendering realistic novel views, and bridging the field of vision and graphics, namely *neural rendering* [12].

However, the original neural fields [2, 13–15] overfit to a single scene to achieve high-quality novel view synthesis. Though achieving perfect photorealism, the neural fields learned in this way cannot be applied to novel scenes, i.e., *a lack of generality*. This deficiency greatly hinders neural fields' applications over downstream 3D editing tasks [16], monocular 3D reconstruction [7], avatar generation [17] and scaling

the learned models to category-free 3D dataset [18]. Going beyond photorealistic reconstruction, recent works have started to delve into the exploration of generalizable neural fields through image-based rendering [3, 19] or distilling 2D model priors [20]. However, these methods are trained with discriminative objectives and yield blurry results when few observations are available [20]. The limitations of the existing works motivate this thesis to conduct studies over better generalizable neural fields to facilitate the downstream applications mentioned above.

With the rapid development of AIGC, 3D generative models such as 3D GANs (Generative Adversarial Networks) [1, 8, 21, 22] and 3D diffusion models [23] propose a new paradigm of learning generalizable models from large-scale data. Motivated by the widespread success of generative models, this thesis focuses more on expanding the generality of neural fields from the *generative model* perspective, where either the pre-trained 3D generative models are used as priors [7, 16] for generalizable neural fields learning, or the novel 3D generative models are carefully developed from scratch [17, 24]. Specifically, this thesis will present its findings from two perspectives: *generalizable neural field algorithms from generative priors* and *generalizable neural field representations for 3D generation* to advance this field, both leveraging the techniques of generative models. By learning a (variational) autodecoder, generalizable neural fields can be achieved by learning a 3D generative model on the latent space. Due to the complexity of the 3D world, this thesis focuses on object-level generalization, such as 3D head and general 3D objects.

Recent advancements have highlighted the immense potential of neural field algorithms across various applications. Beyond creating 3D assets for movie CG and digital games, these techniques now extend into emerging areas such as embodied AI, VR/AR, and digital humans. Moreover, these techniques facilitate applications in cutting-edge fields like autonomous driving scene simulation, visual effects (VFX) for films, and the metaverse virtual world. In these scenarios, neural fields and their corresponding neural rendering techniques are crucial for assisting 3D artists in creating high-quality 3D assets with greater ease. We believe that with more generalizable designs, neural fields will experience even broader adoption across various domains in the future.

1.1 Challenges and Motivations

Modeling the dense 3D correspondences across 3D assets has been a consistent focus in computer graphics and vision. While producing impressive results in traditional 3D representations, e.g., mesh, these methods still fall short of modeling continuous 3D representations and cannot be directly applied to neural fields. Therefore, there is still a need for further exploration in designing algorithms that can establish dense correspondences across neural fields, e.g., NeRF (Neural Radiance Field [2]). Moreover, due to the lack of abundant NeRF-based 3D representations for learning correspondence models, how to resolve the data deficiency remains under investigation.

In addition to directly editing generated or multi-view reconstructed NeRF-based 3D assets, how to inverse a neural field from a single-view image is also extremely important and challenging. Moreover, since 3D space is exponentially more complicated compared to 2D space, the inversion algorithms designed for 2D space are not suitable for achieving high-fidelity 3d reconstruction, especially when only a single-view image is available.

Beyond designing more advanced generalizable algorithms for generic neural fields, this thesis also focuses on developing task-specific generalizable 3D representations for better 3D reconstruction, editing, and content generation. However, the proposed 3D representations shall address three challenges: 1) high-fidelity 3D reconstruction, 2) compatibility with the 3D generation framework, and 3) editing flexibility required by the specific domain.

1.1.1 Dense 3D Correspondence on NeRF

Dense 3D correspondence, as a core problem in computer graphics and vision, has been well-studied over explicit shape representations. While significant progress has been made on traditional 3D representations, as an emerging promising shape representation, the relevant study on the neural fields is missing. This thesis extends such effort to one canonical neural field, i.e., NeRF, and establishes dense 3D correspondence across two objects represented by NeRF. The proposed method potentially enables downstream applications such as texture transfer, texture manipulation, and segmentation transfer.

However, this task is non-trivial. First, existing methods for building dense correspondence across two objects mainly focus on mesh-based representations. It is infeasible to directly apply them to NeRF. Unlike meshes that have explicit vertices and surfaces, Besides, since NeRF lacks an explicit surface, it is hard to resort to derivatives of neural fields [25] as the shape surface descriptors. Moreover, existing methods [26] often require ground-truth correspondence annotations in training, which are hard to obtain for NeRF-based object representations.

1.1.2 Inversion and Editing on 3D GANs

GAN inversion [27] is a challenging task that has been extensively studied for 2D images but remains underexplored in the 3D world. Specifically, a given image is projected to the latent space and obtains an editable latent code with an encoder. The latent code will be fed to a generator to reconstruct the corresponding 3D shape with high-quality shape and texture. Various optimization-based techniques such as PTI [28] and global-local fusion [29] have been well-studied for 2D GAN inversion and editing tasks. However, these methods focus more on the 2D texture side and are not intrinsically designed for 3D GANs. To achieve high-quality 3D GAN inversion, some challenges are posed for 3D GAN inversion: 1) the lack of 2D-3D paired training data, 2) high-fidelity 3D reconstruction given only monocular images, and 3) view-consistent editing.

1.1.3 Avatar 3D Head Representations

Generating and editing photorealistic 3D portraits is one of the cruxes of computer graphics and has tremendous demand in downstream applications, such as embodied AI, VR/AR, digital games, and movie CG. Emerging neural radiance field, 3D-aware GANs [1, 21, 22, 30, 31] have achieved great success in generating high-quality multi-view consistent portraits with volumetric rendering. Editing capabilities for 3D-aware GANs have also been achieved through latent space auto-decoding, altering a 2D semantic segmentation [32, 33], or modifying the underlying geometry scaffold [34]. However, generation and editing quality tends to be unstable and less diversified due to the inherent limitation of GANs, and detailed-level editing is not well supported due to feature entanglement in the compact latent

space or global tri-plane representations. Recently, diffusion models [35, 36] have been proposed for high-quality content generation, achieving competitive performance compared to traditional GAN-based approaches [37]. Efforts have been made on 3D-aware portrait generation by de-noising the global tri-plane representation [23, 38], which, however, do not support animation and region-based editing. Therefore, it is important to devise an effective 3D avatar representation that supports high-quality 3D reconstruction, editing, and 3D generative model training.

1.1.4 3D Representation for General 3D Objects Generation

Though 2D diffusion models [35, 36] have beaten GAN on image synthesis [37] over quality [39], controllability [40] and scalability [41], a unified 3D diffusion pipeline is still unsettled. Generating 3D assets with diffusion models is challenging, and existing methods can be categorized into 2D-lifting methods and feed-forward 3D diffusion models. However, 2D-lifting methods require costly per-instance optimization, are prone to multi-face Janus problem [42], and fail to impose strict view consistency [43]. On the other hand, the feed-forward 3D diffusion models [17, 23, 44–47] enable speedy 3D synthesis without per-instance optimization. While this pipeline is straightforward, it poses extra challenges to achieve high-quality 3D diffusion: 1) Scalability, where existing methods are data inefficient, demanding out-numbered views per instance [44, 46] during training. This significantly hinders the scalability of large-scale 3D datasets. 2) Efficiency, where conductiong diffusion learning on the 3D representations [48–50] is computationally intensive and requires representation-specific design [51]. 3) Generalizability, where existing 3D diffusion models neglecthigh-quality conditional 3D generation (e.g., text-to-3D) across generic, category-free 3D datasets.

1.2 Approaches

This thesis focuses on expanding the generality of neural fields from the *generative model* perspective. The *generative model* can be applied in various ways, and can be categorized into two primary types:

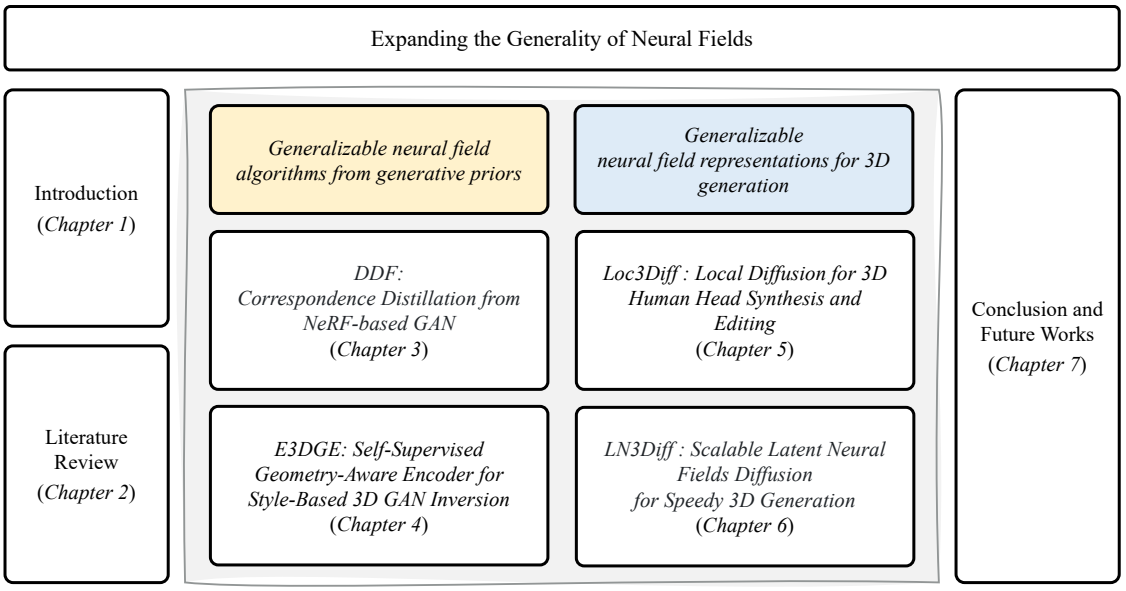


FIGURE 1.1: **Overview of the thesis** The methodology sections are primarily categorized into two major parts, which are 1) Generalizable neural field algorithms from generative priors, and 2) Generalizable neural field representations for 3D generation.

1) *generalizable neural field algorithms from generative priors* - Chapter 3 devotes the efforts to establishing 3D dense correspondences for NeRF with the help of 3D NeRF GANs, giving significant performance improvement in 3D texture transfer and segmentation propagation. Moving on to Chapter 4, beyond 3D editing within synthetic samples, this thesis further sets sight on the 3D reconstruction and editing with real samples. Through the delicately designed algorithms, the monocular images are inverted into semantically editable 3D assets while maintaining appealing efficiency.

2) *generalizable neural field representations for 3D generation* - Chapter 5 explores the development of a novel 3D representations that supports high-quality 3D avatar reconstruction, animation while still supporting 3D diffusion training. The findings highlight the essential role of hybrid 3D representations in achieving an effective solution for this task. Furthermore, this study could offer a canonical solution for further exploration in the 3D avatar field. Moving to Chapter 6, sights are further set on investigating the 3D generation framework for general objects. By incorporating a powerful feed-forward 3D VAE, we compress the multi-view images corresponding to the 3D object into a compact latent code, where efficient latent diffusion learning can be conducted. The resulting model surpasses existing methods in quality, speed, and fidelity, exhibiting great potential for future applications

of 3D AIGC (AI Content Creation).

Specifically, this thesis tackles the four challenges outlined in Sec. 1.1 and presents respective solutions in the following sections. The first two methods, DDF and E3DGE, focus on the preliminary extension of generalizable neural fields over toy datasets, especially FFHQ, and ShapeNet. The proposed method achieves few-shot 3D reconstruction and 3D editing. However, their performance is limited by the existing neural fields design, i.e., the MLP-based representation. To further expand the generality of neural fields, this thesis introduces two new approaches, Loc3Diff and LN3Diff. Specifically, task-specific designs are introduced to the 3D head and general 3D objects. The proposed novel neural field representations achieve better editability, visual quality, and generalizability than previous solutions.

1.2.1 Dense 3D Correspondence on NeRF

To establish the dense 3D correspondences on NeRF, a novel approach is presented that exploits NeRF-based generative adversarial networks (GANs) [21, 30, 52] to facilitate the learning of dense correspondence in NeRF. Specifically, NeRF-based GANs treat image synthesis as novel views rendered from its intermediate NeRF representation. The key idea is to employ the underlying 3D generator, G, to play a **triple role**: **1)** a holistic global structure descriptor for building conditional models that generalize to different object NeRFs of a category of interest. **2)** a robust semantic embedding function that maps corresponding coordinates across different NeRFs into semantically similar features, and **3)** a source of infinite object-specific NeRFs

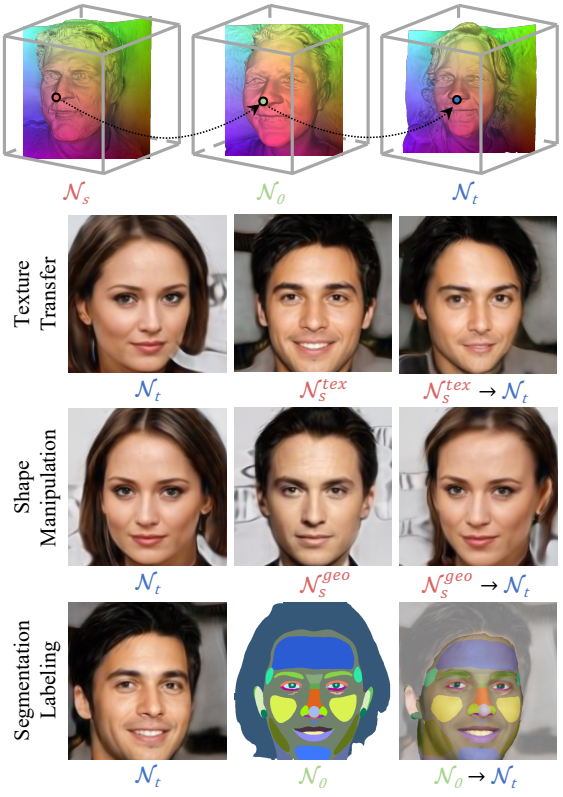


FIGURE 1.2: *DDF* supports high-quality texture transfer, shape manipulation and one-shot segmentation labeling given the established correspondence.

$\mathcal{N}_{i=1}^{\text{inf}}$ for training. The proposed approach is named **Dual Deformation Field** (*DDF*). Without any ground-truth correspondence annotations, dense 3D correspondence is established by mining rich semantic and structural priors from a pre-trained NeRF-based GAN. The performance of the proposed method is shown in Fig. 1.2.

1.2.2 Inversion and Editing on 3D GANs

Compared with 2D GAN inversion and editing, increasing the additional dimension makes inversion more challenging beyond the goal of reconstructing an editable shape with detail preservation. To address these challenges, a novel **E**ncoder-based **3D G**AN inv**E**rsion framework, E3DGE, is proposed, which addresses the aforementioned three challenges. The proposed framework has three novel components with a delicate model design. Specifically: **Learning Inversion with Self-supervised Learning** - We retrofit the generator of a 3D GAN model to provide us with diverse pseudo-training samples, which can then be used to train the inversion encoder in a self-supervised manner. **Local Features for High-Fidelity Inversion** - In addition to inferring an editable global latent code to represent the overall shape of the face, we further devise an hour-glass model to extract local features over the residuals details that the global latent code fails to capture. **Synthesizing View-consistent Edited Output** - We propose a 2D-3D hybrid alignment module for high-quality editing. Specifically, a 2D alignment module and a 3D projection scheme are introduced to align the local features with edited images and inpaint occluded local features in novel view synthesis. Moreover, we propose the following component for EG3D-based E3DGE: **Pose estimation for domain adaptation** - This newly proposed component addresses the problem of noisy poses of 3D GAN inversion over real images, which EG3D is especially sensitive to. With these components, the proposed method achieves 3D GAN inversion with plausible shapes and high-fidelity image reconstruction without affecting editability.

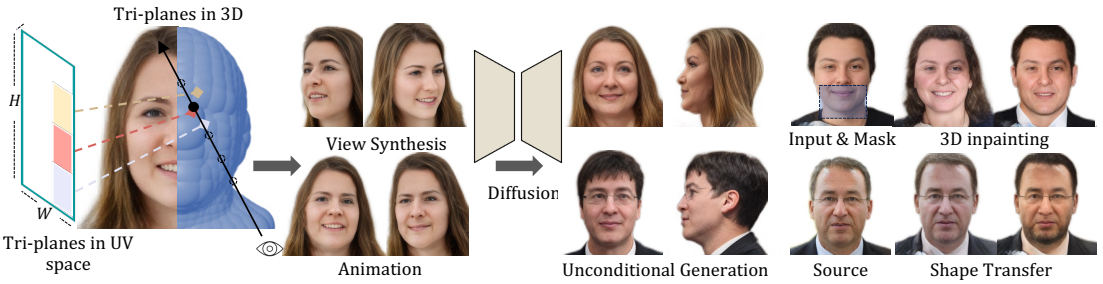


FIGURE 1.3: LOC3DIFF utilizes local tri-planes [1], defined in UV space as the underlying 3D representation for photo-realistic head modeling. It leverages 3D diffusion process to support unconditional generation, high-quality novel view synthesis, 3DMM-based animation, and region-based editing.

1.2.3 Avatar 3D Head Representations

With powerful generation ability, the diffusion model has shown promise in 2D content generation. However, extending these models to 3D avatars is challenging since 3D avatars require versatile capabilities of expression animation, editing, and high-quality reconstruction. Here, a diffusion-based generative model is proposed for the 3D volumetric head. Specifically, a novel representation of the 3D head is proposed, in which complex volumetric geometry and appearance are encoded by a large set of local tri-planes, anchored on the surface of an underlying 3D face parametric model (3DMM). By leveraging the UV space, the proposed framework ensures immediate compatibility with the well-established learning framework of 2D diffusion models [53]. To further create semantically consistent latent space for diffusion training, an auto-decoding framework is devised. The performance of the proposed method is visualized in Fig. 1.3.

1.2.4 3D Representation for Generic 3D Objects Generation

The existing 3D generation framework suffers from efficiency, scalability, and fidelity issues. Here, we propose a novel framework LN3DIFF to address these challenges and enable fast, high-quality, and generic conditional 3D generation. For direct latent space diffusion, the proposed method involves training a 3D variational autoencoder [54] (VAE) to compress input multi-view images into a lower-dimensional 3D-aware latent space, where a cascaded 3D-aware transformer-based decoder gradually decodes the latent into a high-capacity 3D neural field. After

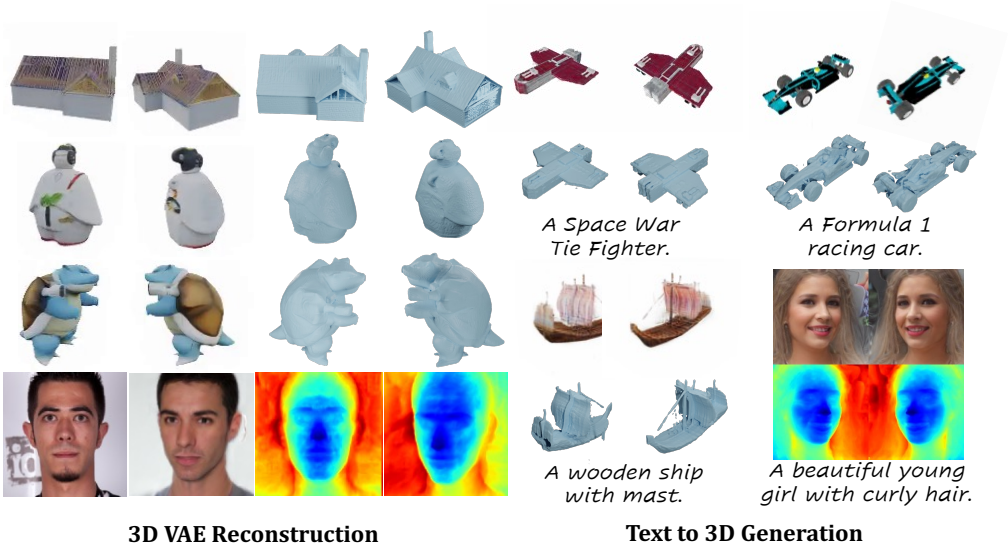


FIGURE 1.4: LN3DIFF performs efficient 3D diffusion learning over a compact latent space. The resulting model enables both high-quality monocular 3D reconstruction and text-to-3D synthesis.

training, we leverage the learned 3D latent space for text-conditioned 3D diffusion learning, ensuring effective utilization of the trained model for high-quality 3D generation. In essence, this work proposes a 3D-representation-agnostic pipeline for building generic high-quality 3D generative models, and provides opportunities to resolve various downstream 3D vision and graphics tasks. The overall performance of the proposed method is shown in Fig. 1.4.

1.3 Outline

This thesis is organized into seven chapters:

Chapter 2 reviews the background of this thesis, including 1) 3D shape correspondence, 2) GAN-based 3D generation and editing, and 3) 3D-aware Diffusion models. Additionally, it motivates the proposed methodologies in this thesis.

Chapter 3 presents the study on the *correspondence-level generalization* of neural fields. By leveraging the priors encoded in a pre-trained NeRF-based GAN, we establish 3D dense correspondence on NeRF that is accu- rate, smooth, and robust. We also show that established dense correspondence across NeRFs can effectively en- able many NeRF-based downstream applications such as texture transfer.

Chapter 4 presents the study on the *object-level generalization* of neural fields Specifically, we introduce an efficient self-training scheme for 3D GAN inversion that does not rely on real-world 2D-3D pairs but utilizes proxy samples generated from a 3D GAN. Furthermore, we introduce a novel pipeline for 3D view-consistent editing. The efficacy of the proposed method is validated on two representative 3D GANs, namely StyleSDF and EG3D. The proposed approach consistently outperforms state-of-the-art inversion methods, delivering superior quality in both shape and texture reconstruction.

Chapter 5 presents the attempt of *generalizable neural field representations* on 3D head. In particular, we present a novel framework for generating photorealistic 3D human heads and subsequently manipulating and reposing them with remarkable flexibility. The proposed method facilitates the creation of diverse and realistic 3D human heads with flexible global and fine-grained region-based editing over facial structures, appearance, and expressions

Chapter 6 delves into the findings on *generalizable neural field representations* for general 3D objects. By harnessing a 3D-aware architecture and variational autoencoder (VAE) to encode the input image into a structured, compact, and 3D latent space, the trained 3D diffusion model surpasses existing methods in terms of inference speed, requiring no per-instance optimization. The proposed method shows a significant advancement in 3D generative modeling and holds promise for various applications in 3D vision and graphics tasks.

Chapter 7 concludes the thesis and discusses potential directions for future work.

Chapter 2

Literature Review

This chapter provides an overview of the relevant literature for the thesis. First, Section 2.1 introduces the technical formulation of neural fields, establishing the foundation for the subsequent chapters. Section 2.2 focuses on neural fields specifically designed for 3D geometry. Section 2.3 delves into the task of 3D correspondence learning, a novel and underexplored area within 3D neural fields. Sections 2.4 and 2.4.2 discuss the techniques related to 3D-aware GANs and existing methods for 2D GAN inversion, respectively. Section 2.5 broadens the scope to cover generalized 3D reconstruction and view synthesis, a highly relevant area for this thesis. Finally, Section 2.6 introduces the technical formulation of diffusion models, a powerful generative modeling framework that can be extended to 3D.

2.1 Neural Fields

As formulated by Xie *et al.* [55], following the universal approximation theorem, any field can be parameterized by a neural network. Therefore, we have:

Definition 2.1. A *neural field* is a field that is parameterized fully or in part by a neural network.

Neural fields offer a unique combination of continuity and adaptability. Unlike discrete parameterizations, whose memory requirements grow with increased spatio-temporal resolution, neural fields' memory needs are tied to the neural network's

parameter count—its complexity. While other continuous parameterizations (e.g., Fourier series) can represent extensive domains, predetermining the optimal complexity for efficient representation is often challenging. Neural fields address this by allocating their parameters dynamically, focusing on areas of field detail. Typically, neural fields are implemented as MLPs with differentiable activation functions. Their analytic differentiability, compatibility with gradient descent, and over-parameterization [56] make them adept at regressing complex signals in otherwise ill-posed optimization problems. The definition of neural fields encompasses neural networks that output spatial data, such as voxel grids. These architectures often employ convolutional layers to produce 2D or 3D grids of various attributes (e.g., RGB color, occupancy, latent features [57], or tri-planes [1]).

2.2 Neural Fields for 3D Geometry

The success of deep learning over 2D domain has spurred a growing interest in the 3D domain. However, traditional explicit representation such as mesh and voxel are hard to fit in deep learning optimization framework due to the varying topology or limited resolution. As a parallel class of shape representation, recent advances in implicit functions [13, 15, 58] have demonstrated their excellence when representing complicated geometry. By representing shapes as a continuous field, implicit representation encodes the geometry properties of a 3D point x using a neural network $f(x)$. Conventional implicit representations were limited by the need of 3D ground-truth. NeRF [2] stands out as a successful variant to support direct learning of 3D scene from multi-view images. Pumarola et al. [59], Park et al. [60] further improve NeRF to model non-rigid and time-varying scenes by equipping static NeRF MLP with an extra deformation field. Mathematically, NeRF [2] proposed an implicit 3D representation for novel view synthesis. Specifically, NeRF defines a scene as $\{\boldsymbol{c}, \sigma\} = F_\Phi(\vec{x}, \boldsymbol{v})$, where $\vec{x}$ is the query point, $\boldsymbol{v}$ is the viewing direction from camera origin to $\vec{x}$, $\boldsymbol{c}$ is the emitted radiance (RGB value), σ is the volume density. To query the RGB value $C(\boldsymbol{r})$ of a point on a ray $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{v}$ shoot from the 3D coordinate origin $\boldsymbol{o}$, we have the volume rendering formulation,

$$C(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\boldsymbol{c}(\boldsymbol{r}(t), \boldsymbol{v})dt, \tag{2.1}$$

where $T(t) = \exp(-\int_{t_n}^{t} \sigma(\boldsymbol{r}(s))ds)$ is the accumulated transmittance along the ray $\boldsymbol{r}$ from t_n to t. t_n and t_f denote the near and far bounds.

Zheng et al. [61], Gafni et al. [62] further augment NeRF MLP with a template shape using 3D basic models, including 3DMM [62, 63], FLAME [61] and SMPL [64] to enable more explicit control. However, they are still limited to overfitting setting and the learned models fail to generalize to novel scenes. Please note that implicit shape representation and neural rendering are still developing rapidly and we refer readers to the survey [12] for more details.

Though great advances have been achieved, building dense correspondence across shapes represented by implicit functions are intrinsically challenging since ground truth correspondence are impossible to acquire. Recent attempts to build correspondence over implicit representations [65, 66] tried to bypass this requirement by defining F as signed distance function (SDF) values of the deformed points and d as the marginal L_1 loss as in [58]. Liu and Liu [67] followed similar principles as functional maps and adopted occupancy loss as supervision, while the basis functions are learned from data. Though dense correspondence over implicit functions could be derived, these methods are unable to establish consistent bijective correspondence and still require 3D supervision during training. Moreover, these methods are all constrained on synthetic dataset [68], which limit the applications on real scenes.

2.3 3D Correspondence Learning

2.3.1 2D Semantics Correspondence

Extracting semantic correspondences between images has long been a central challenge in computer vision. Traditional approaches, such as Scale Invariant Feature Transform (SIFT) [69], rely on extracting local features for matching across images. With the advent of convolutional neural networks (CNNs), deep learning-based feature descriptors have been increasingly used to compute correspondences [70–73]. However, these methods often depend on pseudo correspondence ground truth generated through data augmentation. CoordGAN [74] proposed to leverage GAN features for self-supervised correspondence learning without ground truth.

However, the aforementioned methods deal with correspondence matching across images and could not be easily extended to 3D representations. For example, some of the 2D matching methods [73] requires ground truth mask for supervision, which could not be extended to 3D implicit representations like NeRF. In this work, we focus on dense correspondence matching over the NeRF-based 3D representations without ground truth labels for supervisions.

2.3.2 3D Shape Correspondence

The problem of establishing dense correspondences between 3D shapes is of key importance to a series of downstream tasks [75, 76], and has been studied extensively in recent survey [77, 78]. Traditional approaches build correspondence between shapes represented by mesh or point clouds. They can be roughly divided into registration-based and similarity-based methods, where the former adopts Laplacian coordinate δ_i for vertex v_i as geometric preservation descriptor after registration. Similarity-based solutions do not change the geometry of given shapes and calculate the similarity between vertices with learnable feature descriptors. With recent advances in geometric machine learning [79, 80], researchers extend traditional framework by replacing hand-crafted descriptor with learnable feature descriptors [26, 81, 82]. Halimi et al. [83], Eisenberger et al. [84] further mitigate the requirements of correspondences annotations which builds soft correspondence matrix $\prod \in \mathbb{R}^{m \times n}$ between numerable vertices on the mesh surface. The lack of explicit surface and numerable vertices in NeRF hinders the use of above methods, where correspondence affinity matrix [84] could not be built. Pioneer works [65–67] propose to build correspondences over implicit representations. However, they still rely on ground-truth reconstruction annotations to train the deformation field. Collecting such annotations for NeRF-based representations is infeasible, where there are infinite points with non-zero densities.

2.4 GAN-based Generation and Editing

Generative Adversarial Network [85] has shown promising results in generating photorealistic images [86–88] and inspired researchers to put efforts on 3D aware

generation [89–91].

2.4.1 3D-aware GANs

Motivated by the recent success of neural rendering [2, 13, 14], researchers introduce 3D inductive bias into the generation task [21, 30] and show impressive 3D-awareness synthesis through a hybrid design [1, 8, 22, 52, 92], making it applicable to a series of downstream applications [16, 32, 33, 93]. However, GAN-based methods suffer from mode collapse [94] and fail to model datasets with larger scale and diversity [37]. Besides, 3D reconstruction and editing of GANs require elaborately designed inversion algorithm [7].

2.4.2 2D GAN Inversion

Optimization-based 2D GAN inversion methods [95, 96] achieve photorealistic reconstruction at the cost of slow inference and lack of editability. To speed up, Encoder-based methods [29, 97–100] like pSp [97] and e4e [98] have been developed and show better properties in editing through specific model design [29, 97] and training strategies [98]. However, they [95, 97, 98, 100, 101] all adopt global latent code alone for GAN inversion task, thus failing to recover high-fidelity details. Recently, HFGI [29] introduce an extra spatial consultation map to mitigate this issue, though still designed to restore 2D textures without considering 3D shape modeling. In this work, we propose a delicate design that exploits local features to recover texture details and achieves view-consistent synthesis.

2.4.3 3D Reconstruction and Editing with 3D GANs

Recent development of 3D GANs [1, 8, 21, 22, 30, 52] also calls for corresponding inversion frameworks. π-GAN and EG3D [1] directly adopt 2D inversion method [28, 95], which requires expensive latent or model optimization and still introduces implausible shape artifacts. The most relevant work to ours is Lin *et al.* [102], which employs a computationally expensive optimization-based framework [95] and combines FLAME [103, 104] for portrait animation. However, it fails

to guarantee reasonable shape and is limited to the human face domain. In parallel, Pix2NeRF [105] introduces a feed-forward network to pre-trained π-GAN and enables single-view 3D reconstruction. However, it does not demonstrate its performance in high-quality novel view editing. Pavllo *et al.* [106] proposes a hybrid 3D GAN inversion framework for single-image shape reconstruction. However, the hybrid framework trades off the performance with speed, while our method achieves high-quality inversion and editing performance with fast inference speed. Some other works rely on 3D parametric models [103] or auto-decoder [107] architecture for single-view 3D reconstruction [103] or editing [107], which cannot leverage the strong GAN priors for high-resolution and flexible latent-based editing. A concurrent work, GOAE [108], also proposed a two-stage 3D GAN inversion framework. However, it only demonstrates its ability on tri-plane and image inversion, while our method is verified on both SDF and tri-plane 3D representation over both image and video.

Besides, previous works [109–115] propose to use pretrained GAN to generate training dataset. Through careful design in the sampling strategy [112], loss functions [110] and generation process [114], researches show that off-the-shelf image generators could facilitate a series of downstream visual applications.

3D GANs for Avatar Synthesis. 3D GANs are well developed in the 3D avatar synthesis area, staring from pi-GAN [21]. To further increase the generation resolution, recent works [1, 8, 22, 31, 52, 116–118] resorted a hybrid design to high resolution up to 512. However, samples from these methods cannot easily be edited. On the other hand, FENeRF [32] and IDE-3D [33] proposed to generate, edit, and animate human faces, guided by a segmentation map. However, their support for local editing is still unsatisfactory, as the local geometry cannot be explicitly edited due to the lack of spatial information in the segmentation map. Additionally. Moreover, segmentation-driven animation has several limitations, e.g., can only animate an identity with a similar foreface layout. By contrast, LN3Diff achieves improved performance and flexibility via direct basic-model-driven animation.

Another line of work [7, 16, 109–114] propose to use a pre-trained GAN to generate training data. Through careful design in the sampling strategy [112], loss functions [110] and generation process [114], off-the-shelf generators can facilitate a series of downstream applications. In this work, we also adopt a pre-trained 3D GAN as an "infinite" source of 3D assets.

2.5 Generalized 3D Reconstruction and View Synthesis

To circumvent the per-scene optimization of NeRF, researchers propose to learn a prior model [3, 19, 119] through image-based rendering. However, these methods are designed for view synthesis and lack explicit 3D representations. LoLNeRF [107] learns a prior through auto decoding but is limited to the simple category-specific settings. Moreover, these methods are designed for view synthesis and fail to generate new 3D objects. VQ3D [120] adapts the generalized reconstruction pipeline to 3D generative models. However, they adopt a 2D architecture and work with autoregressive modeling over 1D latent space, which ignores much of the inherent 3D structure. NeRF-VAE [121] directly models 3D likelihood with VAE posterior, but is limited to simple 3D scenes by the deficient capacity of VAE. Concurrently, LRM [122] has proposed a feedforward model for generalized monocular reconstruction. However, its latent space is not specifically designed for learning a generative model, which limits its effectiveness for 3D diffusion learning.

2.6 Diffusion Model

As a new branch of generative models, diffusion models [35, 36, 123] are leading to a series of breakthroughs in various tasks such as image and video generation [37, 39, 53].

The surprising generative power of diffusion models stems from the robust theoretical foundation where the complicated distribution fitting process is transformed into several independent and stable reverse processes via Markov Chain modeling. Compared with other generative models [54, 85], diffusion models are stable during training compared with GAN [85], capable of capturing more accurate distribution than VAE [54] and more feasible without the needs of invertible network architectures compared with Normalizing Flows [124]. In the following sections, we provide a brief introduction to the formulation and approaches related to diffusion models. For detailed theorems and proofs, we refer readers to the original papers [35, 36, 123].

2.6.1 Mathematical Formulation

Diffusion models can be broadly categorized into three types based on their underlying formulations, i.e., denoising diffusion probabilistic models (DDPMs) [36, 123], score-based generative models (SGMs) [125, 126], and stochastic differential equations (Score SDEs) [127, 128].

At the core of DDPMs are two Markov chains: a forward chain that gradually perturbs the data into noise, and a reverse chain that reconstructs the data from noise. The forward process is typically handcrafted, adding noise incrementally to the data, while the reverse process is learned by training a deep neural network to invert the forward chain. In theory, this network can be seen as a series of equally-weighted denoising networks $\epsilon_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ for $t = 1 \dots T$, where T is the length of the Markov chain, $\boldsymbol{x} \sim p(\boldsymbol{x})$ is a sample from the target distribution, and $\boldsymbol{x}_t$ is the noisy version of $\boldsymbol{x}$ at step t. The diffusion training objective can be expressed as follows:

$$L_{DM} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{\epsilon}\sim\mathcal{N}(0,1),t}\left[||\boldsymbol{\epsilon} - \epsilon_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)||_2^2\right], \tag{2.2}$$

where t is uniformly sampled from $1, \dots, T$. Once the model is trained, new data can be generated by sampling a random point from the prior distribution used in the forward Markov chain and performing ancestral sampling through the reverse chain [129].

To further improve the effectiveness and efficiency of applying DDPMs to high-resolution generation, Rombach *et al.* [39] further propose latent DDPMs to conduct the training process of DDPMs in latent space. Specifically, the high-dimensional data $\boldsymbol{x}$ is first compressed into a low-dimensional feature $\boldsymbol{z} = \xi(\boldsymbol{x})$ in latent space using a pretrained perceptual compression autoencoder ξ. Then the training objective can be expressed as:

$$L_{LDM} = \mathbb{E}_{\xi(\boldsymbol{x}),\boldsymbol{\epsilon}\sim\mathcal{N}(0,1),t}\left[||\boldsymbol{\epsilon} - \epsilon_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t)||_2^2\right]. \tag{2.3}$$

2.6.2 3D-aware Diffusion Models

The unprecedented success of the 2D diffusion models has inspired researchers to apply this technique in the context of 3D with several strategies. DreamFusion [42, 130, 131] inspired 3D generation by distilling 2D diffusion model, but

suffers from expensive per-scene optimization, mode collapse, and Janus problem. Some approaches propose to learn the 3D prior and rendering process in a 2D manner [43, 132, 133]. Though photo-realistic, these methods intrinsically lack view consistency and cannot extract the underlying 3D of the synthesized scene. A more canonical 3D diffusion pipeline follows a two-stage training paradigm: after pre-training an auto-decoder as the first stage with multi-view images [17, 23, 38, 44, 45, 134], a group of 3D latent codes will serve as the training corpus for diffusion training in the second stage. However, the auto-decoding stage requires training a tiny shared decoder, which leads to an unclean latent space with limited scalability. Besides, the latent is usually heavy, e.g., $256 \times 256 \times 96$ [23], which hinders efficient diffusion learning [135].

Prior works RenderDiffusion [136, 137] and concurrent DMV3D [122, 138] propose a latent-free 3D diffusion by integrating rendering process in the diffusion sampling. Though straightforward, this design involves time-consuming volume rendering in each denoising step, which significantly slows down the sampling speed. SSDNeRF [46] proposes to jointly learn 3D reconstruction and diffusion with an auto-decoder pipeline. However, it requires a sophisticated training schedule and only demonstrates the performance over a single-category unconditional generation. Comparatively, the proposed LN3Diff trains 3D diffusion on the compressed latent space where no rendering operations are required. We validate in Sec. 6.2 that our method achieves better performance on both 3D generation and monocular 3D reconstruction with 3 times faster speed. Besides, we demonstrate conditional 3D generation over generic, diverse 3D datasets, while RenderDiffusion and SSDNeRF focus on unconditional generation over simple classes.

Chapter 3

Correspondence Distillation from NeRF-based GAN

The success of neural radiance fields (NeRF) [2] has led to remarkable progress in learning 3D representations. Unlike voxel-, point cloud- and mesh-based methods, NeRF represents each 3D object as a distribution of coordinate-based volume densities and view-dependent colors. And by approximating this distribution with a continuous parametric function, NeRF shows great potential in capturing geometric scene details and rendering realistic novel views. current limitations

As a crux in computer vision and graphics, dense 3D correspondence has been well explored over explicit shape representations. As an emerging promising shape representation, it is thus natural to extend such effort to NeRF and establish dense correspondence across two objects represented by NeRF, enabling downstream applications such as segmentation transfer, texture transfer and texture manipulation as shown in Fig. 3.1.

This task is non-trivial. First, existing methods for building dense correspondence across two objects mainly focus on mesh-based representations. It is infeasible to directly apply and adapt them to NeRF. Unlike meshes that have explicit vertices and surfaces, NeRF lacks an explicit surface, preventing us from resorting derivatives of neural fields [25] as the shape surface descriptors. Moreover, existing

The work in this chapter has been published in [16].
Code is publicly available at `https://www.mmlab-ntu.com/project/ddf/index.html`

methods [26] often require ground-truth correspondence annotations in training, which are hard to obtain for NeRF-based object representations.

To overcome the aforementioned limitations, a novel approach is presented that exploits NeRF-based generative adversarial networks (GANs) [21, 30, 52] to facilitate the learning of dense correspondence in NeRF. Specifically, NeRF-based GANs treat image synthesis as novel views rendering from its intermediate NeRF representation.

Our key idea is to employ its generator, G, to play a **triple role**, as shown in Fig. 3.1:

1) Since the generator of a GAN is a latent variable model that learns a mapping $z \rightarrow G(z)$, the associated latent code z shall capture the underlying structure of the generated object NeRF $G(z)$ in a pretrained GAN. Therefore, this latent code naturally serves as a holistic global structure descriptor for building conditional models that generalize to different object NeRFs of a category of interest.

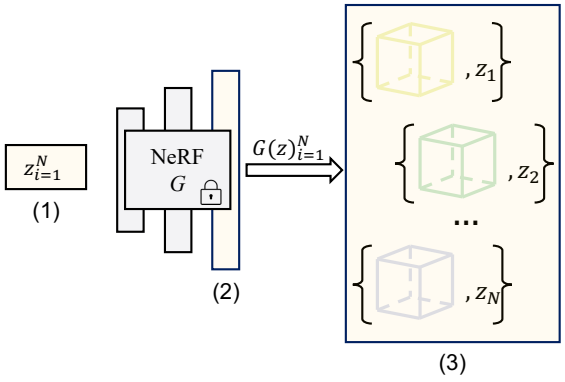


FIGURE 3.1: **The triple role of a NeRF-based GAN**: a pretrained NeRF-based GAN is retrofitted into triple roles: (1) the latent codes $z_{i=1}^{N}$ serve as holistic structure descriptors; (2) the extracted generator features serve as geometry-aware local descriptors; and (3) the sampling space of pretrained G could serve as an infinite object-specific dataset.

2) As a representation learning architecture, G can serve as a robust semantic embedding function that maps corresponding coordinates across different NeRFs into semantically similar features. Based on such cross-instance feature similarity, the features by G can naturally be used as geometric-aware local descriptors.

3) Following the idea of Sim2Real [139], G can also serve as a source of infinite object-specific NeRFs $\mathcal{N}_{i=1}^{\text{inf}}$ for training. Moreover, thanks to the well-learned latent space of GANs, it is flexible to adjust the complexity of sampled NeRFs through the latent codes via the truncation trick [86, 140].

The proposed approach is named as **Dual Deformation Field** (*DDF*). While our *DDF* does not limit the choice of NeRF-based GAN, π-GAN [21] is adopted due to its simplicity and promising synthesis results. Our adaptation of the pre-trained π-GAN starts with considering its first role in model construction, where the latent codes of π-GAN is treated as additional conditions. Specifically, the dense correspondence between NeRFs is regarded as a coordinate-based deformation field from the source NeRF to the target NeRF. Instead of learning a single deformation field conditioned on a pair of source and target latent codes, a fixed template NeRF is used as the bridge and learn two separate deformation fields, namely a backward deformation field B and a forward deformation field F. In our formulation, B always treats the template NeRF as the target, taking only the source latent code as input. Similarly, F always treats the template NeRF as the source, taking the target latent code as the condition. Such a decomposition substantially alleviates the learning complexity. In addition, the dense correspondence between any two NeRFs can be easily established by combining F and B.

Benefit from the second role of π-GAN, *DDF* can learn without ground-truth correspondence annotations. Specifically, for any coordinate in the source NeRF, its corresponding coordinate can be obtained in the target NeRF from *DDF*. Since the features of π-GAN are geometric-aware descriptors, the generator features can be computed for these estimated corresponding coordinates and apply feature-wise cosine similarity as the primary learning objective.

Finally, as π-GAN provides infinite object-specific NeRF samples for training, the complexity of sampled NeRFs can be further controled by mixing the latent codes of sampled NeRFs with that of the template NeRF. Training begins with samples of low deformation complexity and gradually move to samples with higher complexity as training proceeds. This strategy is found to improve the training-time efficiency and stability.

This is an early attempt that establishes the dense correspondence between two NeRF-based object representations. Without any ground-truth correspondence annotations, dense correspondence is established by mining rich semantic and structural priors from a pre-trained NeRF-based GAN. In the challenging category of human faces, the proposed method produces high-quality dense correspondences with promising robustness and generality. Various tasks such as texture transfer and segmentation transfer are tested to demonstrate the potential of our method

in downstream tasks. In general, the proposed method improves the generality of neural fields by establishing the 3D correspondence within intra-class objects. This facilitates downstream 3D editing applications such as texture and segmentation transfer.

3.1 Methodology

In this chapter, we present a new attempt for building dense correspondence between NeRF representations across objects belonging to the same category. Obtaining ground-truth correspondence annotations is infeasible due to the implicit nature of NeRF. Our key insight is to retrofit a generator of a pre-trained NeRF-based π-GAN, denoted as G, into triple roles: 1) the latent codes in G serve as holistic global structure indicators that improve the generality of models; 2) the features of G serve as geometric-aware local descriptors that enable a feature-based learning objective; 3) and the manifold of G serves as a source of infinite training and evaluation samples over a single category.

In the following sections, we first introduce the details of NeRF-based π-GAN in Sec. 3.1.1 as the background knowledge for subsequent sections. Next, we explain the problem formulation and our framework in Sec. 3.1.2, learning objective in Sec. 3.1.3, and training strategy in Sec. 3.1.4.

3.1.1 Background on NeRF-based GANs

Inspired by the success of NeRF as an efficient 3D representation, NeRF-based GANs employ NeRF as their internal representation for 3D-aware image synthesis. We adopt π-GAN [21] in this chapter. Specifically, the generator of the π-GAN contains a mapping network $\mathcal{M}$ and a multi-layer perceptron (MLP) network. Starting from a latent code $\vec{z} \sim p_Z$ that follows the Gaussian prior distribution, the mapping network first maps $\vec{z}$ to a set of modulation signals $\mathcal{M}(z) = \{\boldsymbol{\beta}, \boldsymbol{\gamma}\}$, where $\boldsymbol{\beta} = \{\beta_i\}, \boldsymbol{\gamma} = \{\gamma_i\}$. In π-GAN, a NeRF is obtained by the MLP network, which estimates the view-dependent density $\sigma \in \mathbb{R}^+$ and the color vector $\boldsymbol{c} \in \mathbb{R}^3$ for each 3D point, taking its coordinate $\vec{x} \in \mathbb{R}^3$ and a viewing direction $\boldsymbol{d} \in \mathbb{S}^2$ as input. To associate a latent code to its corresponding NeRF, the modulation

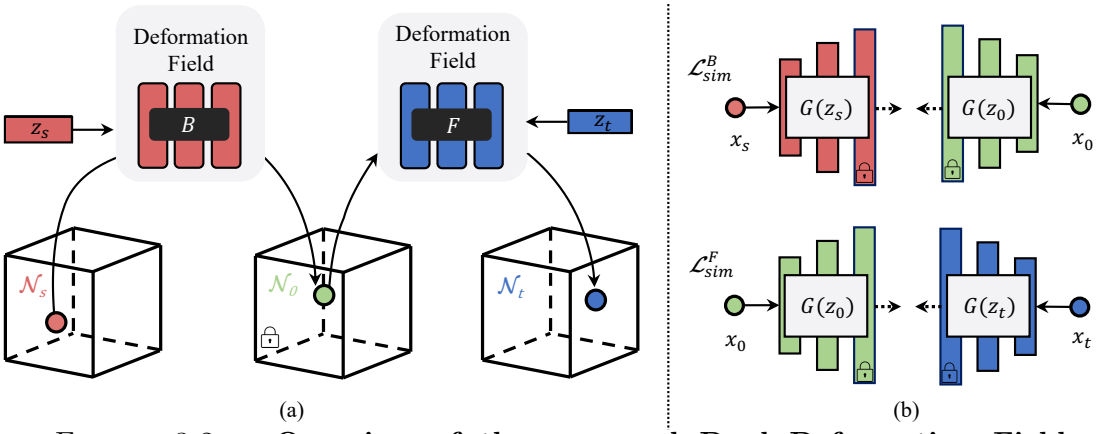


FIGURE 3.2: **Overview of the proposed Dual Deformation Field (*DDF*).** (a) *DDF* consists of two coordinate-based deformation fields, namely the backward B and the forward F. To get the correspondence point given a point $\vec{x}_s$ sampled from the source NeRF $\mathcal{N}_s$, the B model conditions on the $\vec{z}_s$ and learns to deform the input point $\vec{x}_s$ to the correspondence point in the template NeRF $\mathcal{N}_0$. Similarly, the F model conditions on the target latent code $\vec{z}_t$ and learns to deform points from the template NeRF $\vec{x}_0$ to the target NeRF $\mathcal{N}_t$. (b): Feature similarity losses $\mathcal{L}^B_{sim}$ and $\mathcal{L}^F_{sim}$ between features extracted from the generator of the pre-trained π-GAN, G, is adopted as the main loss. Please refer to Fig. 3.4 for the details of the other two supervisions imposed in the training.

signals will be injected into the MLP network, serving as FiLM conditions [141–143] to modulate its features at different layers as $\boldsymbol{f}_{i+1} = \sin(\gamma_i \cdot (\boldsymbol{W}_i \boldsymbol{f}_i + \boldsymbol{b}_i) + \beta_i)$.

Image synthesis in π-GAN is achieved by sampling a latent code and subsequently rendering an image from the corresponding NeRF. Following the volume rendering of NeRF [2], each pixel color C of the image is obtained via sampling a set of points along the ray $r(t) = \boldsymbol{o} + t\boldsymbol{d}$ and accumulating their color vectors weighted by their transmittance:

$$\hat{C}(\boldsymbol{r}) = \sum_{i=1}^{N} T(t_i)(1 - \exp(-\sigma_i \delta_i))\boldsymbol{c}_i, \tag{3.1}$$

where $T(t) = \exp\left(-\sum_{j-1}^{i=1} \sigma_j \delta_j\right)$ and $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples. Using a set of unposed 2D images, π-GAN is trained progressively with the non-saturating GAN loss and the R1 regularization [144].

3.1.2 The Proposed Framework

Problem Formulation. Given any pair of NeRFs $\mathcal{N}_s : \mathbb{R}^3 \mapsto \mathbb{R}^4$ and $\mathcal{N}_t : \mathbb{R}^3 \mapsto \mathbb{R}^4$, our goal is to estimate a 3D deformation residual $H_D : \mathbb{R}^3 \mapsto \mathbb{R}^3$ that deforms NeRF $\mathcal{N}_s$ towards NeRF $\mathcal{N}_t$ via:

$$\mathcal{N}_s \rightarrow \mathcal{N}_t : \vec{x}_t = (\vec{x}_s + H_D(\vec{x}_s)), \forall \vec{x}_s \in \mathcal{N}_s. \tag{3.2}$$

The deformation field H_D represents the residual 3D deformation $D(\vec{x}_s) = \Delta\vec{x}_s$ in the 3D space of the source NeRF $\mathcal{N}_s$. It is an injective mapping that maps each 3D point $\vec{x}_s$ in the source NeRF, $\mathcal{N}_s$, to its corresponding position in the target NeRF, $\mathcal{N}_t$.

Challenges. The problem formulation shown above follows existing attempts [65, 66] that model the dense 3D correspondences between an SDF shape and a shared template via a single deformation field. However, this design does not suit NeRF for the following reasons. First, their parameterization is designed to facilitate shape reconstruction, rather than establishing correspondences between two existing shapes. Second, deforming all the points on a shape to a shared template could only guarantee an injective mapping instead of a bijective mapping, where a random point over the template could not find its correspondence on a target shape. Third, this design limits information (e.g., textures) propagation between NeRFs. Given a ray that intersects with a shape, unlike SDF representation where the shape surface is modeled by a single point on the zero-level iso-surface, the volume-based representation (e.g., NeRF) represents the shape boundary by innumerable points [145]. Therefore, after the source NeRF, $\mathcal{N}_s$, deforms densely sampled near-surface points with texture information to the template, it is computationally intractable for the target NeRF, $\mathcal{N}_t$, to find the precise corresponding texture for points along a ray.

Dual Deformation Field. We propose to fix the above-mentioned issues by lifting the injective mapping to a bijective mapping function. A straightforward solution here is to leverage a single conditional mapping function $D : \mathbb{R}^3 \times \mathbb{R}^{\vec{z}_t} \times \mathbb{R}^{\vec{z}_s} \mapsto \mathbb{R}^3$, which estimates the offset for each point $\vec{x}$ of the source NeRF $\mathcal{N}_s$, taking its coordinate and the latent codes $\vec{z}_t$ and $\vec{z}_s$ of target and source NeRFs as input. However, since the source and target NeRFs vary in each iteration, such a solution requires a large model capacity and fails to converge in practice. A

similar observation has also been proposed in previous work that models dynamic NeRF [59].

To alleviate the computational complexity without sacrificing the bijective property, as illustrated in Fig. 3.2, we sample a fixed NeRF with a latent code $\vec{z}_0$ from G as the intermediate template $\mathcal{N}_0$, and reformulate the deformation field D as the composition of two separate conditional neural deformation fields, namely, a backward deformation field B that estimates the deformation from a source NeRF, $\mathcal{N}_s$, to the template $\mathcal{N}_0$, and a forward deformation field F that estimates the deformation from the template $\mathcal{N}_0$ to a target NeRF, $\mathcal{N}_t$.

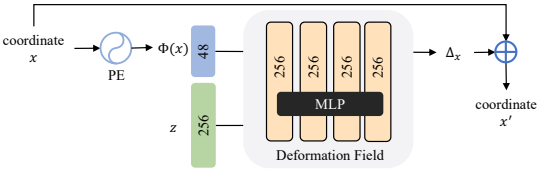


FIGURE 3.3: **A diagram of the deformation field model architecture.** Both the forward deformation field F and the backward deformation field B are implemented as MLPs consisting of four fully-connected layers with residual connections [2, 3]. Both F and B take a latent code $\vec{z}$ of 256 dimensions and a coordinate as input, where the latter is embedded into a 48-dimensional vector via positional encoding [2, 4]

By decomposing the deformation field between two arbitrary NeRFs into two fields B and F bridged by a fixed template NeRF, the overall learning complexity is significantly reduced. In this way we have

$$\vec{x}_0 = B(\vec{x}_s, \vec{z}_s), \qquad B(\vec{x}_s, \vec{z}_s) \coloneqq \vec{x}_s + H_B(\phi(\vec{x}_s), \vec{z}_s), \tag{3.3}$$

$$\vec{x}_t = F(\vec{x}_0, \vec{z}_t), \qquad F(\vec{x}_0, \vec{z}_t) \coloneqq \vec{x}_0 + H_F(\phi(\vec{x}_0), \vec{z}_t), \tag{3.4}$$

where $\vec{x}_s \in \mathcal{N}_s$, $\vec{x}_t \in \mathcal{N}_t$, and $\vec{x}_0 \in \mathcal{N}_0$. And $\phi(\vec{x})$ is the positional encoding [2] of a given point. H_B and H_F are residual functions each implemented as an MLP consisting of four fully-connected layers, as depicted in Fig. 3.3. The correspondence point of $\vec{x}_s \in \mathcal{N}_s$ in a target NeRF $\mathcal{N}_t$ can be retrieved by the composite mapping $F(B(\vec{x}_s, \vec{z}_s), \vec{z}_t)$, as depicted in Fig. 3.2. The latent codes $\vec{z}_s$ and $\vec{z}_t$ serve as the holistic global structure indicators to guide the deformation. Implementation wise, the template NeRF $\mathcal{N}_0$ is chosen as $(\overline{\boldsymbol{\gamma}}, \overline{\boldsymbol{\beta}})$ which can be intuitively seen as the average shape of the trained dataset.

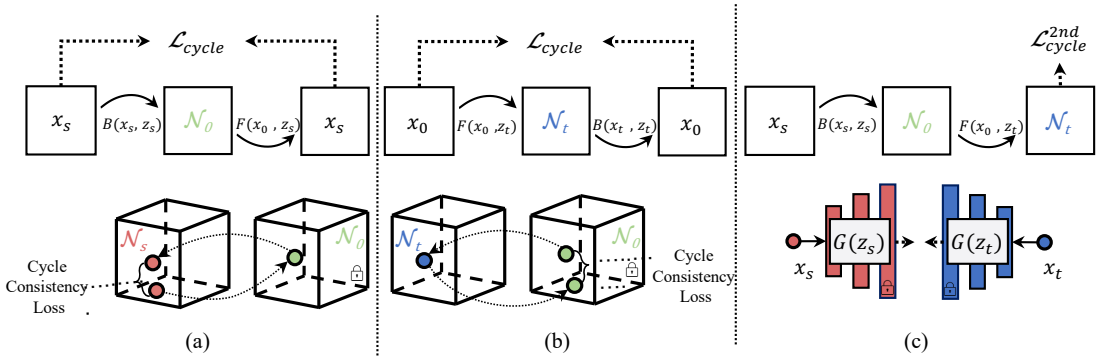


FIGURE 3.4: **Illustration of loss functions used in *DDF*.** (a) Backward cycle-consistency loss: $F(B(\vec{x}_s, \vec{z}_s), \vec{z}_s) \approx \vec{x}_s$, (b) forward cycle-consistency loss: $B(F(\vec{x}_t, \vec{z}_t), \vec{z}_t) \approx \vec{x}_t$ and (c) second-order feature similarity loss: $G(\vec{x}_s, \vec{z}_s) \approx G(F(B(\vec{x}_s, \vec{z}_s), \vec{z}_t), \vec{z}_t)$.

3.1.3 Training Objective

Our overall training objective contains a feature similarity loss for estimated correspondences and three additional regularizations for the deformation fields F and B, namely a cycle-consistency regularization, a second-order feature similarity loss, and a deformation smoothness regularization.

Generator Feature Similarity Loss. Given a collection of n source NeRFs $\{\mathcal{N}_s^{(i)}\}_{i=1}^n$ that are sampled from G with corresponding latent codes $\{\vec{z}_s^{(i)}\}_{i=1}^n$, each of these NeRFs will serve as a source NeRF for B to compute its deformation to the template. For each pair of estimated corresponding points $(\vec{x}_s, \vec{x}_0)$ where $\vec{x}_s$ belongs to one of these source NeRFs and $\vec{x}_0$ belongs to the template, we take a point feature extracted from NeRF generator G as the local geometric descriptor. When the $\vec{x}_s$ and $\vec{x}_0$ are homologous and share similar semantic meanings, the feature similarity loss should be small, and a smaller feature similarity loss in training indicates that the deformation field produces reasonable correspondences. Therefore, for each pair of sampled points, we compare the cosine similarity between two descriptors as their correspondence relevance score and update the network accordingly. Consequently, the feature similarity loss for B can be written as:

$$\begin{aligned}\mathcal{L}_{sim}^{B} = &\frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{|\mathcal{P}_s^{(i)}|}\sum_{\vec{x}_s\in\mathcal{P}_s^{(i)}}\right. \\ &\left. w_{\vec{x}_s} * \frac{1}{2}\left\|(G(\vec{x}_s, \vec{z}_s^{(i)}) - G(B(\vec{x}_s, \vec{z}_s^{(i)}), \vec{z}_0))\right\|_2^2\right],\end{aligned} \tag{3.5}$$

where the loss of each point $\vec{x}_s^{(i)}$ is weighted by $w_{\vec{x}_s} = T(t_{\vec{x}_s})$ defined in Eq. 3.1, so that B is encouraged to focus more on points with large densities, as they are close to the object surface with rich semantics. It is worth noting that to reduce computational redundancy and complexity, we will sample only a subset $\mathcal{P}_s^{(i)}$ of points from each NeRF $\mathcal{N}_s^{(i)}$ by the sampling strategy introduced in the next section. Each of these NeRFs will also serve as a target NeRF for F to compute the deformation of the template to it. The feature similarity loss for F is thus:

$$\begin{aligned}\mathcal{L}_{sim}^{F} =& \frac{1}{m}\sum_{j=1}^{m}\left[\frac{1}{|\mathcal{P}_0^{(j)}|}\sum_{\vec{x}_0\in\mathcal{P}_0^{(j)}}\right. \\ & \left. w_{\vec{x}_0} * \frac{1}{2}\left\|G(\vec{x}_0,\vec{z}_0) - G(F(\vec{x}_0,\vec{z}_t^{(j)}),\vec{z}_t^{(j)})\right\|_2^2\right],\end{aligned} \tag{3.6}$$

where $\vec{x}_0$ stands for a point on the template $\mathcal{N}_0$ and we sample a subset $\mathcal{P}_0^{(j)}$ from $\mathcal{N}_0$ for each different target NeRF $\mathcal{N}_t^{(j)}$. For all the feature similarity supervision, we adopt features of G at multiple layers and concatenate them to better reflect the semantics of a point. We justify our choice in Sec. 3.2.5.

Cycle-Consistency Regularization. Since the conditional deformation fields, F and B, are supposed to restore the original deformation field D, when the same NeRF $\mathcal{N}_i$ is used as both the source and target NeRF, they should satisfy $D(\vec{x},\vec{z}_t,\vec{z}_t) = \vec{x}$ for all valid points $\vec{x}$. As depicted in Fig. 3.4(a,b), we further apply a cycle-consistency regularization for B and F:

$$\begin{aligned}&\mathcal{L}_{cycle} = \\ &\frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{|\mathcal{P}_s^{(i)}|}\sum_{\vec{x}_s\in\mathcal{P}_s^{(i)}}\left\|F(B(\vec{x}_s,\vec{z}_s^{(i)}),\vec{z}_s^{(i)}) - \vec{x}_s\right\|_2^2\right] + \\ &\frac{1}{m}\sum_{j=1}^{m}\left[\frac{1}{|\mathcal{P}_0^{(j)}|}\sum_{\vec{x}_0\in\mathcal{P}_0^{(j)}}\left\|B(F(\vec{x}_0,\vec{z}_t^{(j)}),\vec{z}_t^{(j)}) - \vec{x}_0\right\|_2^2\right].\end{aligned} \tag{3.7}$$

Second-Order Feature Similarity Loss. Apart from the aforementioned point-wise cycle-consistency loss that regularizes the deformation coherency of learned mapping, we also combine it with Eq. 3.5, 3.6 and impose a feature-based cross-instance cycle-consistency loss. Specifically, for a given point $\vec{x}_s$ in a source NeRF $\mathcal{N}_s^{(i)}$ paired with latent code $\vec{z}_s^{(i)}$, beyond imposing the similarity regularization only

over the template NeRF $\mathcal{N}_0$, we further deform its intermediate point $\vec{x}_0 = B(\vec{x}_s, \vec{z}_s)$ to a randomly sampled paired target NeRF $\mathcal{N}_t^{(i)}$ and calculate their feature similarity:

$$\begin{aligned}\mathcal{L}_{cycle}^{2nd} = \frac{1}{n}\sum_{i=1}^{n}\Bigg[\frac{1}{|\mathcal{P}_s^{(i)}|}\sum_{\vec{x}_s\in\mathcal{P}_s^{(i)}} \\ w_{\vec{x}_s} * \frac{1}{2}\left\|G(F(B(\vec{x}_s, \vec{z}_s^{(i)}), \vec{z}_t^{(i)}), \vec{z}_t^{(i)}), G(\vec{x}_s, \vec{z}_s^{(i)})\right\|_2^2\Bigg].\end{aligned} \tag{3.8}$$

We find this auxiliary regularization improves cross-instance deformation consistency.

Deformation Smoothness Regularization. To encourage the smoothness of deformation and reduce spatial distortion, a deformation smoothness regularization is also included. Here we penalize the norm of the Jacobian matrix $\mathbb{J}_D = \nabla D$ of the deformation fields [146] to ensure the learned deformations are physically smooth:

$$\begin{aligned}&\mathcal{L}_{smooth} = \\ &\frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{|\mathcal{P}_s^{(i)}|}\sum_{\vec{x}_s\in\mathcal{P}_s^{(i)}}\max(\left\|\nabla B(\vec{x}_s, \vec{z}_s^{(i)})\right\|_2^2 - \epsilon, 0)\right] + \\ &\frac{1}{m}\sum_{j=1}^{m}\left[\frac{1}{|\mathcal{P}_0^{(j)}|}\sum_{\vec{x}_0\in\mathcal{P}_0^{(j)}}\max(\left\|\nabla F(\vec{x}_0, \vec{z}_t^{(j)})\right\|_2^2 - \epsilon, 0\right],\end{aligned} \tag{3.9}$$

where ϵ is the slack parameter for the smoothness regularization. The final objective is thus $\mathcal{L}_{total} = \mathcal{L}_{sim}^{F} + \mathcal{L}_{sim}^{B} + \lambda_{cycle}\mathcal{L}_{cycle} + \lambda_{cycle}^{2nd}\mathcal{L}_{cycle}^{2nd} + \lambda_{smooth}\mathcal{L}_{smooth}$ where λ_{cycle}, λ_{cycle}^{2nd}, and λ_{smooth} are balancing coefficients, which are respectively set to 1, 0.1 and 10^{-4} in practice. Overall, the first 2 loss terms encourages the establishment of dense NeRF correspondences, and the remaining loss terms regularizes the geometry properties of the learned correspondence, including the cycle consistency and deformatio smoothness.

3.1.4 Training Strategy

While the pre-trained π-GAN G serves as a source of infinite object NeRFs, in each iteration of the training process we will sample a batch of source NeRFs $\{\mathcal{N}_s^{(i)}\}_{i=1}^n$ with corresponding latent codes $\{\vec{z}_s^{(i)}\}_{i=1}^n$, and a batch of target NeRFs

$\{\mathcal{N}_t^{(j)}\}_{j=1}^m$ with the corresponding latent codes $\{\tilde{z}_t^{(j)}\}_{j=1}^m$. To further sample a point set for each sampled NeRF $\mathcal{N}_*^{(i)}$, for each pixel within the resolution $H \times W$ we shoot a ray $r(v) = \boldsymbol{o} + v\boldsymbol{d}$ where $\boldsymbol{d}$ identifies the direction from the camera to the pixel. Subsequently, for each ray we follow Mildenhall et al. [2] and conduct a hierarchical sampling to obtain a *fine* set of points, i.e., points near the object surface. We denote the union of these point sets sampled from source as $\{\mathcal{P}_s^{(i)}\}_{i=1}^n$, which are used to train the B model. Since the points sampled to train F models are all from the template NeRF $\mathcal{N}_0$, here we denote the the point sets paired with target NeRF $\mathcal{N}_t^{(j)}$ as $\{\mathcal{P}_0^{(j)}\}_{j=1}^m$ for clarity.

Curriculum Sampling of NeRFs. In practice, we find the variation between the sampled NeRF and the template NeRF can significantly affect the training process, which may even collapse at the beginning stage if it gets a sampled NeRF that differs substantially from the template.

To improve training stability and efficiency, we adopt a curriculum sampling strategy to obtain NeRFs from G, by gradually morphing the template NeRF in the latent space to sample NeRFs with growing complexity. Specifically, since in π-GAN, the semantics of a sampled NeRF is determined by the modulation signals $(\boldsymbol{\beta}, \boldsymbol{\gamma})$, we can linearly interpolate between two sets of modulation signals to gradually morph one NeRF into another. Inspired by this property of π-GAN, when we sample a set of n NeRFs $\{\mathcal{N}^{(i)}\}_{i=1}^n$, we compute their corresponding modulation signals $\{(\boldsymbol{\beta}^{(i)}, \boldsymbol{\gamma}^{(i)})\}_{i=1}^n$ from their latent codes. Subsequently, we adjust the learning complexity by blending them with the template NeRF as

$$\boldsymbol{\beta}^{(i)}(\alpha) = \boldsymbol{\beta}_0 + \alpha \cdot (\boldsymbol{\beta}^{(i)} - \boldsymbol{\beta}_0) \tag{3.10}$$

$$\boldsymbol{\gamma}^{(i)}(\alpha) = \boldsymbol{\gamma}_0 + \alpha \cdot (\boldsymbol{\gamma}^{(i)} - \boldsymbol{\gamma}_0), \tag{3.11}$$

where $(\boldsymbol{\beta}_0, \boldsymbol{\gamma}_0)$ are the modulation signals of the template NeRF and α controls the learning difficulty. In practice, we start from $\alpha = 0$ and linearly increase the value to 0.6 during training, which is a reasonable value to balance sampling quality and diversity [86]. In this way, the model learns to produce identity deformation first and then gradually evolves to model more complicated deformation when trained on more challenging samples.

TABLE 3.1: Hyper parameters of the sampling and regularization loss weights.

Dataset	Ray Steps	Depth Mask	Sampling Ratio	Batch Size	λ_{cycle}	λ_{smooth}
CelebA [147]	24	1.08	0.2	131,072	0.1	0.1
Carla [30]	48	1.2	0.05	65,536	0.05	0.01
Cats [148]	36	1.08	0.1	49,152	0.1	0.1

3.2 Experiments

3.2.1 Experimental Setup

As discussed in [65, 66], there is no dense correspondence dataset available with ground truth for structure with variations. Therefore, we adopt three proxy tasks as surrogate metrics to evaluate the learned correspondences of *DDF*. In Sec. 3.2.2, we first qualitatively demonstrate the dense correspondences learned by *DDF* through texture transfer. Quantitative results are shown in two alternative tasks, namely fine-grained segmentation transfer and keypoints transfer, in Sec. 3.2.3 and Sec. 3.2.4, respectively. All imagery results shown are rendered at 256^2 resolution.

Implementation Details. In all the experiments, the learning rate is set to 5×10^{-5} and decay in every $5,000$ iteration with gamma=0.5. We adopt Adam [149] optimizer to train the deformation models. In each training iteration, we randomly sample a batch of 10 source NeRFs $\{\mathcal{N}_s^{(i)}\}_{i=1}^{10}$ with corresponding latent codes $\{\vec{z}_s^{(i)}\}_{i=1}^{10}$, and a batch of target NeRFs $\{\mathcal{N}_t^{(j)}\}_{j=1}^{10}$ with corresponding latent codes $\{\vec{z}_t^{(j)}\}_{j=1}^{10}$. For all experiments, we train the *DDF* for $80,000$ iterations, which takes about 8 hours on a single Tesla V100 GPU. The hyperparameter details are listed in Tab. 3.1. We profile our method over the segmentation transfer of a single $128*128$ instance with 96 samples along a ray. With $374.991(K)$ parameters and $582.187(G)$ MACs, *DDF* achieves the latency of $2.145(s)$ on a V100 GPU.

Evaluation Dataset. We extensively demonstrate our approach on human faces from CelebA dataset [147, 150] as the main object category, as human faces are rich in geometric details, making them the best choice for demonstrating the accuracy, smoothness, and robustness of learned correspondences. Moreover, human faces are also rich in downstream tasks, from which we can effectively investigate the potential of learned correspondences. The qualitative results on cats [148] and cars [30, 151] are also included.

Sampling Details. To train the *DDF* network efficiently, we conduct hierarchical sampling to obtain 3D point sets with more specific semantic meanings. As in [2], we first uniformly sample points in 3D space and then sample via importance sampling a more informed fine point set given the density output of the "coarse" point set. These samples are biased toward the more relevant parts of the rendered object. We list the sampling details in Tab. 3.1. Apart from applying a foreground depth mask to filter out background information to increase sampling efficiency, we also control the sampling ratio of remaining rays. By defining a smaller sampling ratio, we could increase the number of NeRF sampled per batch to increase the diversity of training samples. We curtail the sampling points to a certain number so to maintain the stability of training.

NeRF-based GANs. We use the officially released π-GAN pretrained models for dense correspondence learning, and also include EG3D [1] pretrained model for some comparisons. To extract network features, we use the features starting from layer 4. We find the middle layer features have more correlation with the underlying semantics of a given region, while the last few layers are more sensitive to low-level details such as color variations, which could not provide meaningful clues for dense correspondence learning. We further justify our choice in Fig. 3.2.5.

3.2.2 Qualitative Results on Texture Transfer

In this subsection, we qualitatively demonstrate the dense correspondences learned by *DDF* through texture transfer. The results here validate that *DDF* learns accurate underlying structures of NeRFs and their associated correspondences without explicit correspondence supervision provided during training.

Texture Transfer via *DDF*. We denote $\mathcal{N}_s \rightarrow \mathcal{N}_t$ as the process of transferring texture from NeRF $\mathcal{N}_s$ to NeRF $\mathcal{N}_t$ while maintaining the geometry of $\mathcal{N}_t$. To perform the transfer, for each sampling point $\vec{x}_s$ to render NeRF $\mathcal{N}_s$, we first deform $\vec{x}_s$ to the template correspondence $\vec{x}_0$ via B and then deform it to the target NeRF $\mathcal{N}_t$ space correspondence point $\vec{x}'_t$ via F. We query the geometry of $\mathcal{N}_t$ and texture of $\mathcal{N}_s$ to conduct volume rendering in the given view direction. To remove ambiguity, we mask out the hair and background class of the source class using segmentation masks and conduct texture transfer on other semantic regions on the human face.

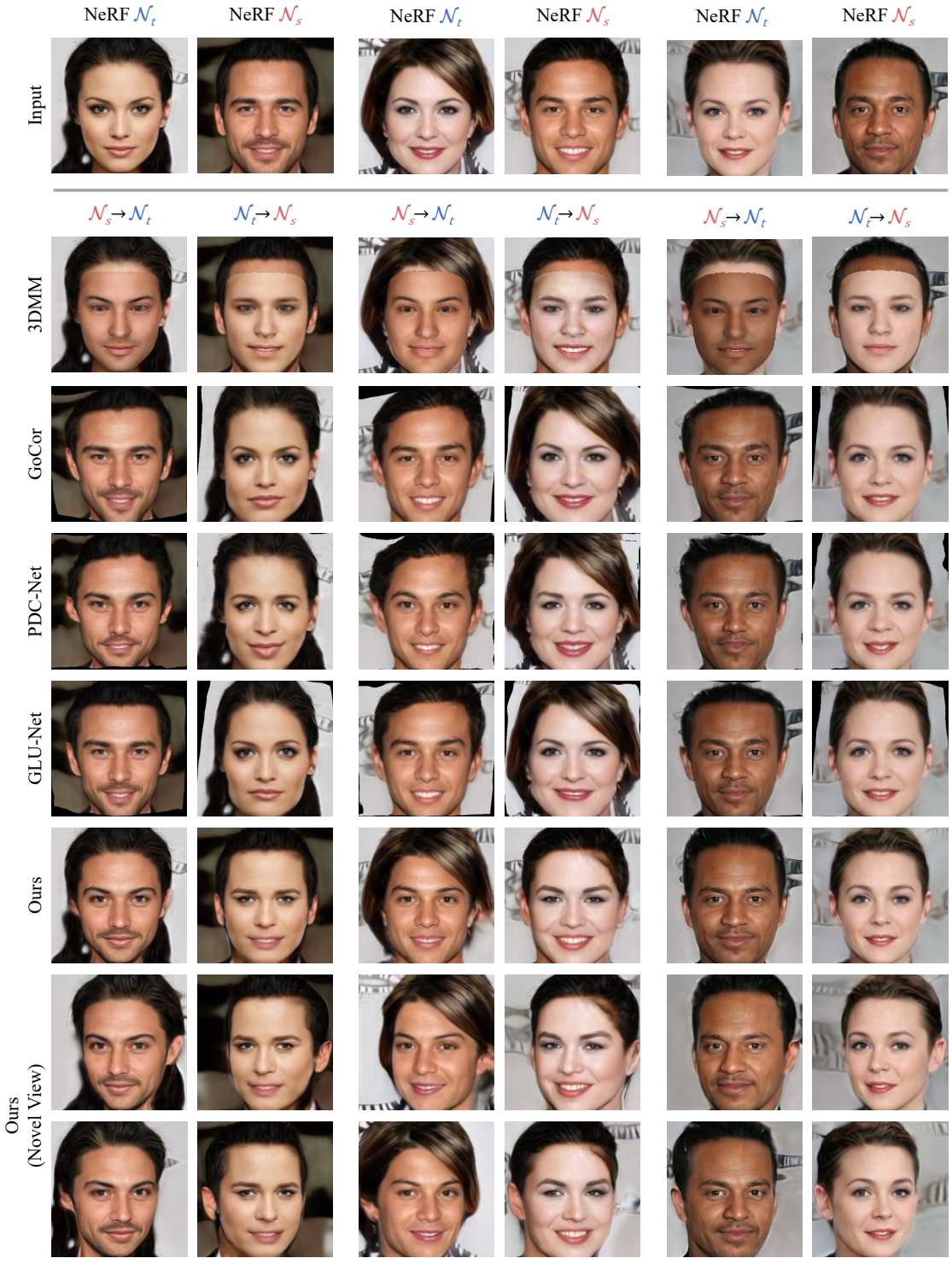


FIGURE 3.5: **Texture transfer through the learned deformation field**. We randomly sample three NeRF pairs here for qualitative evaluation (shown at the top row as inputs). For each NeRF sample $\mathcal{N}_*$ we transfer the texture from the paired NeRF according to their 3D dense correspondences. Specifically, for the column labeled with NeRF $\mathcal{N}_{t(s)}$, we show the texture transfer results from source NeRF $\mathcal{N}_{s(t)} \to \mathcal{N}_{t(s)}$. We conduct dual texture transfer on three pairs (depicted in different separated columns) and show the transferred results over three different angles. The separate line splits the input, the model-based method's output, the learning-based methods' output, and ours. Though not designed for 2D images, our method consistently outperforms the baseline method in terms of fidelity and naturalness.

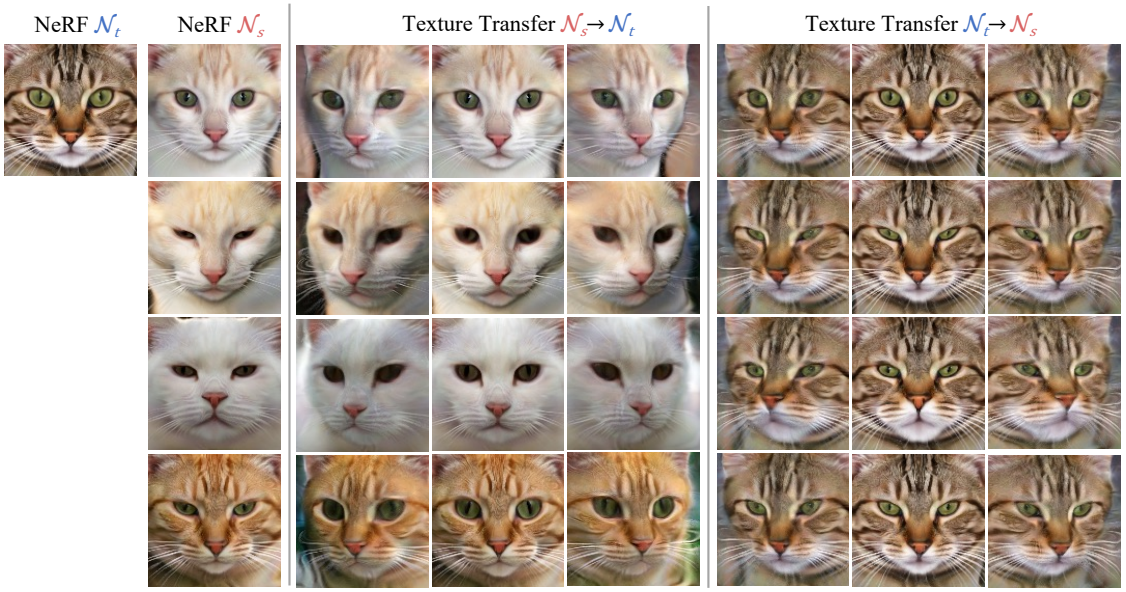


FIGURE 3.6: **Visualization of texture transfer on Cats Dataset.** The size of the eyes and positions of the nose and the overall shape could serve as hints to observe the difference between different cats.

Results on Human Faces. With the above rendering process, we show the cross-instance texture transfer results in Fig. 3.5. We compare our method with two types of baselines, the model-based 3DMM [152] method (row 1) and the state-of-the-art learning-based 2D correspondence matching methods [70–72] (rows 2-4). Visually inspected, our method produces semantic plausible dense correspondences with high-fidelity texture transfer results. We also show our results in multiple views to demonstrate that our method has learned both 3D consistent dense correspondences. Note that good texture transfer results could not be achieved without accurate correspondence matching in 3D space. Our approach shows superior texture transfer in comparison to existing model-based and learning-based methods.

Results on Other Categories. To further illustrate the deformation ability of dual fields in *DDF*, we apply our method on two more pretrained NeRF-based generators, trained respectively on the Cats [148] datasets and the synthetic CARLA [30, 153] dataset. We train the corresponding *DDF* models on the new categories with parameters listed in Tab. 3.1 and conduct texture transfer using the same pipeline. Given the source NeRF $\mathcal{N}_s$ and target NeRF $\mathcal{N}_t$, we show transfer results from both $\mathcal{N}_s \rightarrow \mathcal{N}_t$ and $\mathcal{N}_t \rightarrow \mathcal{N}_s$ to validate the performance of *DDF* on shape categories with larger structure variations.

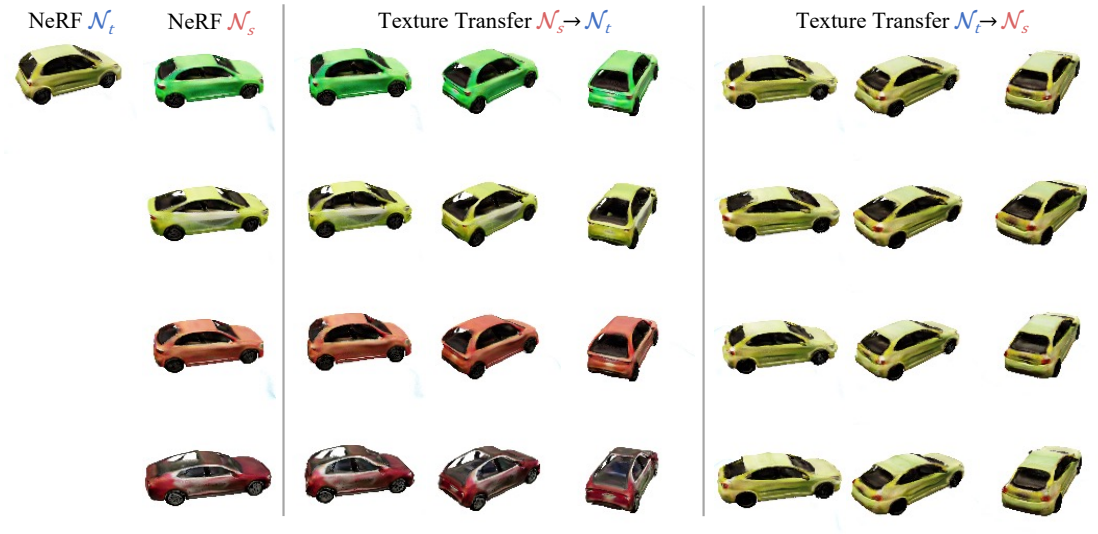


FIGURE 3.7: **Visualization of texture transfer on CARLA Dataset.** In the category with large structure deviations, *DDF* could still generate sound deformation with high fidelity and accuracy.

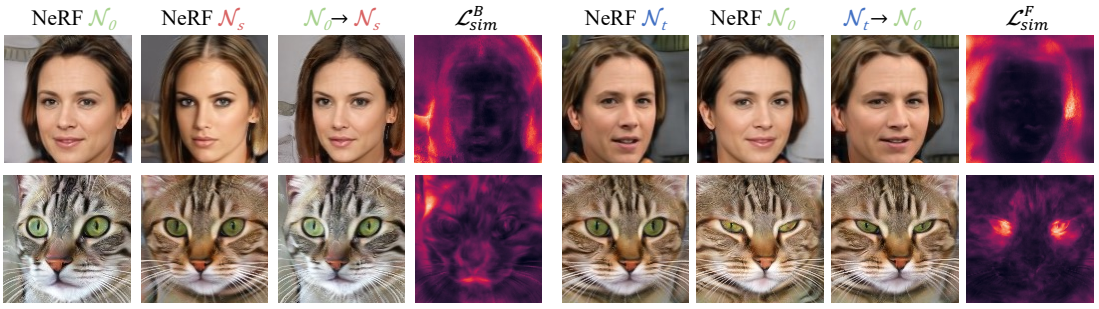


FIGURE 3.8: **Visualization of uncertainty map of learned dense correspondence.** The left and the right column shows the pixel-wise uncertainty map corresponding to backward deformation B and forward deformation F, respectively. The pixel-wise uncertainty is calculated as the integration of the point-wise $\mathcal{L}_{sim}^*$ with Eq. 3.1.

In Fig. 3.6 we show the texture transfer results on the Cats dataset. Though cat's faces have fewer discriminative features compared to human faces, through the overall shape and local details such as the size of the cat's mouth and eye, we could see that the transferred multi-view results share the same texture with the source NeRF, while still matching the geometry of the target NeRF.

In Fig. 3.7 we show the transfer results on synthetic CARLA dataset. Compared to CelebA and Cat datasets, Cars have larger structure variations and larger deformations between different NeRFs, leaving learning accurate deformation on CARLA dataset more challenging. Through the qualitative results, the texture transfer of

$\mathcal{N}_s \rightarrow \mathcal{N}_t$ through *DDF* produces convincing correspondence across two NeRFs that are largely different. The shared semantic components are matched to the maximum extent and also preserve the original geometry pattern of NeRF $\mathcal{N}_t$. The texture transfer of the other direction $\mathcal{N}_t \rightarrow \mathcal{N}_s$ is overall reasonable but produces mismatches in some regions with large deformations such as the car roof, whose size varies evidently across different objects represented in NeRFs.

Uncertainty of Texture Transfer. In Fig. 3.8 we showed the uncertainty heat map and the texture transfer results of the learned *DDF*. After the training of *DDF*, we conduct correspondence inference and calculate the feature similarity loss of the correspondence points and the original points. The feature similarity loss $\mathcal{L}_{sim}^{B}$ and $\mathcal{L}_{sim}^{F}$ between inferred correspondence points could be naturally interpreted as the uncertainty of the learned correspondence. A low feature similarity loss denotes the correctness of deformation and guarantees the visual effects of texture transfer. In Fig. 3.8, we separately show the uncertainty maps corresponding to the backward deformation B and the forward deformation F. As can be seen, the semantic regions of the human face have a low uncertainty score, except for ambiguous regions like hair. For Cat face, the overall uncertainty is low except for the regions where deformations are large such as cat eyes and mouth. After *DDF* converges, the heat map could also serve as the confidence score of the dense correspondences between two NeRFs.

3.2.3 Quantitative Results on Segmentation Label Propagation

To demonstrate the quantitative performance of the learned dense correspondence, following the previous method [66] we resort to segmentation label propagation as the surrogate metric. Intuitively, a 3D point shall share the same segmentation label with its correspondence point from another object with structure variations if being deformed via an accurate correspondence algorithm. Thus, segmentation label propagation could serve as a metric to inspect the performance of learned correspondences. Similar to the texture transfer experiments discussed in Sec. 3.2.2, here we conduct segmentation label propagation on the fine-grained human faces.

Different from explicit-based representations and SDF-based implicit representation [14, 65, 66], NeRF-based representation is designed for view synthesis and has

no clear surface boundary, leaving it hard to directly evaluate the segmentation accuracy in the 3D space. Therefore, we propose to conduct segmentation label propagation in the 3D space and project the propagated labels in the 2D space through volume rendering depicted in Eq. 3.1 for evaluation. We describe how we conduct segmentation label propagation below.

Segmentation Label Propagation. For this task, we first render the front view of our template NeRF $\mathcal{N}_0$ and provide it with the oracle segmentation map acquired from a pretrained DeepLabV3 [5, 154] segmentation model. We refer to this front view as the oracle image, as shown in Fig. 3.9(b). For an unlabeled test image rendered from a NeRF, for each pixel, we cast a ray through this pixel and sample 96 points along the corresponding ray. For a point $\vec{x}$ along the ray, we use the network B to query its correspondence point $\boldsymbol{x}'$ in the template NeRF. The projected segmentation label is thus regarded as the segmentation label prediction of $\vec{x}$. To acquire 2D segmentation predictions for evaluation, we aggregate the predictions of 3D points by rescaling their voting contributions with the transmittance value $T(i)$ defined in Eq. 3.1. The whole segmentation process costs around 3 seconds for a single image on a Tesla V100 GPU. Since we only use the oracle segmentation map for the oracle image, we consider our approach as a 1-shot segmentation method.

Evaluation Settings. We compare *DDF* with DatasetGAN [5], CoordGAN [74] and GANgealing [155], which are respectively the state-of-the-art 2D GAN-based few-shot segmentation method and the concurrent works on establishing 2D correspondences via 2D GANs.

Since *DDF* does not directly accept real images as input, following Zhang et al. [5], we sample $10,000$ image-annotation pairs from pretrained GANs as a dataset and train a segmentation model, as shown in Fig. 3.10. We evaluate the trained segmentation model on the official DatasetGAN test set to

(a) DatasetGAN (b) DDF

FIGURE 3.9: **Canonical segmentation annotation for two 1-shot segmenters.** (a) DatasetGAN [5] and (b) ours. For DatasetGAN we choose the first annotated image in their training set, and for *DDF* we simply use the frontal face of the Template for segmentation transfer. For ease of comparison, the segmentation annotations of the Template are simply acquired through an off-the-shelf pretrained DatasetGAN segmenter, which already provides reasonable results.

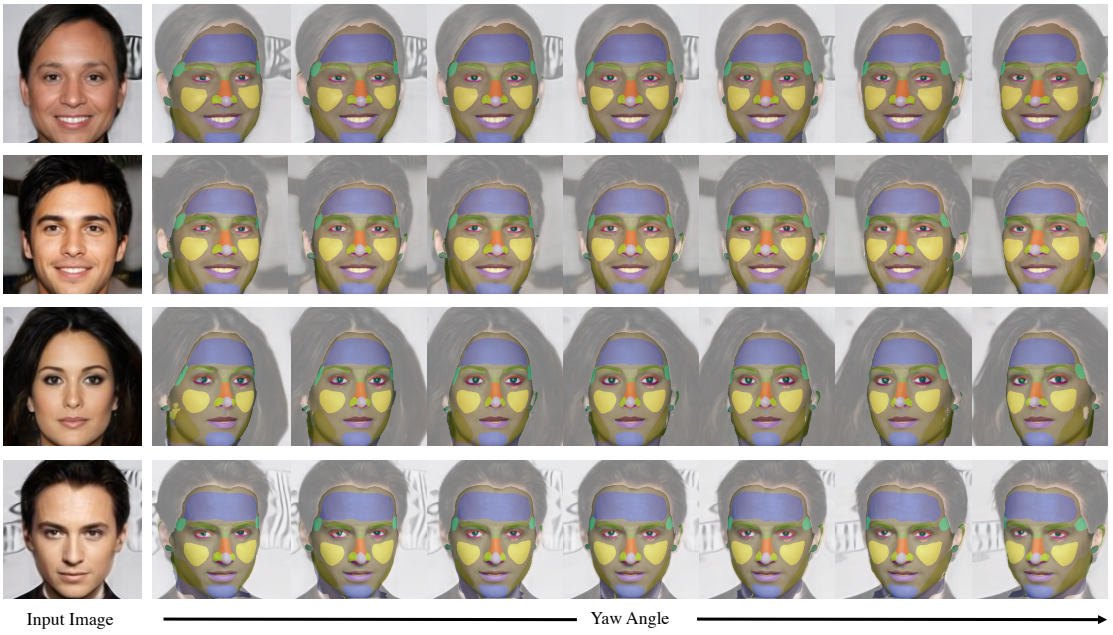


FIGURE 3.10: **3D consistent segmentation label transfer with novel poses**. Given an annotated projection of Template NeRF shown in Fig. 3.9(b), we could derive view-consistent segmentation maps of other NeRF objects through our method. Note that for instances missing teeth class (1st row and 2nd row, segment in white color), our method could still derive accurate correspondences though the teeth class does not exist in the segmentation template. This demonstrates that our method learns consistent 3D dense correspondence.

quantify the segmentation accuracy, which consists of 16 fine-grained annotated 1024^2 real-world images. For a fair comparison, we also train an 1-shot DeepLabV3 model as the baseline, which uses one annotated pair as the sampling source (Fig. 3.9(a)) and follows the data generation pipeline of Zhang et al. [5]. Since Zhang et al. [5] adopts pretrained StyleGAN under 512^2 resolution on the FFHQ dataset while our pretrained GANs are trained over 128^2 resolution CelebA dataset, all the test images are bilinear interpolated with a resolution of 128^2 for evaluation. We adopt the official implementation of DatasetGAN for data generation and use the default settings for all the segmentation models training. The standard mIOU is adopted as the segmentation evaluation metric.

Results. We show the quantitative results the test set of Zhang et al. [5] in Tab. 3.2. As can be seen, our method achieves comparable performance with the baseline, and even performs better over some classes like hair and nose, indicating that our learned correspondences are accurate and smooth. We show the qualitative results in Fig. 3.11. As can be observed, without relying on explicit segmentation supervision, our method can perform 3D consistent segmentation transfer, which is

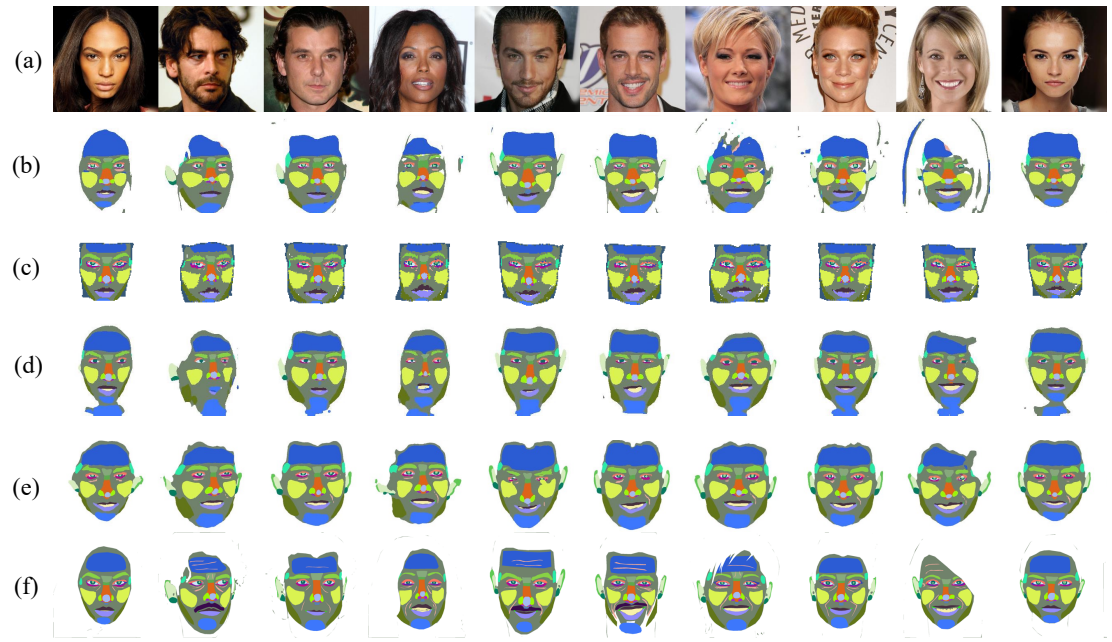


FIGURE 3.11: **Visualization of the 1-shot segmenter prediction.** Starting from the top row, we show the (a) The test set input image, (b) segmentation prediction from DatasetGAN 1-shot segmenter, (c) segmentation prediction from GANgealing 1-shot segmenter, (d) segmentation prediction from *DDF* 1-shot segmenter using $\pi-$GAN as the base model, (e) segmentation prediction from *DDF* 1-shot segmenter using EG3D as the base model and (f) the ground truth segmentation annotation.

not possible with existing 2D correspondence baselines such as DatasetGAN 1-shot segmenter. This is made possible by establishing plausible correspondence between different semantic regions across NeRFs, despite their structure variations in 3D space.

TABLE 3.2: **mIOU scores of two 1-shot segmenters on DatasetGAN [5] test set.** The corresponding segmenters are trained over the synthetic dataset generated by two methods. We show the performance of two versions of *DDF* based on two generators pretrained on different datasets. The 1-shot segmenter trained on our dataset is competitive against the counterpart which is trained in high-resolution images, demonstrating the merit of the learned correspondence.

Method	Mean IOU	Eyes	Mouth	Nose	Cheek	Chin	Hair	Eyebrows	Ears	Jaw	BG
GANgealing 1-shot segmenter	33.8	25.0	36.9	41.7	60.4	7.8	-	32.0	-	18.8	47.6
DatasetGAN 1-shot segmenter	**56.9**	40.5	**62.6**	52.5	61.6	65.5	**72.4**	**59.6**	**49.0**	16.4	**81.4**
DDF 1-shot segmenter	54.6	**51.0**	53.2	**55.4**	**69.2**	**82.6**	67.4	54.2	40.9	**66.9**	75.06

For further comparison with concurrent work that distills correspondences from 2D GANs, Tab. 3.3 presents the mIOU scores over two real-world datasets. Following [74], we train an encoder that predicts the source NeRF code $\vec{z}_s$ using the techniques described in Sec. 3.2.6 and conducts feed-forward inference over the

TABLE 3.3: **IOU comparison for segmentation label propagation**. Our method achieves comparative performance with the 2D representation learning method, and is the only method that supports 3D dense correspondence searching over implicit functions. $*$ means 3D aware.

Method	CelebA-HQ	DGAN-face
Swap AE [157]	24.73	5.48
MoCo [158]	36.19	10.00
VFS [159]	38.10	8.55
ResNet50 [160]	39.48	11.05
Pix2Style2Pix [97]	48.50	20.36
GANgealing [155]	46.61	19.35
CoordGAN [74]	**52.25**	23.78
DDF*	45.32	19.18
DDF$^*_{\text{EG3D}}$	**52.22**	**39.09**

TABLE 3.4: **PCK-Transfer on facial landmarks**. Our method achieves better performance compared to 2D geometry matching methods (row 1,2,3) and competitive performance against SoTA 2D semantic matching method (row 4,5).

Methods	Correspondence Type	PCK@0.05 ↑	PCK@0.01 ↑	AEPE ↓
SIFT Flow [161]	2D Geometry Matching	92.7	32.9	5.22
GoCor [71]	2D Geometry Matching	87.9	24.8	6.24
GLU-Net [70]	2D Geometry Matching	90.0	30.4	5.78
VAT [73]	2D Semantic Matching	79.46	14.24	16.67
CATs [162]	2D Semantic Matching	**99.98**	**55.07**	**2.82**
Ours	3D Semantic Matching	**95.0**	**41.6**	**4.47**

input images. Our method outperforms 2D learning-based models on this task and achieves competitive performance compared with CoordGAN, with the merit of establishing dense correspondences in 3D space. Compared with building correspondences over 2D pixels, establishing correspondences in implicit 3D space is exponentially harder, as explained previously. Moreover, compared with mature 2D GAN families and toolboxes, the development of 3D GANs is still in its early stage. Equipping *DDF* with more developed 3D GANs, i.e. Chan et al. [156], can potentially close the gap.

3.2.4 Quantitative Results on Keypoints Transfer

Though segmentation label propagation is an intuitive and well-adopted surrogate metric for evaluating learned correspondence, we argue that quantitative evaluation

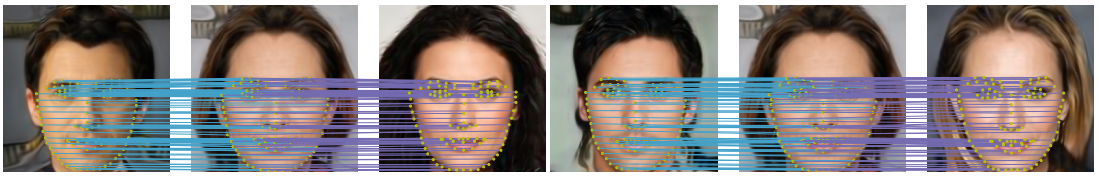


FIGURE 3.12: **Visualization of learned correspondences via landmark transfer.** For each triplet, we first predict 98 facial landmarks of the first column acquired through an off-the-shelf model [6]. We deform the predicted landmarks to the template through network B, and then further deform the landmarks on the template to another sampled face through network F. We sample 3D points near the surface of one NeRF and calculate the dense correspondence point on the target NeRF with our deformation network. Please zoom in for details.

using this metric alone is contrived. Specifically, segmentation label propagation is essentially a pixel-wise classification task, which means any errors in dense correspondences within a segment will not be detected. Moreover, only network B is used in the segmentation label propagation experiment, which could not quantitatively evaluate the forward deformation field F in our method. Therefore, we further evaluate our method via keypoints transfer [65], which is a regression task with independent ground truth for each transferred landmark. In our context, this task can be viewed as few-shot 3D facial landmark transfer learning with 1 sample as training data.

For this task, as in the segmentation label propagation pipeline, we first use an off-the-shelf facial landmarks prediction model [6] to label the template frontal view image with 98 landmarks, which can be seen in the middle of Fig. 3.12. Since these points are in the image space, we first unproject them back to the template NeRF 3D space by appending the corresponding depth values viewing these landmarks positions, which we denote as $\mathcal{P}_0^{lms} = \{\vec{x}_0^{lms(k)}\}_{k=1}^{98}$. After that, we resort to F model and deform these unprojected 3D points to $\mathcal{N}_t$ by $F(\vec{x}_0^{lms(*)}, \vec{z}_t) = \vec{x}_t^{lms(*)}$. The deformed points $\vec{x}_t^{lms(*)}$ are projected back to the image space as the transferred 2D facial landmarks of $\mathcal{N}_t$.

Evaluation Settings We also compare our method with both 3D model-based method (3DMM [152]) as well as current state-of-the-art 2D learning-based matching method [70–72] as our baselines. We regard the output of a representative landmark detector MTCNN [163] as ground-truth. For baselines, we also consider the hand-crated descriptor SIFT Flow [161] as well as several learned descriptors [70–72] that attain state-of-the-art performance on commonly used dense

correspondence benchmarks (e.g., MegaDepth [164]). For the baselines, we use the officially released models to conduct inference in our experiments. We employ the Percentage of Correctly Keypoints (PCK) and Average End Point Error (AEPE) as the evaluation metrics.

Results. We evaluate the performance over 5,000 randomly sampled human faces with different view angles and show the quantitative results in Tab. 3.5. For all the baselines, we adopt the pre-trained weights on SPair-7K [165] dataset for evaluation. At threshold PCK@0.01, *DDF* achieves 41.6 against competitive geometry matching baselines. Comparisons against SoTA semantic matching methods, VAT [73] and CATs [162], are also included. Specifically, CATs achieves better 2D landmarks transfer performance while *DDF* holds the merits of 3D consistency. Note that we attribute the worse performance of VAT to that it is mainly designed for few-shot segmentation and may not generalize well on face landmarks transfer. This result strongly supports the effectiveness of *DDF*.

3.2.5 Ablation Study

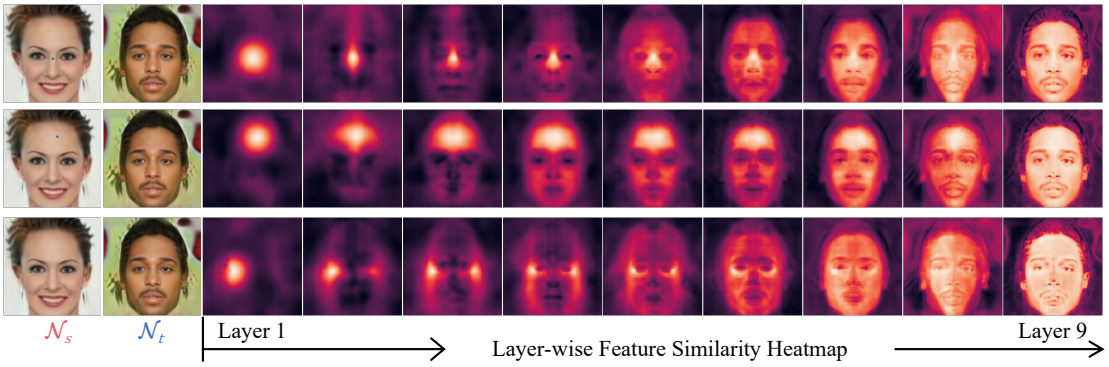
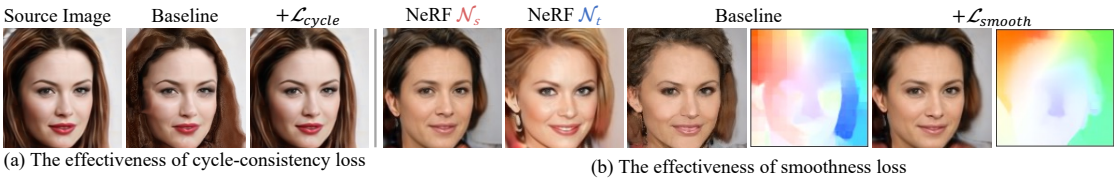


FIGURE 3.13: **Layer-wise feature correlation between projected features of two NeRF.** The NeRFs are sampled from a pretrained π-GAN generator from *shallow (leftmost)* to *deep (rightmost)*. Since $\pi-$GAN generator adopts an 8-layer MLP design appended with a view-dependent MLP layer, here from left to right, we show the feature similarity heatmap from the 1st layer (3rd column) to the 9th layer (last column) of the pretrained generator. We project the features of 3D NeRF to 2D using Eq. 3.1 for better visualization. In each row, a random 2D point from the source NeRF is selected to calculate layer-wise feature similarity heatmaps with the projected feature map of the target NeRF.

Selection of Generator Feature. In our work, we select multiple layers from the generator as the training supervisions of the feature similiary losses depicted in Eq. 3.5 and 3.6. Here we justify the intuition behind this. Different from the feed-forward models [160, 166], generative models like GANs is trained to decode information from a compact latent code. Therefore, features from earlier layers should contain more high-level semantics information while later layers contain more instance-specific texture information. To justify this intuition, we show the layer-wise feature similarity heatmap between the projected 2D feature maps of $\mathcal{N}_s$ and $\mathcal{N}_t$ over a pretrained π-GAN generator in Fig. 3.13. Specifically, given $\mathcal{N}_s$ and $\mathcal{N}_t$, we calculate the 2D feature maps by integrating the features of points along each rays using the volume rendering equation depicted in Eq. 3.1 and get two sets of features maps $F_s = \mathcal{R}^{N*H*W*C}$ and $F_t = \mathcal{R}^{N*H*W*C}$, where $N = 9$ is the layer number of π-GAN generators. Given a 2D coordinate (u, v), we retrieve its corresponding features $F_s^{u,v} \in \mathcal{R}^{N*C}$ from the source feature maps F_s and calculate the cosine similarity with the target feature maps F_t within each layer.

As can be seen, the generator features from different layers encode semantics from different levels, where the semantics compactness linearly decreases as the network goes deeper. Surprisingly, the early generator features are even robust under the symmetric semantics such as the right and left corners of the eyes (3rd row). This is an indispensable property in establishing dense correspondences where a 3D point from the source left eye should not establish correspondence to points in the right eye region of the target. Thus, we choose the normalized features from the first 5 layers as the supervision signals of the *DDF*, which encode unambiguous correspondence information. This property has also been validated in 2D generative models [155, 167, 168], where different layers of pretrained StyleGAN encode different types of information. Quantitatively, when only using the feature from the final layer of $\pi-$GAN generator for *DDF* training, the segmentation transfer performance degrades to 43.79 on CelebA-HQ. Moreover, when only using the feature from the first π-GAN generator layer for training, *DDF* fails to converge.

Deformation Regularization Terms. We validate the efficacy of our regularization terms in terms of qualitative results, including the cycle consistency term and deformation smoothness term A smooth deformation shall yield high-fidelity texture transfer results, while non-smooth deformation yields visual artifacts. To construct a baseline for evaluation, we remove the correspondence deformation



(a) The effectiveness of cycle-consistency loss

(b) The effectiveness of smoothness loss

FIGURE 3.14: **(a) Rendering from the self-reconstructed point through cycle deformation.** From the left is the input image, reconstructed with and without cycle-consistency loss. The deformation model trained with cycle-consistency loss can perfectly reconstruct itself, while the one without cycle-consistency loss leads to distortions. **(b) Output from deformation network trained without and with deformation smoothness loss.** We demonstrate the effectiveness of L_{smooth} via texture transfer from A to B. The corresponding optical flow is also included for visual evaluation. As can be seen, the lack of smoothness regularization leads to distorted visual results. Better zoom in for a better experience.

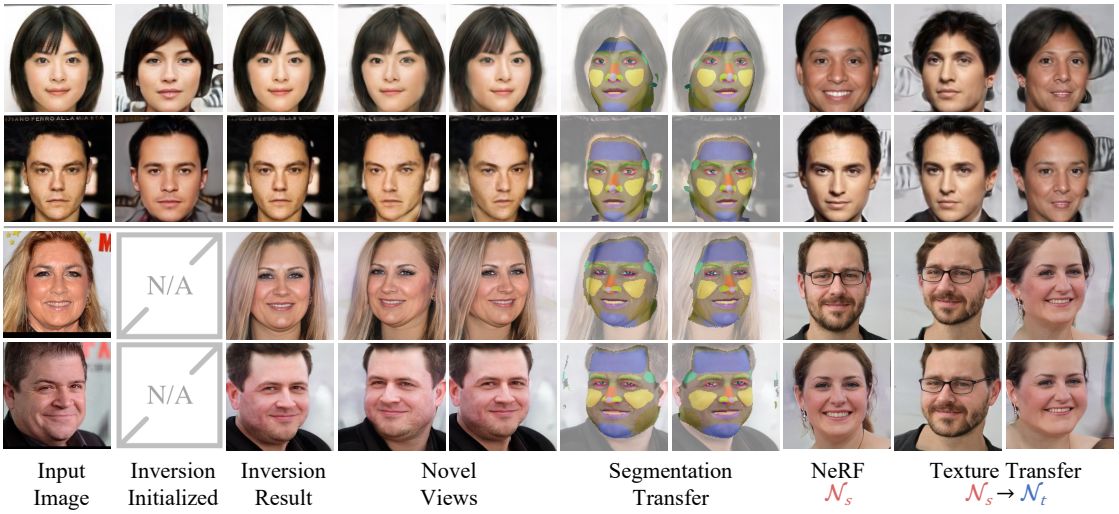


FIGURE 3.15: **Extending *DDF* to real images.** To apply *DDF* to real-world image, we first inverse the real-world images (1st column) into the latent space of the 3D GAN (2nd and 3rd columns). Beyond novel view synthesis (4th and 5th columns), *DDF* also supports 3D consistent segmentation transfer (6th and 7th columns). Given the reference NeRFs (8th column), our method could edit the texture of given identities without changing the overall shape. The first two rows are visualized using π-GAN as the base model and the last two rows are visualized using EG3D as the base model. For EG3D, we directly use E3DGE [7] as the GAN inversion method to get the inversion result.

smoothness loss term and only apply supervision from the feature similarity loss on network training. To evaluate the effect of cycle consistency regularization, we train a baseline without cycle consistency loss term and visualize the self-reconstruction as well as texture transfer results using the trained dual deformation field. As shown in Fig. 3.14(a), cycle-consistency term encourages the consistency property,

that after the forward and backward deformation a point from the source shape will map to itself. Meanwhile, the model trained with cycle consistency term learns less noisy deformation, which is essential when conducting downstream tasks such as texture transfer. As shown in Fig. 3.14(b), without the deformation smoothness regularization, the network tends to learn noisy deformation which leads to distortions in the final rendering. Moreover, we also include the optical flow of the baseline and our method for better qualitative evaluation, where the flow is calculated using RAFT [169]. As can be seen, our method produces smoother 2D flow where the baseline without smoothness terms has visual artifacts on the calculated 2D flow.

For the second order cycle consistency regularization, we find it has similar effect with $\mathcal{L}_{cycle}$ in qualitative performance. Moreover, we set $\lambda^{2nd}_{cycle} = 0$ and conduct the segmentation transfer evaluation as in Tab. 3.2 and observe a mIOU degrade from 56.9 to 55.3, which validates $\mathcal{L}^{2nd}_{cycle}$ could improve the deformation field performance.

TABLE 3.5: **Quantitative results of regularization loss terms**. We evaluate the quantitative ablation of the smoothness terms in the ablation study, including the cycle loss, deformation smoothness and high-order similarity loss. We adopt the criteria used in Tab. 3.2 and report the mIOU. We half the training iterations and report the best mIOU achieved via cross-validation. As can be seen the cycle-based loss terms $\mathcal{L}^{*}_{cycle}$ achieves the best results under appropriate weight, and the smoothness term $\mathcal{L}_{smooth}$ regularizes the deformation performance with larger weight. Therefore, we choose the most balanced weight on different dataset, as reported in Tab. 3.1.

Loss terms	$\mathcal{L}_{cycle}$	$\mathcal{L}^{2nd}_{cycle}$	$\mathcal{L}_{smooth}$
$\lambda = 1e-5$	38.34	37.96	**38.27**
$\lambda = 1e-4$	**38.72**	**38.23**	37.92
$\lambda = 1e-3$	38.47	38.15	37.86
$\lambda = 1e-2$	38.39	37.93	37.50
$\lambda = 1e-1$	38.2	38.12	37.43
$\lambda = 1$	37.9	38.0	37.32

Curriculum Training. To show that our proposed curriculum training strategy could help regularize the training and facilitate convergence, we report the values of feature similarity loss over the evaluation set, with different curriculum steps adopted during training. Using 16/128/1024/4096 steps the loss are respectively 0.455/0.410/0.310/0.287, which demonstrates the effectiveness of our method.

3.2.6 Extending *DDF* to Real Images

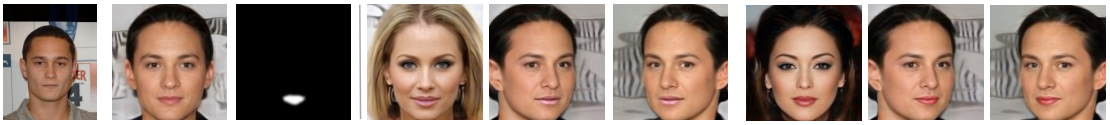


FIGURE 3.16: **Mask-guided texture transfer over real cases.** In the second column, we show the projected image of the GAN inversion of the source image. For mask-guided texture manipulation, we sample two synthetic NeRF from the pretrained GAN (4th and 7th columns) and conduct texture transfer guided by the foreground mask.

Training An Encoder for Inversion. To apply *DDF* real-world images, we use two encoders, namely an encoder denoted by $E(\cdot, \theta_G)$ to invert the input image to the latent space of the NeRF-based GAN and another encoder represented by $E(\cdot, \theta_{DDF})$ to invert the input image to the deformation conditions. Specifically, rather than directly output the low-dimensional deformation code $\vec{z}$, here we follow the observations of Tov et al. [98] which project the $\mathcal{Z}$-space code $\vec{z}$ to the $\mathcal{W}+$ space for better performance. Since the NeRF-based GAN (i.e., π-GAN) already follows this design, here we further augment each of the deformation fields with a mapping function $\mathcal{M}_{DDF}$ [143, 170]. During inversion, the corresponding encoders directly output the $\mathcal{W}+$ space modulations, i.e., $E(I, \theta_G) = \boldsymbol{\beta}^I_G, \boldsymbol{\gamma}^I_G$ and $E(I, \theta_{DDF}) = \boldsymbol{\beta}^I_{DDF}, \boldsymbol{\gamma}^I_{DDF}$.

The encoders are trained in two stages. In the first stage, we train the encoder $E(\cdot, \theta_G)$ where the output latent codes $\boldsymbol{\beta}^I_G, \boldsymbol{\gamma}^I_G$ are fed into the NeRF-based GAN to render a replicate of the images $\hat{I} = G(\boldsymbol{\beta}^I_G, \boldsymbol{\gamma}^I_G, \boldsymbol{\xi}_I)$, where $\boldsymbol{\xi}_I$ is the estimated camera pose of the input image using an off-the-shelf pose estimator. After $E(\cdot, \theta_G)$ converges, in the second stage we train the encoder $E(\cdot, \theta_D)$ to output the corresponding deformations conditions $\boldsymbol{\beta}^I_B, \boldsymbol{\gamma}^I_B$ and $\boldsymbol{\beta}^I_F, \boldsymbol{\gamma}^I_F$. Given the inverted latent code $\boldsymbol{\beta}^I_G, \boldsymbol{\gamma}^I_G$ of the input image, we conduct self texture transfer described in Sec. 3.2.2 to replicate the input image. To stabilize training, we also include synthetic samples from the pre-trained NeRF GAN as training data. Given a latent code $\vec{z} \sim p_{\vec{z}}$, we get the paired modulations $\boldsymbol{\beta}_{\vec{z}}, \boldsymbol{\gamma}_{\vec{z}}$ and synthesized image $I_{\vec{z}} = G(\vec{z}, \boldsymbol{\xi})$ under a random camera pose $\boldsymbol{\xi} \sim p_{\boldsymbol{\xi}}$ as training samples. Apart from image reconstruction loss, the predicted modulations from $E(\cdot, \theta_D)$ are encouraged to mimic the synthetic ground truth. We find the synthetic latent code regularization could stabilize the deformation encoder training. Following Tov et al. [98], the

encoders predict the offsets of the mean modulations of the corresponding mapping function $\boldsymbol{\beta}_0^{DDF}, \boldsymbol{\gamma}_0^{DDF}$ for better initialization. The overall training objectives are: 1) *Image Reconstruction Loss*: We utilize the pixel-wise $\mathcal{L}_2$ as well as the LPIPS loss $\mathcal{L}_{\text{LPIPS}}$ [171] as the image reconstruction supervisions:

$$\begin{aligned}\mathcal{L}_{image} = &\mathcal{L}_2(I, G(E(I, \theta_G), \boldsymbol{\xi}_I)) \\ &+ \mathcal{L}_{\text{LPIPS}}(I, G(E(I, \theta_G), \boldsymbol{\xi}_I)),\end{aligned} \tag{3.12}$$

2) *Latent Codes Regularization*: We regularize the encoded latent codes to match the pseudo ground truth latent codes distribution:

$$\mathcal{L}_{latent} = \mathcal{L}_2((\boldsymbol{\beta}_{\boldsymbol{z}}, \boldsymbol{\gamma}_{\boldsymbol{z}}), E(I_{\boldsymbol{z}}, \theta_D)). \tag{3.13}$$

Results. Here we show the texture transfer results over real images in Fig. 3.15. As can be seen, our hybrid inversion method could faithfully reconstruct the given real images without affecting the view synthesis ability of NeRF. Furthermore, with *DDF*, accurate 3D-consistent segmentation transfer and faithful texture transfer become possible, which is beyond the reach of existing 2D methods. We further show a mask-guided texture transfer applied over real cases in Fig. 3.16, which shows the potential of our method over real-world applications beyond basic texture transfer and segmentation labeling.

3.3 Conclusion

In this chapter, we propose to leverage a pre-trained NeRF-based GAN, π-GAN in our case, to build dense correspondence between NeRF representations of different objects within the same category. The key insight is that the pre-trained GAN possesses three important properties that can help alleviate the challenges of this task, namely 1) instance-specific latent codes that holistically capture the global structure of different NeRFs, 2) geometric-aware generator features that reflect local geometric details of different NeRFs, and 3) the manifold of NeRFs that serves as a source of infinite NeRF samples. Based on the three properties, we respectively propose a generalizable model, referred to as Dual Deformation Field, a learning objective based on generator features that approximate geometric distances in feature space, and finally an effective curriculum training strategy that

feeds samples with growing complexity. To the best of our knowledge, this is the first method that tries to establish dense correspondence across NeRF representations. Our experiments demonstrate that dense correspondences between NeRFs learned from our framework are accurate, smooth, and robust, making them applicable in various downstream applications. Moreover, since *DDF* directly adopts the latent code of GAN as the deformation condition, we believe our method could benefit from later progress in this field and produce more natural-looking results when applied to real images.

Though good performance is acheived, *DDF* still fails to support real-case inversion and editing. In the next chapter, we will delve into the 3D GAN inversion problem, which projects the monocular input into the editable 3D latent space.

Chapter 4

E3DGE: Self-Supervised Geometry-Aware Encoder for Style-Based 3D GAN Inversion

This work aims to devise an effective and generic approach for encoder-based 3D Generative Adversarial Network (GAN) inversion. Specifically, we focus on the reconstruction of 3D faces, requiring just a single 2D face image as the input. In the inversion process, we wish to map a given image to the latent space and obtain an editable latent code with an encoder. The latent code will be further fed to a generator to reconstruct the corresponding 3D shape with high-quality shape and texture. Besides inversion, we aim to further develop an approach to synthesize 3D view-consistent editing results, e.g., driving a neutral expression to smiling, by altering the estimated latent code.

GAN inversion [27] has been extensively studied for 2D images but remains underexplored in the 3D world. Inversion can be achieved via optimization [28, 95, 101], which typically provides a precise image-to-latent mapping but can be time-consuming, or encoder-based techniques [29, 97, 98], which explicitly learn an encoding network that encodes an image into the latent space. Compared to optimization-based methods, encoder-based techniques enjoy the benefits of faster

The work in this chapter has been published in [7].
Code is publicly available at `https://github.com/NIRVANALAN/E3DGE`

inversion and better visual quality [29] when trained with a carefully deisgned framework. In this study, we extend the notion of encoder-based inversion from 2D images to 3D shapes.

Increasing the additional dimension makes inversion more challenging beyond the goal of reconstructing an editable shape with detail preservation. In particular, **1)** Recovering 3D shapes from 2D images is an ill-posed problem, where innumerable compositions of shape and texture could generate identical rendering results. 3D supervisions are crucial to alleviate the ambiguity of shape inversion from images. Though high-quality 2D datasets are easily accessible, owing to the expensive cost of scans there is currently a lack of large-scale labeled 3D datasets. **2)** The global latent code, due to its compact and low-dimensional nature, only captures the coarse shape and texture information. Without high-frequency spatial details, high-fidelity outputs cannot be generated. **3)** Compared with 2D inversion methods where the editing view mostly aligns with the *input view*, in 3D editing the editing results are expected to perform well over the *novel views* with large pose variations. Therefore, 3D GAN inversion is a non-trivial task and could not be achieved by directly applying existing approaches.

To this end, a novel **E**ncoder-based **3D G**AN inv**E**rsion framework, E3DGE is proposed, which addresses the aforementioned three challenges. Our framework has three novel components with a delicate model design. Specifically:

Learning Inversion with Self-supervised Learning - The first component focuses on the training of the inversion encoder. To address the shape collapse of single-view 3D reconstruction without external 3D datasets, the generator of a 3D GAN model is retrofitted to provide us with diverse pseudo-training samples, which can then be used to train our inversion encoder in a self-supervised manner. Specifically, the 3D shapes from the latent space $\mathcal{W}$ is generated of a 3D GAN, and then render diverse 2D views from each 3D shape given different camera poses. In this way, many pseudo-2D-3D pairs can be generated together with the corresponding latent codes. Since the pseudo pairs are generated from a smooth latent space that learns to approximate a natural shape manifold, they serve as effective surrogate data to train the encoder, avoiding potential shape collapse.

Local Features for High-Fidelity Inversion - The second component learns to reconstruct accurate texture details. Our novelty here is to leverage local features

to enhance the representation capacity, beyond just the global latent code generated by the inversion encoder. Specifically, in addition to inferring an editable global latent code to represent the overall shape of the face, an hour-glass model is further devised to extract local features over the residuals details that the global latent code fails to capture. The local features, with proper projection to the 3D space, serve as conditions to modulate the 2D image rendering. Through this effective learning scheme, the benefits of both global and local priors are married and achieve high-fidelity reconstruction.

Synthesizing View-consistent Edited Output - The third component addresses the problem of novel view synthesis, a problem unique to 3D shape editing. Specifically, though high-fidelity reconstruction is achieved through the aforementioned designs, the local residual features may not fully align with the scene when being semantically edited. Moreover, the occlusion issue further degrades the fusion performance when rendering from novel views with large pose variations. To this end, a 2D-3D hybrid alignment module is proposed for high-quality editing. Specifically, a 2D alignment module and a 3D projection scheme are introduced to jointly align the local features with edited images and inpaint occluded local features in novel view synthesis.

Extensive experiments show that our method achieves 3D GAN inversion with plausible shapes and high-fidelity image reconstruction without affecting editability. Owing to the self-supervised training strategy with delicate global-local design, our approach performs well on real-world 2D and 3D benchmarks without resorting to any real-world 3D dataset for training.

Besides studying 3D GAN inversion on MLP-based StyleSDF, this chapter further renovates E3DGE as a generic 3D GAN inversion framework with the following key improvements: 1) *Representation-wise*, the 3D inversion over tri-plane-based GAN is explored, characterized by EG3D [1], the state-of-the-art GAN-based 3D generative model. 2) *Modality-wise*, our proposed method is adapted over video input, facilitating 4D monocular reconstruction. 3) *Application-wise*, a new practical application of our method: 3D toonification. is included.

Since EG3D adopts a new 3D representation, it poses new challenges for 3D GAN inversion: how to design an efficient and high-fidelity encoder-based inversion

pipeline that leverages a triplane-based method. An observation in our experiment is that, though with superior representation capacity, the tri-plane is more sensitive to inaccurate pose during inference. To tackle this challenge, the following component for EG3D-based E3DGE is proposed: **Pose estimation for domain adaptation** - This newly proposed component addresses the problem of noisy poses of 3D GAN inversion over real images, which EG3D is especially sensitive to. Specifically, though the aforementioned designs could achieve high-quality inversion, the assumption that an accurate input pose is available at test time is relied. However, though directly leveraging pseudo samples with ground truth pose for training, the real inputs are inclined to pair with noisy pose labels, which intensifies the domain gap between training and test. To bridge the gap, the predicted pose during training is used with ground-truth poses supervising the pose estimator. Besides, an option is provided to finetune the predicted pose to boost the performance. Additionally, we conduct extensive experiments to assess the new design elements of the extended EG3D-E3DGE. This includes both qualitative and quantitative evaluations of the proposed pose estimator, hybrid fine-tuning methods, and comparisons with relevant baselines. We also explore new applications and extend inversion techniques to a broader range of categories.

To summarize, our main contributions are as follows:

- The learning of an encoder-based 3D GAN inversion framework for high-quality shape and texture inversion is proposed. We show that with careful design, samples synthesized by a GAN could serve as proxy data for self-supervised training in inversion.
- An effective framework that uses local features to complement the global latent code for high-fidelity inversion.
- An effective approach to synthesize view-consistent output with a 2D-3D hybrid alignment.
- The joint training of a pose estimator to address the pose domain gap during inversion.
- The proposed method is evaluated on two representative 3D GAN models, showing its generalizability to different backbones.

4.1 Methodology

4.1.1 Preliminaries

Since recent 3D-aware image generative models are based on neural implicit representations, especially NeRF [2], here we briefly introduce the NeRF-based 3D representation and also hybrid 3D-aware generation based on EG3D/StyleSDF for clarification.

NeRF-based 3D Representation. NeRF [2] proposed an implicit 3D representation for novel view synthesis. Specifically, NeRF defines a scene as $\{\boldsymbol{c}, \sigma\} = F_\Phi(\vec{x}, \boldsymbol{v})$, where $\vec{x}$ is the query point, $\boldsymbol{v}$ is the viewing direction from camera origin to $\vec{x}$, $\boldsymbol{c}$ is the emitted radiance (RGB value), σ is the volume density. To query the RGB value $C(\boldsymbol{r})$ of a point on a ray $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{v}$ shoot from the 3D coordinate origin $\boldsymbol{o}$, we have the volume rendering formulation,

$$C(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\boldsymbol{c}(\boldsymbol{r}(t), \boldsymbol{v})dt, \tag{4.1}$$

where $T(t) = \exp(-\int_{t_n}^{t} \sigma(\boldsymbol{r}(s))ds)$ is the accumulated transmittance along the ray $\boldsymbol{r}$ from t_n to t. t_n and t_f denote the near and far bounds.

Hybrid 3D-aware Generation. To achieve high-resolution novel view synthesis, hybrid 3D-aware generators [1, 8, 22, 52] are proposed. It is typically a cascade model $G = G_\mathbf{1} \circ G_\mathbf{0}$ composed of a NeRF-based renderer $G_\mathbf{0}$ [21] and a 2D super-resolution network $G_\mathbf{1}$, as shown in Fig. 4.1. Both G_0 and G_1 follow the style-based architecture [86, 172] to accept a latent code $\mathbf{w}$ to control the style of the generated object. During generation, G_0 captures the underlying geometry with the full control of $\mathbf{w}$ and camera pose $\boldsymbol{\xi}$, and renders a low resolution image $x_\mathbf{0}$ and an intermediate feature map $\mathbf{F}$. Then, $G_\mathbf{1}$ further upsamples $\mathbf{F}$ to obtain a high-resolution image x with added high-frequency details. Among them, StyleSDF [8] adopts NeRF-based MLP as $G_\mathbf{0}$ for 3D-aware high-quality surface synthesis. In comparison, EG3D [1] introduces axis-aligned plane [173] as $G_\mathbf{0}$ and achieves state of the art performance on several benchmarks with faster rendering efficiency and sharper geometry details compared against previous work [21] and StyleSDF. EG3D also enjoys the flexible style control for semantic editing as in StyleGAN [86]. Therefore, in this work we mainly explore EG3D as the base model for GAN

inversion study (Section 4.1.2). Our method is not limited to EG3D and could be easily extended to other style-based 3D GAN variations [8, 22]. Please refer to Sec. 4.1.7 for the technical details of StyleSDF-based E3DGE study.

4.1.2 E3DGE with EG3D Backbone

An effective 3D GAN inversion shall be capable of **1)** reconstructing plausible 3D shape given single-view input, **2)** maintaining high-fidelity texture, and **3)** allowing view-consistent semantic edits. To achieve these goals, we propose the E3DGE framework with three novel components: In Sec. 4.1.3, we leverage 3D GAN to generate pseudo 2D-3D paired samples for 3D supervisions, and train an inversion encoder E_0 to estimate the latent of plausible 3D shapes from a 2D image; In Sec. 4.1.4, we train a local encoder E_1 to extract pixel-aligned features to enrich texture details for high-fidelity inversion; In Sec. 4.1.5 introduces a hybrid alignment module for view-consistent semantic editing; Finally, in Sec. 4.1.6 we propose to jointly estimate the input camera pose and fine-tune the estimated code to alleviate the domain gap for better performance on the real-world inversion.

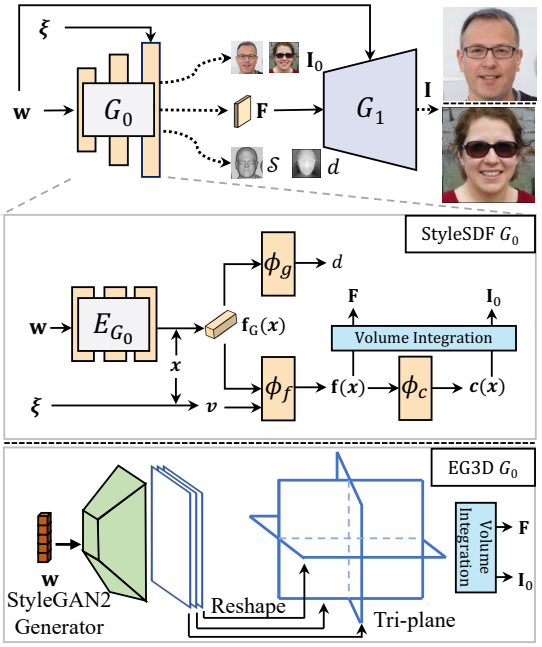


FIGURE 4.1: **StyleSDF and EG3D.** Given a sampled latent code $\mathbf{w}$ and a camera pose $\boldsymbol{\xi}$, StyleSDF [8] generates object SDF d to depict the shape with the corresponding image x, while EG3D [1] adopts density σ as the shape descriptor. Both methods adopt a hybrid synthesis pipeline, where a low resolution image x_0 is first synthesized, and further up-sampled to high-res image x.

4.1.3 Self-supervised Inversion Learning

In this section, we propose to mitigate the lack of large-scale high-quality 2D-3D paired datasets by retrofitting pre-trained 3D GANs to provide pseudo samples for training our inversion encoder. We demonstrate the model trained from pseudo samples can rival and even outperform the methods learned from real data on the 3D GAN inversion task. We detail the process as follows.

Global Encoder for 3D GAN Inversion. With the style-based G, we build our encoder $E_{\mathbf{0}}$ based on pSp [97] for inversion. Given a target image x, $E_{\mathbf{0}}$ predicts its latent code $\hat{\mathbf{w}} = E_{\mathbf{0}}(x)$. Given the corresponding camera pose $\boldsymbol{\xi}$, the reconstructed image is obtained by $\tilde{x} = G(\hat{\mathbf{w}}, \boldsymbol{\xi})$ to approximate x. In addition, we would like its 3D shape predicted by $G_{\mathbf{0}}$ to be plausible enough.

Distill 3D GANs as 3D Supervisions. Different compositions of shape and texture could lead to identical 2D-rendered images. 3D supervision is needed to alleviate such shape-texture ambiguity. In the lack of large-scale high-quality 2D-3D paired samples, we formulate GAN Inversion as a *self-training* task, where samples synthesized from itself are leveraged to boost the reconstruction fidelity in both 2D and 3D domains. As shown in Fig. 4.1 and Fig. 4.2, we synthesize paired 3D shape information $\mathcal{S}$ and 2D image x from latent code $\mathbf{w}$ and camera pose $\boldsymbol{\xi}$ using G to train $E_{\mathbf{0}}$. To extract the 3D shape information $\mathcal{S}$ of each synthetic shape, we first sample a point set $\mathcal{P} = \{\mathcal{P}_{\mathcal{O}}, \mathcal{P}_{\mathcal{F}}\}$ where $\mathcal{P}_{\mathcal{O}}$ and $\mathcal{P}_{\mathcal{F}}$ contain points sampled from the surface and around the surface, respectively. Then, we calculate the geometry descriptor $\boldsymbol{\sigma}_i$ and $\mathbf{DEPTH}_i$ for each 3D point $\vec{x}_i \in \mathcal{P}$, and $\mathcal{S}$ is defined as the set of geometry descriptors of all 3D point in $\mathcal{P}$:

$$\begin{aligned} \mathcal{S} = \{\{\boldsymbol{\sigma}_i, \mathbf{DEPTH}_i\}_{i=1}^{|\mathcal{P}|} \mid \\ \vec{x}_i \in \mathcal{P}, \boldsymbol{\sigma}_i = G_{\mathbf{0}}(\mathbf{w}, \vec{x}_i), \mathbf{DEPTH}_i = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))dt\}, \end{aligned} \tag{4.2}$$

where $\boldsymbol{\sigma}_i$ is the density of point $\vec{x}_i$, $\mathbf{DEPTH}_i$ is the depth of point $\vec{x}_i$ under the given view direction $\boldsymbol{\xi}$ and σ is the density of the point along the ray. Note our method is not limited to the triplane-based shape representation and can be easily extended to SDF-based methods, as detailed in Sec. 4.1.7. Moreover, given different camera poses, we can generate a diverse 2D-3D dataset to help alleviate the shape-texture ambiguity, i.e., for each shape $\mathcal{S}$, various images $x =$

$G(\mathbf{w}, \boldsymbol{\xi})$ can be rendered by randomly sampling $\boldsymbol{\xi}$ from a predefined pose distribution $p_{\boldsymbol{\xi}}$. Finally, we define $\mathcal{X} = \{\mathcal{S}, \boldsymbol{\xi}, x\}$ as a training sample for $E_{\mathbf{0}}$.

3D GAN-Supervised Training. As shown in Fig. 4.2 (a), given a training sample $\mathcal{X}$, the forward process is represented as:

$$\begin{aligned} \hat{\mathbf{w}} &= E_{\mathbf{0}}(x) \\ \{\tilde{x}, \hat{\mathcal{S}}\} &= G(\hat{\mathbf{w}}, \boldsymbol{\xi}, \mathcal{P}) \end{aligned} \tag{4.3}$$

where $\hat{\mathbf{w}}$ is the estimated latent code and $\hat{\mathcal{S}} = \{\{\hat{\boldsymbol{\sigma}}_i, \hat{\mathbf{DEPTH}}_i\}_{i=1}^{|\mathcal{P}|} \mid \vec{x}_i \in \mathcal{P}\}$ is the estimated 3D shape information conditioned on $\tilde{\mathbf{w}}$ and $\mathcal{P}$.

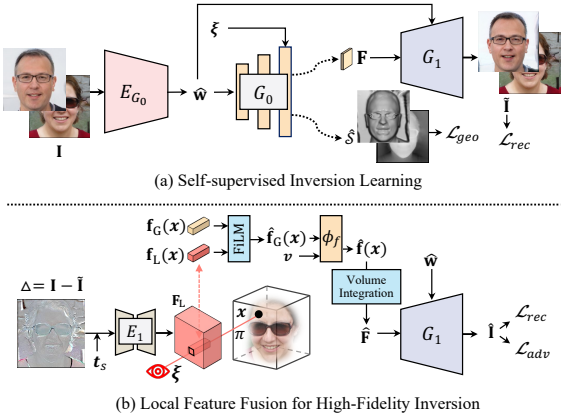


FIGURE 4.2: **E3DGE for 3D GAN inversion**. (a) We augment the training of the encoder $E_{\mathbf{0}}$ with 3D supervision $\mathcal{L}_{geo}$ for plausible 3D shape prediction. (b) We augment the representation capacity of the global latent code $\hat{\mathbf{w}}$ with local point-dependent latent feature $\mathbf{f}_{\mathrm{L}}$ for high-fidelity texture reconstruction. Both StyleSDF and EG3D samples are shown here.

To achieve 3D supervision, for all points we would like the estimated $\hat{\mathcal{S}}$ to approximate the ground truth $\mathcal{S}$. Specifically, we supervise both the predicted density $\hat{\boldsymbol{\sigma}}$ as well as the depth $\hat{\mathbf{DEPTH}}$ [174] for surface points $\vec{x}_i \in \mathcal{P}_{\mathcal{O}}$ and supervise the predicted density for the remaining points $\vec{x}_i \in \mathcal{P}_{\mathcal{F}}$, leading to the geometry loss:

$$\mathcal{L}_{geo}^{\mathcal{O}} = \mathbb{E}_{\mathcal{X}}\left[\frac{1}{|\mathcal{P}_{\mathcal{O}}|}\sum_{i=1}^{|\mathcal{P}_{\mathcal{O}}|} \lambda_{g_1}|\hat{\boldsymbol{\sigma}}_i - \boldsymbol{\sigma}_i| + \lambda_{g_2}\|\hat{\mathbf{DEPTH}}_i - \mathbf{DEPTH}_i\|_1\right] \tag{4.4}$$

$$\mathcal{L}_{geo}^{\mathcal{F}} = \mathbb{E}_{\mathcal{X}}\left[\frac{1}{|\mathcal{P}_{\mathcal{F}}|}\sum_{i=1}^{|\mathcal{P}_{\mathcal{F}}|} \lambda_{g_3}|\hat{\boldsymbol{\sigma}}_i - \boldsymbol{\sigma}_i|\right] \tag{4.5}$$

$$\mathcal{L}_{geo} = \mathcal{L}_{geo}^{\mathcal{O}} + \mathcal{L}_{geo}^{\mathcal{F}}, \tag{4.6}$$

where λs are loss weights and supervision of depth $\hat{\mathbf{DEPTH}}_i$ is only imposed for points over the surface. We also impose code reconstruction loss $\mathcal{L}_{code} = \|\hat{\mathbf{w}} - \mathbf{w}\|_2$ to regularize the learning and 2D supervisions $\mathcal{L}_{rec}$ to minimize the reconstruction error between $\tilde{x}$ and x as in pSp [97]. The overall loss is $\mathcal{L} = \mathcal{L}_{\text{geo}} + \mathcal{L}_{\text{code}} + \mathcal{L}_{\text{rec}}$, which is detailed in Sec. 4.1.8.

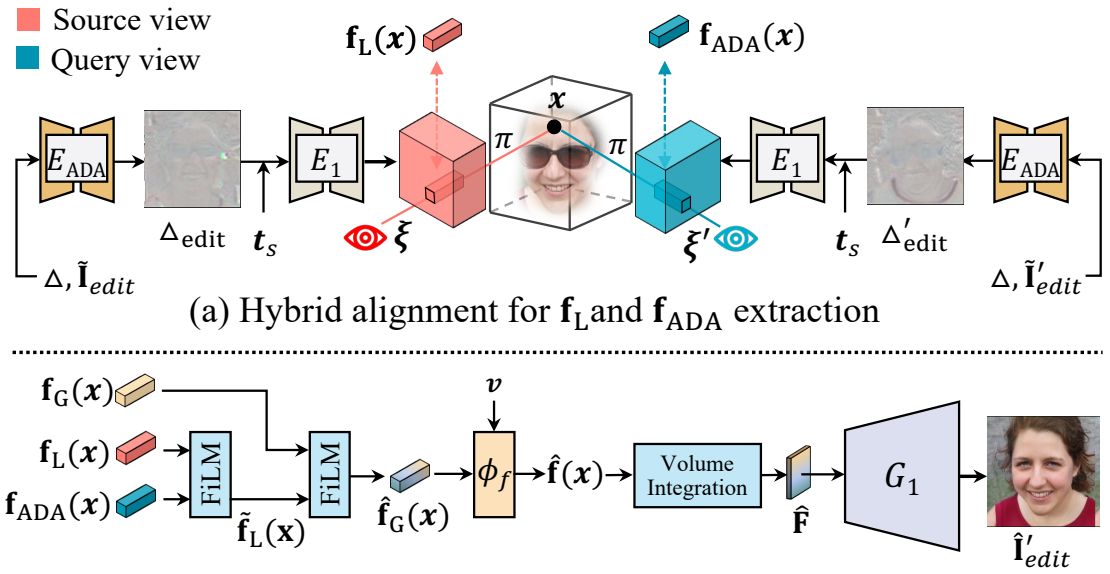


FIGURE 4.3: **Hybrid alignment for high-quality editing.** Given code prediction $\hat{\mathbf{w}}$ from encoder $E_{\mathbf{0}}$ pre-trained in stage-I, we aim to generate high-quality view synthesis over the edited code $\hat{\mathbf{w}}_{\mathrm{edit}}$. In (a), the local details Δ along with the target edited image x'_{edit} and depth map $\boldsymbol{t}_s(\hat{\mathbf{w}}, \boldsymbol{\xi})$ are sent to pre-trained E_{ADA} to predict aligned residual Δ'_{edit}. The original aligned residual Δ along with the 2D auxiliary residual Δ'_{edit} are processed by $E_{\mathbf{1}}$ to recover latent maps $\mathbf{F}_{\mathrm{L}}$ and $\mathbf{F}_{\mathrm{ADA}}$ for later fusion. In (b), the extracted features $\mathbf{f}_{\mathrm{L}}(\vec{x})$ and $\mathbf{f}_{\mathrm{ADA}}(\vec{x})$ are first fused together with a FiLM layer, and the fused result $\hat{\mathbf{f}}_{\mathrm{L}}(\vec{x})$ further serve as conditions to modulate the global feature $\mathbf{f}_{\mathrm{G}}(\vec{x})$. The final modulated feature $\hat{\mathbf{f}}(\vec{x})$ contains complete information, globally and locally. The volume integrated $\hat{\mathbf{F}}$ is sent to $G_{\mathbf{1}}$ for high-resolution synthesis.

4.1.4 Local Features for High-Fidelity Inversion

To facilitate the discussion in the following sections, we first take a look at the details of EG3D. The unique design of EG3D lies in its 3D renderer $G_{\mathbf{0}}$: after a StyleGAN2 generator, the last layer of the synthesized feature maps are reshaped into three orthogonal planes, e.g., tri-plane, where the queried features over three planes are volume rendered into feature map $\mathbf{F}$ and view-consistent image x_0. Specifically, $E_{G_{\mathbf{0}}}$ extracts a global feature $\mathbf{f}_{\mathrm{G}}(\vec{x}) = E_{G_{\mathbf{0}}}(\vec{x}, \mathbf{w})$. Based on $\mathbf{f}_{\mathrm{G}}$, ϕ_g and ϕ_f compute density $\sigma(\vec{x}) = \phi_g(\mathbf{f}_{\mathrm{G}}(\vec{x}))$ and the last-layer feature $\mathbf{f}(\vec{x}, \boldsymbol{v}) = \phi_f(\mathbf{f}_{\mathrm{G}}(\vec{x}), \boldsymbol{v})$ of $G_{\mathbf{0}}$, respectively. $\mathbf{f}$ could be directly transformed to color $\boldsymbol{c}(\vec{x}, \boldsymbol{v}) = \phi_c(\mathbf{f}(\vec{x}, \boldsymbol{v}))$ or being volume integrated to $\mathbf{F}$ and sent to $G_{\mathbf{1}}$ for high resolution synthesis. For simplicity, we omit $\boldsymbol{v}$ in the following. The volume rendering process is the same as StyleSDF, which is depicted in the middle of Fig. 4.1. The main

difference of EG3D-based representation is that $\mathbf{f}_{\mathrm{G}}$ is decoded from tri-plane, while StyleSDF decodes the global feature from a stack of MLP layers.

Local Feature for Detailed Textures. The global latent code $\hat{\mathbf{w}}$ is a compact representation of the predicted scene. However, previous works [29, 99] have validated that a low-dimensional latent code discards high-frequency spatial details and fails to reconstruct high-fidelity outputs. This phenomenon becomes more severe when lifting the 2D image to a 3D scene, which contains exponentially more information. Inspired by recent progress in few-shot 3D reconstruction [3, 19, 175–179], we propose to make up for the lost information by introducing pixel-aligned (local) features. As shown in Fig. 4.2 (b), rather than conditioning all 3D points with the same latent code $\hat{\mathbf{w}}$, we augment the representation capacity with local latent codes $\mathbf{f}_{\mathrm{L}}$ that is dependent on each point $\vec{x}$. We introduce a local hourglass [180] encoder $E_{\mathbf{1}}$ to predict a residual feature map $\mathbf{F}_{\mathrm{L}}$ based on the reconstruction residue $\Delta = x - \tilde{x}$,

$$\mathbf{F}_{\mathrm{L}} = E_{\mathbf{1}}(\Delta, \boldsymbol{t}_s(\hat{\mathbf{w}}, \boldsymbol{\xi})), \tag{4.7}$$

where $\boldsymbol{t}_s(\hat{\mathbf{w}}, \boldsymbol{\xi})$ is the depth map of the scene derived from the predicted density σ to serve as 3D context information. Then, the local latent code of a point $\vec{x}$ is its corresponding value in $\mathbf{F}_{\mathrm{L}}$:

$$\mathbf{f}_{\mathrm{L}}(\vec{x}) = \mathbf{F}_{\mathrm{L}}(\pi(\vec{x})) \oplus \mathbf{PE}(\vec{x}), \tag{4.8}$$

where π maps the 3D point $\vec{x}$ to its corresponding pixel coordinate on 2D feature map $\mathbf{F}_{\mathrm{L}}$. Since in 3D scenes, points along a ray will be projected to the same coordinate on the 2D plane, to differentiate these points, we additionally concatenate their positional encoding $\mathbf{PE}(\vec{x})$ [2] in Eq. (4.8). In this way, the local feature $\mathbf{f}_{\mathrm{L}}$ only encodes the residual information at the projected position $\pi(\vec{x})$ but is also capable of determining where the residual information lies in the 3D scene, as well as inpainting the occluded areas along the ray.

Finally, we fuse the local latent code $\mathbf{f}_{\mathrm{L}}(\vec{x})$ with the global latent code $\mathbf{f}_{\mathrm{G}}(\vec{x}) = E_{G_{\mathbf{0}}}(\vec{x}, \hat{\mathbf{w}})$ to supplement the missing high-frequency details. Specifically, the feature fusion is based on Feature-wise Linear Modulation (FiLM) [181]. As shown in Fig. 4.2, $\mathbf{f}_{\mathrm{L}}(\vec{x})$ is fed into two MLP layers to obtain the scale and bias modulation

parameters $\mathbf{f}_{\mathrm{L}}^{\boldsymbol{\gamma}}(\vec{x})$ and $\mathbf{f}_{\mathrm{L}}^{\boldsymbol{\beta}}(\vec{x})$. Then we modulate $\mathbf{f}_{\mathrm{G}}(\vec{x})$ with FiLM

$$\begin{aligned}\hat{\mathbf{f}}_{\mathrm{G}}(\vec{x}) =& \mathrm{FiLM}(\mathbf{f}_{\mathrm{G}}(\vec{x}), \mathbf{f}_{\mathrm{L}}(\vec{x})) \\ =& \mathbf{f}_{\mathrm{L}}^{\boldsymbol{\gamma}}(\vec{x}) \cdot \mathbf{f}_{\mathrm{G}}(\vec{x}) + \mathbf{f}_{\mathrm{L}}^{\boldsymbol{\beta}}(\vec{x}).\end{aligned} \tag{4.9}$$

The fused $\hat{\mathbf{f}}_{\mathrm{G}}(\vec{x})$ is volume integrated to $\hat{\mathbf{F}}$ and the final high-fidelity reconstructed image is obtained as $\hat{x} = G_{\mathbf{1}}(\hat{\mathbf{F}})$.

Note that through point projection π, the reconstruction with local prior is not limited to the original view, and naturally works for novel views. However, for views with severe occlusions or additional editing, the residual features may not fully align with the scene, leading to a failed feature fusion. We will address this issue in the next subsection with our hybrid feature alignment.

4.1.5 Hybrid Alignment for High-Quality Editing

Though we achieve high-fidelity reconstruction with the aforementioned designs, there is a trade-off between the *input view* reconstruction quality and *novel view* editing performance. We first analyze the reasons behind and propose a hybrid alignment module to address this issue.

Reconstruction Editing Trade-off. Given an input image x with paired reconstruction $\tilde{x}$ and residual map Δ extracted from the input view $\boldsymbol{\xi}$ with the aforementioned method, the reconstruction performance trade-offs the editing performance due to the following two reasons. First, at test time when the input image is edited $\tilde{x}_{\mathrm{edit}}$ or query view $\boldsymbol{\xi}' \neq \boldsymbol{\xi}$, the residual map no longer aligns and is likely to result in wrong predictions. Second, if we supervise the models to reconstruct the input itself, the learned features are *regressive* rather than *generative* since all prediction areas are visible in the inputs. With these above-mentioned challenges, though the model could yield perfect reconstruction at training, it would result in noticeable performance degradation when rendering from novel views at test time.

Hybrid Alignment for High-Quality Editing. To address the first challenge, we propose to infer aligned features with a 2D-3D hybrid alignment. Specifically, given edited latent code $\hat{\mathbf{w}}_{\mathrm{edit}}$, the initial novel-view edited image $\tilde{x}'_{\mathrm{edit}} = G_{\mathbf{0}}(\hat{\mathbf{w}}_{\mathrm{edit}}, \boldsymbol{\xi}')$ is misaligned with Δ. Inspired by HFGI [29], we leverage a 2D alignment module E_{ADA} to address the misalignment. As shown in Fig. 4.3 (a), we first

obtain $\Delta_{\text{edit}} = E_{\text{ADA}}(\Delta, G_{\mathbf{0}}(\hat{\mathbf{w}}_{\text{edit}}, \boldsymbol{\xi}))$, transform it to residual feature map $\mathbf{F}_{\text{L}}^{\text{edit}}$ via Eq. (4.7) and retrieve the view-consistent 3D local feature $\mathbf{f}_{\text{L}}$ via Eq. (4.8). However, to render the high-quality edited image $\hat{x}'_{edit}$ from novel view $\boldsymbol{\xi}'$, $\mathbf{F}_{\text{L}}^{\text{edit}}$ might still suffer from occlusion due to large pose variations. To the end, we propose a hybrid alignment to further refine $\mathbf{F}_{\text{L}}^{\text{edit}}$ with the 2D aligned feature from E_{ADA}. Specifically, we align a 2D residue $\Delta'_{\text{edit}} = E_{\text{ADA}}(\Delta, \tilde{x}'_{\text{edit}})$ and retrieve its corresponding $\mathbf{f}_{\text{ADA}}$ with $E_{\mathbf{1}}$, which fills the occlusion in a 2D manner but lacks 3D consistency. To marry the best of both, as shown in Fig 4.3 (b), we modulate $\mathbf{f}_{\text{L}}$ with $\mathbf{f}_{\text{ADA}}$,

$$\tilde{\mathbf{f}}_{\text{L}}(\vec{x}) = \text{FiLM}(\mathbf{f}_{\text{L}}(\vec{x}), \mathbf{f}_{\text{ADA}}(\vec{x})), \tag{4.10}$$

and further fuse $\tilde{\mathbf{f}}_{\text{L}}$ with $\mathbf{f}_{\text{G}}(\vec{x})$ for final prediction,

$$\hat{\mathbf{f}}(\vec{x}) = \text{FiLM}(\mathbf{f}_{\text{G}}(\vec{x}), \tilde{\mathbf{f}}_{\text{L}}(\vec{x})), \tag{4.11}$$

where $\hat{\mathbf{f}}(\vec{x})$ is then integrated to $\hat{\mathbf{F}}$ for rendering the final novel-view edited image $\hat{x}'_{edit} = G_{\mathbf{1}}(\hat{\mathbf{F}})$.

Novel View Training for Coherent View Synthesis. To address the second challenge and enforce the model to learn generative features, during training, we sample two views $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ for each style code $\mathbf{w}$, and render the corresponding images $x^{\boldsymbol{\xi}_1}$ and $x^{\boldsymbol{\xi}_2}$. Then, we train the models to reconstruct plausible novel views, i.e., $G(E(x^{\boldsymbol{\xi}_1}), \boldsymbol{\xi}_2) \approx x^{\boldsymbol{\xi}_2}$ and $G(E(x^{\boldsymbol{\xi}_2}), \boldsymbol{\xi}_1) \approx x^{\boldsymbol{\xi}_1}$. This training strategy facilitates a high-quality view synthesis over edited scenes.

4.1.6 Pose Estimation for Domain Adaptation

Though the aforementioned strategy could effectively alleviate information loss introduced by the capacity limitation of global latent code, in reality, we observe $E_{\mathbf{1}}$ is likely to fail when the input pose $\boldsymbol{\xi}$ is noisy. This may occur since we directly adopt the ground truth pose $\boldsymbol{\xi} \in \mathcal{X}$ for training, thus the residual Δ is calculated over the reconstruction $\tilde{x}$ with the perfect pose. However, the pose of real-world images estimated from COLMAP [182, 183] or pre-trained model [184] tends to be noisy. In this way, the encoder $E_{\mathbf{1}}$ could not handle the residual Δ caused by slight pose misalignment.

To alleviate this issue, we propose to jointly train a pose estimator $E_{\boldsymbol{\xi}}$ over synthetic samples $\mathcal{X}$ and use the predicted pose $\tilde{\boldsymbol{\xi}}$ to calculate global reconstruction $\tilde{x}_{\tilde{\boldsymbol{\xi}}} = G(\hat{\mathbf{w}}, \tilde{\boldsymbol{\xi}})$. The residual is calculated with $\Delta = x - \tilde{x}_{\tilde{\boldsymbol{\xi}}}$, and the remaining operations defined in Eqs. (4.8) and (4.9) stay the same. Apart from pose estimation loss $\mathcal{L}_{pose} = ||\tilde{\boldsymbol{\xi}} - \boldsymbol{\xi}||_2$, all the aforementioned loss functions in Sec. 4.1.3 are also imposed. We demonstrate in the experiment that this operation is crucial for high-fidelity inversion over EG3D.

Although our pose estimator could alleviate the domain gap between training and testing, we observe that it still cannot fully derive the camera pose accurately enough. Moreover, in some scenarios, the user trades off better fidelity with a reasonable time cost. Therefore, we propose to further finetune the estimated pose for a few steps in the test time, where the estimated identity code $\hat{\mathbf{w}}$ could also be finetuned together. Better quality is achieved via both test-time optimization techniques, as validated in Tab. 4.1 and Fig. 4.9.

4.1.7 E3DGE-StyleSDF Backbone

E3DGE also supports adopting StyleSDF [8] as the base inversion model. Different from EG3D, as shown in the middle of Fig. 4.1, StyleSDF $G_{\mathbf{0}}$ can be further divided into four parts: a 8-layer MLP encoder $E_{G_{\mathbf{0}}}$, a SDF decoder ϕ_g, a feature decoder ϕ_f and a color decoder ϕ_c. To train E3DGE on StyleSDF, as shown in the upper half in Fig 4.1 and Fig. 4.2(a), we synthesize paired 3D shape information $\mathcal{S}$ and 2D image x from latent code $\mathbf{w}$ and camera pose $\boldsymbol{\xi}$ using G to train $E_{\mathbf{0}}$. To extract the 3D shape information $\mathcal{S}$ of each synthetic shape, we first sample a point set $\mathcal{P} = \{\mathcal{P}_{\mathcal{O}}, \mathcal{P}_{\mathcal{F}}\}$ where $\mathcal{P}_{\mathcal{O}}$ and $\mathcal{P}_{\mathcal{F}}$ contain points sampled from the surface and around the surface, respectively. Then, we calculate the geometry descriptor d_i and $\vec{n}_i$ for each 3D point $\vec{x}_i \in \mathcal{P}$, and $\mathcal{S}$ is defined as the set of geometry descriptors of all 3D point in $\mathcal{P}$:

$$\begin{aligned} \mathcal{S} = \{\{d_i, \vec{n}_i\}_{i=1}^{|\mathcal{P}|} \mid \\ \vec{x}_i \in \mathcal{P}, d_i = G_{\mathbf{0}}(\mathbf{w}, \vec{x}_i), \vec{n}_i = \nabla_{\vec{x}_i} d_i\}, \end{aligned} \tag{4.12}$$

where d_i is the distance from $\vec{x}_i$ to the shape surface and $\vec{n}_i$ is the surface normal defined by the gradient of the distance w.r.t. $\vec{x}_i$. Note our method is not limited

to the SDF-based shape representation and can be easily extended to volumetric-based methods [1, 21, 185].

To achieve 3D supervision, we would like the estimated $\hat{\mathcal{S}}$ to approximate the ground truth $\mathcal{S}$. Specifically, for points over the surface, their distances and normal are both considered while for points around the surface, we only supervise their distance following [14, 178], leading to geometry loss:

$$\mathcal{L}_{geo}^{\mathcal{O}} = \mathbb{E}_{\mathcal{X}}\left[\frac{1}{|\mathcal{P}_{\mathcal{O}}|}\sum_{i=1}^{|\mathcal{P}_{\mathcal{O}}|}\lambda_{g_1}|\hat{d}_i| + \lambda_{g_2}\|\hat{\vec{n}}_i - \vec{n}_i\|_1\right] \tag{4.13}$$

$$\mathcal{L}_{geo}^{\mathcal{F}} = \mathbb{E}_{\mathcal{X}}\left[\frac{1}{|\mathcal{P}_{\mathcal{F}}|}\sum_{i=1}^{|\mathcal{P}_{\mathcal{F}}|}\lambda_{g_3}|\hat{d}_i - d_i|\right] \tag{4.14}$$

$$\mathcal{L}_{geo} = \mathcal{L}_{geo}^{\mathcal{O}} + \mathcal{L}_{geo}^{\mathcal{F}}, \tag{4.15}$$

where λs are loss weights and $d_i = 0$ for points over the surface. We also impose code reconstruction loss $\mathcal{L}_{code} = \|\hat{\mathbf{w}} - \mathbf{w}\|_2$ to regularize the learning and 2D supervisions $\mathcal{L}_{rec}$ to minimize the reconstruction error between $\tilde{x}$ and x as in pSp [97]. The overall loss is detailed in Sec. 4.1.8.

Among them, StyleSDF [8] introduces the signed distance function (SDF) to serve as a proxy for the density function $\sigma(\vec{x})$ used for the volume rendering in NeRF. Specifically, StyleSDF uses $G_{\mathbf{0}}$ to predict the distance $d(\vec{x}) = G_{\mathbf{0}}(\mathbf{w}, \vec{x})$ between the query point $\vec{x}$ and the shape surface, where the density function $\sigma(\vec{x})$ can be transformed from $d(\vec{x})$ for NeRF [2] to render. The incorporation of SDF leads to higher-quality geometry in terms of expressiveness view-consistency and clear definition of the surface. StyleSDF also enjoys the flexible style control for semantic editing as in StyleGAN [86]. Therefore, in this chapter, we also use StyleSDF as the base model for GAN inversion study. Please refer to the conference version [7] for the technical details of our method based on StyleSDF. Note that our method is not limited to EG3D/StyleSDF and could be easily extended to other style-based 3D GAN variations [22].

4.1.8 Training

Reconstruction Loss. On the reconstruction side, we adopt the same training objectives as pSp [97] and provide a brief overview of the supervision methods

employed for image reconstruction during both training phases. First, we apply the commonly-used $\mathcal{L}_2$ loss,

$$\mathcal{L}_2(x) = ||x - \hat{x}||_2. \tag{4.16}$$

Additionally, we adopt the LPIPS loss [171], which has demonstrated superior performance in preserving image quality compared to traditional perceptual losses:

$$\mathcal{L}_{\text{LPIPS}}(x) = ||F(x) - F(\hat{x})||_2, \tag{4.17}$$

where $F(\cdot)$ denotes the VGG network.

Moreover, to encourage better ID preservation with the input, we also introduce a recognition loss based on the feature-wise similarity using ArcFace [186] network:

$$\mathcal{L}_{\text{Id}}(x) = 1 - \langle R(x), R(E_g(x)) \rangle, \tag{4.18}$$

In summary, the total loss function is defined as

$$\mathcal{L}_{rec}(x) = \lambda_1 \mathcal{L}_2(x) + \lambda_2 \mathcal{L}_{\text{LPIPS}}(x) + \lambda_3 \mathcal{L}_{\text{Id}}(x),$$

where we set $\lambda_1 = 1$, $\lambda_2 = 0.8$, $\lambda_3 = 0.1$ as the defined loss weights. In $E_{\mathbf{0}}$ training, we supervise images $\hat{x}_{\mathbf{0}}, \hat{x}_{\mathbf{1}}$ of both resolutions. In $E_{\mathbf{1}}$ training, we only supervise the reconstruction of high-resolution images since the network weights to render $\hat{x}_{\mathbf{0}}$ is fixed. Here, We also apply a non-saturating adversarial loss with R1 regularization [144] to enhance the realism of the reconstructed images, defined as:

$$\mathcal{L}_{adv} = -\mathbb{E}[log(D(\hat{x}))], \tag{4.19}$$

$$\mathcal{L}_D = \mathbb{E}[log(D(\hat{x}))] + \mathbb{E}[log(1 - D(x))], \tag{4.20}$$

$$\mathcal{L}_{R1} = \lambda \|\nabla D(\hat{x}; \theta_D)\|_2, \tag{4.21}$$

where θ_D represents the parameters to be optimized. In summary, the total loss is a weighted sum of the previously described loss functions:

$$\mathcal{L} = \mathcal{L}_{geo} + \mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv} + \lambda_D \mathcal{L}_D + \lambda_{R1}\mathcal{L}_{R1}, \tag{4.22}$$

where we set $\lambda_D = \lambda_{adv} = 0.01$ and $\lambda_{R1} = 10$ in the experiments. In general, $\mathcal{L}_{geo}$ encourages reasonable 3D geometry, $\mathcal{L}_{rec}$ encourages high-fidelity texture reconstruction and identity preservation, and $\mathcal{L}_{adv}$, $\mathcal{L}_D$, and $\mathcal{L}_{R1}$ are adversarial training objectives that encourages the naturalness of the rendered results.

4.2 Experiments

4.2.1 Implementation Details

Datasets. We mainly focus on the human face domain and use both 2D and 3D datasets for extensive evaluation. We leverage the 3D GANs pre-trained on FFHQ [86] during training. For inference, to examine 2D reconstruction quality, we adopt CelebA-HQ [150, 187] dataset for source view reconstruction. To further evaluate novel view synthesis performance, we synthesize 100 trajectory videos from a pretrained generator as a proxy test set. For attribute editing, we adopt InterfaceGAN [188] and Talk2Edit [189] to search for the editing directions. To evaluate 3D shape reconstruction quality, we use NoW benchmark [190] that provides a rich variety of face images with ground-truth 3D scans. Note that our method does not rely on any external 3D data during the training process.

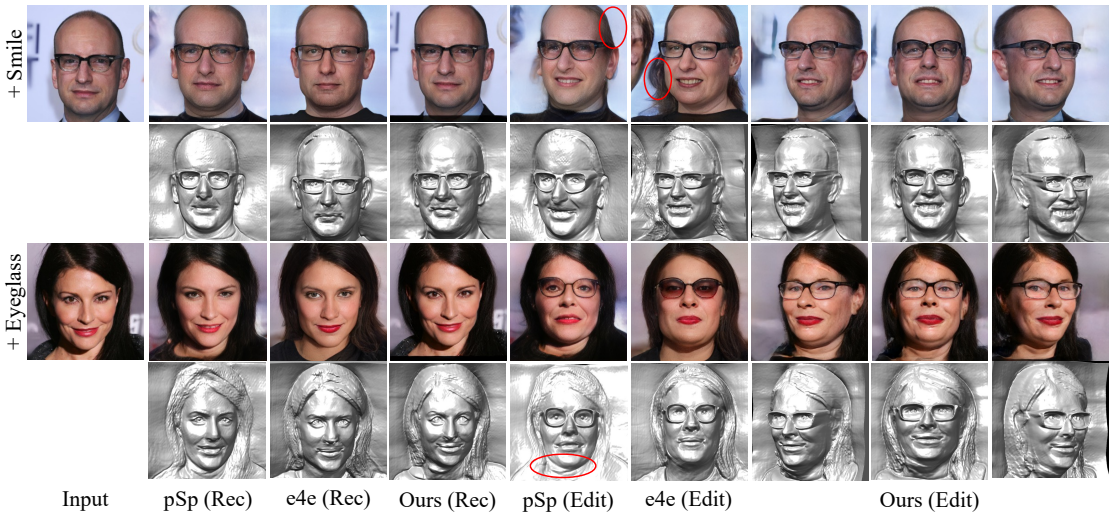


FIGURE 4.4: **Qualitative comparisons on face reconstruction (Rec) and editing (Edit) under novel views.** Rec denotes "Reconstruction" and Edit denotes "Editing". Our method shows both faithful texture preservation and plausible shape reconstruction compared to the baselines.

Network Architecture Details. For $E_{\mathbf{0}}$, a modified version of the pSp encoder [97] is deployed here for a fair comparison with existing work. For EG3D, we introduce 14 prediction heads to the pSp for the latent code prediction. For StyleSDF, since G_0 and G_1 of StyleSDF have 9 and 10 latent codes, respectively, we introduce $9 + 10$ extra prediction heads. We observe that early layers of $G_{\mathbf{0}}$ control the geometry of generated samples, and later $G_{\mathbf{0}}$ layers as well as decoder generator $G_{\mathbf{1}}$ control the texture and high-frequency details. Thus, we adopt the early pSp feature map of resolution 32×32 to predict latent code of G_0 for geometry control, and pSp feature map of resolution 64×64 to predict latent code of G_0 for texture control. We use the highest resolution feature map of pSp with resolution 128×128 to predict the latent code for G_1. We show our FiLM layer where the input features are modulated by the input conditions with predicted γ, and β. The MLP is implemented with two MLP residual blocks [3], which outputs α and β for modulation, respectively.

Training Details. In this work, we directly use the officially released pre-trained GAN models from EG3D and StyleSDF. In self-supervised shape inversion learning (Sec. 4.1), due to GPU memory restriction, we sample 4 shapes per GPU each iteration for training. After $E_{\mathbf{0}}$ converged, we fix the network weights and only train the $E_{\mathbf{1}}$ for high-fidelity inversion. For all the encoder models, we adopt Adam optimizer with a learning rate of $5e-5$ to train the models on 4 NVIDIA Tesla V100 GPUs, with a resolution of 512^2, batch size of 24, and 48 samples along a ray for the recommended $500K$ iterations. Following [175], we filter our invisible 3D points when training from a certain view.

To train the Pose Estimator $E_{\boldsymbol{\xi}}$, we directly append a prediction head behind the pre-trained global encoder $E_{\mathbf{0}}$ and output the spherical pose (θ, ϕ) of the input image. After the base inversion model is trained, we fix the remaining parameters and train the prediction head over synthetic images for $50,000$ iterations, which takes four days over 4 V100 GPUs. The training takes 4 days for StyleSDF-E3DGE and 7 days for EG3D-E3DGE. Code, dataset, and all pre-trained models are publicly available at `https://github.com/NIRVANALAN/CVPR23-E3DGE`.

TABLE 4.1: **Quantitative performance on CelebA-HQ.** 'T' and 'S' denote the time for texture and shape inversion, 'P' denotes finetuning estimated pose, and 'P+C' denotes the finetuning both pose and code, respectively. For the Time(s) column, we show the texture and geometry inversion time separately. The last two rows describe the performance of test-time optimization, where fine-tuning the pose (P) and code (C) further improves the quality. Note that compared with E3DGE-StyleSDF, E3DGE-EG3D does not require post processing to output depth.

Methods	MAE ↓	SSIM ↑	LPIPS ↓	Similarity ↑	Time(s) ↓
$SG2_{StyleSDF}$	.202 ± .063	.650 ± .054	.167 ± .046	.219 ± .106	235
$PTI_{StyleSDF}$	**.062 ± .012**	**.796 ± .017**	**.027 ± .005**	**.892 ± .009**	246
$pSp_{StyleSDF}$	.150 ± .032	.696 ± .048	.270 ± .059	.498 ± .099	.29
$e4e_{StyleSDF}$	.174 ± .049	.669 ± .049	.226 ± .063	.252 ± .107	.29
$E^3DGE_{StyleSDF}$	.103 ± .010	**.769 ± .039**	**.136 ± .039**	.881 ± .041	**.45/.81**
pSp_{EG3D}	.163 ± .024	.689 ± .039	.264 ± .049	.455 ± .096	0.29
$e4e_{EG3D}$	.230 ± .021	.658 ± .019	.425 ± .029	.316 ± .068	0.29
$SG2_{EG3D}$	.241 ± .019	.671 ± .014	.288 ± .019	.434 ± .037	100
PTI_{EG3D}	.079 ± .005	.769 ± .012	.105 ± .011	.779 ± .027	114
E^3DGE_{EG3D}	**.084 ± .012**	**.768 ± .026**	.163 ± .017	**.891 ± .027**	**.38**
$E^3DGE^{P}_{EG3D}$	**.076 ± .004**	**.777 ± .008**	**.153 ± .005**	**.952 ± .004**	45
$E^3DGE^{P+C}_{EG3D}$	**.064 ± .003**	**.795± .009**	**.115 ± .005**	**.974 ± .003**	60

4.2.2 Quantitative Evaluation

For comparison, we implement two canonical encoder-based GAN inversion approaches on StyleSDF [8] and EG3D [1], i.e., pSp [97] and e4e [98], which stress reconstruction and editing quality respectively. Furthermore, we also implement optimization-based methods including SG2 [86] and PTI [28] on EG3D and StyleSDF for extensive comparison.

We report inversion performance for both source view reconstruction and novel view synthesis in Tabs 4.1-4.2. For source view reconstruction, the metrics are calculated on the 2,824 images from CelebA-HQ test set [150]. For novel view synthesis, the metrics are averaged from 100 videos generated from pre-trained 3D GANs, each with 250 frames covering ellipsoid camera poses trajectory. For each video, we randomly pick one image as source view input and the remaining images as ground truths with labeled poses as query views. In this way, we could extensively evaluate the view synthesis ability under occlusions and varied input viewpoints. We also compare E3DGE against two optimization-based methods, SG2 and PTI. As shown in Tab. 4.1, our approach significantly outperforms encoder-based baselines

in terms of reconstruction quality across two settings, while also achieving much faster inference speed against optimization-based methods. Notice that we do not include EG3D in Tab. 4.2 due to its camera pose being misaligned with StyleSDF.

TABLE 4.2: **Quantitative performance on Novel View Synthesis.**

Methods	MAE ↓	SSIM ↑	LPIPS ↓	Similarity ↑
$SG2_{StyleSDF}$	.284 ± .025	.572 ± .006	.244 ± .031	.304 ± .036
$PTI_{StyleSDF}$	.186 ±.016	.652 ± .015	.215 ± .045	.795 ± .040
$pSp_{StyleSDF}$	.201 ± .010	.634 ± .005	.285 ± .029	.559 ± .043
$e4e_{StyleSDF}$	.197 ± .016	.597 ± .011	.212 ± .023	.297 ± .058
E3DGE	**.147 ± .011**	**.694 ± .018**	**.151 ± .024**	**.901 ± .012**

4.2.3 Qualitative Evaluation

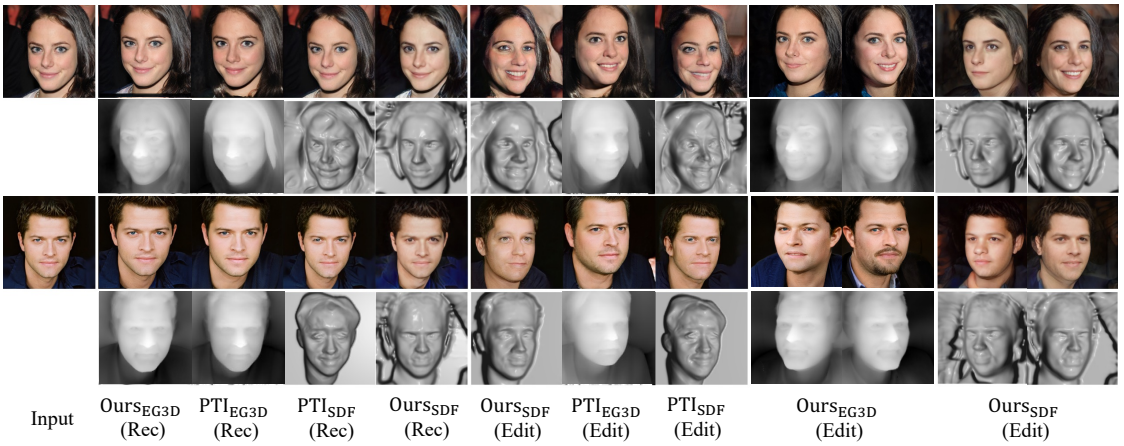


FIGURE 4.5: **Visual comparisons on optimization-based methods. 'Rec' and 'Edit' represent reconstruction and editing, respectively.** For editing, we change the **'Smiling'** attribute for the first instance and **'Age'** attribute for the second instance. Besides, PTI_{EG3D} (column 7) with "Smiling" editing shows geometry-texture misalignment over the teeth, where the geometry fails to show open mouth after editing. However, our $E3DGE_{EG3D}$ shows more coherent geometry and texture editing.

Encoder Baselines. We visualize both inversion and editing results against encoder baselines in Fig. 4.4. *Geometry-wise*, the baseline models without explicit 3D supervision tend to generate implausible intermediate shapes, a 3D plank that

is only plausible from the input view. Besides, their reconstruction is not close to the "ground truth", and the reconstructed surface lacks details. Our method successfully regularizes the intermediate 3D shapes and generates plausible results with surface details and a more complete structure. For instance, our method reconstructs plausible 3D with with faithful identity preservation. Though pSp also preserves identity well, the reconstructed 3D depth is more flat. Although e4e shows better shape reconstruction, its reconstruction fails to maintain the input identity. Corresponding metrics in Tab. 4.4 also validate the usefulness of the direct geometry supervisions and loss designs. *Texture-wise*, existing methods often produce distorted outputs, suffering from artifacts and identity changes. In contrast, by incorporating pixel-aligned features, our method is more robust, delivering high-fidelity results. Specifically, it captures finer details and better preserves identity across different input viewpoints.

To showcase the editing capablity, we select the "Smile" attribute. Our method consistently generates high-quality edited renderings, maintaining view consistency, preserving details, and retaining identity, while also delivering accurate shape reconstruction and high-fidelity texture inversion. In comparison, baseline methods either fail to render plausible novel views (column 5) or struggle to preserve the input identity after editing (column 6), as highlighted.

Optimization Baselines. We also compare our method with the state-of-the-art optimization-based methods PTI and SG2 in Fig. 4.5. We include the performance of PTI [28] on both EG3D and StyleSDF for extensive evaluation. With more than 100× faster inference speed, the proposed method achieves a similar inversion quality. Besides, our editing results successfully preserve the local details with high-fidelity novel view editing performance. Note that the inversion result of StyleSDF tends to be inferior on 3D shape plausibility compared with EG3D, over both encoder-based inversion and optimization-based inversion. Meanwhile, in PTI-based editing, we notice the geometry-texture misalignment of PTI_{EG3D} (column 7), where the "Smiling" texture with teeth does not align with the geometry without teeth. However, our $\text{E3DGE}_{\text{EG3D}}$ shows coherent geometry and texture editing, which demonstrates our proposed method maintains more consistent semantics during editing.

Video Inversion. The efficiency of the encoder-based method also empowers video inversion and editing. Specifically, we experiment on HDTF [191] dataset

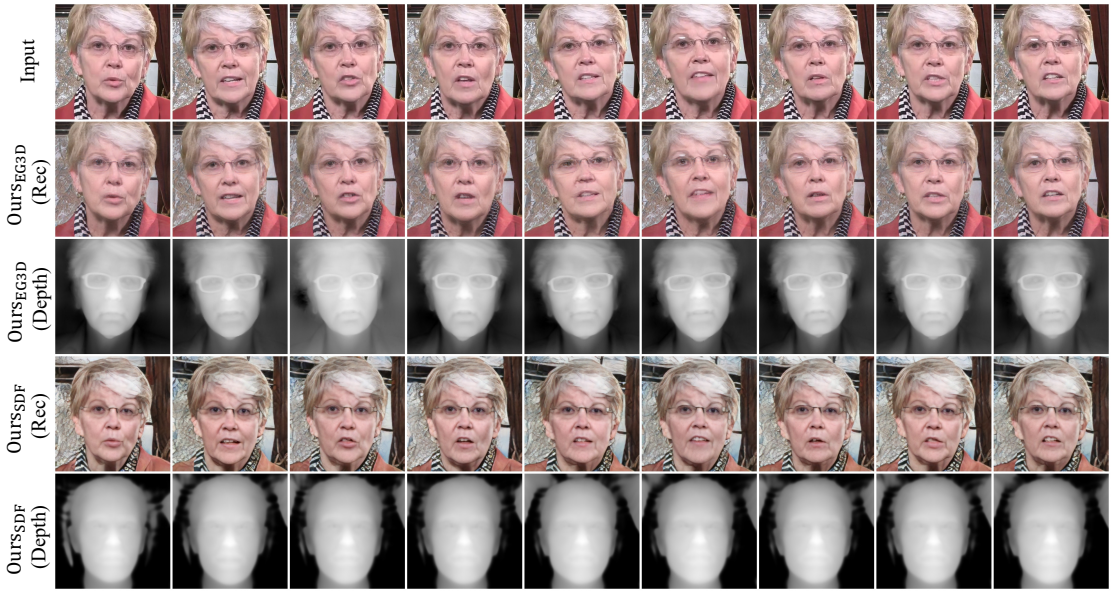


FIGURE 4.6: **Video inversion using E3DGE**. As shown here, the EG3D-based encoder shows better reconstruction texture quality, while StyleSDF-based E3DGE shows smoother surface prediction. Note that we also include the depth of EG3D-E3DGE for pose-aligned texture and shape comparison.

and conduct inversion on each frame using E3DGE. To allow for GAN inversion, we first preprocess the video by cropping the face from each frame using the original alignment standard adopted by StyleSDF and EG3D respectively. To reduce oscillation between frames caused by aligning each frame individually, we smooth the affine transformation of each frame within a small sliding window using a mean filter. This operation effectively improves the visual consistency of inversed video without violating the alignment bias of pre-trained GANs.

We visualize the video inversion results in Fig. 4.6. As can be seen, E3DGE shows consistent video inversion with high-quality texture and geometry. Compared with StyleSDF-based E3DGE, the EG3D-based E3DGE shows consistently better texture and shape reconstruction quality.

4.2.4 Ablation Study

Effect of 3D GAN as Supervisions. We quantitatively validate the effects of 3D supervision in the NoW Challenge validation set and report the corresponding metrics in Tab. 4.4. Compared with 2D supervision only, adding 3D supervisions greatly improves the reconstruction quality. We also validate the benefits of all loss

TABLE 4.3: **Ablations of Local Features and Hybrid Fusion.** Our local-global model design with hybrid alignment achieves the balance of high-quality reconstruction and view synthesis.

	Source View Reconstruction				Novel View Synthesis			
Ablation Settings	MAE ↓	SSIM ↑	LPIPS ↓	ID ↑	MAE ↓	SSIM ↑	LPIPS ↓	ID ↑
Synthetic Training	.245 ± .024	.634 ± .019	.333 ± .029	.369 ± .056	.241 ± .011	.594 ± .008	.366 ± .059	.770 ± .026
+Local Features	**.074 ± .007**	**.811 ± .015**	**.075 ± .010**	**.953 ± .006**	.282 ± .103	.571 ± 0.056	.511 ± 0.031	.608 ± .123
+2D Alignment	.098 ± .005	.774 ± .038	.140 ± .040	.900 ± .032	.178 ± .007	.656 ± .009	.178 ± .012	**.895 ± .018**
+3D Alignment	.102 ± .009	.772 ± .015	.119 ± .016	.818 ± .029	**.150 ± .011**	**.689 ± .022**	**.140 ± .021**	.891 ± .011

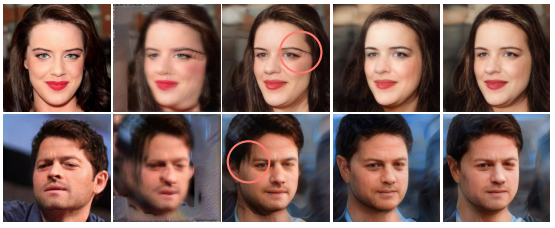


FIGURE 4.7: **Ablation of Hybrid Alignment.** From left to right, we show the novel view synthesis of raw 3D-aligned features w/wo novel view training, synthesis achieved using 2D-aligned features, and the final hybrid features. 3D-aligned features are view-consistent but suffer from occlusions (circled), while 2D features are visually plausible but lack some details (e.g., hair color). Our hybrid fused results share the best of both.

terms in E_0 training. Note that EG3D-based E3DGE achieves worse performance against StyleSDF-based E3DGE, which demonstrates that EG3D achieves better visual quality at the price of less-accurate geometry surface.

Effect of Local Features. As discussed, the local features preserve the image details to facilitate high-fidelity reconstruction. To validate the effectiveness of local features in texture reconstructions, we show the inversion results in Fig. 4.8. With the proposed local-global fusion pipeline, our model captures more details

TABLE 4.4: Effect of 3D Supervisions on the NoW Challenge.

Settings	Median↓	Mean↓	Std
$\text{pSp}_{\text{StyleSDF}}$	1.97	2.43	2.05
$\text{e4e}_{\text{StyleSDF}}$	2.83	3.40	2.67
$E^3DGE_{\text{StyleSDF}} + \mathcal{L}_{geo}^{\mathcal{O}}$	1.75	2.11	1.72
$E^3DGE_{\text{StyleSDF}} + \mathcal{L}_{geo}^{\mathcal{F}}$	1.71	2.09	1.70
$E^3DGE_{\text{StyleSDF}} + \mathcal{L}_{code}$	**1.66**	**2.06**	**1.69**
E^3DGE_{EG3D}	2.29	2.83	2.31

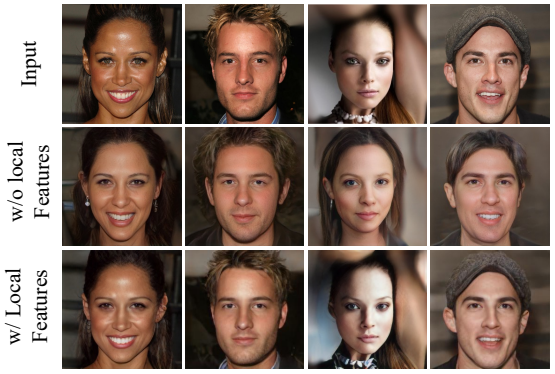


FIGURE 4.8: **Ablation of Local Features.** Our method with pixel-aligned features shows photorealistic reconstructions.

and guarantees photorealistic reconstruction. Quantitative results in Tab. 4.3 also validate the effectiveness of local features in high-quality inversion. The results on the video trajectories also show that without delicate design, e.g., novel view training, local features would fully collapse over novel view synthesis.

Effect of Hybrid Alignment. We show the view synthesis achieved by different alignment methods in Fig. 4.7. To quantitatively analyze the effect of hybrid alignment, in Tab. 4.3 we evaluate the model performance of 3D alignment and 2D alignment individually. For both ablations, novel view training is enabled. As shown here, the 3D alignment model shows better view consistency in video prediction measured by reconstruction metrics, and the 2D alignment model shows better identity preservation. The hybrid alignment model marries the best of both and also enables semantic editing and yields better reconstruction performance on the video predictions.

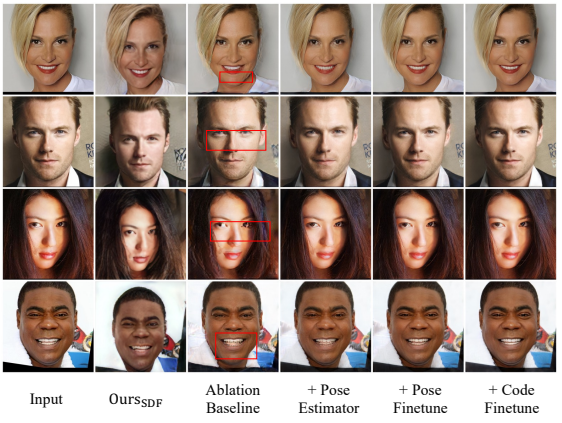


FIGURE 4.9: **Ablation of Hybrid Alignment.** By introducing the pose estimator and further finetuning the pose and latent code, E3DGE achieves better inversion with fewer visual artifacts. Better zoom in.

Effect of Introducing Pose Estimator and Hybrid Optimization. We show the inversion results achieved by different pipelines in Fig. 4.9 and quantitative

analysis in Tab. 4.1. Though EG3D-E3DGE preserves more texture details compared with StyleSDF-E3DGE (column 2), adopting EG3D as the base model introduces more obvious misalignment issues due to the nature of tri-plane, and introducing a jointly trained pose estimator could alleviate this issue during inference. Furthermore, finetuning the predicted camera pose and latent code could further boost the performance.

4.2.5 More Results

E3DGE on Other Categories. Besides FFHQ, the performance of the proposed method on AFHQ-EG3D (Fig. 4.11) and ShapeNet-StyleSDF (Fig. 4.12) is further visualized.

As can be seen, E3DGE also achieves high-quality shape and texture inversion on both cats and chairs, demonstrating the generalizability of our method.

3D Toonification. After training the E3DGE encoder, we show that our method could be directly applied to 3D stylization. We show 3D toonify-stylized results over real-world faces using our proposed method in Fig. 4.10. Following [192], we finetune the pre-trained generator G for 400 iterations with 317 cartoon face images and use our pre-trained encoder E for inference. Visually inspected, the toonified results hold the cartoon style and also preserve the identity of the input image, which demonstrates the potential of applying our method over downstream tasks. Moreover, the toonification of StyleSDF-E3DGE shows richer 3D details compared with EG3D-based E3DGE.

Computational Cost. We include the computational cost of each component in the table below.

TABLE 4.5: Model parameters and computational cost evaluated with MACs.

Component	$E_{\mathbf{0}}$ (pSp)	$E_{\mathbf{1}}$	E_{ADA}
Parameters(M)	219.71	14.06	0.60
MACs(G)	62.95	26.07	4.03

4.2.6 Comparisons of Inversion with MLP and Triplane-based 3D GANs

In this work, we have extended E3DGE from MLP-based 3D GAN model [8] to triplane-based variant [1] and demonstrated the generalization of our design. However, as discussed before, EG3D-based inversion has its new challenges and does not outperform StyleSDF in every perspective. Here we provide our comparisons of conducting 3D GAN inversion on these two representative model architectures.

Regarding the advantages of inversion on triplane-based 3D GANs, we conclude that 1) higher fidelity can be achieved on EG3D as the triplane offers more representation capacity compared to MLP-based counterparts. As shown in Fig. 4.5, inversion of the same input over EG3D yields better 3D and texture details compared with StyleSDF; 2) better 3D inductive bias offers more robust and flexible inversion. Specifically, noticeable shape artifacts are observed when applying 2D inversion methods like pSp, e4e, and PTI on the StyleSDF backbone, as shown in Fig. 4.5. However, thanks to the triplane's strong 3D inductive bias, directly applying 2D inversion methods yields higher-quality performance with plausible shape and textures reconstructed. This also offers the hybrid inversion option, which improves E3DGE's prediction by optimizing the inversed latent code within acceptable overhead without affecting the shape plausibility, which could not be guaranteed by StyleSDF.

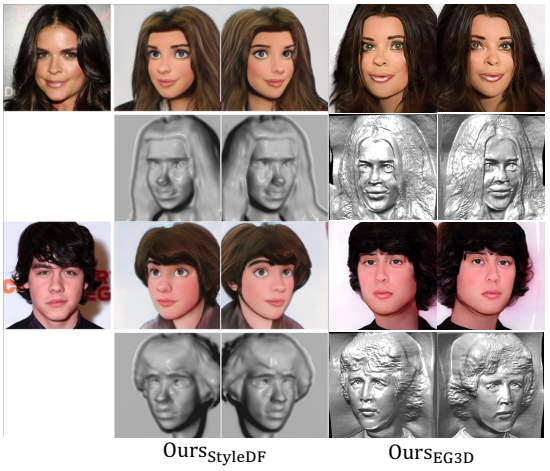


FIGURE 4.10: **Toonification using E3DGE.** From left to right, we show the toonification result of E3DGE based on StyleSDF and EG3D.

However, the superior performance of triplane comes at a cost with observed limitations as the following: 1) Triplane is more sensitive to pose alignment issues, e.g., when the estimated pose is

FIGURE 4.11: **The inversion and view synthesis results of AFHQ cat.**

not accurate enough, the inversion of
E3DGE is likely to fail, as shown in Fig. 4.9. Including an extra pose estimator during training and further finetuning the estimated pose can alleviate this issue. 2) In contrast to StyleSDF, EG3D has more complex latent space and the accurate latent code cannot be accurately acquired via a feed-forward prediction. Further optimization of the estimated poses and latent codes is required to achieve the expected performance in some challenging cases. 3) Training E3DGE on EG3D requires more time to converge, i.e., 7 days compared with 4 days of StyleSDF, which is reasonable due to the increased model capacity and fidelity. Once trained, our model can amortize the inference time on a series of downstream applications. 4) As shown in Tab. 4.4, EG3D-based E3DGE achieves worse performance against StyleSDF-based E3DGE, which indicates that EG3D achieves better visual quality at the price of less-accurate geometry surface.

Overall, triplane-based 3D representation has shown great potential and it is worthwhile to study encoder-based inversion on it. With triplane's unique advantages and affiliated limitations, we hope our study could inspire the research and industry community to choose a suitable representation of their tasks. We also hope our work could motivate future research on improved 3D representations.

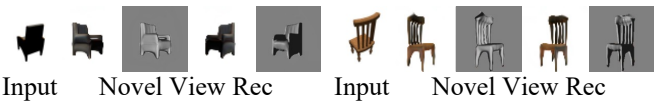


FIGURE 4.12: **E3DGE qualitative performance on ShapeNet Chair.**

4.3 Conclusion

This chapter proposes a novel 3D GAN inversion framework E3DGE for 3D GAN inversion and editing. We marry the benefits of both self-supervised global prior and pixel-aligned local prior for high-quality shape and texture reconstruction. A hybrid alignment that bridges the best of 2D and 3D features is further proposed for view-consistent editing. Benefiting from the overall system design, the proposed method has advantages in terms of both high fidelity and editability. As a pioneer attempt in this direction, we believe this work opens a new line of research direction and will inspire future works on 3D GAN inversion, few-shot 3D reconstruction and 3D-aware learning from 2D images.

Chapter 3 and 4 have demonstrated the significance of designing suitable algorithms to expand the generality of neural fields. The following chapter turns to explore the design of 3D generation framework for both domain-specific 3D generation and general objects 3D generation.

Chapter 5

Loc3Diff: Local Diffusion for 3D Human Head Synthesis and Editing

Generating and editing photorealistic portraits is one of the cruxes of computer graphics and has tremendous demand in downstream applications, such as embodied AI, VR/AR, digital games, and movie CG. Emerging 3D-aware GANs [1, 21, 22, 30, 31] have achieved great success in generating high-quality multi-view consistent portraits with volumetric rendering [2]. Editing capabilities for 3D-aware GANs have also been achieved through latent space auto-decoding, altering a 2D semantic segmentation [32, 33], or modifying the underlying geometry scaffold [34]. However, generation and editing quality tends to be unstable and less diversified due to the inherent limitation of GANs, and detailed-level editing is not well supported due to feature entanglement in the compact latent space or global tri-plane representations. Recently, diffusion models [35, 36] have been proposed for high-quality content generation, achieving competitive performance compared to traditional GAN-based approaches [37]. Efforts have been made on 3D-aware portrait generation by de-noising the global tri-plane representation [23, 38], which, however, do not support animation and region-based editing.

The work in this chapter has been published in [17].
Code is publicly available at `https://nirvanalan.github.io/projects/gaussian3diff/`

This chapter presents LN3DIFF, a diffusion-based generative model designed for 3D volumetric head. This model enables unconditional generation while offering versatile capabilities for flexible global and fine-grained region-based editing, such as change of face shape, expression, or appearance. As the core of our model, a novel representation of the 3D head is proposed, in which complex volumetric geometry and appearance are encoded by a large set of local tri-planes, anchored on the surface of an underlying 3D Morphable Face Models (3DMM) [193]. Each tri-plane is modulated by a 3D Gaussian, refining its position, orientation, and scale, thereby controlling its influence within the 3D space. By anchoring the tri-planes to the 3DMM, each tri-plane and its Gaussian parameters can be projected onto the 2D texture map. This discrete 2D UV format ensures compatibility with the well-studied 2D diffusion frameworks [37, 53].

Through a diffusion process on the UV space, LN3DIFF can generate these UV-anchored local tri-planes and eventually create diverse high-quality head avatars. Due to its rich semantic connection with the 3DMM model, a tri-plane at each location of UV space represents a similar facial region and contains semantic-aligned features across different identities. Leveraging this key attribute, LN3DIFF excels in geometry and appearance editing, which distinguishes it from other generative head models. The shape and texture can be separately interpolated across identities at global or local scales, by swapping Gaussian-modulated tri-planes at the corresponding areas on the UV map. The shape and texture separately transition across identities at global or local scales by linearly interpolating Gaussian-modulated tri-planes at the corresponding areas on the UV map. Moreover, by resampling tri-planes at selected regions through diffusion, LN3DIFF can seamlessly achieve fine-grained editing tasks such as local region-based editing and 3D in-painting with outstanding visual quality.

To train the diffusion network, creating semantically consistent tri-plane ground-truth is a non-trivial challenge. This chapter proposed a novel approach that simultaneously reconstructs large amounts of 3D heads in our representation by learning a shared latent space via an auto-decoder [14] with multi-view supervision. Our main difference with per-example fitting [23, 38] methods is adopting a heavy shared decoder to decode the latent space to the tri-planes across multiple identities. Empirically, it is found that the jointly optimized shared latent space

encourages the alignment of the local tri-planes' features and Gaussian parameters across identities, which is essential to support identity interpolation (Fig. 5.4), 3DMM reanimation (Fig. 5.6) and the success of diffusion training (Sec. 5.1.2).

Our contributions are summarized as follows:

- A novel representation for animatable 3D volumetric head - 3D Gaussian-modulated local tri-plane on 3DMM UV space, which naturally supports flexible editing at global and local scales.
- An auto-decoding-based fitting algorithm to generate training data in our representation and show it benefits diffusion model training.
- Extensive experiments demonstrate that LN3DIFF exhibits superior reconstruction and generation quality, as well as capabilities in various tasks such as shape-texture transfer, 3D inpainting, 3DMM-based editing, and region-based editing.

5.1 Methodology

We propose LN3DIFF, a comprehensive framework designed for the reconstruction and generation of photo-realistic 3D human heads with extensive editing capabilities. To fulfill this objective, we introduce a novel 3D head avatar representation in Sec. 5.1.1. This representation leverages local tri-planes modulated by 3D Gaussians to effectively encode geometric and textural information in local regions. Critically, the local tri-planes are anchored to a 3D Morphable Model (3DMM), allowing for the parameterization of 3D volumetric data into the 2D texture space. This facilitates the application of a 2D diffusion model for the editing

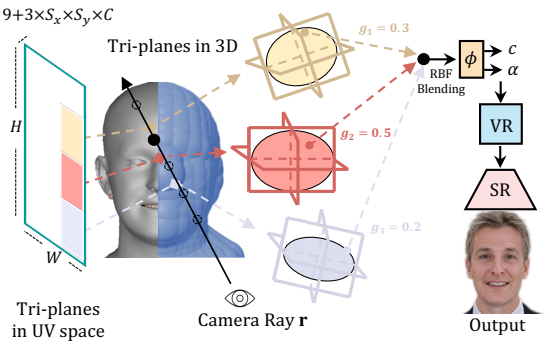


FIGURE 5.1: During volume rendering, tri-plane payloads in UV space are projected onto 3D space with Gaussian pose parameters. For each shading point, we query the texture and geometry information from the three nearest payloads, with influence strength defined using a radial basis function (RBF). The low-res 2D rendering is then upsampled with a CNN-based super-resolution network.

process. In Sec. 5.1.2, we illustrate the diffusion-based avatar editing framework. Initially, we delineate our analysis-by-synthesis approach that concurrently reconstructs a large number of avatars 3D heads of *different expressions and identities/appearance* and learns a shared latent space through multi-view supervision. This ensures that the learned representations of all avatars encapsulate crucial mutual information. Subsequently, we account for the training of a 2D diffusion model that generates 3D heads with *neutral expression*.

In Sec. 5.1.3, we discuss the editing mechanisms to showcase the capabilities of the proposed method.

5.1.1 Avatar Representation

Local Tri-planes in 3D Gaussians. Existing methods represent a 3D head with a global representation [23, 33, 61, 62, 146, 194], where either a single MLP [61, 62, 146, 194] or a tri-plane [23, 33] is employed to encode the entire neural radiance field. However, the global-based representation limits the region-based editing ability and cannot be directly driven by the parametric models [75, 104, 195]. Inspired by previous work [196–198] on representing radiance fields with local primitives, we propose to represent a 3D human head as a set of local tri-planes, each modulated by a 3D Gaussian initialized from a 3DMM. Specifically, the position, orientation, and scale of each tri-plane can be efficiently modeled using a 3D Gaussian.

To elaborate, each 3D Gaussian-modulated tri-plane $\mathcal{G}_i = \{\boldsymbol{\mu}_i, \Sigma_i, P_i\}$ is characterized by 9 pose parameters and a payload - a 3D center $\boldsymbol{\mu}_i$, 3 axis-aligned radii and 3 rotation angles parameterized by a 6-DOF covariance matrix Σ_i, and a tri-plane payload $P_i \in \mathbb{R}^{3\times S_x \times S_y \times C}$. These pose parameters define the local coordinate transform from the world space to the tri-plane space, as well as the influence strength. Each point $\mathbf{x}$ in the world space can be mapped to the canonical local space according to the 3D Gaussian's center $\boldsymbol{\mu}$ and rotation following [199, 200]. The influence strength is defined as an analytic radial basis function (RBF):

$$g(\mathbf{x}) = \exp\left(-\frac{1}{2}\left(\mathbf{x}-\boldsymbol{\mu}\right)^T \Sigma^{-1}\left(\mathbf{x}-\boldsymbol{\mu}\right)\right). \tag{5.1}$$

Given an object integrated by local tri-planes, we can render any view with volumetric rendering [2]:

$$\hat{C}(\mathbf{r}) = \sum_{j=1}^{J} T_j \alpha_j \mathbf{c}_j, \text{where } T_j = \prod_{l=1}^{j-1}(1-\alpha_l). \tag{5.2}$$

where $\hat{C}(\mathbf{r})$ is the rendered color from the ray $\mathbf{r}$, T_j is transmission at the j-th sample along the ray, α_j, $\mathbf{c}_j$ are the opacity and color of the sample, and J is the total number of samples along the ray. To efficiently compute c_j and α_j of each sample point $\mathbf{x}_j$, we only query K nearest tri-planes $\mathcal{G}_k$ measured by the Euclidean distance to the Gaussian centers $\boldsymbol{\mu}_k$. The queried features are transformed to the corresponding $\mathbb{R}^4$ values via a shared tiny rendering MLP ϕ. We then take a weighted average of the k individual color and opacity:

$$\mathbf{c}_j = \sum_{k=1}^{K} \hat{g}_k(\mathbf{x}_j)\mathbf{c}_{j,k}, \alpha_j = \sum_{k=1}^{K} g_k(\mathbf{x}_j)\alpha_{j,k}, \tag{5.3}$$

$$\text{where } \hat{g}_k(\mathbf{x}_j) = \frac{g_k(\mathbf{x}_j)}{\sum_{k=1}^{K} g_k(\mathbf{x}_k) + \epsilon}, \tag{5.4}$$

where $\mathbf{c}_{j,k}$ and $\alpha_{j,k}$ represent the color and opacity of point $\mathbf{x}_j$ queried from a tri-plane $\mathcal{G}_k$. $\hat{g}_k(\mathbf{x}_j)$ denotes the normalized inference strength and ϵ serves as a factor allowing smooth decay. Note that we do not normalize $g_i(\mathbf{x}_k)$ when computing opacity α_j. This choice allows the opacity α_j to naturally decay in empty space. This strategy acts as a window function [201], encouraging the tri-planes to focus on the local surface region.

Our approach is thus more efficient than previous 3D Gaussian-based representations [200, 202] that require millions of tiny blobs where each only stores the spherical harmonic (SH) coefficients and opacity value.

UV Space Representation. By anchoring tri-planes on a 3DMM, each tri-plane now corresponds precisely to a specific 2D location on the texture map. Consequently, these tri-planes stored on the UV space can be processed with the U-Net-based diffusion framework [39]. Furthermore, the semantically aligned texture map facilitates a range of editing operations.

Specifically, following previous work on the dynamic avatar reconstruction [196], we first register a 3DMM model, *e.g.*, FLAME [104] for each identity instance

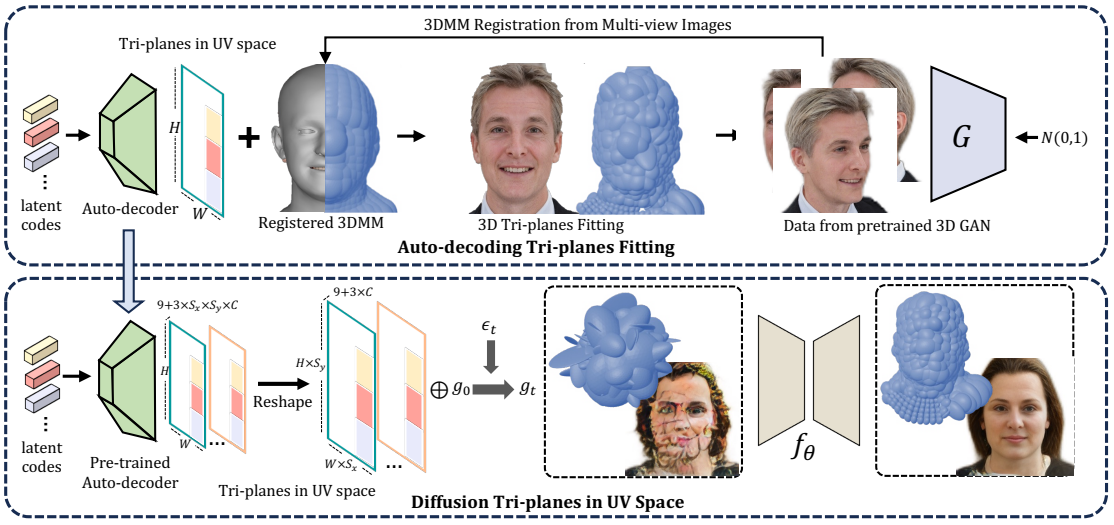


FIGURE 5.2: Pipeline of learning 3D head generation. Top: An autoencoder is trained to generate tri-planes modulated by 3D Gaussians in UV space from a dataset produced by a pretrained 3D GAN; Bottom: a diffusion model is trained by a diffusion and denoising process on tri-planes with Gaussian parameters represented in UV space, generated by the auto-decoder trained in the previous step.

generated from pretrained 3D GAN. Vertices on fitted 3DMM model can be directly rasterized onto the UV space, where a local tri-plane is attached to each rasterized vertex. Specifically, we rasterize the 3DMM mesh vertices to the UV space, where each texel on the UV space anchors a flattened vector including the tri-plane embeddings and its 3D Gaussian parameters. Besides, each tri-plane's position, orientation, scale and blending function is efficiently modulated by the corresponding 3D Gaussian. We utilize the vertex positions and face normals to initialize the centers $\boldsymbol{\mu}_i$ and the rotations of 3D Gaussian-modulated tri-planes. The axis-aligned anisotropic scaling is initialized proportionally to the area of the corresponding faces on the mesh. Moreover, to maintain flexibility over regions such as glasses and hair, all of the Gaussian parameters are allowed to be optimized during reconstruction.

The overall trainable parameters of each identity consist of local tri-planes over the UV grid: $P \in \mathbb{R}^{H\times W\times 3\times S_x\times S_y\times C}$, and the corresponding 9-DOF Gaussian parameters: $\boldsymbol{\mu} \in \mathbb{R}^{H\times W\times 3}$ and $\Sigma \in \mathbb{R}^{H\times W\times 6}$.

5.1.2 Learning 3D Head Generation

Reconstruction 3D Heads with an Auto-Decoder. To effectively train the diffusion model, it is essential to have a large dataset of high-quality photorealistic 3D head assets. To address this issue, we employ the DatasetGAN [7, 16, 114] paradigm and utilize Panohead [31], a state-of-the-art 3D GAN for generating human heads, as our data generator. This approach enables us to prepare a sufficient number of 3D assets for local tri-plane fitting and diffusion training.

Fitting 3D assets individually involves costly reconstruction over dense multi-view images from scratch, making it data-intensive and inefficient. To overcome this challenge, we adopt an auto-decoding design [107, 203, 204] that learns a shared decoder to reconstruct 3D heads by optimizing a latent code from multi-view images. Specifically, each 3D instance is associated with a latent code $\mathbf{z} \in \mathbb{R}^{512}$ during the optimization process. This latent code can be decoded into the local tri-planes in UV space through a convolutional decoder $D : \mathbb{R}^{512} \rightarrow \mathbb{R}^{H\times W\times 3\times S_x\times S_y\times C}$. Unlike previous work [23] that fits tri-planes independently, our shared decoder is trained from multiple instances, enabling faster convergence and improved generalizability. Furthermore, decoding all local payloads from a shared decoder results in a smooth latent space suitable for diffusion training.

Similar to PanoHead [31], to reduce the memory consumption and computation cost, we render the color image in low resolution from 3D Gaussian tri-planes and upsample them to high resolution with a super-resolution module.

During the training process, we jointly optimize all network parameters and the latent code. The loss function is decomposed into RGB loss, opacity regularization, and latent code regularization.

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{code}}. \tag{5.5}$$

where $\mathcal{L}_{\text{rgb}}$ is the RGB loss measured with L1 and LPIPS [171] between the synthesized color $\hat{C}$ and the ground truth color C within each patch, $\mathcal{L}_{\text{reg}}$ regularizes a compact 3D representations and $\mathcal{L}_{\text{code}}$ penalize the norm of the latent code given a normal prior [203]. More details can be found in supplementary materials.

Local Tri-plane Diffusion in the UV Space. After the local tri-planes are prepared in UV space, we could learn a diffusion prior on the tri-planes with Gaussian

parameters in UV to support 3D avatar generation. Specifically, a diffusion model generates data by reversing a gradual destruction process. This process typically involves adding Gaussian noise over time. The process can be expressed using marginals $q(\mathcal{G}_t|\mathcal{G}_0)$, defined by:

$$q(\mathcal{G}_t|\mathcal{G}_0) = \mathcal{N}(\mathcal{G}_t|\alpha_t\mathcal{G}_0, \sigma_t^2\mathbf{I}) \tag{5.6}$$

Here, $\alpha_t, \sigma_t \in (0, 1)$ are hyperparameters between 0 and 1 that control how much signal is destroyed at timestep t. Commonly, a variance-preserving [35] process is used where $\alpha_t^2 = 1 - \sigma_t^2$. Before training diffusion model, $\mathcal{G}_0$ is set to *neutral expression*. This allows the generated samples can be easily manipulated using the expression basis [104].

FIGURE 5.3: **Auto-Decoder Results.** With the design of local tri-planes on 3DMM, our auto-decoder D yield high-quality and view-consistent reconstructions. Moreover, LOC3DIFF intrinsically support novel expression animation by moving the positions of the tri-planes. Note that we do not rely on multi-expression dataset during training.

Forward Process. Assuming the diffusion process is Markov, the forward transition is given by:

$$q(\mathcal{G}_t|\mathcal{G}_s) = \mathcal{N}(\mathcal{G}_t|\alpha_{ts}\mathcal{G}_s, \sigma_{ts}^2\mathbf{I}), \tag{5.7}$$

where $\alpha_{ts} = \alpha_t/\alpha_s$ and $\sigma_{ts}^2 = \sigma_t^2 - \alpha_{t|s}^2\sigma_s^2$ and $t > s$. To improve the performance of the diffusion model, which favors narrower input channels [39], we unfold local tri-planes onto UV space along the x and y dimensions. This operation reshapes the Gaussian parameters on UV from $\mathbb{R}^{H\times W\times(9+3\times S_x\times S_y\times C)}$ to $\mathbb{R}^{(H\times S_y)\times(W\times S_x)\times(9+3\times C)}$, and the 9-DOF Gaussian parameters are replicated $S_x \times S_y$ times within each local tri-plane during the unfolding.

Denoising Process. Conditioned on a single datapoint $\mathcal{G}$, the denoising process can be written as:

$$q(\mathcal{G}_s|\mathcal{G}_t, \mathcal{G}_0) = \mathcal{N}(\mathcal{G}_t|\boldsymbol{\mu}_{t\to s}, \sigma_{t\to s}^2\mathbf{I}). \tag{5.8}$$

where $\boldsymbol{\mu}_{t\to s} = \frac{\alpha_{ts}\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t + \frac{\alpha_s\sigma_{ts}^2}{\sigma_t^2}\boldsymbol{x}$ and $\sigma_{t\to s} = \frac{\sigma_{ts}^2\sigma_s^2}{\sigma_t^2}$. The literature shows [35] that by approximating $\mathcal{G}_0$ by a denoiser $\hat{\mathcal{G}}_0 = f_\theta(\mathcal{G}_t)$, we can define the learned distribution $p(\mathcal{G}_s|\mathcal{G}_t) = q(\mathcal{G}_s|\mathcal{G}_t, \boldsymbol{x} = \hat{\mathcal{G}}_0)$ without loss of generality as $s \to t$.

Here, we train the denoiser f_θ to output the input tri-plane $\mathcal{G}_0$ such that:

$$\mathcal{L}_{\text{t}}^{\text{ddpm}} := \mathbb{E}_{\mathcal{G},\epsilon\sim\mathcal{N}(0,1),t}\Big[w_t\|\mathcal{G}_0 - f_\theta(\mathcal{G}_t, t)\|_2^2\Big]. \tag{5.9}$$

where the denoiser $f_\theta(\circ, t)$ is realized as a time-conditional U-Net [205]. We choose an empirical $w_t = \text{S}(\text{SNR}(t))$ where $\text{SNR}(t) = \alpha_t^2/\sigma_t^2$ and S is the sigmoid function, as in [206].

5.1.3 Editing Mechanism

We emphasize three key advantages of our proposed method and explore their potential applications. **1)**: Local tri-planes with 3DMM template. In contrast to global-based 3D representations [31, 33, 146, 206–208] where each attribute is intricately entangled, our method gains advantage by integrating 3D scenes with local tri-planes. This approach allows for isolating and controlling local edits without unintended propagation to the global representation. Additionally, anchoring the tri-planes over 3DMM inherits the benefits of a 3DMM, and enables direct identity and expression editing. **2)**: UV-space parameterization. By rasterizing the 3DMM onto semantically consistent UV space, our method facilitates flexible region-based editing Specifically, we can directly transfer specific semantic regions [16], such as the mouth or nose, across identities by swapping the corresponding anchored Gaussians and tri-plane payloads. . Leveraging the trained diffusion model, we can further edit the region by diffusing the masked region in UV space to while keeping the remaining areas frozen. **3)** Geometry-texture disentanglement. Empowered by floatable tri-planes modulated by 3D Gaussians, a noteworthy byproduct benefit of LN3DIFF is the support of geometry-texture disentanglement. All the aforementioned editing applications can be conducted on either geometry, texture, or both.

5.2 Experiments

Dataset. To maintain quality and diversity, we sample 10,000 3D portraits from pre-trained Panohead [31] with diverse identities and expressions. For each identity, we render 50 images and depths with given camera poses. During reenacement inference, we select the real-world video used in IDE-3D [33] for visualization. We use the 64×64 view-consistent 3D renderings for tri-plane fitting, and the 512×512 samples for super-resolution training. As data pre-processing, we apply multi-view 3DMM fitting over each sample to initialize the anchored Gaussians. We filter out the low-quality samples using CLIP [209].

Implementation Details. We use $N = 1024$ local tri-planes to represent each 3D identity, given $H = W = 32$. During rendering, we adopt $k = 3$ for nearby tri-plane blending. For each local tri-plane, we have $S_x = S_y = C = 8$. The autodecoder D is implemented similarly to StyleGAN [86] with noise injection removed. After the D is trained, we further stack a $\times 4$ super-resolution model above it with the architecture from ESRGAN [210]. The denoiser f_θ is implemented as a 2D U-Net with architecture from Imagen [53]. The decoded UV maps of all instances are exported from the trained autodecoder D as the training corpus of the diffusion model f_θ.

Training Details. For autodecoder, we adopted AdamW optimizer with no weight decay for all the experiments, and set $lr = 0.05$ for optimizing $\mathbf{z}$, $lr = 0.0025$ for optimizing autodecoder D and $lr = 1e-5$ for optimizing Gaussians and $lr = 5e-2$ for optimizing a tiny 2-layer rendering MLP with $8 \times 32 \times 4$ shape. We decay all the learning rates at 100k steps by a factor of 0.5. For diffusion model training, we adhere to the diffusion model implementation as outlined in Imagen [211]. Note that we normalize the Gaussians and payload features to $[-1, 1]$ for better diffusion learning. Besides, we set the expression of the reconstructed 3D identities to the neutral state. The training is conducted in two stages: stage-1 auto-decodign triplane fitting takes 3 days, and stage-2 3D diffusion training takes 4 days. All experiments are performed using 2 A6000 GPUs. During inference, a 3D avatar head can be synthesized in around 45 seconds on a single A6000 GPU.

Evaluation Metrics. We select a series of proxy metrics to benchmark our method. Following [33], we evaluate view consistency assessed by multi-view facial identity consistency (ID) [186] rendered from random camera poses. To evaluate

TABLE 5.1: **Quantitative performance.** LOC3DIFF achieves competitive performance over 3D-related metrics (ID, Depth) and SoTA performance on the expression editing (PCK) performance. Additionally, yields faster rendering with less storage required with competitive 3D metrics.

Methods	ID ↑	Depth ↓	PCK@2.5 ↑	PCK@5↑	FPS ↑	Storage(MB)↓
FENeRF	0.61	2.71	-	-	1.2	10
Panohead	**0.80**	2.32	-	-	19	72
IDE-3D	0.76	**1.71**	0.16	0.33	25.1	48
Ours	0.78	2.58	**0.783**	**0.99**	**64/37**	**8.25**

the synthesized 3D geometry, we follow EG3D [1] to use an off-the-shelf tool to estimate depth maps from renderings and compute L2 distance against rendered depths. Moreover, we adopt an avatar-centric metric, Percentage of Correct Keypoints (PCK) [212] to evaluate the expression editing ability. The rendering speed and storage are also included.

5.2.1 Quantitative Comparisons

The results of numerical comparisons are presented in Tab. 5.1. Given that our method leverages Panohead data for training, it exhibits similar performance on ID and Depth metrics. In terms of expression editing ability, conventional global-based methods such as FeNeRF and Panohead do not support animation. Though IDE-3D supports segmentation-based reenactment, it lacks identity preservation and falls behind the PCK metric. LN3DIFF stands out as the only method that supports 3DMM-driven expression editing and achieving better PCK performance under both thresholds. Moreover, LN3DIFF supports faster rendering with less storage required for synthesized human heads.

5.2.2 Qualitative Evaluations

Auto-decoded Local Tri-planes. We first visualize the local tri-plane reconstruction from the autodecoder D in columns $1-4$ of Fig. 5.3. The reconstruction produces high-fidelity and view-consistent view synthesis. Additionally, the corresponding optimized Gaussians align with the identity shape of the input, showcasing the robust capacity of our design. Besides, though the tri-plane payloads are

centered on the 3DMM mesh surface, it can naturally support regions beyond the surface through volumetric rendering, in which way the hair and teeth are handled.

Expression Editing. We include the novel expression editing performance in columns $5-8$ Fig. 5.3. Despite being trained on collections of identities with a single expression, LN3Diff inherently supports 3DMM-based expression editing by manipulating the underlying Gaussian parameters of each local tri-plane. Furthermore, owing to the autodecoder design, LN3Diff can learn diverse expressions across identities, yielding natural-looking results under novel expressions.

FIGURE 5.4: We visualize the intermediate trajectory of transferring both texture and geometry from the source to the target. Both shape and texture interpolation results preserve high-fidelity during the middle state.

Shape-Texture Transfer. LN3Diff naturally supports geometry-texture disentanglement, where Gaussians manage the geometry and attached local tri-planes determine the texture within a local region defined on the UV map. We present the interpolation trajectory of the shape-texture transfer in Fig. 5.4, where both shape and texture are gradually added from the source identity to the target. The semantically meaningful intermediate results in both shape and texture interpolation validate the effectiveness of our design.

Unconditional Generation. Thanks to the compact UV space design, we can directly leverage powerful 2D diffusion architectures for 3D-aware generation. Specifically, we train a diffusion model $\boldsymbol{\epsilon}_\theta$ over the exported UV maps from the autodecoder D and include the diffusion generation results in Fig. 5.5. Visually inspected, the diffusion-generated results maintain the same high-fidelity and view-consistent renderings as the reconstruction results with diverse sampling. Compared with the previous tri-plane-based method [23, 31], LN3Diff maintains high capacity, and flexibility and intrinsically avoids Janus problem [42]. Besides, diffusion models process better editing ability compared with GAN-based methods.

Comparison with Existing Methods. We include the qualitative reenactment comparison of IDE-3D [33] to our method in Fig. 5.6. Both methods can accurately reflect the input expression in their reenactment results, although IDE-3D exhibits

FIGURE 5.5: **Unconditional Diffusion Sampling.** We showcase four samples with diverse poses. The compact UV space design allows us to leverage 2D diffusion architectures for 3D aware synthesis.

a deficiency in preserving the input identity, attributed to its segmentation-based animation paradigm. This difference accounts for their inferior PCK performance compared to the proposed method.

Moreover, IDE-3D and FE-NeRF lack support for region-based transfer between identities. This limitation arises from the fact that global-based representations such as tri-plane and MLP define all the 3D components in the 3D space. Attempting a direct copy-paste of one tri-plane region to another results in inconsistencies, especially when there are variations in their absolute sizes.

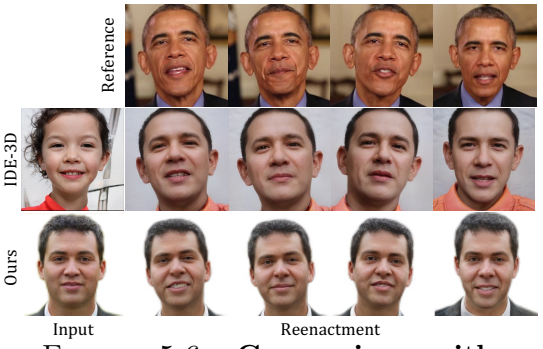


FIGURE 5.6: **Comparison with IDE-3D on Reenactment.** Employing four reference images (row-1), we animate the identities synthesized by IDE-3D and our method to the target expression. Given IDE-3D's segmentation-based nature, utilizing a reference identity with a different shape inevitably alters the facial layout. Nevertheless, LOC3DIFF preserves of the input identity through the reenactment process.

5.2.3 Applications

3D Inpainting using Diffusion Model. First, we showcase both the geometry- and texture-based inpainting in Fig. 5.7, where unmask the *upper face in the UV space* and let the diffusion model inpaints the remaining areas. Both yield holistically reasonable results while keeping the corresponding inputs within the mask sharing a similar pattern.

3DMM-based Editing. LN3DIFF marries the best of both the model-based 3DMM and neural representations through the rasterized UV space and naturally

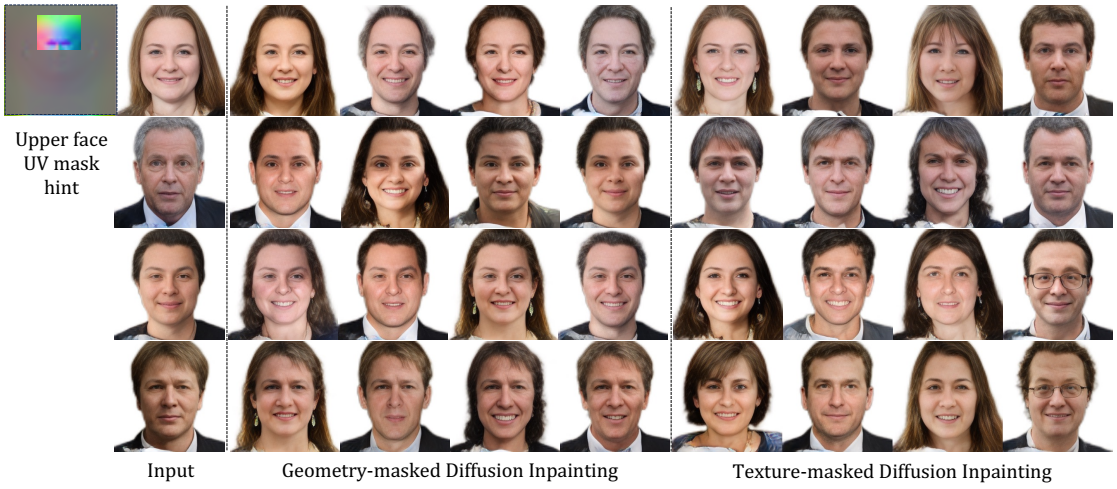


FIGURE 5.7: **Diffusion-based Inpainting given Geometry or Texture Mask.** We provide either the geometry part (first 9 channels of the $256 \times 256 \times 33$ tensor $\mathcal{G}_0$) or the texture part (last 24 channels of the $256 \times 256 \times 33$ tensor $\mathcal{G}_0$) of the *upper face* as hints, and take the diffusion model f_θ in-paints the remaining details. As illustrated in columns $2-5$, all the generated results exhibit a well-maintained upper-face shape, including the layout of the eyes and forehead, but with different textures. Conversely, the diffusion-inpainted results with texture masks (columns $6-10$) showcase an well-preserved upper-face texture, encompassing features such as hair and forehead color, while varying in shape.

supports 3DMM-based editing by changing the shape and expression codes. Specifically, by driving the anchored Gaussians w.r.t the vertices offsets decoded from changed 3DMM expression code, we support 3DMM-based reenactment. We showcase this ability in Fig. 5.6, where we reenact the given reference expression to the target. As can be seen, the reenactment results maintain their original identity but accurately follow the expression of the given source input. This application has the potential to facilitate avatar editing in game engines and media creation.

Regional-based Editing. In addition to global interpolation and transfer capabilities, LN3DIFF provides support for region-based editing, allowing modification exclusively within the semantic region defined by the UV mask. This functionality is illustrated in Fig. 5.8, where we showcase the transfer of corresponding source Gaussians (geometry) to the target, guided by the provided mask. The transferred results exhibit the same shape as the source within the defined semantic region, while the remaining areas remain unchanged. Benefiting from the UV space design, regional editing consistently produces semantically consistent results when transferring between mouth/nose regions of varying sizes across different identities.

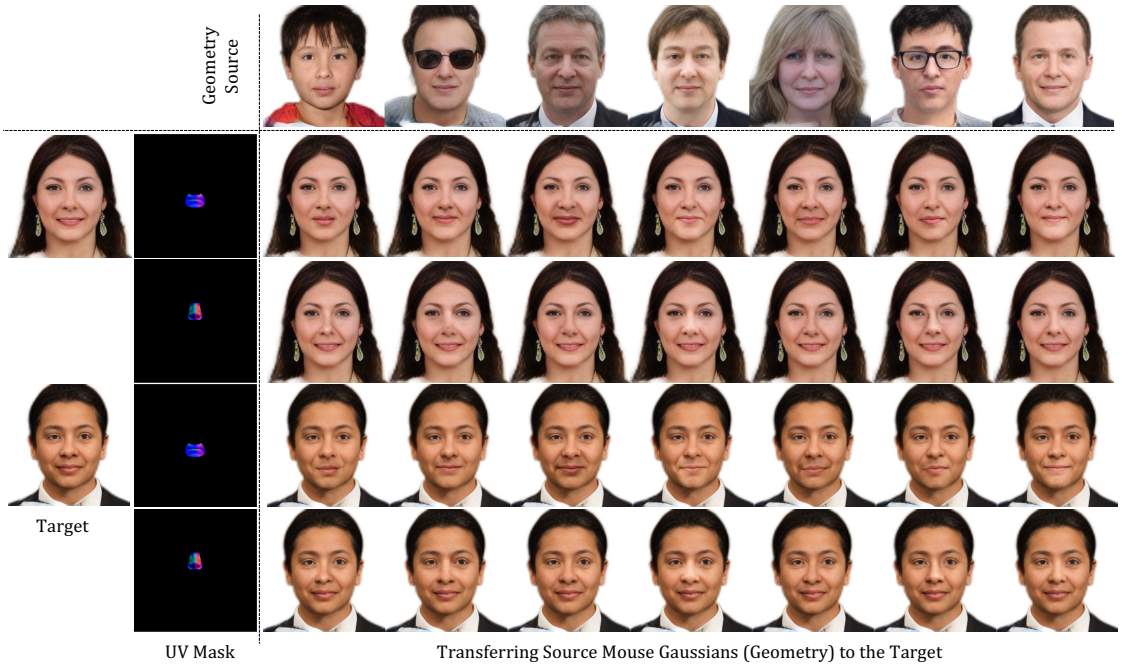


FIGURE 5.8: **Regional Editing with UV Mask**. The UV-space design in LOC3DIFF enables the editing of specific semantic regions defined on the UV map. In this example, we transfer the geometry shapes of the "nose" and "mouth" from the source identities to the target while leaving the remaining areas unchanged. It is important to note that all the source and target identities presented here are diffusion-sampled.

Furthermore, this demonstration underscores that LN3DIFF can surpass 3DMM constraints, enhancing controllability and exhibiting significant potential for avatar personalization.

5.2.4 Ablation Study

We ablate the detailed design choice of local tri-plane payload here and the choice of K value in tri-plane blending.

Local Tri-plane. In our early experiments, we opted for a pure feature vector [213] as the local payload to represent the textures within a local region. However, we observed that the reconstruction performance consistently reached limitations. As visualized in Fig. 5.9, this motivated us to employ a tiny tri-plane as the texture payload. For both settings, we utilized 1024 local payloads to represent a scene and trained two variations till convergence. The results indicate that using a pure feature vector as the payload results in blurry view synthesis with a PSNR 21db and noisy depths. Conversely, our local tri-plane payload variations exhibit improved

fidelity with PSNR 32db and cleaner surface reconstruction. We also ablate detailed choice of the latent space with the same configuration as Tab. 5.3. Adopting $N = 32 \times 32$ with $S_x = S_y = C = 8$ achieves the best trade-off between speed, VRAM usage and performance. Moreover, a smaller UV space with $N = 32 \times 32$ facilitates efficient diffusion training.

TABLE 5.2: **Quantitative Ablation of k in Gaussian Blending.** Larger k improves reconstruction quality, albeit requiring extra computational resources. However, a performance decline is observed for larger k.

k	PSNR ↑	MAE ↓	Speed ↓
1	27.72	0.0421	0.4
2	27.92	0.0412	0.53
3	**28.39**	**0.0392**	0.67
4	28.28	0.0400	0.8
5	28.25	0.0395	0.87

TABLE 5.3: **Ablation of N, S_x, S_y, C.** We follow the same ablation setting as Tab. 6.3. Increasing the UV size N and spatial tri-plane size S_x, S_y improves the quality of reconstruction, albeit with an associated increase in the computational resources. Thus, we choose a balanced setting for all the experiments.

N	S_x, S_y, C	PSNR ↑	Speed ↓	VRAM ↓
1024	4, 4, 4	26.89	0.44	4985
1024	4, 4, 8	27.54	0.61	5358
1024	8, 8, 8	28.39	0.67	5810
2304	8, 8, 8	28.95	0.99	9872

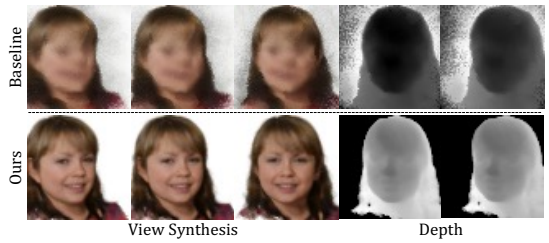


FIGURE 5.9: **Ablation Study on Local Tri-plane.** Using raw feature vectors as the payload lacks the ability to encode spatial information, and our local tri-plane design holds larger capacity and better reconstruction results.

Details Selection of k Nearby Tri-planes for Blending. We investigate the choice of k value in Gaussian blending and include the quantitative results in Tab. 5.2. As it can be observed, the reconstruction performance initially improves steadily with increasing k, as does the computational cost. To trade off the overall performance, we adopt $k = 3$ in our experiments. However, efficient training with larger k is still worth exploring.

5.3 Conclusion

This chapter have introduced Loc3Diff, a new 3D generative framework, and demonstrated its promising results across various scenarios. A novel representation based on 3DMM anchored by 3D Gaussians-modulated local tri-planes is first introduced, which allows us to decouple the underlying smooth geometry and deformation from the complex volumetric appearance. Importantly, the proposed representation can be stored in the UV space that is amenable to generative modelling. We then proposed a method to simultaneously reconstruct and learn a latent space for our 3D representations via multi-view supervision, upon which we train a 2D diffusion model to perform various editing tasks. In general, the proposed method improves the generality of neural fields within the 3D head domain. Compared to existing per-instance optimization-based methods [146], the proposed method achieves 3D generaiton from a random Gaussian noise. Besides, the proposed method supports 3D editing (inpainting), texture transfer, and reenactment across identities, which cannot be achieved with previous per-instance optimization methods.

Though performing good on the avatar, a 3D diffusion framework for general objects is still necessary. In the following chapter, this thesis will thoroughly investigate a unified 3D framework to efficiently generate high-quality 3D objects within seconds.

Chapter 6

LN3Diff: Scalable Latent Neural Fields Diffusion for Speedy 3D Generation

The advancement of generative models [36, 85] and differentiable rendering [214] has paved the way for a new research direction called neural rendering [214]. This field is continuously pushing the limits of view synthesis [2], editing [16], and particularly 3D object synthesis [1]. While 2D diffusion models [35, 36] have outperformed GANs in image synthesis [37] in terms of quality [39], controllability [40], and scalability [41], a unified 3D diffusion pipeline has yet to be established.

3D object generation methods using diffusion models can be categorized into 2D-lifting and feed-forward 3D diffusion models. In 2D-lifting methods, score distillation sampling (SDS) [42, 131] and Zero-123 [43, 215] achieve 3D generation by leveraging pre-trained 2D diffusion models. However, SDS-based methods require costly per-instance optimization and are prone to the multi-face Janus problem [42]. Meanwhile, Zero-123 fails to enforce strict view consistency. On the other hand, feed-forward 3D diffusion models [17, 23, 44–47] enable fast 3D synthesis without per-instance optimization. However, these methods typically involve a two-stage pre-processing approach. First, during the data preparation stage, a shared decoder

The work in this chapter has been published in [24].
Code is publicly available at `https://github.com/NIRVANALAN/LN3Diff`

is learned over a large number of instances to ensure a shared latent space. This is followed by per-instance optimization to convert each 3D asset in the datasets into neural fields [9]. After this, the feed-forward diffusion model is trained on the prepared neural fields.

While the pipeline above is straightforward, it poses extra challenges to achieve high-quality 3D diffusion: **1) Scalability.** In the data preparation stage, existing methods face scalability issues due to using a shared, low-capacity MLP decoder for per-instance optimization. This approach is data inefficient, requiring over 50 views per instance [44, 46] during training. Consequently, computation cost scales linearly with the dataset size, hindering scalability for large, diverse 3D datasets. **2) Efficiency.** Employing 3D-specific architectures [48–50] is computationally intensive and necessitates representation-specific designs [51]. Consequently, existing methods compress each 3D asset into neural fields [9] before training. However, this compression introduces high-dimensional 3D latent, increasing computational demands and training challenges. Limiting the neural field size [46] might mitigate these issues but at the cost of reconstruction quality. In addition, the auto-decoding paradigm can result in an unclean latent space [38, 45, 216], unsuitable for 3D diffusion training [39]. **3) Generalizability.** Existing 3D diffusion models primarily focus on unconditional generation over single classes, neglecting high-quality conditional 3D generation (e.g., text-to-3D) across generic, category-free 3D datasets. Furthermore, projecting monocular input images into the diffusion latent space is crucial for conditional generation and editing [7, 40], but this is challenging with the shared decoder designed for multi-view inputs.

In this study, a novel framework called **L**atent **N**eural fields **3**D **Diff**usion (LN3Diff) is proposed to address these challenges and enable fast, high-quality and generic conditional 3D generation. Our method involves training a variational autoencoder [54] (VAE) to compress input images into a lower-dimensional 3D-aware latent space, which is more expressive and flexible compared to pixel-space diffusion [35–37, 135]. From this space, a 3D-aware transformer-based decoder gradually decodes the latent into a high-capacity 3D neural field. This autoencoding stage is trained amortized with differentiable rendering [214], incorporating novel view supervision for multi-view datasets [68, 217] and adversarial supervision for monocular dataset [187]. Thanks to the high-capacity model design, our method

is more *view efficient*, requiring only two views per instance during training. After training, the learned 3D latent space is leveraged for conditional 3D diffusion learning, ensuring effective utilization of the trained model for high-quality 3D generation. The pre-trained encoder can amortize the data encoding over incoming data, thus streamlining operations and facilitating efficient 3D diffusion learning while remaining compatible with advances in 3D representations.

To enhance efficient information flow in the 3D space and promote coherent geometry reconstruction, a novel 3D-aware architecture is introduced, which is tailored for fast and high-quality 3D reconstruction while maintaining a structured latent space. Specifically, a convolutional tokenizer is employed to encode the input image(s) into a *KL*-regularized 3D latent space, leveraging its superior perceptual compression ability [218]. The transformers [219, 220] is employed to enable flexible 3D-aware attention across 3D tokens in the latent space. Finally, the 3D latent is upsampled and apply differentiable rendering for image-space supervision, making our method a self-supervised 3D learner [221].

In summary, a 3D-representation-agnostic pipeline is proposed for building generic, high-quality 3D generative models. This pipeline provides opportunities to resolve a series of downstream 3D vision and graphics tasks. Specifically, a novel 3D-aware reconstruction model is proposed that achieves high-quality 3D data encoding in an amortized manner. Learning in the compact latent space, our model demonstrates state-of-the-art 3D generation performance on the ShapeNet benchmark [68], surpassing both Generative Adversarial Network (GAN)-based and 3D diffusion-based approaches. Our method shows superior performance in monocular 3D reconstruction and conditional 3D generation on ShapeNet, FFHQ, and Objaverse datasets, with a fast inference speed, e.g., 3× faster against existing latent-free 3D diffusion methods [136].

6.1 Methodology

This section introduces our latent 3D diffusion model, which learns efficient diffusion prior over the compressed latent space by a dedicated variational autoencoder. Specifically, the goal of training is to learn a variational encoder $\mathcal{E}_{\phi}$ that maps a set of posed 2D image(s) $\mathcal{I} = \{x_i, ..., x_V\}$, of a 3D object to a latent code $\mathbf{z}$, a

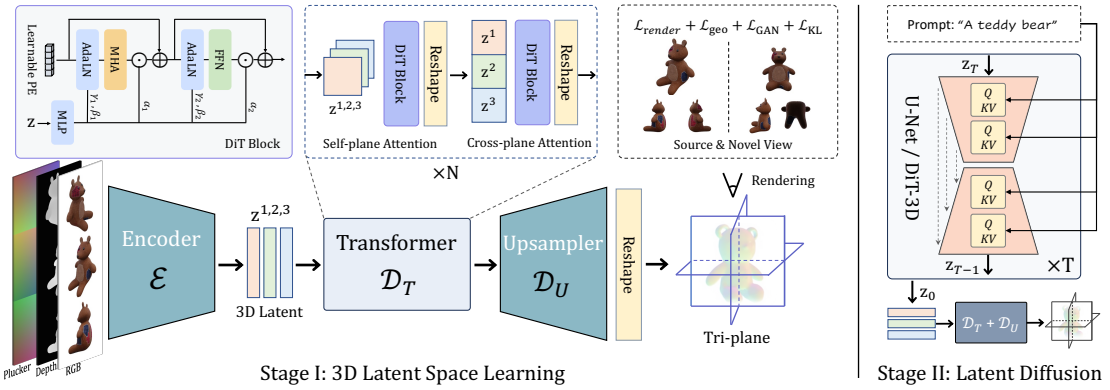


FIGURE 6.1: **Pipeline of LN3Diff.** In the 3D latent space learning stage, a convolutional encoder $\mathcal{E}_\phi$ encodes a set of images $\mathcal{I}$ into the KL-regularized latent space. The encoded 3D latent is further decoded by a 3D-aware DiT transformer $\mathcal{D}_T$, in which we perform self-plane attention and cross-plane attention. The transformer-decoded latent is up-sampled by a convolutional upsampler $\mathcal{D}_U$ towards a high-res tri-plane for rendering supervisions. In the next stage, we perform conditional diffusion learning over the compact latent space using either U-Net or DiT.

denoiser $\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t)$ to denoise the noisy latent code $\mathbf{z}_t$ given diffusion time step t, and a decoder $\mathcal{D}_{\psi}$ (including a Transformer $\mathcal{D}_T$ and an Upsampler $\mathcal{D}_U$) to map $\mathbf{z}_0$ to the 3D tri-plane $\widetilde{\mathcal{X}}$ corresponding to the input object.

Such design offers several advantages: (1) By explicitly separating the 3D data compression and diffusion stage, we avoid representation-specific 3D diffusion design [38, 44, 45, 51, 136] and achieve 3D representation/rendering-agnostic diffusion, which can be applied to any neural rendering techniques. (2) By leaving the high-dimensional 3D space, we reuse the well-studied Latent Diffusion Model (LDM) architecture [39, 219, 222] for computationally efficient learning and achieve better sampling performance with faster speed. (3) The trained 3D compression model in the first stage serves as an efficient and general-purpose 3D tokenizer, whose latent space can be easily re-used over downstream applications or extended to new datasets [18, 223].

In the following subsections, we first discuss the compressive stage with a detailed framework design in Sec. 6.1.1. Based on that, we introduce the 3D diffusion generation stage in Sec. 6.1.2 and present the condition injection in Sec. 6.1.3. The method overview is shown in Fig. 3.2.

6.1.1 Perceptual 3D Latent Compression

As analyzed previously, directly leveraging neural fields for diffusion training hinders model scalability and performance. Inspired by previous work [39, 218], we propose to take multi-view image(s) as a proxy of the underlying 3D scene and compress the input image(s) into a compact 3D latent space. Though this paradigm is well-adopted in the image domain [39, 218] with similar trials in specific 3D tasks [7, 105, 120, 224], we, for the first time, demonstrate that a high-quality compression model is feasible, whose latent space serves as a compact proxy for efficient diffusion learning.

Encoder. Given a set of image(s) $\mathcal{I}$ of an 3D object where each image within the set $x \in \mathbb{R}^{H\times W\times 3}$ is an observation of an underlying 3D object from viewpoints $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_V\}$, LN3DIFF adopts a convolutional encoder $\mathcal{E}_\phi$ to encode the image set $\mathcal{I}$ into a latent representation $\mathbf{z} \sim \mathcal{E}_\phi(\mathcal{I})$. To inject camera condition, we concatenate Plucker coordinates $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{p}_i \times \mathbf{d}_i) \in \mathbb{R}^6$ channel-wise as part of the input [225], where $\mathbf{d}_i$ is the normalized ray direction and $\mathbf{p}_i$ is the camera origin corresponding to the camera $\mathbf{c}_i$, and $\times$ denotes the cross product. For challenging datasets like Objaverse [217], we also concatenate the rendered depth map, making our input a dense 3D colored point cloud [226].

Unlike existing works [120, 218] that operate on 1D order latent and ignore the internal structure, we choose to output 3D latent $\mathbf{z} \in \mathbb{R}^{h\times w\times d\times c}$ to facilitate 3D-aware operations, where $h = H/f, w = W/f$ are the spatial resolution with downsample factor f, and d denotes the 3D dimension. Here we set $f = 8$ and $d = 3$ to make $\mathbf{z} \in \mathbb{R}^{h\times w\times 3\times c}$ a tri-latent, which is similar to tri-plane [1, 173] but in the compact 3D latent space. We further impose *KL-reg* [54] to encourage a well-structured latent space to facilitate diffusion training [39, 222].

Decoder Transformer. The decoder aims to decode the compact 3D codes $\mathbf{z}$ for high-quality 3D reconstruction. Existing image-to-3D methods [1, 7, 105, 216] adopt convolution as the building block, which lacks 3D-aware operations and impede information flow in the 3D space. Here, we adopt ViT [219, 220] as the decoder backbone due to its flexibility and effectiveness. Inspired by Rodin [23], we made the following reformulation to the raw ViT decoder to encourage 3D inductive bias and avoid the mixing of uncorrelated 3D features: (1) *Self-plane Attention Block*. Given the input $\mathbf{z} \in \mathbb{R}^{l\times 3\times c}$ where $l = h \times w$ is the sequence

length, we treat each of the three latent planes as a data point and conduct self-attention within itself. This operation is efficient and encourages local feature aggregation. (2) *Cross-plane Attention Block*. To further encourage 3D inductive bias, we roll out $\mathbf{z}$ as a long sequence $l \times 3 \times c \rightarrow 3l \times c$ to conduct attention across planes, so that all tokens in the latent space could attend to each other. In this way, we encourage global information flow for more coherent 3D reconstruction and generation. Compared to Rodin, our design is fully attention-based and naturally supports parallel computing without the expensive axis pooling aggregation.

Empirically, we observe that using DiT [219] block and injecting the latent $\mathbf{z}$ as conditions yields better performance compared to the ViT [220, 227] block, which takes the latent $\mathbf{z}_0$ as the regular input. Specifically, the adaptive layer norm (adaLN) layer [219] fuses the input latent $\mathbf{z}$ with the learnable positional encoding for attention operations. Moreover, we interleave the two types of attention layers to make sure the overall parameters count consistent with the pre-defined DiT length, ensuring efficient training and inference. As all operations are defined in the token space, the decoder achieves efficient computation against Rodin [23] while promoting 3D priors.

Decoder Upsampler. After all the attention operations, we obtain the tokens from the last transformer layer $\widetilde{\mathbf{z}}$ as the output. The context-rich tokens are reshaped back into spatial domain [228] and up-sampled by a convolutional decoder to the final tri-plane representation with shape $\hat{H} \times \hat{W} \times 3C$. Here, we adopt a lighter version of the convolutional decoder for efficient upsampling, where the three spatial latent of $\widetilde{\mathbf{z}}$ are processed in parallel.

Learning a Perceptually Rich and Intact 3D Latent Space. Adversarial learning [85] has been widely applied in learning a compact and perceptually rich latent space [39, 218]. In the 3D domain, the adversarial loss can also encourage correct 3D geometry when novel-view reconstruction supervisions are inapplicable [21, 120, 229], e.g., the monocular dataset such as FFHQ. Inspired by previous research [120, 229], we leverage adversarial loss to bypass this issue. Specifically, we impose an input-view discriminator to maintain perceptually-reasonable input view reconstruction, and an auxiliary novel-view discriminator to distinguish the rendered images between the input and novel views. We observe that if asking the novel-view discriminator to differentiate novel-view renderings and real images instead, the reconstruction model will suffer from *posterior collapse* [230], which

outputs input-irrelevant but high-fidelity results to fool the novel-view discriminator. This phenomenon has also been observed by Kato *et al.* [229].

Training. After the decoder $\mathcal{D}_{\psi}$ decodes a high-resolution neural fields $\hat{\mathbf{z}}_0$ from the latent, we have $\hat{x} = \mathtt{R}(\widetilde{\mathcal{X}}) = \mathtt{R}(\mathcal{D}_{\psi}(\mathbf{z})) = \mathtt{R}(\mathcal{D}_{\psi}(\mathcal{E}_{\phi}(\S)))$, where $\mathtt{R}$ stands for differentiable rendering [214] and we take $\mathcal{D}_{\psi}(\mathbf{z}) = \mathtt{R}(\mathcal{D}_{\psi}(\mathbf{z}))$ for brevity. Here, we choose $\widetilde{\mathcal{X}}$ as tri-plane [1, 173] and $\mathtt{R}$ as volume rendering [2] for experiments. Note that our compression model is 3D representations/rendering agnostic and new neural rendering techniques [200] can be easily integrated by alternating the decoder architecture [231]. The final training objective reads as

$$\mathcal{L}(\phi, \psi) = \mathcal{L}_{\text{render}} + \lambda_{\text{geo}}\mathcal{L}_{\text{geo}} + \lambda_{\text{kl}}\mathcal{L}_{\text{KL}} + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}}, \tag{6.1}$$

where $\mathcal{L}_{\text{render}}$ is a mixture of the L_1 and perceptual loss [171], $\mathcal{L}_{\text{reg}}$ encourages smooth geometry [232], $\mathcal{L}_{\text{KL}}$ is the *KL-reg* loss to regularize a structured latent space [39], and $\mathcal{L}_{\text{GAN}}$ improves perceptual quality and enforces correct geometry for monocular datasets. Note that $\mathcal{L}_{\text{render}}$ is applied to both input-view and randomly sampled novel-view images.

For category-specific datasets such as ShapeNet [68], we only supervise *one* novel view, which already yields good enough performance. For category-free datasets with diverse shape variations, e.g., Objaverse [217], we supervise *four* novel views. Our method is more data-efficient against the existing state-of-the-art 3D diffusion method [44, 46], which requires 50 views to converge. The implementation details are included in the supplementary material.

6.1.2 Latent Diffusion and Denoising

Latent Diffusion Models. LDM [39, 222] is designed to acquire a prior distribution $p_{\boldsymbol{\theta}}(\mathbf{z}_0)$ within the perceptual latent space, whose training data is the latent obtained online from the trained $\mathcal{E}_{\phi}$. Here, we use the score-based latent diffusion model [222], which is the continuous derivation of DDPM variational objective [36]. Specifically, the denoiser $\boldsymbol{\epsilon}_{\theta}$ parameterizes the score function score [35] as $\nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t) := -\boldsymbol{\epsilon}_{\theta}(\mathbf{z}_t, t)/\sigma_t$, with continuous time sequence t. By training to predict a denoised variant of the noisy input $\mathbf{z}_t$, $\boldsymbol{\epsilon}_{\theta}$ gradually learns to denoise from a standard Normal prior $\mathcal{N}(\mathbf{0}, \mathbf{I})$ by solving a reverse SDE [36].

Following LSGM [222], we formulate the learned prior at time t geometric mixing $\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t) := \sigma_t(1-\boldsymbol{\alpha}) \odot \mathbf{z}_t + \boldsymbol{\alpha} \odot \boldsymbol{\epsilon}'_\theta(\mathbf{z}_t, t)$, where $\boldsymbol{\epsilon}'_\theta(\mathbf{z}_t, t)$ is the denoiser output and $\alpha \in [0, 1]$ is a learnable scalar coefficient. Intuitively, this formulation can bring the denoiser input closer to a standard Normal distribution, which the reverse SDE can be solved faster. Similarly, Stable Diffusion [39, 233] also scales the input latent by a factor to maintain a unit variance, which is pre-calculated on the billion-level dataset [41]. The training objective reads as

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{\mathcal{E}_\phi(x), \boldsymbol{\epsilon} \sim \mathcal{N}(0,1), t} \left[\frac{w_t}{2} \| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t) \|_2^2 \right], \tag{6.2}$$

where $t \sim \mathcal{U}[0, 1]$ and w_t is an empirical time-dependent weighting function.

The denoiser $\boldsymbol{\epsilon}_\theta$ is realized by a time-dependent U-Net [234] or DiT [219]. During training, we obtain $\mathbf{z}_0$ online from the fixed $\mathcal{E}_\phi$, roll-out the tri-latent $h \times w \times 3 \times c \rightarrow h \times (3w) \times c$, and add time-dependent noise to get $\mathbf{z}_t$. Here, we choose the importance sampling schedule [222] with *velocity* [235] parameterization, which yields more stable behavior against ϵ parameterization for diffusion learning. After training, the denoised samples can be decoded to the 3D neural field (i.e., tri-plane here) with a single forward pass through $\mathcal{D}_\psi$, on which neural rendering can be applied.

6.1.3 Conditioning Mechanisms

Compared with the existing approach that focuses on category-specific unconditional 3D diffusion model [136], we propose to inject CLIP embeddings [209] into the latent 3D diffusion model to support image/text-conditioned 3D generation. Given input condition $\mathbf{y}$, the diffusion model formulates the conditional distribution $p(\mathbf{z}|\mathbf{y})$ with $\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, t, \mathbf{y})$. The inputs $\mathbf{y}$ can be text captions for datasets like Objaverse, or images for general datasets like ShapeNet and FFHQ.

Text Conditioning. For datasets with text caption, we follow Stable Diffusion [39] to directly leverage the CLIP text encoder CLIP_T to encode the text caption as the conditions. All the output tokens 77×768 are used and injected to diffusion denoiser with cross attention blocks.

Image Conditioning. For datasets with images only, we encode the input x corresponding to the latent code $\mathbf{z}_0$ using CLIP image encoder CLIP_I and adopt

the output embedding as the condition. To support both image and text conditions, we re-scale the image latent code with a pre-calculated factor to match the scale of the text latent.

Classifier-free Guidance. We adopt *classifier-free guidance* [236] for latent conditioning to support conditional and unconditional generation. During diffusion model training, we randomly zero out the corresponding conditioning latent embeddings with 15% probability to jointly train the unconditional and conditional settings. During sampling, we perform a linear combination of the conditional and unconditional score estimates:

$$\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_t, \boldsymbol{\tau}_\theta(y)) = s\boldsymbol{\epsilon}_\theta(\mathbf{z}_t, \boldsymbol{\tau}_\theta(y)) + (1-s)\boldsymbol{\epsilon}_\theta(\mathbf{z}_t), \tag{6.3}$$

where s is the guidance scale to control the mixing strength to balance sampling diversity and quality.

6.2 Experiments

Datasets. Following most previous work, we use ShapeNet [68] to benchmark the 3D generation performance. We use the Car, Chair, and Plane categories with 3514, 6700, and 4045 instances correspondingly. Each instance is randomly rendered from 50 views following a spherical uniform distribution. The evaluation is conducted over the same dataset following the split of SRN [221]. Moreover, to evaluate the performance over diverse high-quality 3D datasets, we also include the experiments over Objaverse [217], which is the largest 3D dataset with challenging categories and complicated geometry. We use the renderings provided by G-Objaverse [237] and choose a high-quality subset with around $176K$ 3D instances for training, where each consists of 40 random views.

Training Details. For ShapeNet and FFHQ training, we adopt a monocular input setting with $V = 1$ and target rendering size 128×128. For Objaverse, we adopt multi-view inputs with $V = 6$ and target rendering size 192×192. All training images are resized to $H = W = 256$ as input. The encoder $\mathcal{E}_\phi$ has down-sample factor $f = 8$ and the decoder upsampler $\mathcal{D}_U$ outputs tri-plane with size $\hat{H} = \hat{W} = 128$ and $C = 32$. To trade off rendering resolution and training batch size, we impose supervision over 80×80 randomly cropped patches. For

adversarial loss, we use DINO [227] in vision-aided GAN [238] with non-saturating GAN loss [239] for discriminator training. For conditional diffusion training, we use the CLIP image embedding for ShapeNet and FFHQ, and CLIP text embedding from the official text caption for Objaverse. Both the autoencoding model and diffusion model are trained for $800K$ iterations, which take around 7 days with 8 A100 GPUs in total.

Metrics. Following prior work [38, 44, 46, 240], we adopt both 2D and 3D metrics to benchmark the generation performance: Fréchet Inception Distance (FID@50K) [241] and Kernel Inception Distance (KID@50K) [242] to evaluate 2D renderings, as well as Coverage Score (COV) and Minimum Matching Distance (MMD) to benchmark 3D geometry. We calculate all metrics under 128×128 for fair comparisons across all baselines.

6.2.1 Evaluation

In this section, we compare our method with both state-of-the-art GAN-based methods: EG3D [1], GET3D [240] as well as recent diffusion-based methods: DiffRF [44], RenderDiffusion [136] and SSDNeRF [46]. Since LN3Diff only leverages $v = 2$ for ShapeNet experiments, for SSDNeRF, we include both the official 50-views trained $\text{SSDNeRF}_{\text{V=50}}$ version as well as the reproduced $\text{SSDNeRF}_{\text{V=3}}$ for fair comparison. We find SSDNeRF fails to converge with $V = 2$. We set the guidance scale in Eq. (6.3) to $s = 0$ for unconditional generation, and $s = 6.5$ for all conditional generation sampling.

Unconditional Generation on ShapeNet. To evaluate our methods against existing 3D generation methods, we include the quantitative and qualitative results for unconditional single-category 3D generation on ShapeNet in Tab. 6.1 and Fig. 6.2. We evaluate all baseline 3D diffusion methods with 250 DDIM steps and GAN-based baselines with $psi = 0.8$ to guarantee each sample is intact for COV/MMD evaluation. For FID/KID evaluation, we re-train the baselines and calculate the metrics using a fixed upper-sphere ellipsoid camera trajectory [221] across all datasets. For COV/MMD evaluation, we randomly sample 4096 points around the extracted sampled mesh and ground truth mesh surface and adopt Chamfer Distance for evaluation.

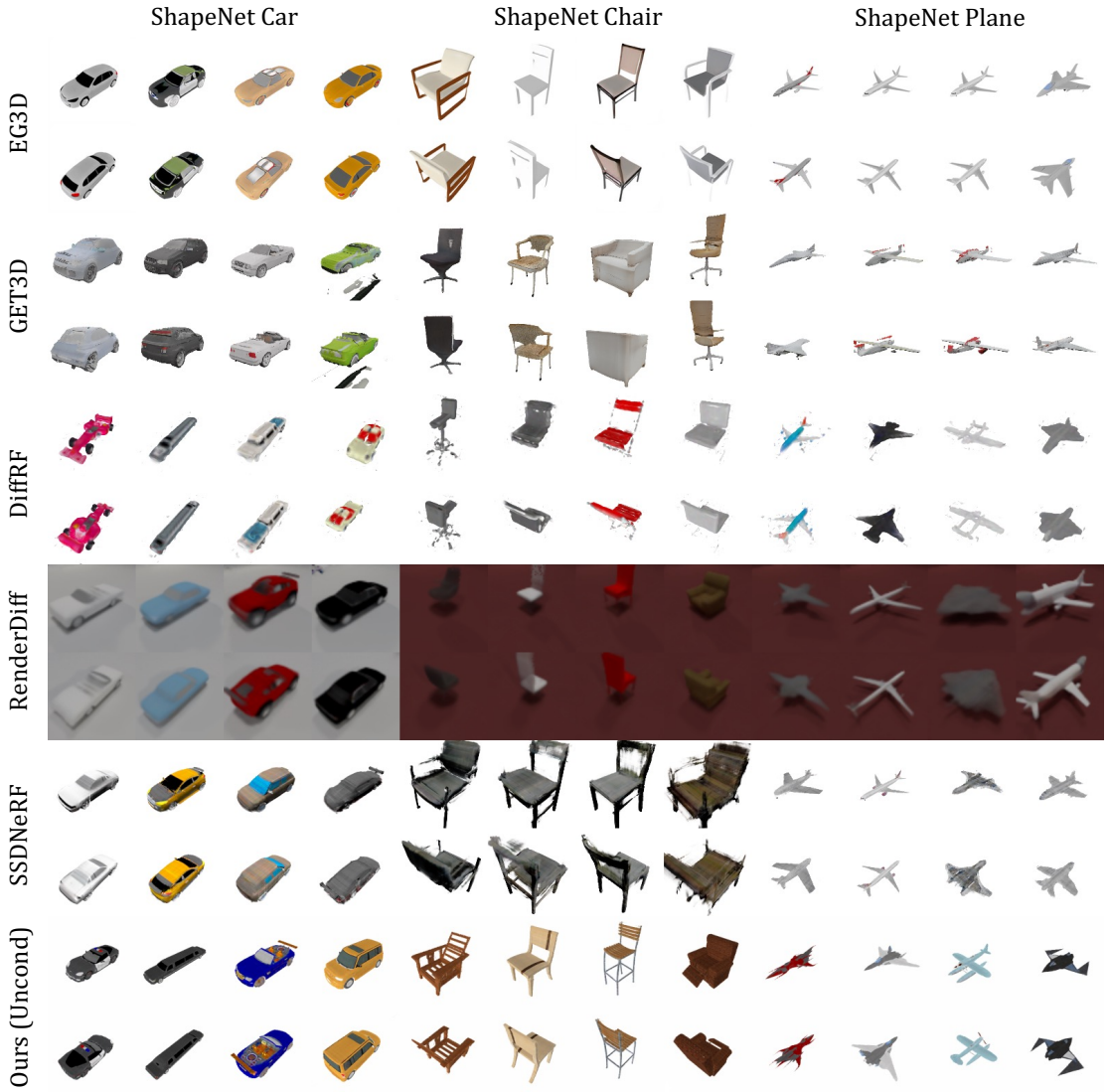


FIGURE 6.2: **ShapeNet Unconditional Generation.** We show four samples for each method. Zoom in for the best view.

As shown in Tab. 6.1, LN3DIFF achieves quantitatively better performance against all GAN-based baselines regarding rendering quality and 3D coverage. Fig. 6.2 further demonstrates that GAN-based methods suffer greatly from mode collapse: in the ShapeNet Plane category, both EG3D and GET3D are limited to the white civil airplanes, which is fairly common in the dataset. Our methods can sample more diverse results with high-fidelity texture.

Compared against diffusion-based baselines, LN3DIFF also shows better visual quality with better quantitative metrics. SSDNeRF$_{V=50}$ shows a better coverage score, which benefits from leveraging more views during training. However, our

TABLE 6.1: **Quantitative Comparison of Unconditional Generation on ShapeNet**. It proves the impressive performance of our LN3DIFF on the single-category generation.

Category	Method	FID↓	KID(%)↓	COV(%)↑	MMD(‰)↓
Car	EG3D [1]	33.33	1.4	35.32	3.95
	GET3D [240]	41.41	1.8	37.78	3.87
	DiffRF [44]	75.09	5.1	29.01	4.52
	RenderDiffusion [136]	46.5	4.1	-	-
	$\text{SSDNeRF}_{\text{v=3}}$ [46]	47.72	2.8	37.84	3.46
	$\text{SSDNeRF}^{*}_{\text{v=50}}$ [46]	45.37	2.1	**67.82**	2.50
	LN3DIFF**(Ours)**	**17.6**	**0.49**	43.12	**2.32**
Plane	EG3D [1]	14.47	0.54	18.12	4.50
	GET3D [240]	26.80	1.7	21.30	4.06
	DiffRF [44]	101.79	6.5	37.57	3.99
	RenderDiffusion [136]	43.5	5.9	-	-
	$\text{SSDNeRF}_{\text{v=3}}$ [46]	21.01	1.0	42.50	2.94
	LN3DIFF**(Ours)**	**8.84**	**0.36**	**43.40**	**2.71**
Chair	EG3D [1]	26.09	1.1	19.17	10.31
	GET3D [240]	35.33	1.5	28.07	9.10
	DiffRF [44]	99.37	4.9	17.05	14.97
	RenderDiffusion [136]	53.3	6.4	-	-
	$\text{SSDNeRF}_{\text{v=3}}$ [46]	65.04	3.0	**47.54**	6.71
	LN3DIFF**(Ours)**	**16.9**	**0.47**	47.1	**5.28**

TABLE 6.2: **Quantitative Metrics on Text-to-3D.** The proposed method outperforms Point-E and Shape-E on CLIP scores over two different backbones.

Method	VIT-B/32	VIT-L/14
Point-E [243]	26.35	21.40
Shape-E [45]	27.84	25.84
Ours	**29.12**	**27.80**

method with $V = 2$ shows comparative performance against $\text{SSDNeRF}_{\text{V=3}}$ on the ShapeNet Chair and even better performance on the remaining datasets.

Conditional 3D Generation. Conditional 3D generation has the potential to streamline the 3D modeling process in both the gaming and film industries. As visualized in Fig. 6.3, we present our conditional generation performance on the ShapeNet dataset, where either text or image serves as the input prompt. Visually inspected, our method demonstrates promising performance in conditional generation, closely aligning the generated outputs with the input conditions. For

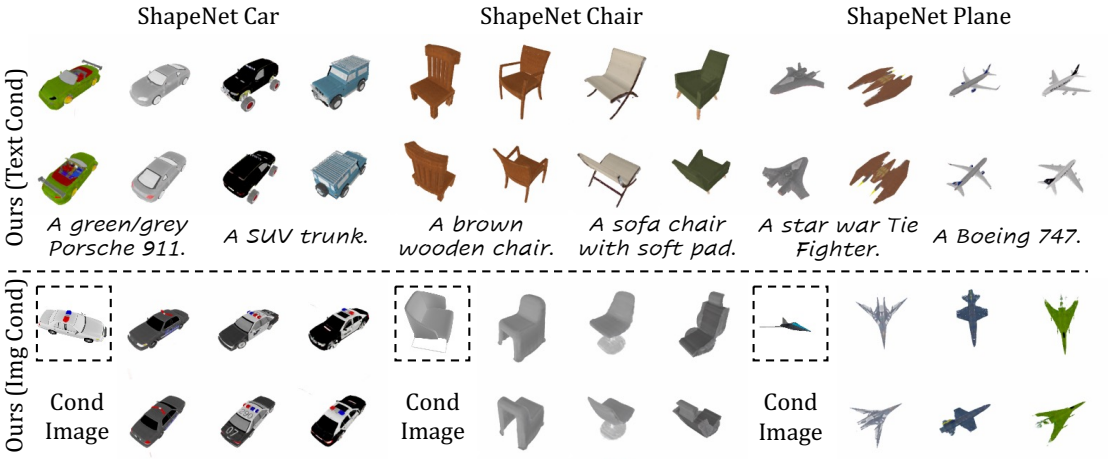


FIGURE 6.3: **ShapeNet Conditional Generation.** We show conditional generation with both texts and image as inputs. Zoom in for the best view.

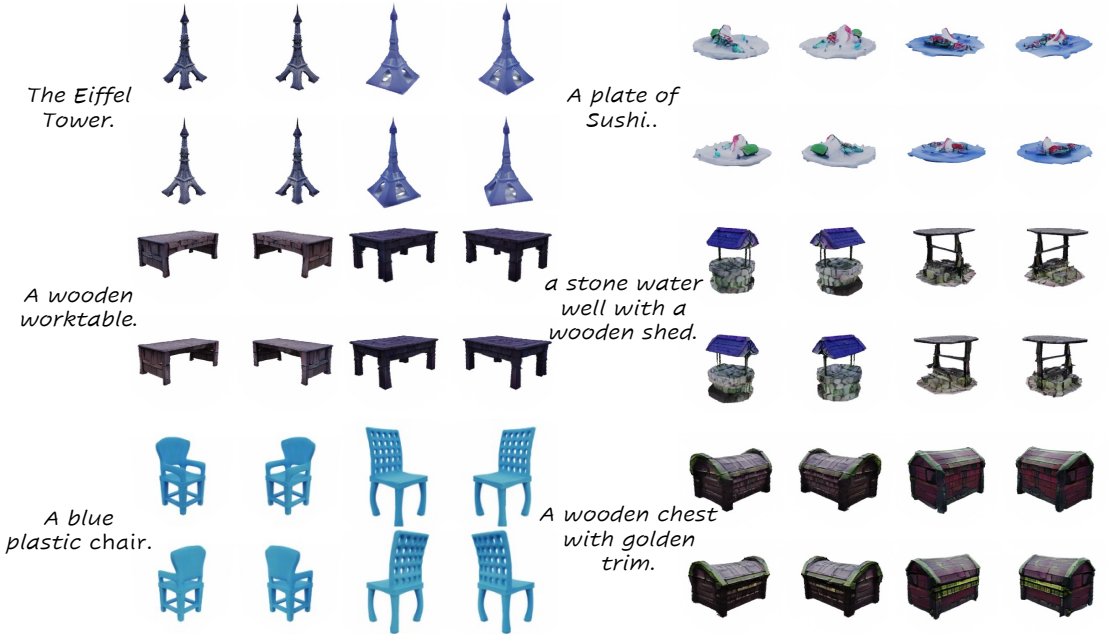


FIGURE 6.4: **Objaverse Conditional Generation Given Text Prompt.** We show two samples for each prompt. Zoom in for the best view.

the image-conditioned generation, our method yields semantically similar samples while maintaining diversity.

We also demonstrate the text-conditioned generation of Objaverse in Fig. 6.4 and Tab. 6.2. As shown, the diffusion model trained over LN3DIFF's latent space enables high-quality 3D generation over generic 3D datasets. This ability is unprecedented among existing 3D diffusion baselines and and marks a significant step toward highly controllable 3D generation. Qualitative comparisons against Shape-E and Point-E are provided in the supplementary materials.

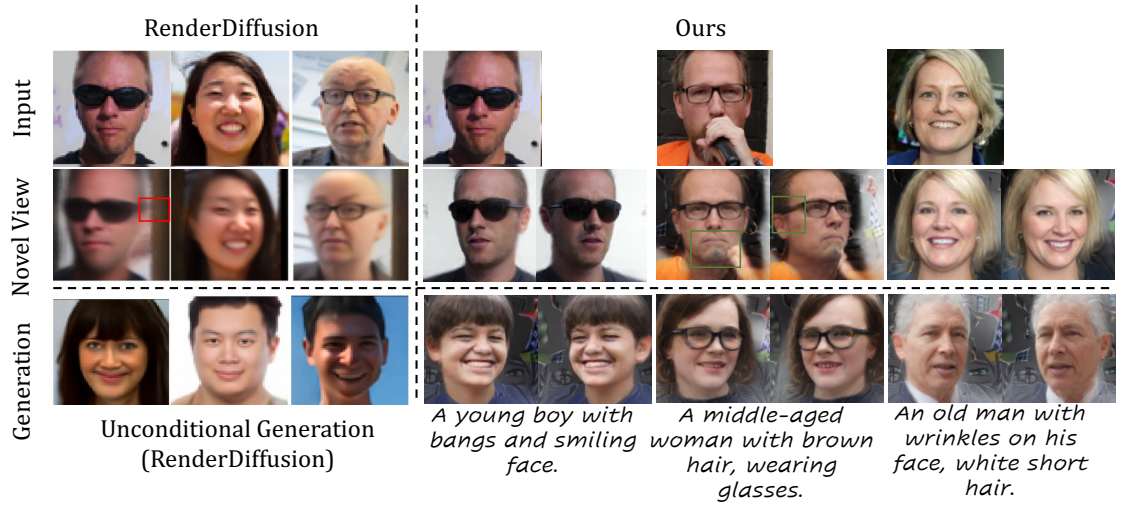


FIGURE 6.5: **FFHQ Monocular Reconstruction (upper half) and 3D Generation (lower half).** For monocular reconstruction, we test our method with hold-out test set and visualize the input-view and novel-view. Compared to baseline, our method shows consistently better performance on both reconstruction and generation.

We compare our method against RenderDiffusion, the only 3D diffusion method that supports 3D generation over FFHQ. As shown in the lower part in Fig. 6.5, beyond view-consistent 3D generation, our method further supports conditional 3D generation at 128×128 resolution, while RenderDiffusion is limited to 64×64 resolution due to the expensive volume rendering integrated into diffusion training. Quantitatively, our method achieves an FID score of 36.6, compared to 59.3 by RenderDiffusion.

Monocular 3D Reconstruction. We also include monocular reconstruction results over FFHQ datasets in the upper half of Fig. 6.5 and compare against RenderDiffusion. As can be observed, our method demonstrates high fidelity and preserves semantic details even in self-occluded images. The novel view generated by RenderDiffusion appears blurry and misses semantic components that are not visible in the input view, such as the leg of the eyeglass.

6.2.2 Ablation Study and Analysis

Reconstruction Arch Design. In Tab. 6.3, we benchmark each component of our auto-encoding architecture over a subset of Objaverse with $7K$ instances

TABLE 6.3: **Ablation of Reconstruction Arch Design.** We ablate the design of our auto-encoding architecture. Each component contributes to a consistent gain in the reconstruction performance, indicating an improvement in the modeling capacity.

Design	PSNR@100K
2D Conv Baseline	17.46
+ ViT Block	18.92
ViT Block → DiT Block	20.61
+ Plucker Embedding	21.29
+ Cross-Plane Attention	21.70
+ Self-Plane Attention	**21.95**

TABLE 6.4: **Diffusion Sampling Speed and Latent Size**. We provide the sampling time per instance evaluated on V100, along with the latent size. Our method achieves faster speed and better generation performance.

Method	Seconds	Latent Size
Get3D/EG3D	0.5	256
SSDNeRF	8.1	$128^2 \times 18$
RenderDiffusion	15.8	-
DiffRF	18.7	$32^3 \times 4$
LN3DIFF$_{\text{uncond}}$	**5.7**	$32^2 \times 12$
LN3DIFF$_{\text{cfg}}$	**7.5**	$32^2 \times 12$

and record the PSNR at $100K$ iterations. Each component introduces consistent improvements with negligible parameter increases.

Novel View Discriminator for Monocular Dataset. Novel view discriminator is crucial for monocular datasets like FFHQ. As shown in Fig. 6.6, without it, the VAE model fails to yield a plausible novel view.

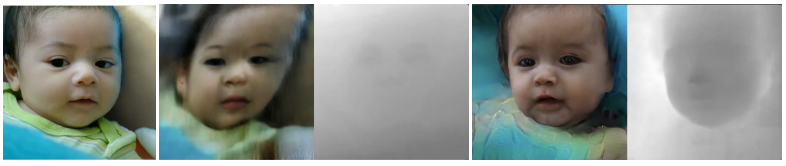


FIGURE 6.6: Ablation of novel view discriminator. Imposing novel view discriminator facilitates better 3D awareness.

Diffusion Sampling Speed and Latent Size. We report the sampling speed and latent space size comparison in Tab. 6.4. By performing on the compact latent space, our method achieves the fastest sampling while keeping the best generation performance. Though RenderDiffusion follows a latent-free design, its intermediate 3D neural field has a shape of $256^2 \times 96$ and hinders efficient diffusion training.

6.3 Conclusion

This chapter presents a new paradigm of 3D generative model by learning the diffusion model over a compact 3D-aware latent space. A dedicated variational

autoencoder encodes (multi-view) image(s) into a low-dim structured latent space, where conditional diffusion learning can be efficiently performed. We achieve state-of-the-art performance over ShapeNet and demonstrate our method over generic category-free Objaverse 3D datasets. The proposed work in this chapter potentially facilitates numerous downstream applications in 3D vision and graphics tasks. In the future, LN3Diff can be further extended to generate dynamic 3D assets by introducing a learnable temporal layer [244] over the pre-trained 3D diffusion model. Besides, it can be directly trained on the dynamic 3D datasets [18].

Chapter 7

Conclusion and Future Work

This thesis has investigated how to expand the generality of neural fields by leveraging the techniques of generative models. Specifically, the thesis primarily delved into two avenues to improve the neural fields' generality: *generalizable neural field algorithms from generative priors*, focusing on designing delicate algorithms to learn generalizable neural fields with the help of generative priors from pre-trained 3D generative models. *generalizable neural field representations for 3D generation*, instead, propose novel 3D representations that are designed for training 3D generative models.

The exploration of generalizable neural fields techniques has opened up numerous intriguing and meaningful avenues and a preliminary investigation has been made from only two perspectives. Though some progress has been made in this thesis, there is significant room for further unleashing the potential of neural fields by proposing more delicately designed algorithms and 3D representations for better 3D reconstruction, editing, and content generation. Additionally, beyond leveraging generative models, there remain other meaningful solutions that warrant further exploration. In this chapter, the findings are summarized within the thesis and discuss the potential future directions.

7.1 Dense 3D Correspondence from NeRF GANs

This thesis proposes to leverage a pre-trained NeRF-based GAN, π-GAN in our case, to build dense correspondence between NeRF representations of different objects within the same category. In Chapter 3, the key insight is that the pre-trained GAN possesses three important properties that can help alleviate the challenges of this task, namely 1) instance-specific latent codes that holistically capture the global structure of different NeRFs, 2) geometric-aware generator features that reflect local geometric details of different NeRFs, and 3) the manifold of NeRFs that serves as a source of infinite NeRF samples. Based on the three properties, a generalizable model is proposed, referred to as the Dual Deformation Field, a learning objective based on generator features that approximate geometric distances in feature space, and finally an effective curriculum training strategy that feeds samples with growing complexity. To the best of our knowledge, this is the first method that tries to establish dense correspondence across NeRF representations. Our experiments demonstrate that dense correspondences between NeRFs learned from our framework are accurate, smooth, and robust, making them applicable in various downstream applications. Moreover, since *DDF* directly adopts the latent code of GAN as the deformation condition, the proposed method could benefit from later progress in this field and produce more natural-looking results when applied to real images.

Despite the notable improvements achieved by the proposed method, there are three potential directions for further improvement for practical use:

1. *Using better 3D GAN model* - since our method relies on pre-trained 3D GANs, the performance of the learned correspondence may be limited by the performance of the GANs used. This could be addressed using a better 3D GAN design, e.g., EG3D [1], as shown in Tab. 3 and Fig. 12 and Fig. 16.

2. *Handling missing correspondence points* - our method may not find correct correspondence when the semantic part from source NeRF $\mathcal{N}_s$ does not exist in the target NeRF $\mathcal{N}_t$, e.g., transferring facial texture from a person without teeth to a smiling face with teeth. This could be addressed by filtering out searched correspondence points with high similarity loss.

3. *Real case inversion* - applying our method to real cases relies on 3D GAN inversion, which is an open problem to study. A potential solution is to leverage 3D GAN inversion [7] to alleviate this shortcoming.

7.2 Inversion and Editing on 3D GANs

This thesis proposes a novel 3D GAN inversion framework E3DGE for 3D GAN inversion and editing. As detailed in Chapter 4, the benefits of both self-supervised global prior and pixel-aligned local prior is jointly considered for high-quality shape and texture reconstruction. A hybrid alignment that bridges the best of 2D and 3D features is further proposed for view-consistent editing. Benefiting from the overall system design, the proposed method has advantages in terms of both high fidelity and editability. As a pioneer attempt in this direction, this work opens a new line of research direction and will inspire future works on 3D GAN inversion, few-shot 3D reconstruction, and 3D-aware learning from 2D images.

While our proposed method supports 3D reconstruction and editing given a monocular image, some improvements can potentially lead to better performance:

1. *Better global-local fusion design* - The fusion of global and local texture in EG3D sometimes lead to visual artifacts as shown in Fig 4.9. Though a solution is proposed to alleviate this issue with an extra pose estimator and fine-tuning, how to resolve this challenge in the single stage is worth future investigation.

2. *Data bias of synthetic data* - the proposed method may suffer from data bias introduced by the synthetic data. As the synthetic data lacks complex details and poses variations compared with real-world data, our method trained with it tends to generate a simple background and fail on extreme samples. This issue is particularly noticeable in StyleSDF-E3DGE as shown in 4.6, where all the backgrounds are more blurry compared to EG3D-E3DGE. A future direction is to leverage real data for semi-supervised training.

3. *Faster model speed* - having two models (global and local) to model the texture introduces extra computational cost. A potential solution is to leverage

the hyper network [245] for efficient local feature incorporation to alleviate the added computational cost of the 2D alignment module.

4. *Support of more 3D GANs* - it is worth exploring the potential of the proposed framework on other 3D GANs and shapes and other editing methods uniquely designed for 3D GANs.

7.3 Avatar 3D Head Representations

This thesis has introduced Loc3Diff, a new 3D generative framework, and demonstrated its promising results across various scenarios. As detailed in Chapter 5, a novel representation based on 3DMM anchored by 3D Gaussians-modulated local tri-planes is first introduced, which allows the decoupling of the underlying smooth geometry and deformation from the complex volumetric appearance. Importantly, the proposed representation can be stored in a UV space and is amenable to generative modeling. Moreover, a method is devised to simultaneously reconstruct and learn a latent space for our 3D representations via multi-view supervision, upon which a 2D diffusion model is trained to perform various editing tasks.

To further improve along this line of work, some directions are worth investigation:

1. *Faster rendering speed* - since volume rendering is still adopted as the rendering technique here, a potential solution is to directly leverage efficient 3D Gaussian rasterization technique [200] to accelerate the model training and inference.

2. *Single-stage training* - currently the proposed method requires two stages: stage 1 auto-decoder training to get the learned UV features, and stage 2 diffusion training for prior learning. A meaningful direction is to study whether the two stages can be merged into a single stage.

3. *Full body avatar modeling* - a natural follow-up will be extending our method to full body and introduce text/segmentation control [40] on local tri-planes.

7.4 3D Representation for General 3D Objects Generation

This thesis presents a new paradigm of the 3D generative model by learning the diffusion model over a compact 3D-aware latent space. As discussed in Chapter 6, a dedicated variational autoencoder encodes (multi-view) image(s) into a low-dim structured latent space, where conditional diffusion learning can be efficiently performed. The state-of-the-art performance is achieved over ShapeNet and demonstrates our method over generic category-free Objaverse 3D datasets. The proposed framework potentially facilitates numerous downstream applications in 3D vision and graphics tasks.

Though the proposed method is a step towards a general 3D diffusion model and can inspire future research in this direction, a few potential paths are outlined here for future exploration:

1. *Efficient VAE training* - the volume rendering is memory-consuming. Extending our decoder to more efficient 3D representations such as 3DGS [200] shall alleviate this issue.

2. *Support of latent-based 3D editing* - currently the 3D latent space is still triplane, which cannot be directly used for intuitive editing. A more versatile 3D latent space, e.g., point cloud-structured latent space, is indispensable for more diffusion-based 3D editing.

3. *Data bias of synthetic data* - adding more real-world data such as MVImageNet [223] and more control conditions [40] is also worth exploring.

List of Author's Awards, Patents, and Publications

Conference Proceedings

- **Yushi Lan**, Feitong Tan, Qiangeng Xu, Di Qiu, Kyle Genova, Zeng Huang, Sean Fanello, Rohit Pandey, Thomas Funkhouser, Chen Change Loy and Yinda Zhang. Loc3Diff: Local Diffusion for 3D Human Head Synthesis and Editing. In *ECCV*, 2024.

- **Yushi Lan**, Fangzhou Hong, Shuai Yang, Shangchen Zhou, Xuyi Meng, Bo Dai, Xingang Pan and Chen Change Loy. LN3Diff: Scalable Latent Neural Fields Diffusion for Speedy 3D Generation. In *ECCV*, 2024.

- Junzhe Zhang*, **Yushi Lan***, Shuai Yang, Fangzhou Hong, Quan Wang, Chai Kiat Yeo, Ziwei Liu, Chen Change Loy. DeformToon3D: Deformable 3D Toonification from Neural Radiance Fields. In *ICCV*, 2023.

- **Yushi Lan**, Xuyi Meng, Shuai Yang, Chen Change Loy and Dai, Bo. E3DGE: Self-Supervised Geometry-Aware Encoder for Style-based 3D GAN Inversion. In *CVPR*, 2023.

- Fangzhou Hong, Zhaoxi Chen, **Yushi Lan**, Liang Pan, Ziwei Liu. EVA3D: Compositional 3D Human Generation from 2D Image Collections. In *ICLR*, 2023.

- Songlin Yang, Wei Wang, **Yushi Lan**, Xiangyu Fan, Bo Peng, Lei Yang, Jing Dong. Learning Dense Correspondence for NeRF-Based Face Reenactment. In *AAAI*, 2024.

- **Yushi Lan***, Yuan Liu*, Xinchi Zhou, Maoqing Tian, Xuesen Zhang, Shuai Yi, Hongsheng Li. Magnifier: Towards Semantic Adversary and Fusion for Person Re-identification. In *BMVC*, 2020.

Journals

- **Yushi Lan**, Chen Change Loy, Bo Dai. DDF: Correspondence Distillation from NeRF-Based GAN. In *IJCV*, 2022.

Preprints

- **Yushi Lan**, Shangchen Zhou, Zhaoyang Lyu, Fangzhou Hong, Shuai Yang, Bo Dai, Xingang Pan, Chen Change Loy. GaussianAnything: Interactive Point Cloud Latent Diffusion for 3D Generation. In *arXiv*, 2024.

- Yongwei Chen, **Yushi Lan**, Shangchen Zhou, Tengfei Wang, Xingang Pan. SAR3D: Autoregressive 3D Object Generation and Understanding via Multi-scale 3D VQVAE. In *arXiv*, 2024.

- Zhouxia Wang, **Yushi Lan**, Shangchen Zhou, Chen Change Loy. ObjCtrl-2.5D: Training-free Object Control with Camera Poses. In *arXiv*, 2024.

- Honghua Chen, **Yushi Lan**, Yongwei Chen, Yifan Zhou, Xingang Pan. MVDrag3D: Drag-based Creative 3D Editing via Multi-view Generation-Reconstruction Priors. In *arXiv*, 2024.

- Zhaoxi Chen, Jiaxiang Tang, Yuhao Dong, Ziang Cao, Fangzhou Hong, **Yushi Lan**, Tengfei Wang, Haozhe Xie, Tong Wu, Shunsuke Saito, Liang Pan, Dahua Lin, Ziwei Liu. 3DTopia-XL: Scaling High-quality 3D Asset Generation via Primitive Diffusion. In *arXiv*, 2024.

Bibliography

[1] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. xix, xxii, 2, 4, 9, 14, 17, 18, 35, 55, 57, 58, 66, 70, 77, 81, 91, 99, 103, 105, 108, 110, 116

[2] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. xx, 1, 3, 14, 17, 23, 27, 29, 33, 35, 57, 62, 66, 81, 85, 99, 105

[3] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. PixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. xx, 2, 19, 29, 62, 69

[4] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *NeurIPS*, 2020. xx, 29

[5] Yuxuan Zhang, Huan Ling, Jun Gao, K. Yin, Jean-Francois Lafleche, Adela Barriuso, A. Torralba, and S. Fidler. Datasetgan: Efficient labeled data factory with minimal human effort. In *CVPR*, 2021. xx, xxvii, 40, 41, 42

[6] Xinyao Wang, Liefeng Bo, and Li Fuxin. Adaptive wing loss for robust face alignment via heatmap regression. In *ICCV*, October 2019. xxi, 44

[7] Yushi Lan, Xuyi Meng, Shuai Yang, Chen Change Loy, and Bo Dai. E3dge: Self-supervised geometry-aware encoder for style-based 3D gan inversion. In *CVPR*, 2023. xxii, 1, 2, 17, 18, 47, 53, 66, 87, 100, 103, 117

[8] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. StyleSDF: High-resolution 3D-consistent image and geometry generation. In *CVPR*, 2021. xxii, 2, 17, 18, 57, 58, 65, 66, 70, 77

[9] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41, 2021. 1, 100

[10] Julien N. P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: Adaptive coordinate networks for neural scene representation. *TOG*, 40(4), 2021. 1

[11] Weihao Xia and Jing-Hao Xue. A survey on deep generative 3d-aware image synthesis. In *ACM Computing Surveys (CSUR)*, 2023. 1

[12] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B Goldman, and M. Zollhöfer. State of the Art on Neural Rendering. *Computer Graphics Forum*, 2020. 1, 15

[13] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy Networks: Learning 3D reconstruction in function space. In *CVPR*, 2019. 1, 14, 17

[14] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, pages 165–174, 2019. 17, 39, 66, 82

[15] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, pages 5932–5941, 2019. 1, 14

[16] Yushi Lan, Chen Change Loy, and Bo Dai. DDF: Correspondence distillation from nerf-based gan. *IJCV*, 2022. 1, 2, 17, 18, 23, 87, 89, 99

[17] Yushi Lan, Feitong Tan, Di Qiu, Qiangeng Xu, Kyle Genova, Zeng Huang, Sean Fanello, Rohit Pandey, Thomas Funkhouser, Chen Change Loy, and Yinda Zhang. Gaussian3Diff: 3D gaussian diffusion for 3D full head synthesis and editing. *ECCV*, 2024. 1, 2, 5, 21, 81, 99

[18] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663*, 2023. 2, 102, 114

[19] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P. Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas A. Funkhouser. IBRNet: Learning Multi-View Image-Based Rendering. In *CVPR*, 2021. 2, 19, 62

[20] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *ICCV*, pages 5885–5894, October 2021. 2

[21] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and G. Wetzstein. Pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. In *CVPR*, 2021. 2, 4, 7, 17, 18, 24, 25, 26, 57, 66, 81, 104

[22] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. StyleNeRF: A style-based 3D-aware generator for high-resolution image synthesis. In *ICLR*, 2021. 2, 4, 17, 18, 57, 58, 66, 81

[23] Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, et al. RODIN: A generative model for sculpting 3D digital avatars using diffusion. In *CVPR*, 2023. 2, 5, 21, 81, 82, 84, 87, 92, 99, 103, 104

[24] Yushi Lan, Fangzhou Hong, Shuai Yang, Shangchen Zhou, Xuyi Meng, Bo Dai, Xingang Pan, and Chen Change Loy. Ln3diff: Scalable latent neural fields diffusion for speedy 3d generation. In *ECCV*, 2024. 2, 99

[25] Guandao Yang, Serge Belongie, Bharath Hariharan, and Vladlen Koltun. Geometry processing with neural fields. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 4, 23

[26] O. Litany, Tal Remez, E. Rodolà, A. Bronstein, and M. Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *ICCV*, pages 5660–5668, 2017. 4, 16, 24

[27] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN Inversion: A Survey. *PAMI*, 2022. 4, 53

[28] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Trans. Graph.*, 2021. 4, 17, 53, 70, 72

[29] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. High-Fidelity GAN inversion for image attribute editing. In *CVPR*, 2022. 4, 17, 53, 54, 62, 63

[30] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3D-aware image synthesis. In *NeurIPS*, 2020. 4, 7, 17, 24, 34, 37, 81

[31] Sizhe An, Hongyi Xu, Yichun Shi, Guoxian Song, Umit Y. Ogras, and Linjie Luo. Panohead: Geometry-aware 3D full-head synthesis in 360deg. In *CVPR*, pages 20950–20959, June 2023. 4, 18, 81, 87, 89, 90, 92

[32] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. FENeRF: Face editing in neural radiance fields. In *arXiv*, 2021. 4, 17, 18, 81

[33] Jingxiang Sun, Xuan Wang, Yichun Shi, Lizhen Wang, Jue Wang, and Yebin Liu. Ide-3d: Interactive disentangled editing for high-resolution 3D-aware portrait synthesis. *ACM Transactions on Graphics (TOG)*, 41(6):1–10, 2022. 4, 17, 18, 81, 84, 89, 90, 92

[34] Jingxiang Sun, Xuan Wang, Lizhen Wang, Xiaoyu Li, Yong Zhang, Hongwen Zhang, and Yebin Liu. Next3d: Generative neural texture rasterization for 3d-aware head avatars. In *CVPR*, 2023. 4, 81

[35] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021. 5, 19, 81, 88, 89, 99, 100, 105

[36] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 5, 19, 20, 81, 99, 105

[37] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021. 5, 17, 19, 81, 82, 99, 100

[38] Jessica Shue, Eric Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *CVPR*, 2022. 5, 21, 81, 82, 100, 102, 108

[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 5, 19, 20, 85, 88, 99, 100, 102, 103, 104, 105, 106

[40] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023. 5, 99, 100, 118, 119

[41] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. In *arXiv*, 2022. 5, 99, 106

[42] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. *ICLR*, 2022. 5, 20, 92, 99

[43] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object, 2023. 5, 21, 99

[44] Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulo, Peter Kontschieder, and Matthias Nießner. DiffRF: Rendering-guided 3D radiance field diffusion. In *CVPR*, 2023. 5, 21, 99, 100, 102, 105, 108, 110

[45] Heewoo Jun and Alex Nichol. Shap-E: Generating conditional 3D implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 21, 100, 102, 110

[46] Hansheng Chen, Jiatao Gu, Anpei Chen, Wei Tian, Zhuowen Tu, Lingjie Liu, and Hao Su. Single-stage diffusion nerf: A unified approach to 3D generation and reconstruction. In *ICCV*, 2023. 5, 21, 100, 105, 108, 110

[47] Biao Zhang, Jiapeng Tang, Matthias Nießner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Trans. Graph.*, 42(4), jul 2023. ISSN 0730-0301. doi: 10.1145/3592442. URL `https://doi.org/10.1145/3592442`. 5, 99

[48] Charles Qi, Hao Su, Kaichun Mo, and Leonidas Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. *arXiv*, 2016. 5, 100

[49] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. KPConv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019.

[50] SpConv Contributors. SpConv: Spatially sparse convolution library. `https://github.com/traveller59/spconv`, 2022. 5, 100

[51] Linqi Zhou, Yilun Du, and Jiajun Wu. 3D shape generation and completion through point-voxel diffusion. In *ICCV*, 2021. 5, 100, 102

[52] Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 7, 17, 18, 24, 57

[53] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L. Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, Seyedeh Sara Mahdavi, Raphael Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022. 9, 19, 82, 90

[54] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *arXiv*, 2013. 9, 19, 100, 103

[55] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41, 2022. 13

[56] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *ICLR*, 2019. 14

[57] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. DeepVoxels: Learning persistent 3D feature embeddings. In *CVPR*, 2019. 14

[58] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 14, 15

[59] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *CVPR*, 2020. 14, 29

[60] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, 2021. 14

[61] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. I M Avatar: Implicit morphable head avatars from videos. In *CVPR*, 2022. 15, 84

[62] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *CVPR*, pages 8649–8658, June 2021. 15, 84

[63] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *CVPR*, 2022. 15

[64] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Neural articulated radiance field. In *ICCV*, 2021. 15

[65] Zerong Zheng, Tao Yu, Qionghai Dai, and Yebin Liu. Deep implicit templates for 3D shape representation. In *CVPR*, pages 1429–1439, 2021. 15, 16, 28, 34, 39, 44

[66] Yu Deng, Jiaolong Yang, and Xin Tong. Deformed implicit field: Modeling 3D shapes with learned dense correspondence. In *CVPR*, pages 10286–10296, 2021. 15, 28, 34, 39

[67] Feng Liu and Xiaoming Liu. Learning implicit functions for topology-varying dense 3d shape correspondence. In *NIPS*, Virtual, December 2020. 15, 16

[68] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*, 2015. 15, 100, 101, 105, 107

[69] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*, volume 2, pages 1150–1157. IEEE, 1999. 15

[70] Prune Truong, Martin Danelljan, and Radu Timofte. Glu-net: Global-local universal network for dense flow and correspondences. In *CVPR*, pages 6257–6267, 2020. 15, 37, 43, 44

[71] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. GOCor: Bringing globally optimized correspondence volumes into your neural network. In *NeurIPS*, 2020. 43

[72] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning accurate dense correspondences and when to trust them. In *CVPR*, pages 5710–5720, 2021. 37, 44

[73] Sunghwan Hong, Seokju Cho, Jisu Nam, Stephen Lin, and Seungryong Kim. Cost aggregation with 4d convolutional swin transformer for few-shot segmentation. In *ECCV*, pages 108–126. Springer, 2022. 15, 16, 43, 45

[74] Jiteng Mu, Shalini De Mello, Zhiding Yu, Nuno Vasconcelos, Xiaolong Wang, Jan Kautz, and Sifei Liu. Coordgan: Self-supervised dense correspondences emerge from gans. In *CVPR*, 2022. 15, 40, 42, 43

[75] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *TOG*, 34(6): 1–16, 2015. 16, 84

[76] B. Egger, W. Smith, Ayush Tewari, Stefanie Wuhrer, M. Zollhöfer, T. Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, S. Romdhani, C. Theobalt, V. Blanz, and T. Vetter. 3d morphable face models—past, present, and future. *TOG*, 39:1 – 38, 2020. 16

[77] O. V. Kaick, Hao Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30, 2011. 16

[78] Y. Sahillioglu. Recent advances in shape correspondence. *The Visual Computer*, 36:1705 – 1721, 2019. 16

[79] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, M. Bronstein, and J. Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 38:1 – 12, 2019. 16

[80] C. Qi, Hao Su, Kaichun Mo, and L. Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, pages 77–85, 2017. 16

[81] Zhenfeng Fan, Xiyuan Hu, C. Chen, and S. Peng. Boosting local shape matching for dense 3D face correspondence. In *CVPR*, pages 10936–10946, 2019. 16

[82] Tinghui Zhou, Philipp Krähenbühl, Mathieu Aubry, Qixing Huang, and Alexei A. Efros. Learning dense correspondence via 3D-guided cycle consistency. In *CVPR*, pages 117–126, 2016. 16

[83] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *CVPR*, pages 4370–4379, 2019. 16

[84] M. Eisenberger, David Novotný, Gael Kerchenbaum, Patrick Labatut, N. Neverova, D. Cremers, and A. Vedaldi. Neuromorph: Unsupervised shape interpolation and correspondence in one go. In *CVPR*, 2021. 16

[85] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 16, 19, 99, 104

[86] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 16, 24, 33, 57, 66, 68, 70, 90

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

[88] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020. 16

[89] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yongliang Yang. HoloGAN: Unsupervised Learning of 3D Representations From Natural Images. In *ICCV*, 2019. 17

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

[91] Xingang Pan, Bo Dai, Ziwei Liu, Chen Change Loy, and Ping Luo. Do 2D GANs know 3D shape? Unsupervised 3D Shape Reconstruction from 2D Image GANs. In *ICLR*, 2021. 17

[92] Fangzhou Hong, Zhaoxi Chen, Yushi Lan, Liang Pan, and Ziwei Liu. EVA3D: Compositional 3D human generation from 2d image collections. In *ICLR*, 2022. 17

[93] Junzhe Zhang, Yushi Lan, Shuai Yang, Fangzhou Hong, Quan Wang, Chai Kiat Yeo, Ziwei Liu, and Chen Change Loy. Deformtoon3d: Deformable 3D toonification from neural radiance fields. In *ICCV*, 2023. 17

[94] Hoang Thanh-Tung and T. Tran. Catastrophic forgetting and mode collapse in gans. *IJCNN*, pages 1–10, 2020. 17

[95] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2StyleGAN: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 17, 53

[96] Jianglin Fu, Shikai Li, Yuming Jiang, Kwan-Yee Lin, Chen Qian, Chen Change Loy, Wayne Wu, and Ziwei Liu. Stylegan-human: A data-centric odyssey of human generation. In *ECCV*, 2022. 17

[97] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a StyleGAN encoder for image-to-image translation. In *CVPR*, 2021. 17, 43, 53, 59, 60, 66, 69, 70

[98] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for StyleGAN image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 17, 49, 53, 70

[99] Kelvin C.K. Chan, Xiangyu Xu, Xintao Wang, Jinwei Gu, and Chen Change Loy. GLEAN: Generative latent bank for large-factor image super-resolution and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 62

[100] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain GAN Inversion for Real Image Editing. In *ECCV*, 2020. 17

[101] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2StyleGAN++: How to edit the embedded images? In *CVPR*, 2020. 17, 53

[102] Connor Z. Lin, David B. Lindell, Eric Chan, and Gordon Wetzstein. 3D GAN Inversion for Controllable Portrait Image Animation. *arXiv*, abs/2203.13441, 2022. 17

[103] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. In *SIGGRAPH*, volume 40, 2021. 17, 18

[104] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *TOG*, 36(6), 2017. URL `https://doi.org/10.1145/3130800.3130813`. 17, 84, 85, 88

[105] Shengqu Cai, Anton Obukhov, Dengxin Dai, and Luc Van Gool. Pix2NeRF: Unsupervised Conditional p-GAN for Single Image to Neural Radiance Fields Translation. In *CVPR*, 2022. 18, 103

[106] Dario Pavllo, David Joseph Tan, Marie-Julie Rakotosaona, and Federico Tombari. Shape, pose, and appearance from a single image via bootstrapped radiance field inversion. In *CVPR*, 2023. 18

[107] Daniel Rebain, Mark Matthews, Kwang Moo Yi, Dmitry Lagun, and Andrea Tagliasacchi. LOLNeRF: Learn from one look. In *CVPR*, 2022. 18, 19, 87

[108] Ziyang Yuan, Yiming Zhu, Yu Li, Hongyu Liu, and Chun Yuan. Make encoder great again in 3d gan inversion through geometry and occlusion-aware encoding. *ICCV*, 2023. 18

[109] Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick P'erez. This Dataset Does Not Exist: Training Models from Generated Images. *ICASSP*, 2020. 18

[110] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting Deep Generative Prior for Versatile Image Restoration and Manipulation. *PAMI*, 44:7474–7489, 2022. 18

[111] Ali Jahanian, Lucy Chai, and Phillip Isola. On the" steerability" of generative adversarial networks. *ICLR*, 2020.

[112] Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning. *ICLR*, 2022. 18

[113] Shuai Yang, Liming Jiang, Ziwei Liu, , and Chen Change Loy. VToonify: Controllable high-resolution portrait video style transfer. *ACM Transactions on Graphics (TOG)*, 41(6):1–15, 2022. doi: 10.1145/3550454.3555437.

[114] Yuxuan Zhang, Huan Ling, Jun Gao, Kangxue Yin, Jean-Francois Lafleche, Adela Barriuso, Antonio Torralba, and Sanja Fidler. DatasetGAN: Efficient labeled data factory with minimal human effort. In *CVPR*, 2021. 18, 87

[115] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. In *NeurIPS*, 2021. 18

[116] Feitong Tan, Sean Fanello, Abhimitra Meka, Sergio Orts-Escolano, Danhang Tang, Rohit Pandey, Jonathan Taylor, Ping Tan, and Yinda Zhang. Voluxgan: A generative model for 3D face synthesis with hdri relighting. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022. 18

[117] Jianfeng Xiang, Jiaolong Yang, Yu Deng, and Xin Tong. Gram-hd: 3d-consistent image generation at high resolution with generative radiance manifolds. In *ICCV*, pages 2195–2205, October 2023.

[118] Xingyu Chen, Yu Deng, and Baoyuan Wang. Mimic3d: Thriving 3d-aware gans via 3d-to-2d imitation. In *ICCV*, 2023. 18

[119] Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas Funkhouser, and Andrea Tagliasacchi. Scene Representation Transformer: Geometry-free novel view synthesis through set-latent scene representations. *CVPR*, 2022. 19

[120] Kyle Sargent, Jing Yu Koh, Han Zhang, Huiwen Chang, Charles Herrmann, Pratul P. Srinivasan, Jiajun Wu, and Deqing Sun. VQ3D: Learning a 3D-aware generative model on imagenet. *ICCV*, 2023. 19, 103, 104

[121] Adam R. Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Sovna Mokr'a, and Danilo Jimenez Rezende. NeRF-VAE: A geometry aware 3D scene generative model. *ICML*, 2021. 19

[122] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. In *ICLR*, 2024. 19, 21

[123] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 19, 20

[124] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015. 19

[125] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019. 20

[126] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. In *NeurIPS*, 2020. 20

[127] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2020. 20

[128] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In *NeurIPS*, 2021. 20

[129] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 2023. 20

[130] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *CVPR*, 2022. 20

[131] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3D generation with variational score distillation. In *NeurIPS*, 2023. 20, 99

[132] Eric R. Chan, Koki Nagano, Matthew A. Chan, Alexander W. Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In *arXiv*, 2023. 21

[133] Ayush Tewari, Tianwei Yin, George Cazenavette, Semon Rezchikov, Joshua B. Tenenbaum, Frédo Durand, William T. Freeman, and Vincent Sitzmann. Diffusion with forward models: Solving stochastic inverse problems without direct supervision. In *NeurIPS*, 2023. 21

[134] Emilien Dupont, Hyunjik Kim, S. M. Ali Eslami, Danilo Jimenez Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. In *ICML*, 2022. 21

[135] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *ICML*, 2023. 21, 100

[136] Titas Anciukevičius, Zexiang Xu, Matthew Fisher, Paul Henderson, Hakan Bilen, Niloy J Mitra, and Paul Guerrero. RenderDiffusion: Image diffusion for 3D reconstruction, inpainting and generation. In *CVPR*, 2023. 21, 101, 102, 106, 108, 110

[137] Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy Mitra. Holodiffusion: Training a 3D diffusion model using 2D images. In *CVPR*, 2023. 21

[138] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, and Kai Zhang. DMV3D: Denoising multi-view diffusion using 3D large reconstruction model. In *ICLR*, 2024. 21

[139] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Matthew Johnson, Virginia Estellers, Thomas J. Cashman, and Jamie Shotton. Fake it till you make it: Face analysis in the wild using synthetic data alone. In *ICCV*, 2021. 24

[140] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019. 24

[141] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, volume 32, 2018. 27

[142] Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. Feature-wise transformations. *Distill*, 2018. doi: 10.23915/distill.00011. https://distill.pub/2018/feature-wise-transformations.

[143] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. 27, 49

[144] Lars M. Mescheder, Andreas Geiger, and S. Nowozin. Which training methods for GANs do actually converge? In *ICML*, 2018. 27, 67

[145] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv*, 2020. 28

[146] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, 2021. 32, 84, 89, 97

[147] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, December 2015. 34

[148] Weiwei Zhang, Jian Sun, and Xiaoou Tang. Cat head detection - how to effectively exploit shape and texture features. In *ECCV*, 2008. 34, 37

[149] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, volume abs/1412.6980, 2015. 34

[150] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. MaskGAN: Towards diverse and interactive facial image manipulation. In *CVPR*, 2020. 34, 68, 70

[151] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017. 34

[152] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH*, 1999. 37, 44

[153] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Proc. CoRL*, 2017. 37

[154] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv*, abs/1706.05587, 2017. 40

[155] William Peebles, Jun-Yan Zhu, Richard Zhang, Antonio Torralba, Alexei Efros, and Eli Shechtman. Gan-supervised dense visual alignment. In *CVPR*, 2022. 40, 43, 46

[156] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. 43

[157] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A. Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In *NeurIPS*, 2020. 43

[158] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv*, 2019. 43

[159] Jiarui Xu and Xiaolong Wang. Rethinking self-supervised correspondence learning: A video frame-level similarity perspective. *arXiv*, 2021. 43

[160] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 43, 46

[161] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *PAMI*, 33:978–994, 2011. 43, 44

[162] Seokju Cho, Sunghwan Hong, Sangryul Jeon, Yunsung Lee, Kwanghoon Sohn, and Seungryong Kim. Cats: Cost aggregation transformers for visual correspondence. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 43, 45

[163] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23:1499–1503, 2016. 44

[164] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *ICCV*, pages 2041–2050, 2018. 45

[165] Juhong Min, Jongmin Lee, Jean Ponce, and Minsu Cho. Spair-71k: A large-scale benchmark for semantic correspondence. *ArXiv*, abs/1908.10543, 2019. 45

[166] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *CoRR*, volume abs/1409.1556, 2015. 46

[167] Shuai Yang, Liming Jiang, Ziwei Liu, , and Chen Change Loy. Unsupervised image-to-image translation with generative prior. In *CVPR*, 2022. 46

[168] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *ICLR*, 2021. 46

[169] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 48

[170] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. In *CVPR*, 2021. 49

[171] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 50, 67, 87, 105

[172] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020. 57

[173] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 57, 103, 105

[174] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, June 2022. 60

[175] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, October 2019. 62, 69

[176] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. PIFuHD: Multi-level pixel-aligned implicit function for high-resolution 3D human digitization. In *CVPR*, June 2020.

[177] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J. Black. ICON: Implicit Clothed humans Obtained from Normals. In *CVPR*, June 2022.

[178] Thiemo Alldieck, Mihai Zanfir, and Cristian Sminchisescu. Photorealistic monocular 3D reconstruction of humans wearing clothing. *CVPR*, 2022. 66

[179] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo Radiance Fields (SRF): Learning View Synthesis for Sparse Views of Novel Scenes. *CVPR*, 2021. 62

[180] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016. 62

[181] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 62

[182] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 64

[183] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 64

[184] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3D face reconstruction with weakly-supervised learning: From single image to image set. In *CVPR*, 2019. 64

[185] Xingang Pan, Xudong Xu, Chen Change Loy, Christian Theobalt, and Bo Dai. A Shading-Guided Generative Implicit Model for Shape-Accurate 3D-Aware Image Synthesis. In *NeurIPS*, 2021. 66

[186] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *CVPR*, 2019. 67, 90

[187] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 68, 100

[188] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. InterFaceGAN: Interpreting the disentangled face representation learned by GANs. *PAMI*, PP, 2020. 68

[189] Yuming Jiang, Ziqi Huang, Xingang Pan, Chen Change Loy, and Ziwei Liu. Talk-to-Edit: Fine-grained facial editing via dialog. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 68

[190] Soubhik Sanyal, Timo Bolkart, Haiwen Feng, and Michael Black. Learning to regress 3D face shape and expression from an image without 3D supervision. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 68

[191] Zhimeng Zhang, Lincheng Li, Yu Ding, and Changjie Fan. Flow-guided one-shot talking face generation with a high-resolution audio-visual dataset. In *CVPR*, pages 3661–3670, 2021. 72

[192] Justin NM Pinkney and Doron Adler. Resolution dependent gan interpolation for controllable image synthesis between domains. *arXiv preprint arXiv:2010.05334*, 2020. 76

[193] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH*, 1999. 82

[194] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. In *CVPR*, pages 18653–18664, 2022. 84

[195] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *CVPR*, pages 10975–10985, 2019. 84

[196] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4), jul 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459863. URL `https://doi.org/10.1145/3450626.3459863`. 84, 85

[197] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In *CVPR*, pages 5449–5458, 2022.

[198] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local SDF priors for detailed 3D reconstruction. In *ECCV*, 2020. 84

[199] Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova. Nerflets: Local radiance fields for efficient structure-aware 3D scene representation from 2d supervision. In *CVPR*, 2023. 84

[200] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023. 84, 85, 105, 118, 119

[201] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (TOG)*, 40(4):1–13, 2021. 85

[202] Leonid Keselman and Martial Hebert. Approximate differentiable rendering with algebraic surfaces. In *European Conference on Computer Vision*, pages 596–614. Springer, 2022. 85

[203] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *ICLR*, 2018. 87

[204] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, June 2019. 87

[205] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 89

[206] Jiatao Gu, Qingzhe Gao, Shuangfei Zhai, Baoquan Chen, Lingjie Liu, and Josh Susskind. Learning controllable 3d diffusion models from single-view images. *arXiv preprint arXiv:2304.06700*, 2023. 89

[207] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *TOG*, 40(6), dec 2021.

[208] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *CVPR*, 2021. 89

[209] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 90, 106

[210] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCVW*, September 2018. 90

[211] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 35:36479–36494, 2022. 90

[212] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *PAMI*, 35:2878–2890, 2013. 91

[213] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *CVPR*, volume abs/2201.08845, 2022. 95

[214] Anju Tewari, Otto Fried, Justus Thies, Vincent Sitzmann, S. Lombardi, Z Xu, Tanaba Simon, Matthias Nießner, Edgar Tretschk, L. Liu, Ben Mildenhall, Pranatharthi Srinivasan, R. Pandey, Sergio Orts-Escolano, S. Fanello, M. Guang Guo, Gordon Wetzstein, J y Zhu, Christian Theobalt, Manju Agrawala, Donald B. Goldman, and Michael Zollhöfer. Advances in neural rendering. *Computer Graphics Forum*, 41, 2021. 99, 100, 105

[215] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. In *arXiv*, 2023. 99

[216] Jiatao Gu, Qingzhe Gao, Shuangfei Zhai, Baoquan Chen, Lingjie Liu, and Josh Susskind. Learning controllable 3D diffusion models from single-view images. *arXiv preprint arXiv:2304.06700*, 2023. 100, 103

[217] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3D objects. *arXiv preprint arXiv:2212.08051*, 2022. 100, 103, 105, 107

[218] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 101, 103, 104

[219] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022. 101, 102, 103, 104, 106

[220] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 101, 103, 104

[221] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene Representation Networks: Continuous 3D-structure-aware neural scene representations. In *NeurIPS*, 2019. 101, 107, 108

[222] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *NeurIPS*, 2021. 102, 103, 105, 106

[223] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Tianyou Liang, Guanying Chen, Shuguang Cui, and Xiaoguang Han. MVImgNet: A large-scale dataset of multi-view images. In *CVPR*, 2023. 102, 119

[224] Lu Mi, Abhijit Kundu, David Ross, Frank Dellaert, Noah Snavely, and Alireza Fathi. im2nerf: Image to neural radiance field in the wild. In *arXiv*, 2022. 103

[225] Vincent Sitzmann, Semon Rezchikov, William T. Freeman, Joshua B. Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In *NeurIPS*, 2021. 103

[226] Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3D reconstruction. *arXiv preprint arXiv:2301.08247*, 2023. 103

[227] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision, 2023. 104, 108

[228] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 104

[229] Hiroharu Kato and Tatsuya Harada. Learning view priors for single-view 3D reconstruction. In *CVPR*, 2019. 104, 105

[230] James Lucas, G. Tucker, Roger Baker Grosse, and Mohammad Norouzi. Understanding posterior collapse in generative latent variable models. In *ICLR*, 2019. 104

[231] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3D reconstruction. In *arXiv*, 2023. 105

[232] Chung-Yi Weng, Pratul P. Srinivasan, Brian Curless, and Ira Kemelmacher-Shlizerman. PersonNeRF: Personalized reconstruction from photo collections. In *CVPR*, pages 524–533, June 2023. 105

[233] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *arXiv*, 2023. 106

[234] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 106

[235] Chenlin Meng, Ruiqi Gao, Diederik P. Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *CVPR*, pages 14297–14306, 2022. 106

[236] Jonathan Ho. Classifier-free diffusion guidance. In *NeurIPS*, 2021. 107

[237] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d. *arXiv preprint arXiv:2311.16918*, 2023. 107

[238] Nupur Kumari, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Ensembling off-the-shelf models for gan training. In *CVPR*, 2022. 108

[239] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein GANs. In *NeurIPS*, 2017. 108

[240] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3D: A generative model of high quality 3D textured shapes learned from images. In *NeurIPS*, 2022. 108, 110

[241] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017. 108

[242] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *ICLR*, 2018. 108

[243] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts, 2022. 110

[244] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *ICLR*, 2024. 114

[245] Tan M. Dinh, Anh Tuan Tran, Rang Nguyen, and Binh-Son Hua. HyperInverter: Improving stylegan inversion via hypernetwork. In *CVPR*, 2022. 118