INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Learning-Based 3D Reconstruction and Synthesis Problems

Doctoral Thesis Submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

Aradhya Neeraj Mathur

aradhyam@iiitd.ac.in

Indraprastha Institute of Information Technology

Supervisor:

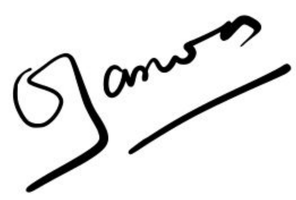
Prof. Ojaswa Sharma

September 20, 2025

Certificate

This is to certify that the thesis titled **Learning-Based 3D Reconstruction and Synthesis Problems** being submitted by **Aradhya Neeraj Mathur** (PHD17008) to the Indraprastha Institute of Information Technology-Delhi, for the award of the degree of **Doctor of Philosophy**, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards, fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or in full to any other university or institute for the award of any degree/diploma.



Prof. Ojaswa Sharma

Thesis Supervisor

Associate Professor (CSE)

Department of Computer Science,

Indraprastha Institute of Information Technology Delhi,

Okhla Industrial Estate, Phase III, New Delhi, Delhi 110020.

Acknowledgement

The journey of a PhD is not as solo as most think it is. It is a collaborative effort and a partnership between an advisor and a student to culminate the essential skillset and imbibe the necessary mindset needed to become an independent researcher. My sincerest gratitude is to my advisor, Dr. Ojaswa Sharma, who not only guided me with the important aspects of research but also helped me develop a mindset needed to survive the long, tedious journey in a much more resilient and positive manner. Who not only taught me to humbly accept the wins but also to get back up from the defeats. His thorough support and freedom helped me fulfill my curiosity and build trust to pursue my offbeat ideas. I would also like to thank my committee members, Dr. Saket Anand, Dr. A V Subramanyam, and Dr. Kaushik Kalyanaraman, who have always guided and provided constructive ideas that have helped me refine my work along the way. I would also like to express my gratitude to my external collaborator, Dr. Aniket Bera from Purdue University, for his insights. A special thanks to my project partners Apoorv Khattar, Anish Madan, Phu An Pham, and Pragya Sethi, who always showed unwavering support.

I would also like to thank my mother, Aradhana N. Mathur, and father, Dr. Neeraj Mathur who have always encouraged me to learn endlessly from the very inception of my life. All my work is dedicated to them. The two pillars who have always stood by me and my sister, Mayuri, who has always encouraged me to become better.

I would also like to express my thanks to all the friends I made, especially Sharat, Pravin bhaiya & Anupriya di, Gaurav, Shounak, Anshul, Shyama, Venky, Akshit, Karan, Sunil, Khushboo, Sagnik, Tharma, Sourav, Vatsala and Nilay.

I would also like to convey my sincerest gratitude to Ph.D. administration and members of the IT staff, Bhawani for continued support and assistance.

Aradhya Neeraj Mathur

Publications

The overall theme of the research pertains to the class of reconstruction problems using deep learning methods. Reconstruction is performed across different modalities.

Published

1. Mathur, Aradhya Neeraj, Apoorv Khattar, and Ojaswa Sharma. "2D to 3D medical image colorization." Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2021.

2. Mathur, Aradhya Neeraj, Devansh Batra, Yaman Kumar Singla, Rajiv Ratn Shah, Changyou Chen, Roger Zimmermann. "Lifi: Towards linguistically informed frame interpolation." ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021.

3. Mathur, Aradhya Neeraj, Anish Madan, and Ojaswa Sharma. "SLI-pSp: Injecting Multi-Scale Spatial Layout in pSp." Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2023.

4. Mathur, Aradhya Neeraj, Apoorv Khattar, and Ojaswa Sharma. "Curvy: A Parametric Cross-Section Based Surface Reconstruction". Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - GRAPP, INSTICC, 2025, pp. 139–150, https://doi.org/10.5220/0013300500003912.

Under Submission

Phu Pham, Mathur, Aradhya N., Aniket Bera, Ojaswa Sharma. "MV-Gaussian: High-Fidelity text-to-3D Content Generation with Multi-View Guidance and Surface Densification" under submission at WACV 2026.

Additional Works

Mathur, Aradhya N., Phu Pham, Aniket Bera, Ojaswa Sharma. "RL Dreams: Policy Gradient Optimization for Score Distillation based 3D Generation." arXiv preprint arXiv:2312.04806 (2023).

Table of Contents

Page

1 Introduction **1**

2 Background **6**

2.1 Modality Transitions in 3D Reconstruction 6

2.2 Approaches for Reconstruction and Synthesis 8

3 Multi-Modal Medical Image Colorization **16**

3.1 Related Work 17

3.2 Background 20

3.2.1 Poisson Image Editing 20

3.3 Proposed Approach 21

3.3.1 Dataset 21

3.3.2 MRI to grayscale conversion 21

3.3.3 Style transfer for generating hints 24

3.3.4 Volume colorization using hints 24

3.3.5 Upsampling 25

3.4 Results and comparisons 27

3.5 Discussion and Challenges 29

3.6 Closing Notes 29

3.7 Acknowledgement 30

4 *LiFi*: Towards Linguistically Informed Frame Interpolation **31**

4.1 Related Work 32

4.1.1 Video Interpolation 33

4.1.2 Facial Video Generation 34

4.2 Evaluation Methods 34

4.3 Approach 40

4.3.1 Dataset 40
4.3.2 Models 41
4.4 Results and Discussion 43
4.5 Closing Notes 46

5 SLI-pSp:Injecting Multi-Scale Spatial Layout in pSp 47
5.1 Related Work 49
5.1.1 GANs 49
5.1.2 Conditional GANs & Image-to-Image Translation 50
5.1.3 Self-Attention Methods 51
5.2 Approach 52
5.2.1 Overview 52
5.2.2 Baseline Method 53
5.2.3 Spatial Layout Injection 53
5.2.4 Variant - Attentive Spatial Layout Injection 54
5.2.5 Loss Functions 54
5.3 Experimental Setup 55
5.3.1 Datasets 55
5.3.2 Training Details 56
5.4 Results and Discussion 56
5.4.1 Segmentation Map to Face Image 56
5.4.2 Super-Resolution 58
5.4.3 Edges to RGB Image 59
5.5 Closing Notes 61

6 Curvy: A Parametric Cross-section based Point Cloud Generation 63
6.1 Related work 64
6.1.1 Pointcloud generation 65
6.1.2 Surface reconstruction 65
6.2 Approach 66
6.2.1 Parametric curve fitting 67
6.2.2 Training on parametric space 70

6.3 Results and Discussion 75
6.3.1 Cross-section dependence 75
6.3.2 Attention 77
6.3.3 Changes in Embedding 78
6.3.4 Edge Cases 78
6.4 Ablation Study 79
6.4.1 Variance in Loss 80
6.5 Comparisons 81
6.6 Failure Cases 83
6.7 Conclusion and Future Scope 84

7 RLQuad: Policy Gradient Optimization for Score Distillation based 3D Generation 85

7.1 Related Work 87
7.1.1 NeRFs 87
7.1.2 Diffusion Models 88
7.2 Approach 90
7.2.1 Quad-Tree based Densification 94
7.3 Experiments, Results, and Discussion 94
7.3.1 Impact of AES Term 96
7.3.2 QuadTree based Densification 99
7.3.3 Limitations & Future Scope 99
7.4 Conclusion 102

8 MVGaussian: High-Fidelity text-to-3D Content Generation 104

8.1 Related work 105
8.1.1 Text-to-image generation 105
8.1.2 3D Representations 106
8.1.3 Lifting to 3D 106
8.2 Method 107
8.2.1 Multi-view guidance for consistent 3D generation 108
8.2.2 Gaussian Alignment for Optimal Geometry 111
8.2.3 Surface densification and pruning 114

8.3 Experiments and Results 115
8.4 User Study 116
8.5 Limitations 117
8.6 Conclusion 119
8.7 Ethical concern and societal impact 119

9 Conclusion **120**

Appendices **123**

A Multi-Modal Medical Image Colorization **124**
A.1 Preventing unwanted textures in modality conversion using GANs 124
A.2 Testing on Unknown Volumes 125

B SLI-pSp:Injecting Multi-Scale Spatial Layout in pSp **127**

C MVGaussian: High-Fidelity text-to-3D Content Generation **134**
C.1 Quantitative Evaluation 134
C.2 Ablation studies 135
C.3 More Qualitative Comparisons 140
C.4 Implementation Details 142
C.5 Parameter analysis 145
C.6 Additional Results 146
*

List of Tables

Table **Page**

1.1 Overview of Thesis and summary of contributions. 4

3.1 Quantitative comparison with recent style transfer methods. . . . 29

4.1 Statistics for the challenge datasets made for each task. 35
4.2 Overview of LRS-3 dataset used for experimentation. 35
4.3 SSIM obtained upon the reconstruction of corrupted videos in the Visemic Corruption dataset. 37
4.4 Metrics obtained on the test set using FCN3D + ROI. 37
4.5 Overview of LRS-3 dataset used for experimentation. 38
4.6 Metrics obtained on the newly introduced challenge datasets. . . 43
4.7 Detailed metric table for reconstruction. 44
4.8 Reconstruction Metrics across different methods. 45

5.1 Quantitative evaluation of our method and pSp on the test set. . 57
5.2 Quantitative Results for Super-resolution task on CelebA. 59
5.3 Quantitative Results for Edges to RGB image task. 61

6.1 Per-class Chamfer Distance corresponding to the variation in the number of cross-sections. 78
6.2 Ablation results with varying attention mechanisms and connectivity. 79
6.3 Standard deviation of Chamfer loss across varying cross-sections. 81

7.1 Qualitative comparison with other text-to-3D methods. 95
7.2 CLIP scores and aesthetic scores for the models. 96

8.1 Comparison of different methods based on human evaluation scores. 115

C.1 Additional comparison of different text-to-3D methods based on the given prompt set. 134
C.2 Comparison of densification methods. 140
C.3 Sensitivity analysis of surface regularization weight w_s. 145
C.4 Sensitivity analysis of the smoothness regularization weight w_c. . 145

List of Figures

Figure **Page**

1.1 Research works grouped by modality focus. 3

2.1 Timeline showing the evolution of different modalities. 7
2.2 Key architectural difference between Autoencoders and Variational Autoencoders. 9
2.3 The GAN framework, with generator model and discriminator model. 10
2.4 Working of a diffusion model leveraging forward and backward process. 11
2.5 NeRF Architecture. 12
2.6 Pipeline of Gaussian Splatting based rendering. 13

3.1 Pipeline for 3D Volumetric Colorization. 17
3.2 Poisson Image Editing. 20
3.3 Modality conversion and hint slice generation for colorization. . . 25
3.4 Colorization result with a cryo style image. 26
3.5 3D rendering of the synthesized volume generated with the style image in Inviwo. 26
3.6 Comparison with exemplar colorization methods. 28
3.7 Comparison against style transfer methods. 28

4.1 Synthetic video clip generation metrics. 38
4.2 Super-SloMo interpolation results. 39
4.3 Demonstration of effective synthesis: high SSIM and PSNR from only two input frames. 39
4.4 Sample Completion using ROI stream. 40
4.5 LiFi-architecture of the proposed ROI loss. 41

5.1 Impact of SLI-pSp approach on samples, demonstrating generation of finer details. 48
5.2 Proposed SLI-pSp framework for conditional image generation. . 52
5.3 We show results on the Seg2Face task using the CelebAMask-HQ dataset and compare it to some previous works. 57
5.4 SLI-pSp Superresolution Results for 8×, 16× and 32×. 58
5.5 SLI-pSp AFHQ super-resolution results. 60
5.6 SLI-pSp results on Places2 dataset. 60

6.1 Proposed GCN-reconstruction approach for parametric cross-section based point cloud generation. 63
6.2 Overview of our parametric reconstruction approach. 66
6.3 Converting a piecewise parametric representation of a cross-section to a graph. 68
6.4 Overall parametric training reconstruction pipeline in the GAN framework. 73
6.5 Results of the proposed model trained only on a single class of objects using parametric reconstruction. 75
6.6 Comparison of reconstruction quality with an increasing number of cross-sections. 76
6.7 Dependence of Chamfer loss on sampling conditions 76
6.8 t-SNE plots of the embeddings obtained from the ground truth complete point cloud. 79
6.9 Comparison with the state-of-the-art methods. The inset shows point-wise surface error compared with the GT. The blue regions indicate areas of low error, while brighter areas indicate higher error. 82
6.10 Failure cases resulting in incorrect shapes. 83

7.1 Architecture diagram of RL-based optimization. 90
7.2 Assets generated using AES (GSGEN, Dreamfusion). 97
7.3 DDPSO3D : MVDream Guidance. 97
7.4 Blue Jay DDPO: Zoom-in-View. 98
7.5 Parrot, Cupake AES stage 2 optimization. 98
7.6 QuadTree-based densification scheme. 99
7.7 Image-to-3D QuadTree vs Original Densification. 100

7.8 Image-to-3D quadtree densification sample. 100
7.9 Image-to-3D QuadTree vs Original Densification under constrained setting. 101
7.10 Text-to-3D QuadTree vs Original Densification under constrained setting. 101
7.11 Changes in the appearance with increasing weightage of the AES term. 103
7.12 Failure Cases: Artifacts observed in RLQuad samples. 103

8.1 Overview of our MVGaussian framework. 108
8.2 Extensive qualitative comparisons on object and scene prompts. . 109
8.3 Additional qualitative comparisons on character and object prompts. 110
8.4 Additional qualitative comparisons with several state-of-the-art methods. 111
8.5 Runtime comparison against ProlificDreamer. 117
8.6 Human assessment on different text-to-3D models. 118
8.7 Failure cases containing spike-like artifacts. 118

A.1 Convergence of training loss. 125
A.2 Colorization results on MIDAS dataset. 126
A.3 Rendering results with Inviwo renderer. 126

B.1 Additional Results generated by ASLI and SLI-pSp on Places2 dataset. 128
B.2 Multi-Modal Synthesis Results on Seg2Face task. 129
B.3 Additional Results generated by SLI-pSp on Seg2Face task. . . . 130
B.4 Additional Results generated by ASLI-pSp on Seg2Face task. . . 131
B.5 Additional Results generated by SLI-pSp on Super resolution task. 131
B.6 Additional Results generated by SLI-pSp on Super resolution task (16x downsampling). 132
B.7 Additional Results generated by ASLI-pSp on Super resolution task (8x downsampling). 132
B.8 Additional Results generated by ASLI-pSp on Super resolution task (16x downsampling). 133
B.9 Additional Results generated by SLI-pSp for Super-resolution task (8x downsampling). 133

C.1 Ablation study on the surface and flattening loss. 136
C.2 Ablation with regularization loss $\mathscr{L}_s$. 137
C.3 Ablation study on the s_g loss term. 138
C.4 Ablation study on the color and depth smoothness loss. 138
C.5 Artifacts observed when densification is based on 2D Gaussian splatting (2DGS). 139
C.6 Qualitative comparison of densification strategies. 140
C.7 Additional comparison results with existing methods. 141
C.8 Qualitative comparison with MVDream. 142
C.9 Effect of pruning threshold τ on the prompt. 146
C.10 Additional results generated using generic prompt template. . . . 147

Chapter 1

Introduction

Motivation

> "I stand upon my desk to remind myself that we must constantly look at things in a different way."- John Keating

Computer graphics is a field that brings art to life by bridging aesthetics with science, drawing from physics, engineering, and mathematics, to create immersive visual experiences. Data for such systems can be collected through a diverse range of acquisition frameworks, leading to a wide range of modalities acting as input signals. These include multi-sensory data, such as medical datasets comprising images and volumetric scans, as well as linguistic data like textual information [117]. The *reconstruction* and *synthesis* of visual data are fundamental problems in the fields of computer vision and graphics. *Reconstruction* refers to the restoration of a signal, whereas *synthesis* refers to the generation of a novel signal derived from specific information. These input signals are referred to as *prior* for generation. Despite their distinct goals, both tasks often use similar data and techniques. With the advent of deep learning, these processes have advanced rapidly, blurring the lines between them, both in terminology and methodology, with the terms *reconstruction* and *synthesis* often being used interchangeably in the literature.

The geometry and appearance of a shape are the two primary factors that determine the perceived quality of its synthesis. While geometry governs key aspects such as topology, shape, and spatial arrangement, radiometry governs perceptual properties such as color, texture fidelity, and lighting coherence. The geometric and radiometric guidance together form what we refer to as *structural priors* in reconstruction and synthesis problems. Inaccuracies in either aspect often result in perceptually unsatisfactory outcomes. Deep learning has allowed the field to move beyond conventional input signals such as images, videos, and volumes, which we shall hereafter refer to as *input modalities*, to more abstract priors such as text. Images, volumes, and videos are rich in geometric and radiometric details, whereas natural language inputs only provide general descriptions

without tight spatial correspondences, thus being a lot more abstract and often being subject to multiple interpretations.

This thesis dives deep into structurally consistent generation across modalities, whereby structural consistency encompasses both geometric integrity and radiometric fidelity. The work proposes consistent geometric and radiometric reconstruction and synthesis across different priors with varying levels of abstraction, ranging from low-level signals to only high-level semantic cues.

Recent AI-based Generative methods have enabled adaptation across diverse visual modalities, contributing to a wide range of applications and broadening access to 2D and 3D content generation for everyone. Social media, arts, entertainment, and engineering have been profoundly impacted by these methods, resulting in billions of 2D and 3D assets being generated by AI [4]. While the incorporation of different kinds of inputs as priors has expanded accessibility for everyone, it has brought newer challenges to the forefront, particularly to ensure structural consistency. The thesis addresses these challenges in several modern deep learning-based approaches, proposing solutions that yield high-quality results.

This thesis examines a range of problems, primarily considering two modalities - *image* and *surface*, encompassing both discretised and continuous 3D data. The thesis views videos and volumes as special cases of images: videos extend images by incorporating temporal information along with spatial details, whereas volumes extend images into three dimensions, capturing spatial information across depth. For the surface modality, the thesis investigates point clouds and Gaussian splatting-based representations that closely capture surface geometry, and further explores several recent techniques, such as differentiable rendering with text conditioning. While point clouds are points spatially sampled in three dimensions, Gaussians can be viewed as point clouds with additional shape and appearance properties such as scale, color, and opacity. The thesis provides a comprehensive set of tools and techniques to enhance the quality of results, ensuring structural consistency across various tasks. The thesis begins with methods where the prior is highly structured and *close* to the target modality, such as input and output both being images, as in several image-to-image translation methods [169]. Further, the thesis then investigates reconstruction from abstract inputs, which are *distant*, so to say, from the target modality, such as text-to-3D generation [179], making the task of reconstruction highly challenging. As the gap between the input and the target increases, designing constraints to ensure structural consistency becomes significantly difficult. Thus, as a secondary contribution, the thesis also investigates reconstruction and synthesis from sparse inputs. In this context, sparsity refers to a lack of sufficient geometric and radiometric information. There are multiple forms of sparsity that the thesis explores. For instance, cross-sections exhibit geometric sparsity while inputs like grayscale volumes lack radiometric richness. Similarly, modalities like

segmentation maps, edge maps, or text provide limited cues in both geometry and appearance.

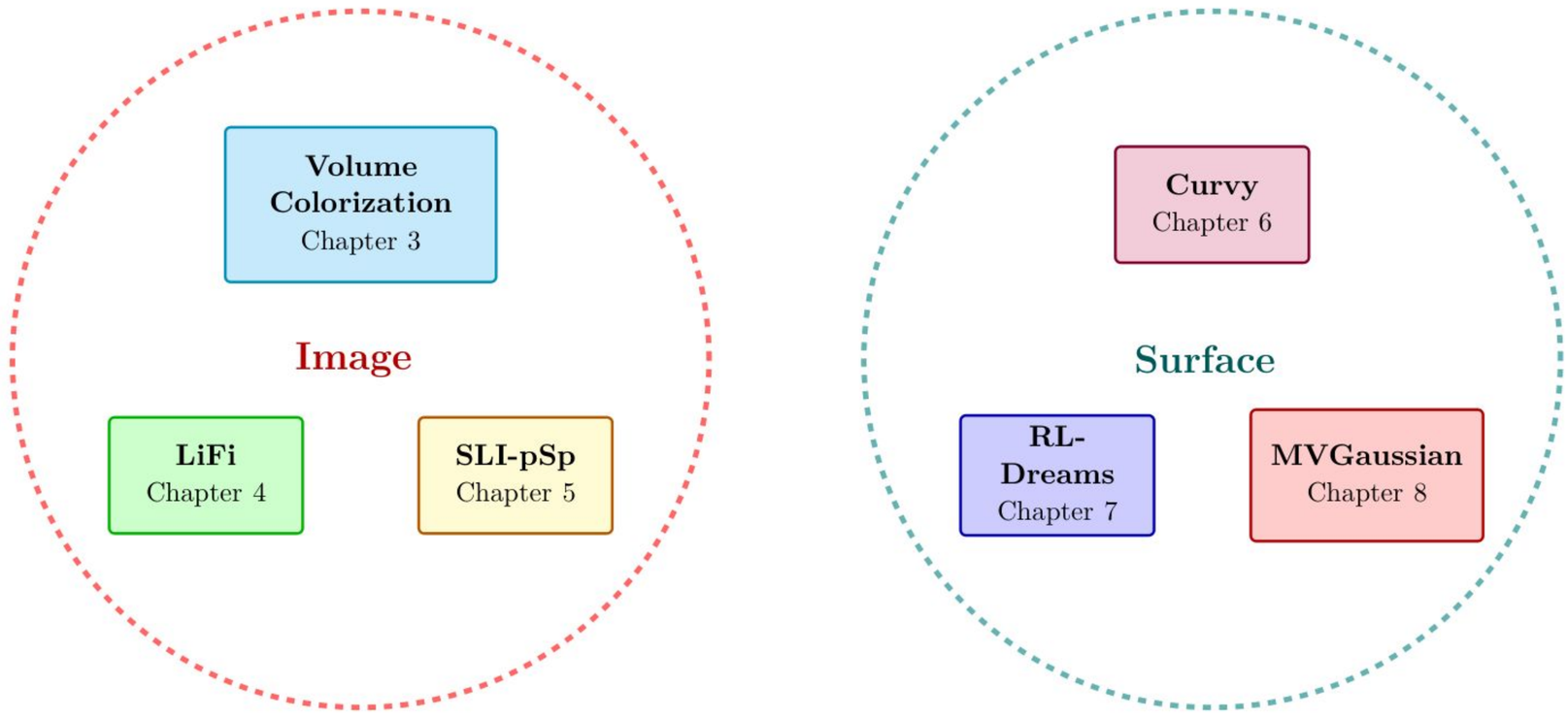


Figure 1.1: Research works grouped by modality focus.

Contributions and Organization of Thesis

Post a gentle introduction to the problem statement in this chapter, Chapter 2 presents a comprehensive overview of key deep learning-based reconstruction and synthesis techniques that serve as the foundation for this thesis, providing the necessary background for the proposed solutions. The key contributions, illustrated in Figure 1.1, span multiple modalities and outline proposed works and are further summarized in Table 1.1. The contributions span diverse problems involving different representations, from dense volumetric medical data to abstract natural language, revealing the growing complexity of the generative task as input becomes more sparse or structurally *distant* from the input.

Image as Modality

The first three works discuss image as a modality, covering volume, video, and image reconstruction and synthesis problems. The image-based methods cover spatial (volume), temporal (video), and semantic (label maps) prior-based generation methods. Chapter 3 discusses extension of classical image-based generative models to volumetric data, motivated by longstanding challenges in medical volume visualization. Specifically, it addresses the task of photorealistic medical volume colorization, a domain that has traditionally relied on manually designed transfer functions. The method presented introduces a novel deep learning-based pipeline for voxel-wise synthesis of photorealistic colored volumes from grayscale MRI data. It emphasizes the preservation of anatomical geometry while achieving a photorealistic appearance. This work handles radiometric sparsity by draw-

Table 1.1: Overview of thesis and summary of contributions.

Chapter	**Area of contribution**	**Input modality**	**Output modality**	**Target consistency**	**Sparsity of input**
Image as modality					
Chapter 3	Photorealistic volume colorization (MRI)	Grayscale volumetric data	Colored volumetric data	Radiometric	Lacks color/texture information; low resolution and low details.
Chapter 4	Viseme-aware video Frame interpolation	Sparse video frames with linguistic cues	Video frames	Geometric & radiometric	Sparse temporal sampling; incomplete visual information; linguistic ambiguity
Chapter 5	StyleGAN-based conditional image generation	Segmentation masks & edge maps	High-resolution color images	Geometric & radiometric	Structurally sparse; lacks high-frequency detail and photometric richness
Surface as modality					
Chapter 6	Point cloud generation	2D planar cross-sections	3D point clouds	Geometric	Sparse geometry
Chapters 7 and 8	Text-to-3D generation with SDS	Free-form text	Textured 3D meshes or rendered images	Geometric & radiometric	Extremely sparse: abstract semantic descriptions with no direct 3D or appearance cues

ing inspiration from image colorization and style transfer methods, performing hint-based colorization. This work achieves high-fidelity volume rendering while preserving geometric detail, offering one of the first data-driven solutions for photorealistic medical volume colorization.

The following chapters explore surface as a modality, investigating point cloud and recent state-of-the-art neural rendering-based techniques. Building upon these 2D and 3D synthesis techniques, video reconstruction is explored in Chapter 4, particularly in the context of a linguistically informed frame interpolation. This work focuses on reconstructing speech videos from sparsely sampled frames. By incorporating viseme-aware conditioning, the method enhances the quality and naturalness of frame interpolation, especially relevant during the COVID pandemic when video communication was both prevalent and bandwidth-constrained. This chapter serves as a bridge between image inputs and video outputs, leveraging linguistic cues to guide reconstruction.

Further exploring the broader landscape of generative techniques, the subsequent work notes the limitations in high-resolution texture synthesis, particularly in StyleGAN's [98] ability to perform high-resolution image generation from low-frequency conditional inputs. In Chapter 5, the thesis delves into conditional StyleGAN architecture pSp-StyleGAN [188], and proposes enhancements for synthesizing detailed, high-resolution imagery from sparse inputs such as semantic segmentation maps and edge maps. The method demonstrates significant improvements over prior approaches, especially in synthesizing fine-grained textures for complex domains such as human faces, and ensuring stronger geometric consistency.

Surface as Modality

Recognizing that while volumetric grids offer dense representations for anatomical structures, a large portion of real-world 3D data is captured and utilized as point clouds or meshes, the subsequent work focuses on point representation. Drawing upon insights from volumetric synthesis and noting that cross-section-based reconstruction remains a relatively unexplored area in deep learning, the thesis focuses on point cloud generation. Chapter 6, introduces a novel reconstruction method that maps sparse parametric curves to point clouds using a graph neural network in a GAN-based setting. This work bridges implicit and explicit 3D representations, providing a flexible and learnable reconstruction pipeline that performs competitively against optimization-based and learning-based state-of-the-art methods.

Finally, in Chapters 7 and 8, the works extend the modality-centric framework to natural language, the most abstract modality considered in this thesis. These works address the task of text-to-3D generation, which presents unique challenges due to the lack of explicit geometric grounding. The contributions include a novel back-projection mechanism, depth-based regularization, and a reinforcement learning-based perceptual scoring function. Additionally, the proposed work mitigates the Janus problem, commonly observed in multi-view synthesis, by strategically selecting a backbone architecture that promotes stable, consistent 3D generation. These works focus on enforcing both geometric and radiometric consistency in the newly developed field of neural rendering with text-based inputs.

From volumetric and point-based 3D representations to viseme-aware video reconstruction and abstract text-to-3D synthesis, the chapters are unified by a modality-centric philosophy that guides the design and development of each method to ensure structural consistency within the broader scope of multimodal generation.

Chapter 2

Background

In the field of computer graphics, both reconstruction and synthesis lie within the subset of modeling problems associated with the mathematical specification of the shape and appearance of data. However, this modeling is far from monolithic: data can take on a variety of forms and representations, each offering unique advantages while introducing distinct challenges.

The performance of reconstruction or synthesis is measured based on both geometric integrity and radiometric fidelity, which drive how the energy functions for optimizing these methods are defined. Each representation imposes its own constraints on generative modeling in terms of structure, expressivity, and feasibility, thus requiring uniquely designed algorithmic strategies. Given the recent success of deep learning methods owing to the data-driven effectiveness and rapid development of GPUs for tensor operations [144] supporting it, we focus on deep learning-based solutions for generative modeling. Careful consideration of constraints imposed by the unique inputs allows for a plausible, structurally consistent generation, enhancing visual fidelity.

2.1 Modality Transitions in 3D Reconstruction

To understand the evolution of modality in 3D reconstruction, we revisit the field's origins in the 1960s with the development of tools like Sketchpad [216], which enabled direct manipulation of geometric objects and laid the groundwork for CAD systems. The 1970s and 1980s saw the rise of procedural modeling, with methods such as L-Systems [192] allowing for the automatic generation of complex 3D content through mathematical rules. In the 1990s and 2000s, rapid advancements in computer vision, particularly techniques leveraging epipolar geometry [219], helped reconstruct 3D geometry from 2D images. Conditional 3D reconstruction methods have traditionally operated on input-output pairs that are closely aligned in terms of information content and/or structure, such as stereo image pairs, depth maps, point clouds, cross-sections, or volumetric medical images [76, 78, 93, 213]. This tight coupling of inputs and outputs

has enabled the development of methods grounded in multi-view consistency, epipolar geometry, or interpolation [37, 48, 66].

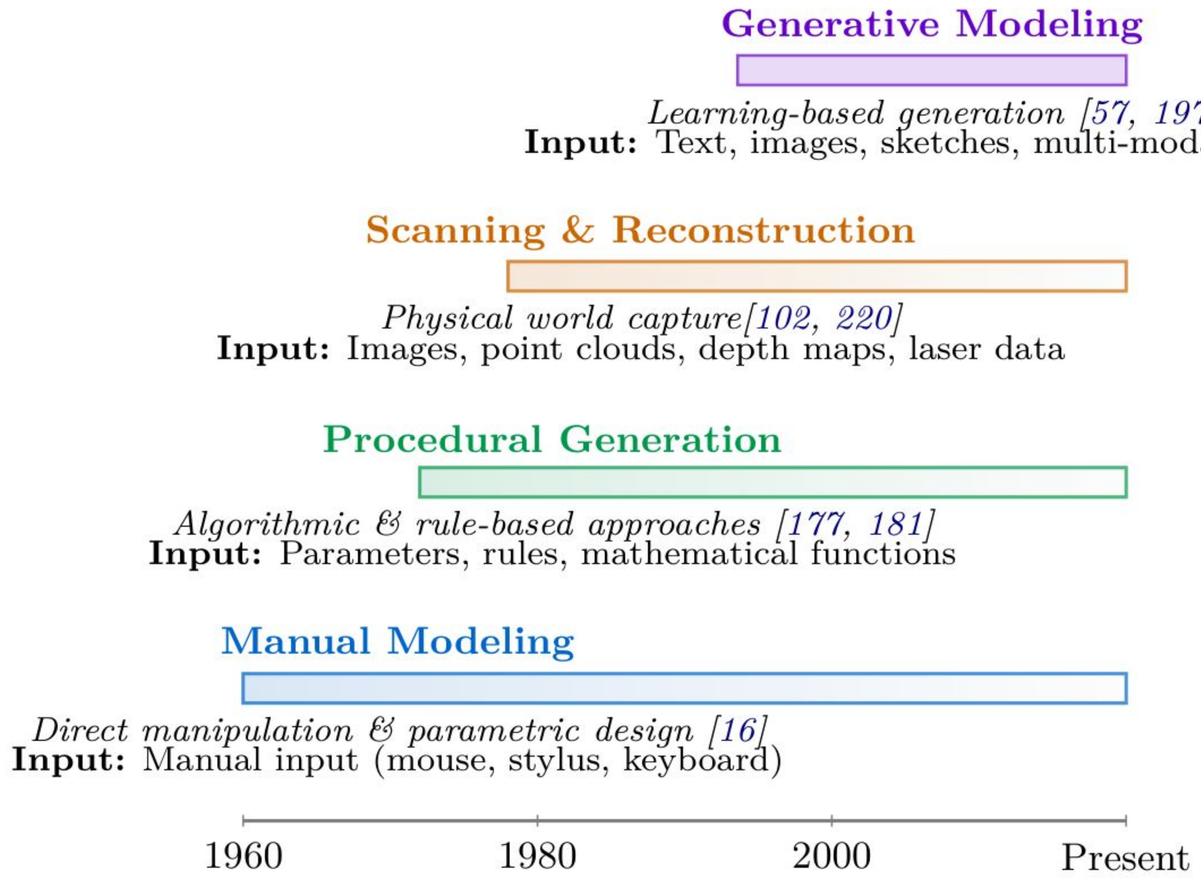


Figure 2.1: Timeline showing the evolution of different approaches for 3D content generation from 1960 to present [56, 192, 216, 219], denoting when active development began for each method.

Deep learning initially leveraged data-driven methods to perform reconstruction and synthesis for inputs that were closely aligned with the target modalities [56]. Similar to the aforementioned techniques, these early data-driven 3D reconstruction and synthesis methods exhibited a strong correspondence between input and output pairs. They were built upon structural constraints used in classical approaches and adapted data-driven frameworks. We follow a similar problem setting in Chapters 3, 4, and 5, where synthesis leverages classical methods such as Poisson Image Editing [176] and the tight coupling between input and output for energy function design.

However, in recent years, the field has advanced significantly, with breakthroughs in neural 3D synthesis and text-to-3D modeling further expanding the space of usable inputs and modalities for 3D content creation. This variation across different inputs is summarized over time in Figure 2.1. Over time, the field has witnessed a dramatic shift from relying on traditional methods and towards a new class of inputs that are no longer tightly coupled to the output in terms of geometry or structure, providing only high-level semantic cues. By approximating functions over data distributions, generative models enable the synthesis of plausible 3D structures even when the input is sparse, noisy, or highly abstract [106, 233].

Leveraging large-scale datasets, these models can hallucinate complete 3D objects that are consistent with the natural distribution of the data [75, 170, 246].

This allows a deep learning model to perform synthesis from *distant* inputs such as free-form text [179], which, despite lacking explicit 3D signals, enables generation based on the semantic information captured by the underlying networks. Thus, driving a shift in modality, where the input no longer needs to have a tight geometric alignment with the output and is explored for Chapters 6, 7, and 8 whereby newer methods such as Score Distillation Sampling (SDS) [218] are leveraged for generation, discussed further in detail.

The following sections discuss several reconstruction and synthesis methods that form the foundation for the methods that we leverage to develop our solutions.

2.2 Approaches for Reconstruction and Synthesis

Given some input signal x', from a target signal (complete signal for reconstruction or plausible target signal for synthesis) $x \sim p(x)$, the function f parameterized by θ aims to retrieve the target signal x. Thus, the objective is to find the optimal parameters θ^* that minimize the reconstruction/synthesis loss given by d between the generated signal $f(x';\theta)$ and the ground truth signal x, formulated as

$$\theta^* = \arg\min_{\theta} d(f(x';\theta), x).$$

Images and volumetric data involve textures, intensities, and geometric details to be reconstructed. Point clouds and meshes, on the other hand, require accurate geometric details and point densities to achieve high-quality generation. For distance field approximation, one needs to capture the underlying implicit representation precisely. In order to do this, either unconditionally or conditionally (on some prior $y \sim p(y)$), it is necessary to model the underlying distribution. Generative methods refer to the class of methods that focus on learning the underlying data distribution. Unlike discriminative classifiers that usually model the conditional distribution, generative models aim to learn the joint distribution $p_{data}(x, y)$ from which one can derive both the marginal $p(x)$ and the conditional $p(y|x)$ [17].

The following sections explore key methods that form the basis for reconstruction conditioned on various forms of input. The section begins with several deep learning-based generative methods, followed by recent methods that leverage more distant modalities for 3D synthesis as the field evolves. Deep learning-based methods have been used heavily for developing techniques for colorization [269], image generation [58], image-to-image translation [167], and style transfer [91] and further extended to volumes [38] and videos [88].

Auto-Encoders, Variational Auto-Encoders & GANs

Given a data sample x drawn from an underlying data distribution $p(x)$, an autoencoder as shown in Figure 2.2, aims to learn a neural network function $f(.;\theta)$, parameterized by θ that represents the network mapping a compressed representation of this sample x by learning to encode a much lower-dimensional representation of x which is then decoded to yield $\hat{x}$ such that the generated sample $\hat{x}$ and x are close in some metric space. This is typically done by minimizing the Euclidean or Manhattan distance between the generated sample $\hat{x}$ and the ground truth/input x to attain the optimal parameters θ^*.

$$\theta^* = \arg\min_{\theta} ||f(x;\theta) - x||_2^2.$$

This has further been modified to allow for denoising and restoration of the sample x by training the network $f(.;\theta)$ on a noisy data sample x' to generate a clean sample by learning to map a noisy sample to the denoised sample, with a minor change in the above equation as:

$$\arg\min_{\theta} ||f(x';\theta) - x||_2^2.$$

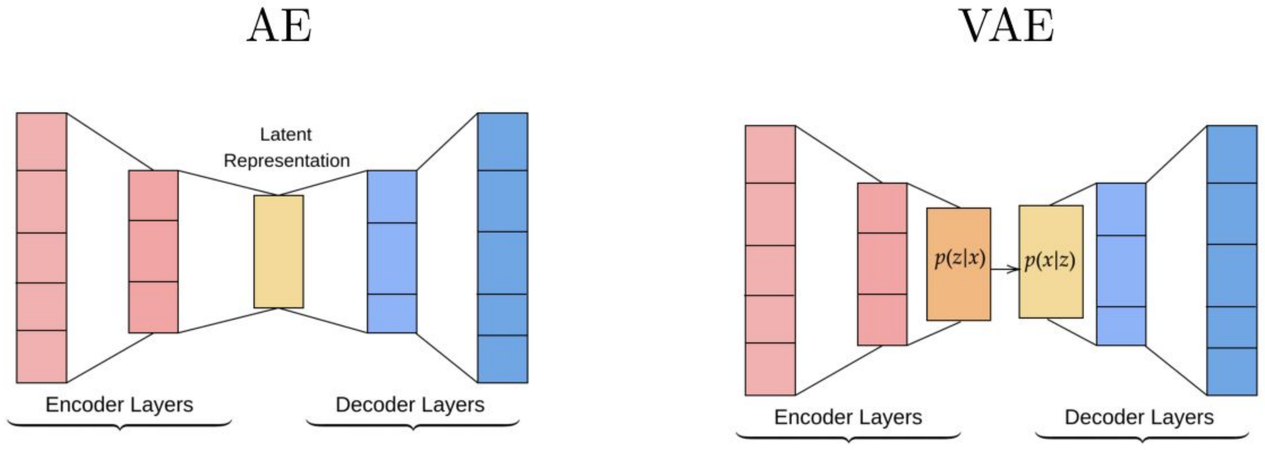


Figure 2.2: An Autoencoder (left) maps an input signal to a compressed latent representation and then decodes it to match the original signal, whereas Variational Autoencoders (VAE) (right) predict the parameters of the latent distribution from which the latent variables can be sampled and the data can be reconstructed via decoding.

Improving upon autoencoder, Kingma et al. [109] introduced the Variational Autoencoders, in which evidence lower bound (ELBO) is used as the objective function. The ELBO includes the KL-Divergence, D_{KL} term that regularizes the approximate posterior towards a Gaussian prior. The proposed method not only allows for compression of the data sample x by creating a *bottleneck* as done in autoencoders, but restricts the latent space to a Gaussian, from which new samples can then be drawn post-training, unlike vanilla auto-encoders. This allows for generating novel samples from Gaussian noise in both conditioned/unconditioned settings. The authors achieve this with the help of the reparameterization trick. Following which, the mean and covariance predicted by the encoder are used

to scale and shift the input Gaussian noise, which is then decoded to yield the sample as shown in Figure 2.2. In a variational autoencoder, the intractable true posterior distribution $p_\theta(z|x)$ is approximated by a parametric encoder $q_\phi(z|x)$, where ϕ are the encoder parameters. The decoder network, parameterized by θ, models the likelihood $p_\theta(x|z)$, and learns a continuous latent representation z of the input x. The parameters θ and ϕ are jointly optimized by minimizing the variational loss:

$$\arg\min_{\theta} \log p_\theta(x|z) - D_{KL}\left[q_\phi(z|x) \| p_\theta(z)\right].$$

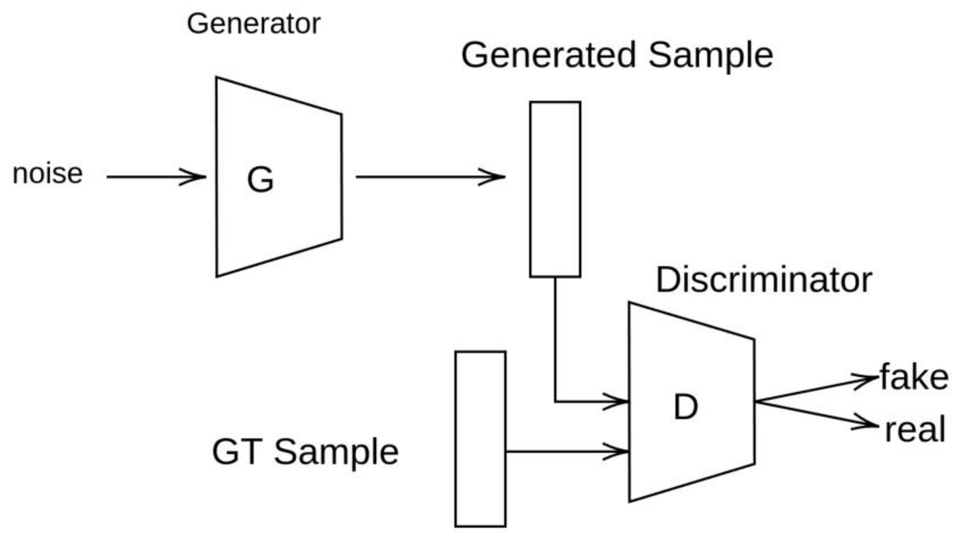


Figure 2.3: The GAN framework uses a generator and a discriminator in a min-max game setting to learn the target distribution.

Generative Adversarial Networks (GANs) were proposed by Goodfellow et al. [57] as a method for learning to model the data distributions by drawing inspiration from game theory. The authors proposed an adversarial training regime for transforming a given distribution, such as a Gaussian, to a target distribution. This method involves two neural networks, namely the generator G and the discriminator D. The generator and the discriminator play a two-player minimax game, which upon convergence yields a trained model G. The learned generator G maps a sample $z \sim p_z(z)$ from a Gaussian or some other distribution p_z to a sample in the target distribution p_{data} given by

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))].$$

The task of the generator, as shown in Figure 2.3, is to generate a sample that matches the target distribution by fooling the discriminator, which trains to differentiate between the ground truth data and the generated sample.

Diffusion Models

GANs allow for diverse, high-quality generation but are notoriously hard to train. Thus, the technique of diffusion was proposed by Ho et al. [71], where the authors introduce a denoising mechanism that trains a network. The network converts Gaussian noise into a sample from the target distribution by gradually denoising the input. The model is trained by progressively reversing the noisy data back to the original sample across different time steps. To prepare the training data, samples are corrupted with Gaussian noise following a scheduling scheme, known as the forward process. The network is trained on this noisy data sample ϵ_t conditioned on the time step, to predict the noise of the previous time step, ϵ_{t-1}. Longer time steps have empirically been shown to boost performance by enabling more gradual conversion from the underlying data distribution to a Gaussian. The data can further be conditioned on other priors to allow for conditional generation, which has led to massive strides in text-to-image and text-to-3D content generation through the development of score distillation methods [179] based on this technique. The diffusion network $f_\theta(.)$ takes a noisy sample x_t at time step t and predicts the noise $\hat{\epsilon}_t$, which, when close to the ground truth noise ϵ_t can be used to yield a cleaner sample at time step $t-1$ learned by minimizing the objective function

$$\theta^* = min_\theta \; ||f_\theta(x_t, t) - \epsilon_t||_2^2,$$

where $\theta*$ denotes the optimal network parameters.

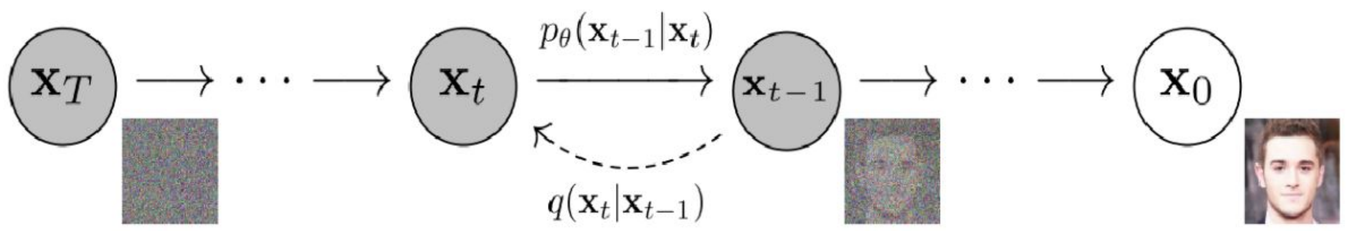


Figure 2.4: A diffusion model utilizes a scheduled forward noising and a learnable backward denoising process to map noise to target distribution [image source].

The process of diffusion begins by mapping the data sample $\mathbf{x}_0$ to a latent representation $\mathbf{z}_0$ using an encoder E, such that $\mathbf{z}_0 = E(\mathbf{x}_0)$. The forward diffusion process in the latent space involves gradually adding Gaussian noise to the latent representation over a series of discrete time steps t, creating a noisy latent variable $\mathbf{z}_t$ at each step. This can be mathematically represented as:

$$\mathbf{z}_t = \sqrt{\bar{\alpha}_t}\mathbf{z}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon,$$

where $\bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s$ and α_s represents the noise schedule at time $t = s$, and $\epsilon \sim \mathcal{N}(0, I)$ is Gaussian noise.

The objective function for training the denoising model involves minimizing the expected L_2 loss between the predicted and the true noise:

$$\mathscr{L} = \mathbb{E}_{t,\mathbf{z}_0,\epsilon}\left[||\epsilon_\theta(\mathbf{z}_t, t) - \epsilon||^2\right].$$

After the reverse diffusion process is completed, the denoised latent variable $\mathbf{z}_0$ is mapped back to the data space using a decoder D, resulting in $\hat{\mathbf{x}}_0 = D(\mathbf{z}_0)$. This process enables the model to iteratively restore the original data sample from its corrupted representation, as shown in Figure 2.4.

NeRFs

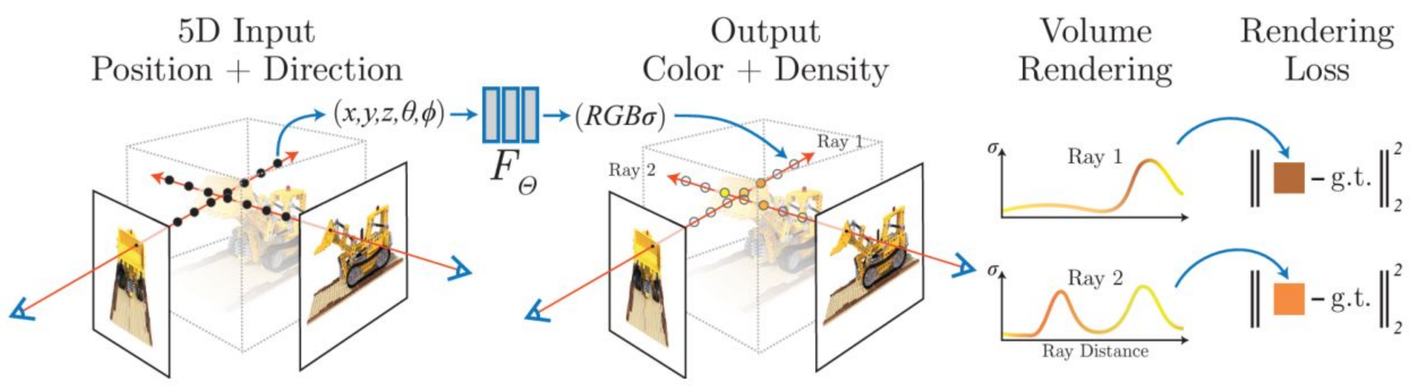


Figure 2.5: NeRF Architecture, a neural network is learned to predict the color and density at a point in space [image source].

One of the seminal works by Mildenhall et al. [155], as shown in Figure 2.5, introduced novel view synthesis by using shallow MLPs and allowing them to overfit on a set of views conditioned on the respective camera poses. The input to such a network is a 5D point (position $x(x, y, z)$ and viewing direction $d(\theta, \phi)$), and the output is an implicit field consisting of RGB color and density σ at any point. For rendering a view after optimization, a volume raymarching process evaluates the MLP at various ray positions to compose an image. The expected color $C(r)$ for a camera ray $r(t) = o + td$ traveling between the near and far bounds t_n and t_f is computed using the transmittance T, emitted color $c(r, g, b)$:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)dt\,, \text{ where } T(t) = \exp\left\{-\int_{t_n}^{t} \sigma(r(s))ds\right\}.$$

The transmittance T gives the probability that the ray travels from t_n to t without hitting any other particle. The neural network $f(;\theta)$ provides a mapping from the position $\mathbf{x}$ and the viewing direction d to the color C optimized by minimizing MSE between the predicted and the color from the rendered image.

$$\mathscr{L} = \sum_{\mathbf{r}\in\mathscr{R}} \left\|\hat{C} - C(\mathbf{r})\right\|_2^2$$

where $\hat{C}$ is the predicted color.

Gaussian splatting

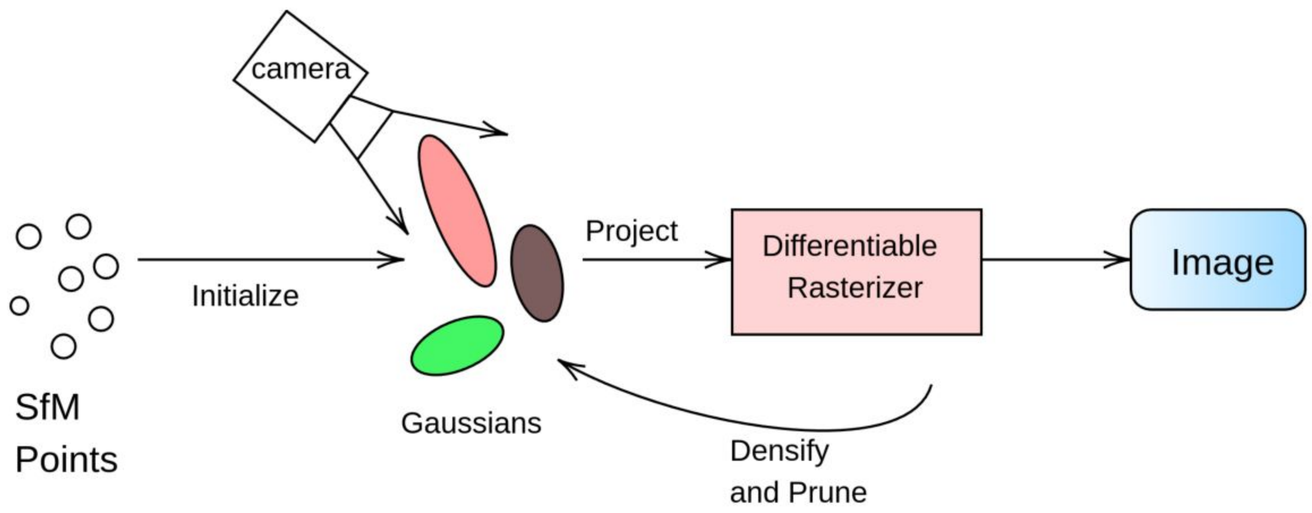


Figure 2.6: Using an SFM-based point cloud for initializing the Gaussians, the method optimizes the Gaussians and the density to render the target images [image source].

Kerbl et al. [104] present a technique to enable fast high-resolution novel view synthesis using Gaussian Splatting, which has emerged as a faster alternative for NeRF-based approaches. The technique, as shown in Figure 2.6, first generates a sparse point cloud from a set of multi-view images using Structure-from-Motion (SFM) and uses that for initializing 3D Gaussians that are differentiable and allow for easy projection to 2D splats, resulting in fast α-blending for real-time performance.

The seminal work presented in [105] introduces an innovative explicit approach for representing and rendering three-dimensional objects using Gaussian functions as the fundamental building blocks. 3D Gaussian splatting employs continuous Gaussian distributions to define the geometry and appearance of a 3D model. Each Gaussian is parameterized by its center position or mean μ, color c, opacity σ, and a full covariance matrix Σ. This covariance matrix Σ is decomposed into a rotation matrix R and a scaling matrix S, formulated as:

$$\Sigma = RSS^T R^T.$$

The parameters R and S are stored and optimized independently. Consequently, a 3D Gaussian can be defined as:

$$G(x) = \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

where $x \in \mathbb{R}^3$ represents a point in 3D space, $\mu \in \mathbb{R}^3$ is the Gaussian's center, and $\Sigma \in \mathbb{R}^{3\times3}$ is the covariance matrix that defines the shape, orientation, and scale of the Gaussian. Each Gaussian splat also includes a color vector $c \in \mathbb{R}^3$,

representing the RGB values and a scalar $\sigma \in [0, 1]$, indicating the transparency level.

To render a scene using Gaussian splats, the 3D Gaussians are projected onto a 2D image plane. The contribution of each splat to the final image is determined by integrating the Gaussian over the pixels it influences. The final color of a pixel p is a combination of the influences from all depth-sorted points x that project to pixel p

$$C = \sum_i c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j).$$

Unlike implicit representations used in NeRF models, Gaussian splatting is an explicit method that requires a mechanism to manage the number of Gaussians. This is achieved through a unique densification and pruning scheme, discussed subsequently. The authors further propose a method for densifying the Gaussians to enhance the information content in the image. This method addresses two specific conditions: under-reconstruction and over-reconstruction. Under-reconstruction occurs when additional Gaussians are needed in a region to complete the image. This is addressed by cloning existing Gaussians in that area. Over-reconstruction, on the other hand, occurs when a large Gaussian encompasses small-scale geometry and is handled by splitting the large Gaussian. Both these conditions lead to large view-space positional gradients.

While these methods learn from a set of images and generate novel views, several works in the realm of generative modeling have also utilized them for the same purpose and have been leveraged extensively later in the thesis.

Diffusion+NeRF/Gaussian Splatting Models for 3D Rendering

Numerous diffusion models have undergone training on extensive image datasets, such as LAION [202], containing over a billion meticulously curated training samples sourced from the web. These prompt-based models exhibit remarkable generalization capabilities. Leveraging the strengths of prompt-based diffusion models, researchers have employed them to train Neural Radiance Fields (NeRF). Poole et al. [179] introduced Score Distillation Sampling, which involves utilizing a pre-trained diffusion model as a frozen critic to guide the NeRF in image generation. To incorporate pose information, prompts are enriched with pose-centric phrases and keywords, enhancing the NeRF model's ability to distill accurate pose-dependent details.

Score Distillation Sampling (SDS) [179] is an optimization strategy that integrates the generative capability of pre-trained 2D diffusion models into the synthesis of 3D objects. This is achieved by optimizing a 3D model such that

its 2D projections, when viewed from random angles, mimic the image distribution learned by a diffusion model. The key aspect of SDS is its utilization of score functions derived from the diffusion process to inform the optimization, thereby ensuring that the generated 3D models align with the statistics of images conditioned on textual prompts.

The score functions $s_\phi(\mathbf{z}_t; \theta) = -\epsilon_\phi(\mathbf{z}_t; \theta)/\sigma_t$ derived from the diffusion model are used to compute updates for the Gaussian parameters θ, where s is the diffusion model or the score function parameterized by ϕ, ϵ_ϕ is the predicted noise and σ_t is the noise standard deviation pre-determined by the diffusion scheduler. These functions provide gradients of the log probability with respect to the latents, guiding the optimization to areas of higher probability under the model's distribution.

Then, as shown in [179], the gradient of L_{SDS} for a text-to-3D pipeline can be computed as

$$\nabla_\theta L_{\text{SDS}} = \mathbb{E}_{t,\epsilon}\left[w(t)(\hat{\epsilon}_\phi(\mathbf{z}_t; c, t) - \epsilon)\frac{\partial \mathbf{x}}{\partial \theta}\right],$$

where $\hat{\epsilon}_\phi$ is the noise prediction by the model, $w(t)$ is a weighting function depending on the time step t, ϵ is the noise added in the forward process, $\mathbf{x}$ is the rendered image of the 3D model represented by θ and z_t is the latent vector that is being denoised. Additionally, the score function s is conditioned on the context vector c obtained from the embeddings of the prompt provided along with the latent vector z_t. However, since the guidance is from text-to-image diffusion models, it lacks explicit 3D supervision, leading to face repetition, commonly known as the "Janus" problem, and blurring. Subsequent works in the field leveraging deep learning are further discussed in Chapters 7 and 8.

Chapter 3

Multi-Modal Medical Image Colorization

Colorization involves the synthesis of colors while preserving structural content as well as the semantics of the target image. This problem has been well studied for 2D photographs with many state-of-the-art solutions. We explore a new challenge in the field of colorization where we aim at colorizing multi-modal 3D medical data using 2D style exemplars.

Image colorization enables transferring the radiometric characteristics of one image onto another. From an artistic perspective, it has opened new horizons in content creation and has seen rapid strides with the advent of digital media. The problem of colorization has been extensively explored for 2D photographs; however, the same approaches do not work well for 3D medical images. Our current research aims at finding a robust approach to apply colorization in the 3D domain for multi-modal medical data. Automatic and realistic colorization of 3D medical data is required to improve the volume visualization capabilities of current systems. Our approach aims at automating colorization in medical imaging visualizations that are mostly manually controlled currently. Furthermore, what makes this problem setting more unique is the ability of our approach to generate photorealistic 3D volumes that can be used for better visualizations compared with existing approaches.

We view our approach as a suitable replacement to the current colorization methods prevalent in the volume visualization community, such as transfer functions (TF) based methods that are heavily used in the visualization of 3D CT and MRI modalities. A TF maps intensity values to colors; however, the creation of such a mapping is mostly manual, making it cumbersome and time-consuming. Automatic and semi-automatic synthesis of TFs have been explored in the volume visualization community, and many such ideas are summarized by Ljung et al. [139], however, these approaches still rely on the creation of a TF. Colorization can be seen as an effective alternative to using a transfer function, but it is an ill-posed problem since MRI or CT medical data is a different modality compared to a photograph.

In this work, we propose a framework to perform modality conversion of a 3D MRI to a photographic volume. While doing so, we also address the challenges of handling 3D data in using deep learning for the medical domain. We show multi-modal volume colorization using only a single 2D style image. Such a colorization can be used to create better 3D visualizations without the need to create time-consuming transfer functions. Our approach can be used to colorize other modalities like 3D CT as well.

To the best of our knowledge, this work is the first of its kind and poses challenges related to the modality (medical MRI) and dimensionality (3D volumetric images) of the data. Our approach to colorization is motivated by modality conversion, which highlights its robustness in handling multimodal data.

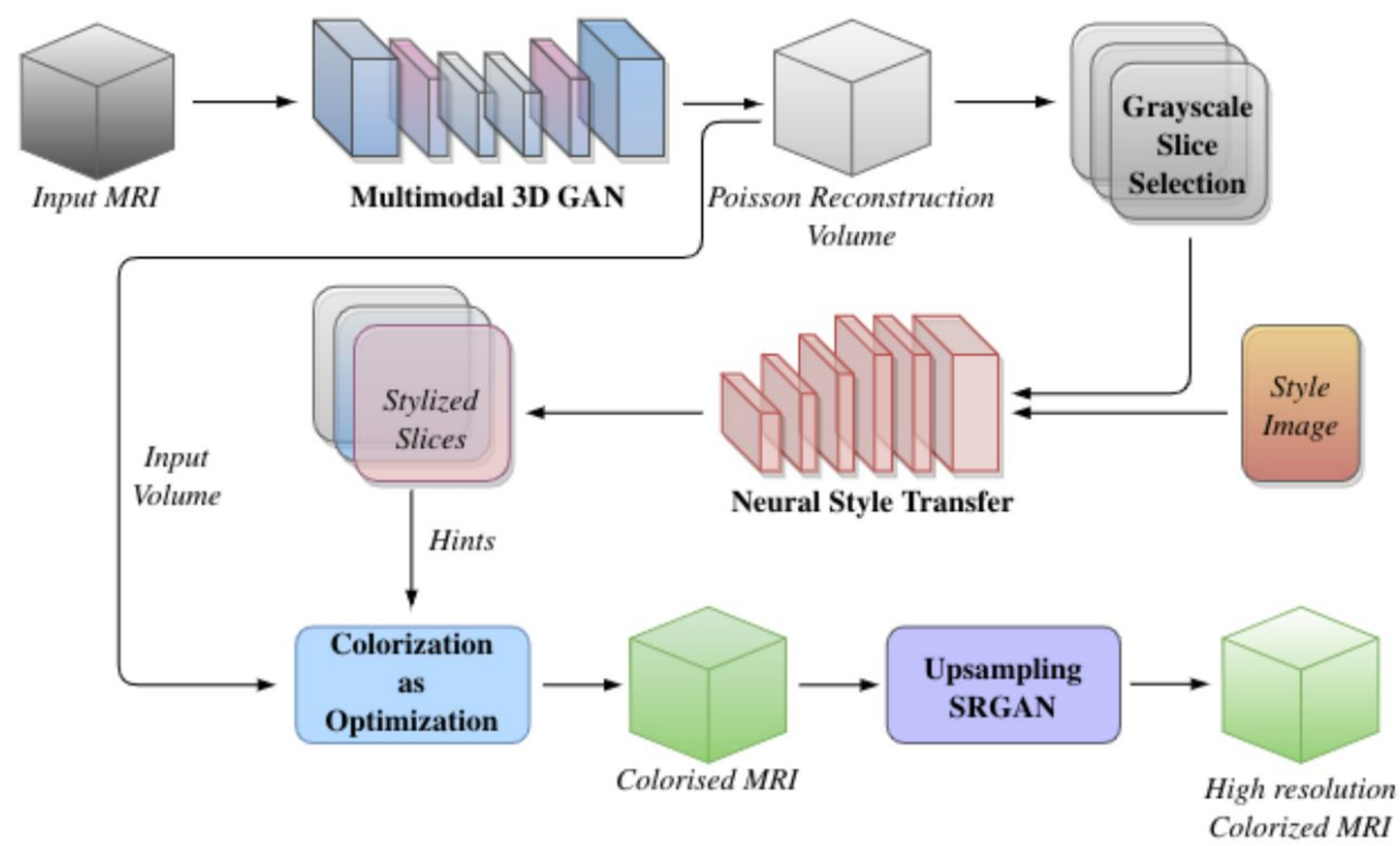


Figure 3.1: Our colorization framework is based on the modality conversion of input MRI. The first stage uses 3D GANs to remap the MRI intensities, allowing for a more flexible colorization. Some of the selected slices from the produced output are stylized. These stylized slices serve as hints in the final stage, where volumetric colorization takes place.

3.1 Related Work

We address the problem of multi-modal colorization where the source and the target images belong to different modalities, which are 3D MRI scans and photographic volumes in our case. There are many approaches for the colorization

of grayscale images, and they mostly differ in the additional input required along with a grayscale image for generating results.

Classical approaches to colorization

Several classical methods look into solving color transfer by treating it as a texture transfer problem. Welsh et al. [236] propose a patch matching algorithm that requires a source image from which color information has to be transferred to the target image. Their approach performs a jittered sampling on the source image to get color patches, finds the closest matching patch from the sampled data, and transfers colors onto the target image. Levin et al. [121] view colorization as an optimization problem where, given some hints in the form of scribbles, colorization is performed with the constraint that similar intensities should have similar colors. Nie et al. [164] develop on the above idea and use quad-tree image decomposition for cost function minimization.

Automatic Colorization

With the advent of Convolutional Neural Networks (CNNs), visual tasks have seen rapid strides. Iizuka et al. [84] leverage the estimation capability of CNNs and perform colorization using both local and global information present in an image. Their approach relies on the classification of an image to generate global priors in the form of class labels that are added to the colorization network for better results. Zhang et al. [269] propose an automatic colorization algorithm where colorization is treated as a multinomial classification problem. The authors employ a CNN to perform colorization in the CIELab color space by predicting only the chroma channel. Zhang et al. [271] propose an interactive colorization algorithm that utilizes user-generated hints and demonstrates the generation of different results for the same grayscale image. Yang et al. [256] propose GANs for colorizing 3D objects. They perform voxel colorization using a GAN consisting of a 3-component architecture. The first part learns a compressed representation, and the second component decompresses and adds information from the latent z vector to generate colors. The last component uses a mask for refinement and generates the final colorized volume.

Exemplar Colorization

Exemplar colorization involves the transfer of colors from a source image to a target grayscale image. These methods rely on CNNs and make use of the semantic context in the source and target images to produce meaningful and coherent colored images. In one of the recent works, He et al. [68] present a deep exemplar-based colorization in which they use a similarity sub-net and

a colorization sub-net to perform exemplar colorization. The similarity subnet computes the bidirectional similarity maps and aligned chrominance channels for the input and reference images, which are then passed to the colorization network. Li et al. [122] present a cross-scale texture matching method using a multi-label graph-cut algorithm. Xiao et al. [248] propose a novel pyramidal network to perform exemplar colorization as a multinomial classification problem. They propose a hierarchical encoder-decoder filter to pass color distributions from lower to higher levels. Xu et al. [252] propose a stylization-based approach for fast deep exemplar colorization. They achieve this with the help of a transfer sub-net to extract coarse chrominance information referred to as *ab* map for the target gray image using WCT [127]. For faster feature matching, they use Adaptive Instance Normalization proposed by Huang et al. [79]. They use this generated *ab* map for subsequent colorization using a colorization net which is optimized using Huber Loss [83].

Style transfer as a generalization

Style transfer forms an integral part of our proposed method to generate hints for colorizing a volume. Prior to the appearance of neural style transfer, a number of approaches to perform artistic stylization and non-photorealistic rendering were explored. Gatys et al. [55] perform style transfer by minimizing reconstruction losses for feature and style with a pre-trained VGG network [207]. Their seminal work demonstrated that CNNs are capable of extracting content information from a photograph and style information from an artwork image to produce a stylized image. Li et al. [127] perform image reconstruction along with whitening and coloring transforms (WCT). The authors use VGG convolutions as an encoder and train a decoder symmetrical to VGG to produce photorealistic stylizations. Jamriška et al. [87] propose a non deep learning based approach for stylizing videos. For an input video, they require some hand-crafted frames as stylization hints. A patch-based optimization uses a mask guide for an object of interest, a color guide for capturing color information, and a positional guide to capture structural information.

Modality conversion

For a more realistic colorization, we perform modality conversion on the input MRI volume. GANs have been explored for converting the modality of medical volumes, commonly known as medical image synthesis. Nie et al. [165] propose a method to convert MRI to CT using GANs. Their method takes an MRI image and generates the corresponding CT image using a fully convolutional network and uses a gradient difference loss function. This work is further extended by Nie et al. in [166] where they combine this approach with Auto-Context [222] and create a context-aware GAN to generate 7T MRI from 3T MRI and CT

from MRI. In a two-step approach, Guibas et al. [62] use a DCGAN to generate synthetic segmentation maps and teach a GAN to generate photorealistic images.

3.2 Background

One of the early works by Gangnet et al. [176] aims to view style transfer as an image editing operation, such as seamless cloning and provides a mathematical machinery for various modifications that can be performed on images. This work forms an integral part of our proposed pipeline.

3.2.1 Poisson Image Editing

Poisson Image Editing work revolves around the main idea that any scalar function on a bounded domain can be identified by the boundary and its Laplacian in the interior, as shown in Figure 3.2.

This allows for the Poisson equation to achieve seamless filling. The Poisson equation can be interpreted as computing a function whose gradient is close to a given vector field in the L_2 norm. The authors define the simplest interpolant

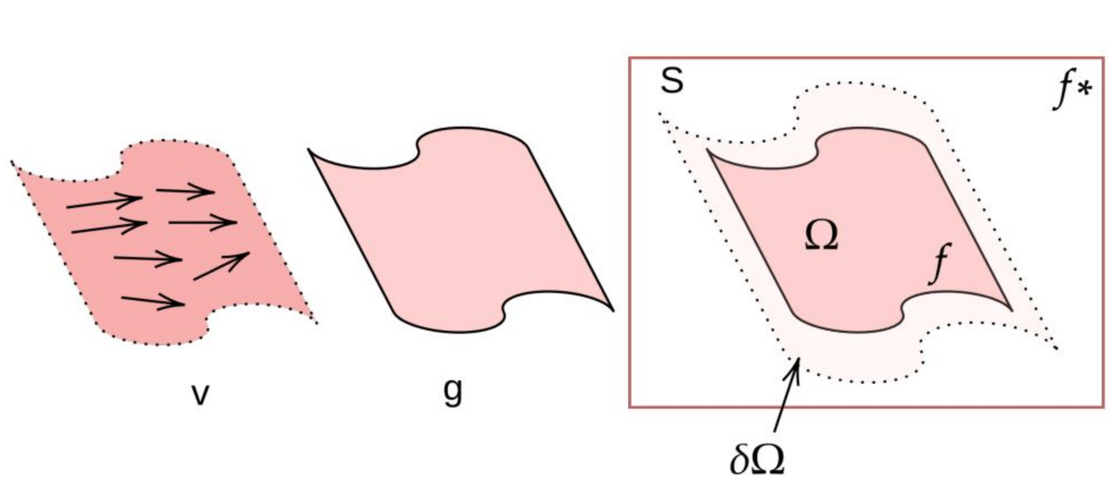


Figure 3.2: Function f which is unknown interpolates in the domain Ω, f^* is a known scalar function referred to as destination function under the guidance of the vector field v.

f^* of an unknown scalar function f, defined over the interior of Ω, by solving the following minimization problem:

$$min_f \int \int_\Omega |\nabla f|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega},$$

where $\nabla = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}]$ it the gradient operator.

This method can be used for editing operations such as texture flattening, inserting objects with holes, feature exchanges, insertion of transparent objects, and monochromatic transfer.

3.3 Proposed Approach

Our approach enables us to bypass designing a transfer function for color volume visualization. The colorization of the MRI volume shows the capability of our approach in handling multi-modal data. In our three-step approach, we use Generative Adversarial Networks (GANs) [57] to convert the modality of the input MRI volume to a corresponding gray cryosection (cryo) volume. This is followed by style transfer on selected slices in the generated gray volume, and a subsequent colorization of the entire volume. Figure 3.1 shows an overview of our colorization approach.

3.3.1 Dataset

We use the Visible Human 2 [3] dataset, which contains unregistered 16-bit MRI and 24-bit color cryo imaging data of the human head. It is necessary to perform the registration between the two modalities in order to get correspondence between the two. We perform a rigid registration of MRI with cryo volume using fiducial markers. We also perform a marker-based segmentation of the MRI volume into foreground and background. The binary background segmentation mask is used in removing the background for visualization purposes.

3.3.2 MRI to grayscale conversion

A direct colorization from MRI to color will not yield good results, given substantial differences in the radiometric characteristics of MRI imaging and photographs. An MRI response of tissue is different from that of visible light, i.e., the tissues in an MRI will look different from how they will appear in a photograph. Therefore, the first step of colorization is to convert a given MRI to a corresponding grayscale photographic volume (hereafter referred to as a grayscale volume). In essence, this step performs modality conversion and predicts how the grayscale intensity of the same tissue will appear under visible light.

MRI aligned grayscale volume for training

In a real scenario, a cryo volume for an MRI will not be available; therefore, to generalize our method, we train a neural network that learns to convert an MRI to a grayscale volume. In order to train such a network, both an MRI and a corresponding grayscale volume are needed. Unfortunately, a grayscale cryo volume is not suitable for this purpose, making the problem non-trivial due to the following reasons:

- Even after careful registration, the features in cryo and MRI do not align perfectly, and there are errors of a few voxels at some places, and

- The MRI volume has a lower level of detail in comparison with the cryo volume, which manifests as ambiguity in the tissue boundaries.

Poisson volume synthesis To create a grayscale volume that matches the features and scale of the input MRI volume, we perform a Poisson reconstruction [176]. Here, we reconstruct a volume I_{gray} from gradients of the MRI volume I_{mri} treating intensities from grayscale cryo volume I_{cryo} as boundary condition by solving the following Poisson equation:

$$\Delta I_{gray} = \boldsymbol{\nabla} \cdot \boldsymbol{\nabla} I_{mri}, \text{s.t. } I_{gray}|_{\partial\Omega} = I_{cryo}|_{\partial\Omega}.$$

This helps us obtain the radiometry of cryo volume while maintaining the geometry of the structures in MRI, thus creating a new modality of photo-realistic MRI (see Figure 3.3(a)). However, in a real-life scenario, such correspondence might not be available; thus, we require a method that can learn this conversion of modality. We achieve this with the help of a 3D GAN. This modality plays a key role in contributing to the visual perception and greater stability of the GANs by providing a tighter structural correspondence to the MRI, thus reinforcing the preservation of structures in the MRI.

Segmentation mask generation For visualization, background removal is usually required. Therefore, we remove the background of the Poisson volume using a segmentation mask. The segmentation is performed by the graph cut algorithm given by Jirik et al. [90], which uses the region and boundary properties of the image to perform segmentation. The segmentation is performed using manual hints for demarcating internal and external sections of the volume.

Training data We generate about 3000 sub-volumes of size $32 \times 32 \times 32$ for training the network. These volumes are taken from the MRI and the Poisson grayscale volumes by performing random rotations and translations to augment the training dataset. The final results and illustrations are generated by passing the complete volume of size $128 \times 128 \times 88$ from the trained network. We further compute the metrics from the generated Poisson volume w.r.t. the MRI volume as shown in Table 3.1, which serves as the upper bound for our GAN outputs. Since the Poisson volume contains the radiometry corresponding to the cryo volume and gradients from the MRI, it serves as the ideal candidate for realistic ground truth.

GAN network for modality conversion

We use the sub-volumes generated from the MRI and Poisson grayscale volume to train a GAN intended to perform modality conversion of any MRI volume.

The GAN learns to generate a grayscale volume corresponding to an input MRI. The generator is based on a 3D-FCN (Fully Convolutional Network) without any max-pooling layers. This plays an important role in the selection of architecture for our method since the MRI and grayscale volumes have a strict geometric correspondence. Avoiding the use of max-pooling layers helps the network essentially learn the remapping of the intensity values without distorting the structures. The generator architecture consists of 8 3D convolutional layers; each subsequent convolution is followed by an activation function. In our case, it is Leaky ReLU with a negative slope of 0.01, except for the last layer, where sigmoid activation is used. The generator architecture can be summarized in the table below.

Layer	**Filter size**	**Stride**	**kernel size**	**activation**
Conv3D	1×64	1	3	LeakyReLU
Conv3D	64×128	1	3	LeakyReLU
Conv3D	128×256	1	3	LeakyReLU
Conv3D	256×512	1	3	LeakyReLU
Conv3D	512×256	1	3	LeakyReLU
Conv3D	256×128	1	3	LeakyReLU
Conv3D	128×64	1	3	LeakyReLU
Conv3D	64×1	1	3	Sigmoid

In order to ensure better structures, we use Structural Similarity (SSIM) loss [232] and an L_1 loss term. The SSIM loss enables the network to learn corresponding cryo structures more robustly, while the L_1 loss helps in further improving the intensities of the generated voxels. The SSIM loss $\mathscr{L}_s$ between two volumes I_1 and I_2 is given by,

$$\mathscr{L}_s(I_1, I_2) = 1 - \frac{(2\mu_1\mu_2 + c_1)(2\sigma_{1,2} + c_2)}{(\mu_1^2 + \mu_2^2 + c_1)(\sigma_1^2 + \sigma_2^2 + c_2)},$$

where μ_1 and σ_1^2 are the mean and variance of I_1, μ_2 and σ_2^2 are the mean and variance of I_2 and $\sigma_{1,2}$ is the covariance of I_1 and I_2. c_1 and c_2 are constants to prevent overflow in the case of small denominators based on the dynamic range of voxel values. The complete generator loss can be formulated as,

$$\begin{aligned}\mathscr{L}_G = \mathscr{L}_s(G(I_{mri}), I_{gray}) + ||G(I_{mri}) - I_{gray}||_1 \\ + \log(1 - D(G(I_{mri}))),\end{aligned}$$

where I_{mri} is the input MRI subvolume, I_{gray} is the corresponding ground truth Poisson grayscale subvolume, $\log(1 - D(G(I_{mri})))$ [57] is the generator loss as described in equation (3.3.2) for generator G and discriminator D for input volume. The discriminator loss can be formulated as, $\mathscr{L}_D = \log(D(G(I_{mri})))$. Our discriminator architecture is as follows:

Layer	**Filter size**	**Stride**	**kernel size**
Conv3D + Batchnorm	1×256	2	3
Conv3D + Batchnorm	256×256	1	3
Conv3D + Batchnorm	256×128	2	3
Conv3D + Batchnorm	128×64	4	3

The last layer being a linear of size 512×1 with Leaky ReLU as an activation function for all layers except the last where sigmoid is used. The results of MRI to grayscale conversion are shown in Figure 3.3(a). The performance of our network can be seen by its ability to generate a grayscale volume that is as close as possible to the ground truth (GT) Poisson grayscale volume.

3.3.3 Style transfer for generating hints

For generating planar hints, we use the style transfer method proposed by Gatys et al. [55]. The objective is to minimize the style loss and content loss, for which convolution layers of a pre-trained VGG19[207] are used. We use the features from the first five convolutions to characterize the style and content loss. Figure 3.3(b) shows sample colorization.

Since the VGG is not trained on medical data, it is not aware of the tissue structures present in our data. This results in subsequent irregularities in colors and color bleeding. To control this to some extent, we further penalize the construction of color channels by using total variation loss [194]. This is done by converting the resultant image to YUV colorspace and applying total variation loss over the color channels U and V. This additional loss is combined with the content and style losses of the style transfer method.

Upon the application of Gatys' style transfer on the complete volume, slice by slice, it is observed that it leads to incoherent results because the loss function is defined for images, and it ignores the content and style information along the third dimension in the volume. Therefore, as explained next, we utilize a separate colorization step that completely preserves the MRI structures while propagating colors from the hints.

3.3.4 Volume colorization using hints

To perform colorization, we leverage the work by Levin et al. [121], performing colorization using optimization with hints given by the user. We extend this method to 3D volume colorization and use the stylized images as 2D hints for colorization in 3D. While the original method uses hints in 2D, we directly extend the optimization to 3D and observe that it gives suitable results. The colorization thus performed maintained coherence across all axes. To select slices as hints, we choose the slices along any of the axes with a high standard deviation to ensure

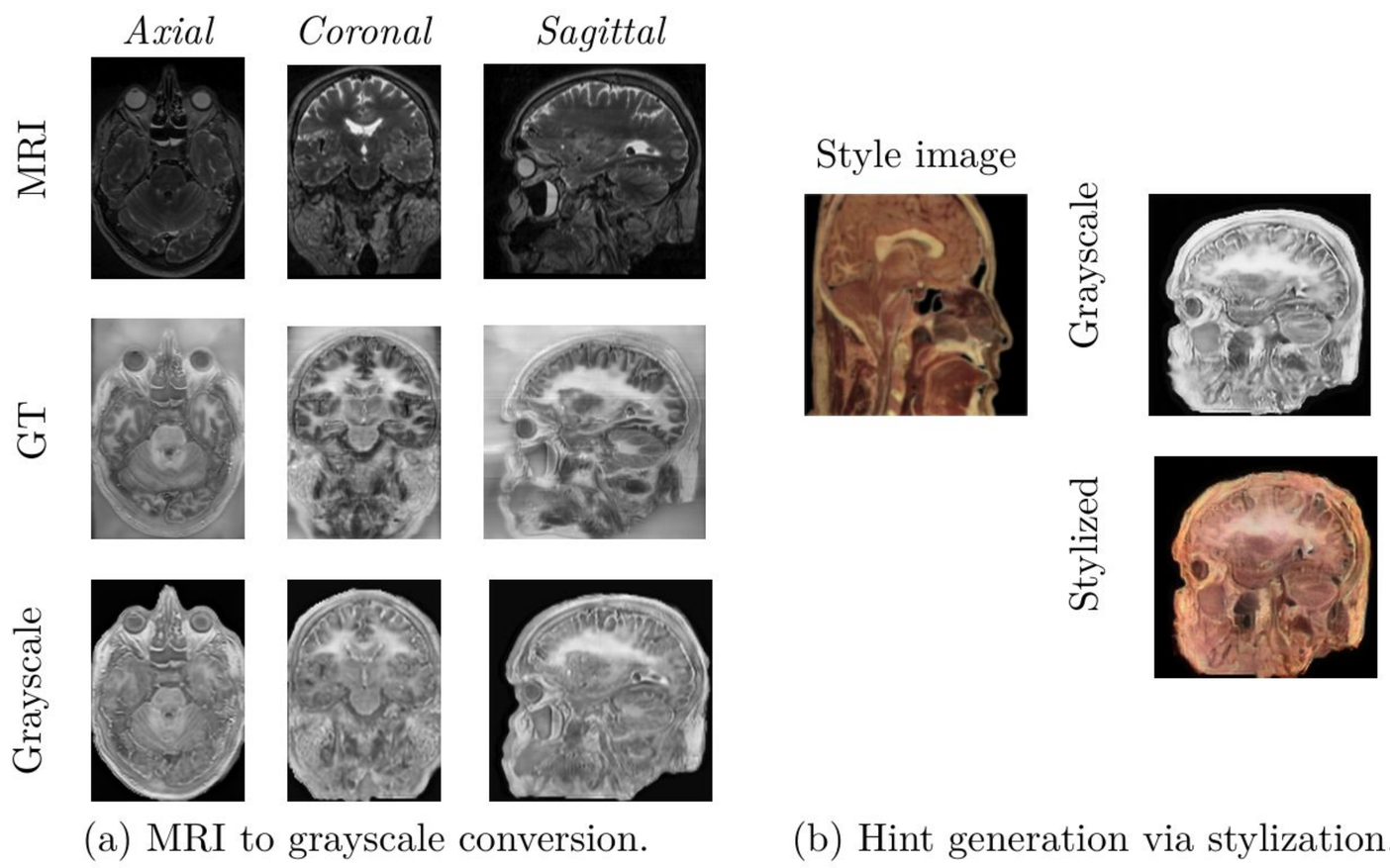
good contrast. The colorization is solved as a constrained optimization problem where two neighboring voxels in a volume, p and q should have the same colors if their intensities are similar. The colorization is performed in the YUV color space, where the Y channel is the same as the one generated using GAN, and the cost function for U and V is defined as

$$J(U) = \sum_{p} \left(U(p) - \sum_{q \in N(p)} w_{pq} U(q) \right)^2,$$

where $N(p)$ is a set of all voxels that lie in a $3 \times 3 \times 3$ window around p. w_{pq} is a weighting function that is large for similar values of $Y(p)$ and $Y(q)$. Here we used $w_{pq} = e^{-(Y(p)-Y(q))^2/2\sigma_p^2}$, where σ_p is the the variance along $N(p)$. Given the quadratic cost function subject to the defined constraints, we minimize $J(U)$ and $J(V)$. Since the optimization problem creates a large system of sparse equations, we use a multi-grid solver [168] to obtain an optimal solution.

3.3.5 Upsampling

Due to hardware restrictions, the maximum cube size that the GAN in our case could process is $128 \times 128 \times 88$. Therefore, we separately train an SRGAN [119] to upscale the slices from the output and obtain the high-resolution volume from the output. The SRGAN is trained to perform 8x upsampling of the image patches of cryo images of size 128×128. The final colorized volume is upsampled in a slice-by-slice manner along the axial view.



(a) MRI to grayscale conversion. (b) Hint generation via stylization.

Figure 3.3: Modality conversion and hint slice generation for colorization.

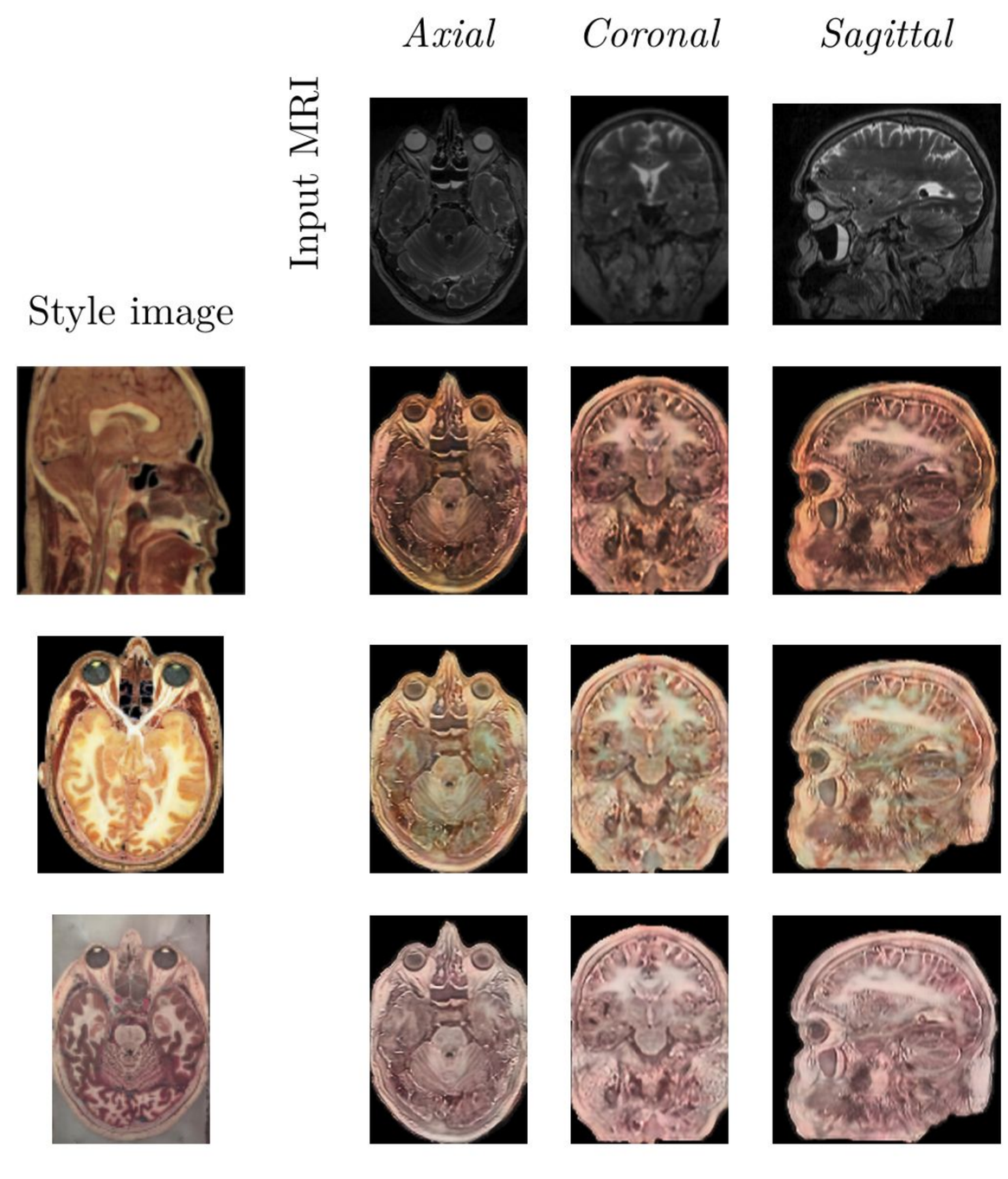


Figure 3.4: Colorization result with a cryo style image.

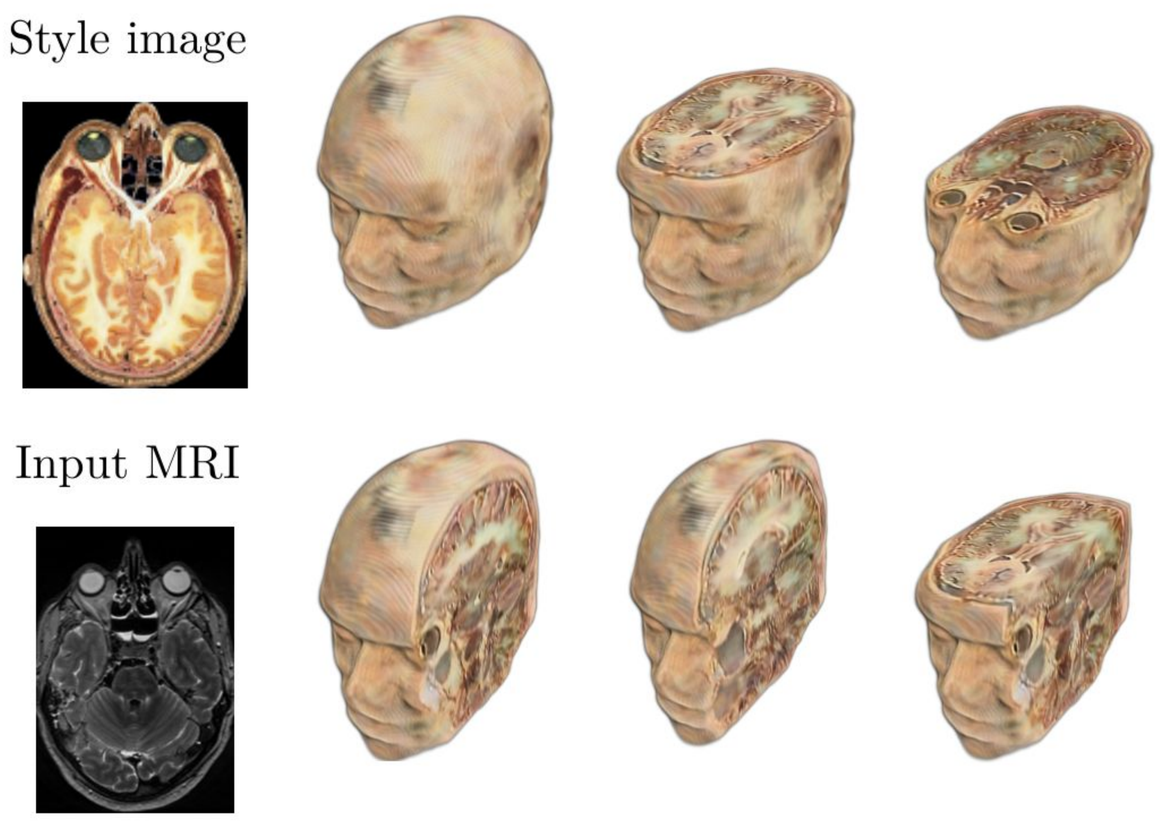


Figure 3.5: 3D rendering of the synthesized volume generated with the style image. RGBA volume visualized using Inviwo renderer. [214].

3.4 Results and comparisons

We train our method using the Visible Human 2 [3] 16-bit MRI and color cryo dataset. Our code is written in Python, where we use PyTorch [173] for training the GAN and PyAMG [168] for solving the optimization problem to generate the colorized volume. Our computational setup consisted of an Intel Xeon 2.4 GHz processor with 128 GB of memory and an Nvidia Quadro P6000 GPU with 24 GB of memory. Our code will be made available for research purposes.

Once our network is trained with $32 \times 32 \times 32$ sized sub-volumes, we pass the complete $128 \times 128 \times 88$ MRI volume for colorization at test-time, which is the maximum size of the volume that our 3D GAN network can handle due to available GPU memory resources. It took 0.165 sec for the modality conversion on the content volume of size $128 \times 128 \times 88$ by the GAN, and upsampling takes approximately 1.5 seconds per image on our hardware. The high-resolution samples in the paper are of size 512×352 obtained by the trained SRGAN, and the overall size of the resulting high-resolution color volume is $512 \times 512 \times 352$. In Figure A.2 and Figure 3.6(e), it can be seen that the tissue intensities in MRI have been correctly mapped to tissue colors of the cryo style image, and structures of the input MRI are maintained very well. In general, MRI intensities do not map to cryo intensities due to the difference in the capturing methods, while cryo images are captured by a camera containing an RGB array sensor, the MRIs are captured by magnetic fields and magnetic field gradients, and our colorization takes care of such differences in the two modalities.

To the best of our knowledge, there are no existing methods that directly aim at solving the aforementioned task for medical volumes. Most existing colorization methods work on 2D photographs and none on medical images. We first compare these methods with state-of-the-art exemplar colorization methods since that is similar to our objective. However, we observe that the modality of the data poses a major challenge for these state-of-the-art methods. In Figure 3.6, we show the results of exemplar colorization and observe that the results of colorization are not visually perceptible. We suspect that this is due to the different intensity distribution of MRI (w.r.t a photograph) and the dependence of these methods on a feature extraction network such as VGG. Further, these methods generate only the chroma channels without any changes to the luminosity channel, which is not suitable for our objective. Hence, we are motivated to compare our approach with four state-of-the-art methods for style transfer after suitable modifications need to be applied for medical imaging data (see Figure 6.6). All of these methods require 2D images as input, and therefore, we process the MRI volume slice-by-slice along the axial plane. In practice, we suggest a full 3D processing as with our algorithm since that leads to a better coherency in the volume compared to a slice-by-slice approach.

In the comparisons, it can be seen that the results obtained by using the

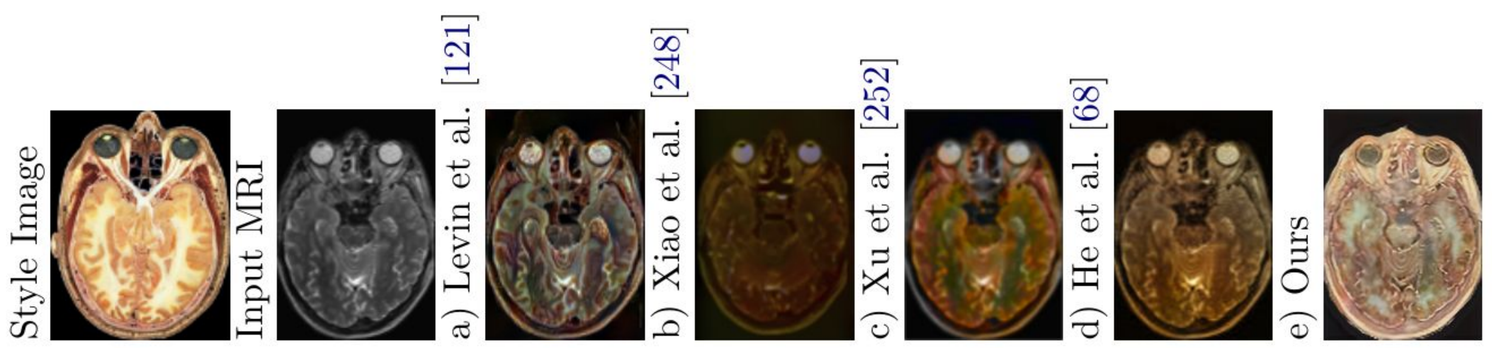


Figure 3.6: Comparison with exemplar colorization methods. For a particular style image, our method results in a much better visual perception. This demonstrates that exemplar colorization cannot be performed on the MRI directly due to the nature of the image modality.

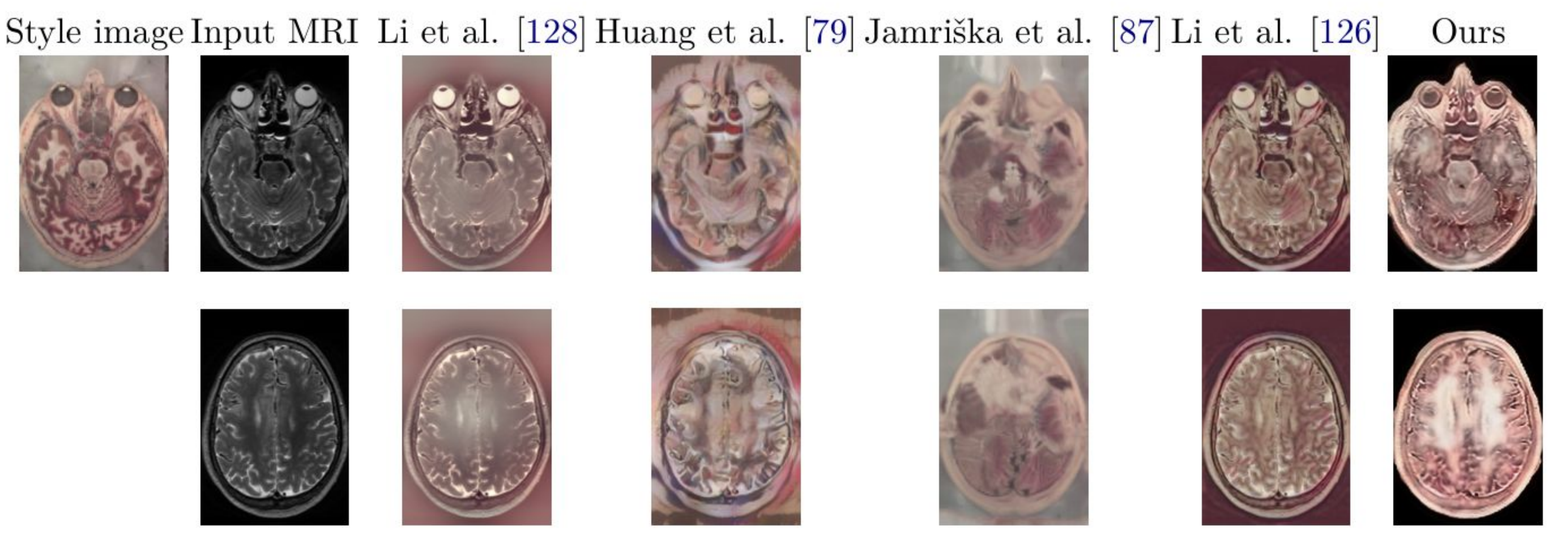


Figure 3.7: Comparison with style transfer methods. Our method results in the most photorealistic colors while preserving the structures of the input data. This balance of structural and radiometric reconstruction is lacking in most other methods, which mostly focus on only one of these aspects.

method proposed by Huang et al. [79], the outputs lack coherency across different slices and also contain undesirable artifacts. Li et al. [128] also do not maintain such a coherency. In addition, they perform photorealistic smoothing on the stylized image that blurs the structures of the input volume. Since Li et al. [126] operates on a video sequence, it preserves coherency across slices but does not fully transfer the style details. Results of Jamriška et al. [87] are best at reproducing the style, but the method requires hints that are completely aligned with the input slices that may not be available for every style image. It also fails to preserve the structures in the input frames. We also perform a quantitative comparison of the generated images with respect to the input MRI and RGB Poisson volume. Table 3.1 shows average SSIM, PSNR, and MSE metrics computed for the entire volume and averaged over RGB color channels. SSIM is calculated between the generated volume and the input MRI volume to ensure that the structures present in the MRI are preserved. PSNR and MSE are calculated between the generated volume and the RGB Poisson volume to measure the color consistency and overall coherence across the volume. Please refer to the appendix for additional results.

Table 3.1: Quantitative comparison with recent style transfer methods. The comparisons were done on a $128 \times 128 \times 88$ cubic volume generated using each method before any upsampling and segmentation mask was applied for the sake of a uniform comparison.

Method	**SSIM** $\uparrow$	**PSNR** $\uparrow$	**MSE** $\downarrow$
Li et al. [128]	0.313	7.57	0.177
Huang et al. [79]	0.313	7.02	0.201
Jamriška et al. [87]	0.314	6.03	0.255
Li et al. [126]	0.314	7.97	0.163
Ours	**0.319**	**9.72**	**0.144**

3.5 Discussion and Challenges

The unique setting of our problem brings forward several key challenges in medical data colorization. The 3D nature of the data places severe restrictions on computational resources. The difference in modality leads to a lack of direct translation between MRI and cryo volume, thus requiring the need for grayscale synthesis. A key limitation of our method is the lack of semantic information for the MRI volume that prevents context-aware stylization of hints. Gatys' style transfer method and other stylization techniques are not suitable for medical data and do not seem to perform well, leading to issues such as patchy colors and color bleeding due to a lack of semantic information. Further, the lack of suitable colorization metrics for medical images makes comparison and evaluation difficult.

3.6 Closing Notes

Automatic colorization of 3D imaging datasets has not been much explored and has been highly dependent on manual methods. The lack of methods for 3D volumetric colorization can be attributed to the absence of a model such as VGG that is trained on datasets such as ImageNet [41], which provides an important backbone for a myriad of visual computing tasks. Our primary contribution is to propose a framework with the aim that it could be used on multi-modal data in 3D for performing colorization, keeping in view the photorealism of the generated outputs. To do so, we demonstrated how a synthetic dataset can be generated to create a different modality that is more suited to visual perception. Further, we extend this method to newer unseen data using the generalization capabilities of a GAN. Overall, the framework allows us to perform colorization from contextual images and extend it to 3D. We also demonstrate how several state-of-the-art methods in 2D fail considerably on medical imaging data due to their multi-modal and 3D nature. Our approach takes into account all these factors, demonstrating robustness in terms of both metrics and visual perception. Though our method has been demonstrated to work on 3D medical volumes, the overall framework is general enough to operate on other 3D multi-modal data.

Our approach opens up several broad aspects of multi-modal colorization that have applications in high-fidelity photorealistic visualization and colorization.

3.7 Acknowledgement

We would like to express our gratitude to NVIDIA for providing the GPU grants to facilitate this research.

Chapter 4

LiFi: Towards Linguistically Informed Frame Interpolation

In this work, we explore a new problem of frame interpolation for speech videos. Such content today forms the major form of online communication. We try to solve this problem by using several deep-learning video generation algorithms to generate the missing frames. We provide examples where computer vision models despite showing high performance on conventional non-linguistic metrics fail to accurately produce faithful interpolation of speech videos. With this motivation, we provide a new set of linguistically-informed metrics specifically targeted to the problem of speech video interpolation. We also release several datasets to test computer vision video generation models of their speech understanding.

Video frame interpolation and extrapolation is the task of synthesizing new video frames conditioned on the context of a given video [138]. Contemporary applications of such interpolation are video playback software for increasing frame rates, video editing software for creating slow-motion effects, and virtual reality software to decrease resource usage. From the early per-pixel phase-based shifting approaches such as by [151], current approaches have shifted to generating video frames using techniques like optical flow or stereo methods [88, 138]. These types of approaches typically involve two steps: motion estimation and pixel synthesis. While motion estimation is needed to understand the movement of different objects of the images across frames, pixel synthesis focuses on the generation of new data.

A related task to video frame interpolation is talking face generation. Here, given an audio waveform, the task is to automatically synthesize a talking face [28, 115]. In recent times, these approaches have become popular for both academic and non-academic purposes [209]. While, on one hand, they are being used to extend speechreading models to low-resource languages [116], on the other hand, many of them are also used to generate media content as well.

The literature on talking face generation, as well as video interpolation and

extrapolation, uses the conventional video quality metrics such as root mean squared (RMS) distance, structural similarity index, *etc.* to measure the quality of generated videos [232]. Although this evaluates the quality of generated pixels well, none of the research works show how it translates to linguistically plausible video frames. This is important since speech, being a linguistic problem, cannot be solely addressed by non-linguistic metrics like RMS distance. The goal of this paper is to investigate how the different video interpolation and extrapolation algorithms are able to capture linguistic differences between the generated videos.

Speech as a natural signal is composed of three parts [145]: visual modality, audio modality, and the context in which it was spoken (crudely, the role played by language). Correspondingly, there are three tasks for modeling speech: speech-reading (popularly known as lipreading) [35, 116], speech recognition (or ASR) [263], and language modeling [153]. The part of speech that is closest to the speech video generation task is the visual modality of speech, and visemes are the fundamental units of this part of speech. Calculating metrics such as mean squared error (MSE) over the entire video does not directly yield any information on these aspects of speech, which raises questions about the faithfulness of the thus attained reconstruction. Therefore, the focus of this work is to investigate the video generation model's understanding of the visual speech modality. To this end, we propose six tasks for looking at different aspects of the visual speech modality. Hence, with this work, we try to make the following contributions:

1. We explore different video interpolation and extrapolation networks for usage on speechreading videos and propose a new auxiliary method of ROI loss using a visemic ROI unit for attaining faithful reconstructions.

2. For the first time in literature, we test the video frame generation algorithms on different aspects of speech such as visemic completion, generating prefixes and suffixes from context, word-level understanding, *etc.* These facets of language are critical to a model's language understanding. We show that most of these networks are not able to capture the language aspect of a speech video.

3. We release six new challenge datasets corresponding to different language aspects. These have been verified automatically and manually and are meant to facilitate reproduction, follow-up testing, and interpretation.

4.1 Related Work

Our problem is closely related to the task of video interpolation and can be viewed as a specialized variant focused primarily on speech videos. Several prior

works have investigated this domain from various perspectives. Early methods leveraged simple interpolation-based methods, which were superseded by flow-based methods. More recently, deep learning-based techniques have significantly advanced the state of the art by leveraging spatiotemporal patterns using flow-based features more effectively.

4.1.1 Video Interpolation

Video interpolation has tremendous applications in entertainment, surveillance, and general world understanding due to the incorporation of temporal information.

Classical Methods

Early works in the field of video interpolation were built upon extending image interpolation methods and image warping [20, 239]. One of the early works by Zitnick et al. [277] focuses on video interpolation from a perspective of viewpoint control using a small number of video cameras, using novel image-based modeling and rendering algorithms. They first leverage a novel color segmentation-based stereo algorithm to generate correspondences, followed by a novel two-layer compressed representation that handles matting. Methods such as the one proposed by Yu et al. [231] focus on motion-based video targeting. They perform optimized crop-and-warp that focuses on maintaining the persistence of video content and present a novel cropping algorithm for the same. Other directions include phase-based and path-based methods [143, 152]. Phase-based approaches estimate motion by analyzing shifts in frequency components, whereas path-based methods model motion by explicitly estimating pixel or feature trajectories across frames.

Deep learning based methods

Deep learning based methods such as DeepFlow [235] and FlowNet [45] paved the way for modern video interpolation by leveraging CNNs for optical flow estimation. PhaseNet [152] incorporates the aforementioned phase-based methods with deep learning to perform video interpolation. The state-of-the-art Super Slow-MO [89] method leverages deep learning based methods for generating intermediate frames given two consecutive frames. They train a bi-directional optical flow generator network and a warping network to generate the intermediate frames and train across 300k individual frames extracted from 1132 240fps videos. Video Frame Interpolation Via Residue Refinement [125] proposes a novel network structure for residue refinement and adaptive weight generation for synthesizing intermediate frames. The proposed method generates intermediate frames by first estimating bi-directional optical flow and further refining it

to warp the input frames forward and backward. These frames are then linearly combined and the weight map is learnt by the U-Net.

4.1.2 Facial Video Generation

Lip generation is an important generation problem due to the specific challenges it faces and the wide variety of applications. Lip generation has found tremendous applications in dubbed movies, translation, animation, news media, video communication, and several other forms of entertainment. Most of the works take into account the conditioning on the basis of audio, images, and other modalities. The seminal work "You Said that?" [34] focuses on generating video of a talking face. The method takes in an input image and an audio speech segment. They leverage a CNN model that utilizes the joint embeddings from the two modalities, followed by a deblurring module inspired by VDSR [107]. KR et al. [113] develop a pipeline for Face-to-Face translation in a multi-lingual setting, whereby the input facial video is converted to the target language with the help of LipGAN and text-to-speech. Prajwal et al. [180] propose using a two-stream network, one for the audio signal provided in the form of a melspectrogram and the video frame segment for the second stream. The method uses L1 reconstruction loss in a GAN framework with a visual quality discriminator and a pre-trained lip-sync expert to obtain the sync loss. However, most of the aforementioned methods depend on additional priors, which either incur higher computational costs or may not always be accessible. As a result, in more constrained scenarios, applying these methods becomes challenging. To overcome this, we introduce a novel approach that eliminates the need for audio or external priors and instead relies solely on visual input to generate reliable videos.

4.2 Evaluation Methods

Most of the previous research on speech video reconstruction and interpolation focuses on conventional metrics like MSE, SSIM, and PSNR, which, although ensuring good quality reconstructions [88, 240], can be misleading in terms of the nuances of the underlying dataset. Videos of the same person saying two different things can have fairly high values of the mentioned metrics thus indicating faithful reconstruction, which makes the evaluation difficult (Figures 4.1, 4.2)[1]. We propose the following evaluation methods and corresponding datasets [2] (Table 4.1) for the speech videos to take into account the underlying language and speech information.

[1] We provide two examples in the Figures 4.1 and 4.2 and a few more in the Figure 4.3.

[2] We make the datasets available at `http://bit.ly/lifi-dataset`. Training scripts for the models in Section 4.3.2 are available at `http://bit.ly/lifi-code`

Corruption Types	**# Sp**	**# Ut**	**# WI**	**Vo**
RandomFrame and Extreme Sparsity	4,004	31,982	356,940	17,545
Visemic	2,883	6,152	338,207	16,663
InterWord	3,008	10,421	141,850	12,621
IntraWord	2,756	10,360	141,296	11,824
Prefix and Suffix	4,004	31,982	356,940	17,545
POS Tagged Corruptions	4,004	31,982	356,940	17,545

Table 4.1: Statistics for the challenge datasets made for each task. Legend: # Sp: Number of speakers, # Ut: Number of utterances, # WI: Number of word instances, Vo: Vocabulary.

Metrics	**Train**	**Eval**	**Test**
# Speakers	5091	4005	413
# Utterances	118,516	31,982	1321
# Word instances	3,894,800	807,375	9890
Vocab	51,211	17,546	2002
Avg. Video Length	XXX	XXX	XXX

Table 4.2: Overview of LRS-3 dataset used for experimentation.

1. RandomFrame: In this type of frame corruption, the frames at random timestamps are replaced with white noise. In a real scenario, the missing frames are dropped however, the indices of the dropped frames are generally available [191]. To indicate the missing frames, we insert white noise images in their place. We do this type of corruption in two settings: 40% and 75% corruption rate. We call the 75% case one with the *extreme sparsity*. A model that can work in such conditions can deal with extremely sparse speech videos and high dropout rates. Random corruption tests whether the model effectively interpolates between different poses, facial expressions, and probable word completions.

2. Visemic Corruption: A viseme is a specific facial image corresponding to a particular sound [145]. In this type of corruption, we choose visemically equivalent sub-words in two or more words and corrupt the first viseme within the subword. For example, consider the words 'million' and 'billion', both are visemically equivalent. We corrupt the first common viseme - 'p' and show the results for such corruption in Table 4.6. The dataset also includes the cases where the visemic equivalence does not occur at the start of the word. For example, the first common viseme which is corrupted is '@', in 'Probably' and 'Possibly', when the phonetic phrase "ebli" is spoken in both. The timestamps of specific phonemes are obtained through the Montreal Forced Aligner [148]. After that, the phonemes were converted to visemes. Through this type of corruption, we test the model's knowledge of specific visemes given the context. We list the SSIM metrics after reconstructing each viseme as detailed in Table 4.7.

3. InterWord and IntraWord Corruptions: In this type of corruption, we corrupt 80% of approximately 20,000 unigram and bigram combinations. Corruption was done such that for a particular gram, 10% each of the prefixes and suffixes of the overall gram remains in the original state after the corruption. For example, consider the bigram 'United States'; the corruption of this bigram results in 'Un*********es'. A model that works well on such corruptions understands the context of the words well and is robust to corruptions during the transition from one word to another.

4. Prefix Completion: For this corruption, we remove the start frames of a video. We do this for two levels of corruption- a low level, 40%, and a high level of corruption (75%).

5. Suffix Completion: Similar to the PREFIX COMPLETION test, we corrupt the ending frames from the sequence of input, which is also performed for 40% and 75% corruption. These two tests make sure that the model understands both the word start and word end and is not biased towards either. While predicting word start requires context from the previous word and the remaining part of the word, predicting the end of the word requires context from the prefix and the next word.

6. POS Tagged Corruptions: In this type of corruption, we identify nouns, pronouns, verbs, adjectives, and determinants using Spacy's English language POS tagger[3]. The corresponding timestamps for all such tagged words were obtained through MFA. Following this, we decided to corrupt these words for further study. For example, in a video from LRS3, the speaker says *'The short answer to the question is that no, it's not the same thing'*. The POS tagger identifies the three words *'answer'*, *'question'*, and *'thing'* as nouns. Thus, all of the frames belonging to these three words were corrupted while inducing corruptions to nouns.

By introducing such corruptions, we test how well the models understand the semantic features of the language spoken. For example, for a particular model, proper nouns may be more difficult to predict than pronouns. This test helps to bring such nuances to the forefront.

Viseme	**FCN3D**	**LSTM**	**FCN3D (ROI)**	**Super SloMo**
Viseme Corruption				
@	0.8974	0.7489	0.9086	0.9660
a	0.8838	0.7410	0.9152	0.9566
E	0.8902	0.7425	0.9231	0.9656
f	0.8922	0.7465	0.9197	0.9617
i	0.8923	0.7387	0.9239	0.9761
k	0.8910	0.7434	0.9242	0.9684
O	0.9059	0.7513	0.9213	0.9558
p	0.8921	0.7415	0.9256	0.9715
r	0.8998	0.7446	0.9255	0.9731
s	0.8920	0.7497	0.9238	0.9662
S	0.8829	0.7398	0.9237	0.9556
t	0.8945	0.7443	0.9268	0.9702
T	0.9087	0.7436	0.9185	0.9771
u	0.8918	0.7503	0.8936	0.9395

Table 4.3: SSIM obtained upon the reconstruction of corrupted videos in the Visemic Corruption dataset, averaged across the videos being corrupted for each viseme.

Model	**SSIM**	**PSNR**	**SSIM**	**PSNR**
Corruption	40%		75%	
Random Corruption				
FCN3D	0.9271	23.7161	0.8554	21.8354
BDLSTM	0.8119	21.6103	0.8199	22.3904
FCN3D + ROI	**0.9326**	**28.3123**	**0.8654**	**24.7521**
Super SloMo	**0.9849**	**30.3459**	**0.9603**	**28.2660**
Prefix Corruption				
FCN3D	0.7680	18.4513	0.5230	14.7208
BDLSTM	**0.8119**	**21.6103**	**0.6288**	**17.6097**
FCN3D + ROI	0.7721	19.6718	0.5208	15.3935
Super SloMo	**0.8411**	**23.1864**	**0.7387**	**20.8218**
Suffix Corruption				
FCN3D	0.7816	18.7518	0.5301	14.8344
BDLSTM	**0.7682**	**20.4863**	**0.6281**	**17.8500**
FCN3D + ROI	0.7774	19.7790	0.5161	15.3414
Super SloMo	**0.8140**	**22.4334**	**0.6952**	**19.8319**

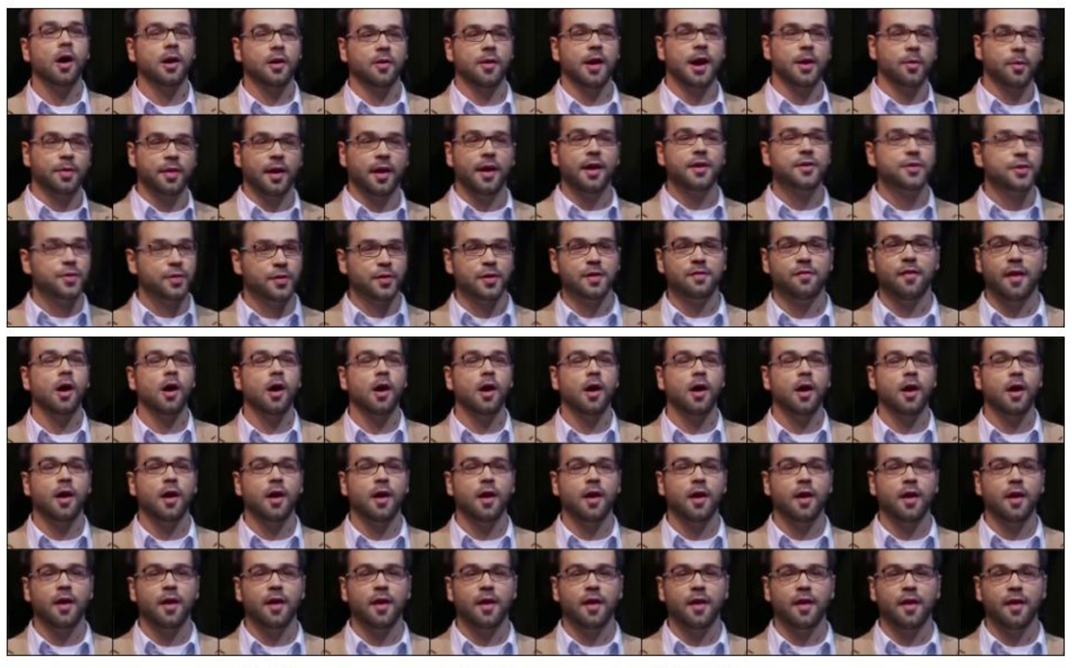
Table 4.4: Metrics obtained on the test set for the different experiments conducted. For each experiment, two different levels of corruption were tested, a low 40% and a high 75% corruption. It also highlights how the behaviors of models vary across different experiments, thus further reaffirming the need for a more comprehensive test suite. We share more detailed metrics in Table 4.7.

MSE: 0.0093898 **PSNR**: 20.273426 **SSIM**: 0.6174879

Figure 4.1: To show the ineffectiveness of the existing metrics such as PSNR, SSIM, and MSE, we create a new synthetic video clip of size 32 frames by replicating the first frame 16 times and the last frame 16 times in the second image. We notice that even though contextually the information is incorrect, since only two frames are used to construct the complete video, the metrics show high values, indicating that the synthetic video is a faithful reconstruction of the original video. This result reinforces the fact that we need better metrics to compensate for the underlying context and comes on par with the metrics obtained by training FCN3D in Table 4.4. However, the proposed method takes into account the visemic reconstruction, thus enabling to get a more faithful reconstruction.

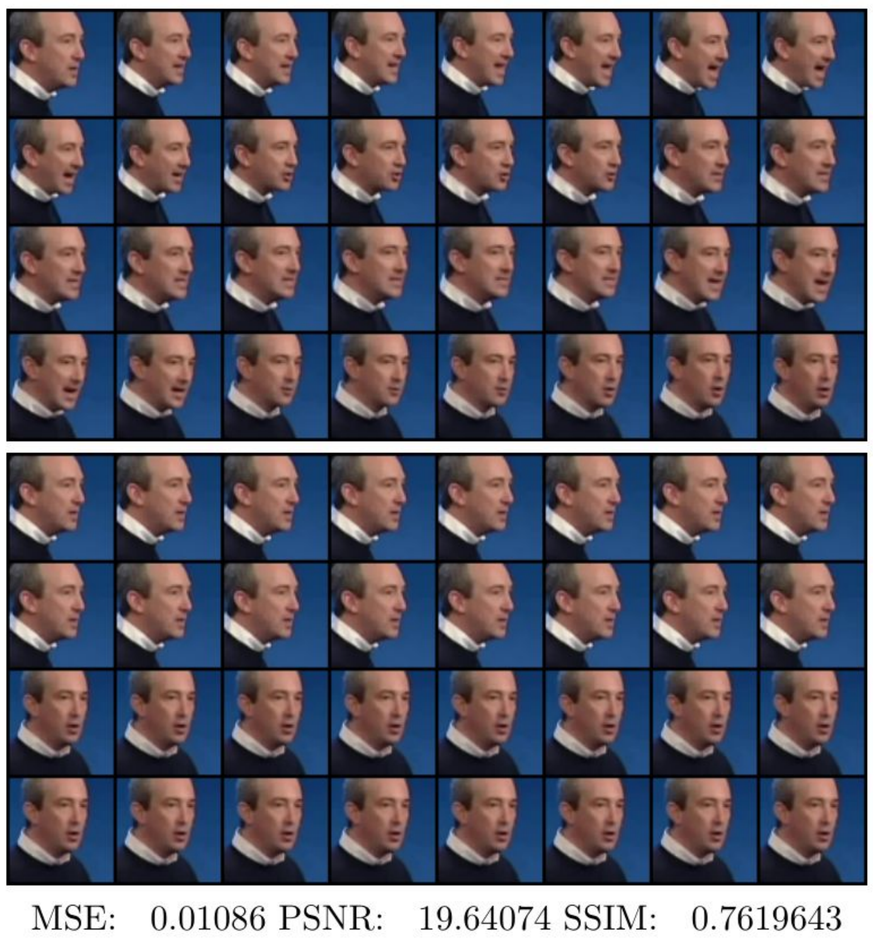
Metrics	**Pretrain**	**Trainval**	**Test**
# Speakers	5091	4005	413
# Utterances	118,516	31,982	1321
# Word instances	3,894,800	807,375	9890
Vocab	51,211	17,546	2002

Table 4.5: Overview of LRS-3 dataset used for experimentation. To train our models, we used the 'Pretrain' section of the dataset and the 'trainval' section as the testing set. We also sample the linguistically aware evaluation datasets (Section 4.2) from the 'trainval' section itself. The 'test' set of LRS3 has very few videos (1321) and was dropped to simplify the distribution.



MSE: 0.0088 **PSNR**: 20.5415 **SSIM**: 0.9521

Figure 4.2: The above images are part of the original and Super-SloMo interpolated videos for the phrase 'I don't exactly walk around with a hundred and thirty-five million dollars in my wallet'. Although the interpolated frames have an excellent quality and achieve very high scores on the metrics, zooming in on them reveals that they are not able to reproduce the lip movements. For instance, in the figure above, the speaker closes his mouth several times, but in the bottom one, the speaker does not close his mouth at all. This interpolated video, despite achieving excellent scores, will look asynchronous to a viewer.



MSE: 0.01086 PSNR: 19.64074 SSIM: 0.7619643

Figure 4.3: Additional example to show the case where we get relatively high SSIM and PSNR values despite the usage of only two frames to create the synthetic clip. This reinforces the fact that we need better metrics to compensate for the underlying context.

4.3 Approach

4.3.1 Dataset

We choose the LRS3 dataset [5] for testing out the speech video frame generation models and making the six challenge sets as described in the last section. We chose LRS3 for this task since it has a high variety of speakers (5091), number of utterances (118k) and a longer average speech length. Other datasets are generally much less diverse both in terms of speakers and number and length of videos [8, 255]. In LRS3, the standard frame rates are 24, 30 and 60 fps. Therefore, we choose a standard frame rate of 32 to train our models and pad it with random noise. We train and evaluate our models using the LRS3 dataset with slight modifications, as explained in Table 4.5. We evaluate the models' generated videos on MSE, SSIM and PSNR metrics [88].

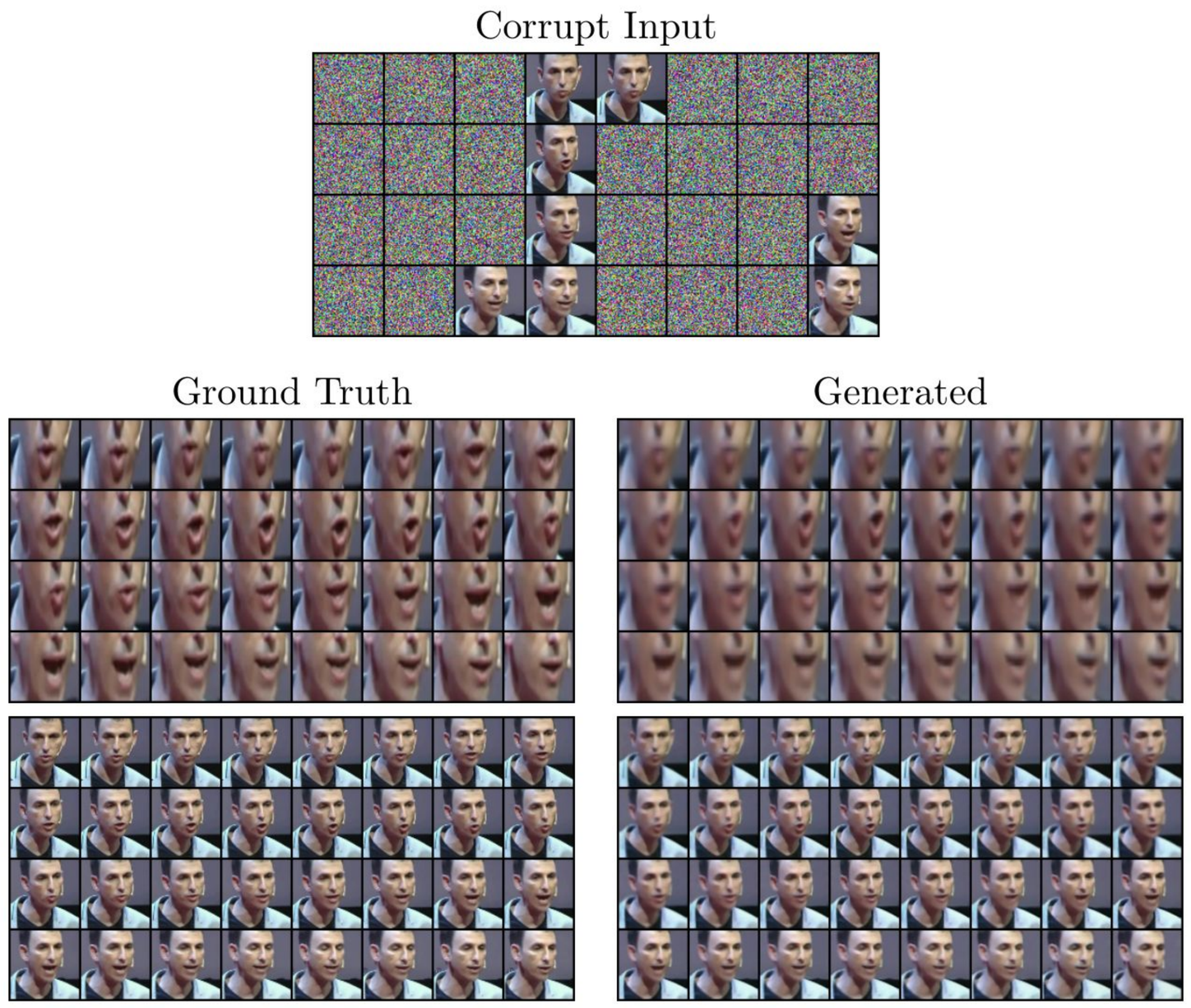


Figure 4.4: Sample Completion using ROI stream, the corrupt input spans two columns at the top, the left column contains the ground truth and the right column contains the generated output ROI and frame reconstructions.

[3] https://spacy.io/api/tagger/

4.3.2 Models

We perform the tests on 4 different models. We perform the baseline tests on standard methods comprising Convolutional Bi-Directional LSTM (BDLSTM) and Fully Convolutional Neural Networks consisting of 3D convolutions (FCN3D). Further, we compare them with the proposed 3D Fully Convolutional Network with ROI unit. FCN3D has been heavily used in 3D image synthesis-based tasks such as [240] while convolutional BDLSTM has been used for sequential image synthesis [247]. For the FCN3D model, we select a 3D Denoising auto-encoder-based approach, which has been known to be robust for video and 3D image synthesis-related tasks [62, 261]. We also compare the results with NVIDIA's Super-Slo-Mo model [88], which is a state-of-the-art method for interpolating frames with suitable modifications for the test suite.

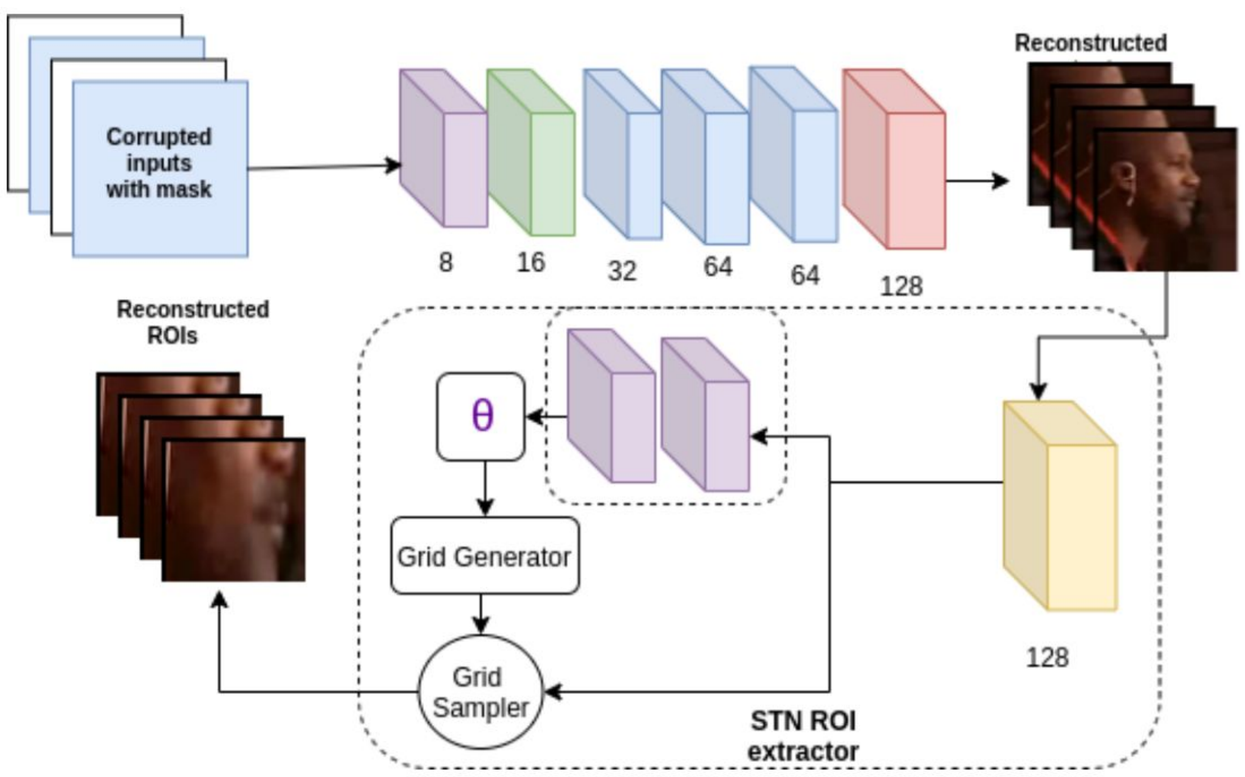


Figure 4.5: The architecture of the proposed ROI loss network. The generated frames are passed through the secondary stream, which learns to extract the mouth ROI. This is aimed at making the network focus more on the visemic information that each frame carries and learn the reconstruction of the ROI.

In this approach, we take a set of noisy frames as input, generate the denoised frames, and train the network as a denoising autoencoder [226]. The network is expected to learn interpolations based on the available frames in the video sequence without any additional information. We test the model on two different percentages of corruption and show the results for sparse reconstruction in Figure 4.4. The network is trained with the corrupted frames with the original frames as the output. We tried for two kinds of losses, L1 and L2; since L2 is well known to produce blurring, we define the reconstruction loss in equation 4.1.

$$\mathscr{L}_{frame} = \frac{1}{N} \sum_{x \in X} ||f(x_{corr}) - y||_1, \tag{4.1}$$

where x_{corr} is the corrupted input image for an image x in the set of images X. The network learns to interpolate the missing frames based on the available set of uncorrupted frames.

Region of Interest (ROI) Unit

We noticed that all the models described before and their losses pay attention to images as a whole, whereas the speech videos have an important component in the form of visemes. Since there are no viseme-centric losses in conventional approaches, they are largely governed by other aspects of the images and not the visemes. To counter this problem, we propose an ROI loss, which is calculated by computing the reconstruction loss between the ROI, *i.e.*, the mouth region of the reconstructed output, and the ground truth. The mouth region is extracted using the facial landmarks obtained by the Blaze package [15].

From the facial landmarks, we extract the mouth region as a proxy for the viseme for a particular frame and then compute the L1 loss between the mouth region extracted for the ground truth and the generated videos. This loss helps us understand the discrepancy that might arise when reconstruction is trained for full-frame instead of being trained on specific aspects, like the viseme for speech video reconstruction.

Further, we introduce a new unit for extracting ROI from the reconstructed frames (as shown in Figure 4.5). During the experiments, we noticed that the ROI extraction performed best from the last three output channels. The ROI extraction unit consists of 2 convolutional layers and a spatial transformer layer proposed by [86]. The key idea is that the convolutional layers learn to focus on the ROI while the STN learns to extract only the mouth region. We show that the FCN3D model shows a much greater boost in terms of metrics for even sequential tasks such as prefix and suffix corruptions, coming close to the performance of bi-directional LSTMs for sequential reconstructions.
The ROI loss can be defined as:

$$\mathscr{L}_{roi} = \frac{1}{N} \sum_{x \in X} ||f_{stn}(x_{corr}) - y_{roi}||_1, \tag{4.2}$$

where y_{roi} are the ground truth ROI features extracted from the frames and f_{roi} is the ROI stream of the architecture.
Thus, the total loss during training is

$$\mathscr{L}_{total} = \mathscr{L}_{frame} + \mathscr{L}_{ROI}. \tag{4.3}$$

However, for computing metrics, we take into account only the reconstructed frames since ROI loss is only an auxiliary for frame reconstruction.

Model	**SSIM**	**PSNR**
Viseme Corruption for Viseme		
FCN3D	0.8385	18.5201
BDLSTM	0.7269	17.2827
FCN3D + ROI	**0.9106**	**26.5401**
Super SloMo	**0.9129**	**25.2605**
Word Level Corruption		
FCN3D	0.8109	18.4554
BDLSTM	0.7411	18.066
FCN3D + ROI	**0.8804**	**25.3279**
Super SloMo	**0.8800**	**24.5971**
Multi Word Corruption		
FCN3D	0.8064	19.6111
BDLSTM	0.7411	18.0665
FCN3D + ROI	**0.8308**	**23.1354**
Super SloMo	**0.8481**	**23.8295**

Table 4.6: Metrics obtained on the newly introduced challenge datasets. It can be seen that even the simpler models trained with ROI unit embedding give results close to the baseline (Super SloMo). More details can be found in Table 4.8.

4.4 Results and Discussion

In this section, we present the experimental results for the models given in Section 4.3.2 on the tests explained in Section 4.2. Results of the test suite are given in Table 4.4. We notice that for random corruption, though fully convolutional models beat bi-directional LSTM for low levels of noise, the bi-directional LSTMs give a much better performance for longer sequential frame reconstructions. We see that the ROI extraction unit helps boost the performance of the FCN model. The ROI stream network helps attain a better PSNR and SSIM comparable with LSTM. This can be attributed to the model's ability to focus more on visemes indirectly by learning a better reconstruction of the ROI for speech videos. The secondary stream is a very shallow network consisting of two convolutions and a spatial transformer, which helps extract the mouth ROI from the already generated output. Thus, ROI loss further helps the network learn more about the visemic structure of the reconstruction and helps attain better scores on the metrics.

We also show a quantitative comparison with Super SloMo [88]. Since Super SloMo only allows for the prediction of the contiguous frame, we had to significantly modify the experimental setup to make any comparisons with it. The architecture did not allow the first and last frames of the batch to be corrupted. Therefore, though the quantitative results are the best for it, Super-Slomo suffers from severe experimental restrictions, unlike the other models presented. In the

Model	**Corruption** (%)	**MSE**	**L1**	**SSIM**	**PSNR**
Random Corruption					
FCN3D	75	0.008719796199	0.05576719156	0.7754931869	20.78947158
BDLSTM	75	0.00585898655	0.03845565588	0.8575272247	23.27513855
FCN3D + ROI	75	0.0042412771	0.03334277668	**0.8654025606**	**24.75212616**
Super SloMo	75	**0.0032095665**	**0.01857670849**	**0.9603758130**	**28.26607934**
FCN3D	40	0.004914738255	0.03917299796	0.8867508521	23.30322587
BDLSTM	40	0.004940670419	0.03570744986	0.8905155776	24.38299802
FCN3D + ROI	40	**0.001789543358**	0.02319063034	**0.9326486659**	**28.31231905**
Super SloMo	40	0.001827087549	**0.01178585826**	**0.9849396967**	**30.34593392**
Prefix Corruption					
FCN3D	75	0.03761557525	0.1411872019	0.4735313937	14.27897795
BDLSTM	75	0.01357646521	0.06795089568	**0.6898999835**	**19.08365961**
FCN3D + ROI	75	0.03076031727	0.1183280068	0.5208106449	15.39350073
Super SloMo	75	**0.01147114887**	**0.0528111567**	**0.7387903618**	**20.82188895**
FCN3D	40	0.0176491743	0.08077385376	0.7199462504	17.5958132
BDLSTM	40	0.007748413963	0.0462029091	**0.8265586414**	**21.73186513**
FCN3D + ROI	40	0.01163159572	0.05775547531	0.7721097014	19.67188354
Super SloMo	40	0.00629138803	0.03642695826	**0.8411455717**	**23.1864927**
Suffix Corruption					
FCN3D	75	0.036539371	0.1388430108	0.48635537	14.39668367
BDLSTM	75	0.01277490123	0.06604159133	**0.6951813442**	**19.21521863**
FCN3D + ROI	75	0.03108713292	0.1189973814	0.5161932313	15.34146382
Super SloMo	75	**0.01606990903**	**0.0645460370**	**0.6952581282**	**19.83196098**
FCN3D	40	0.01651665661	0.07831238874	0.7360948577	17.85544448
BDLSTM	40	0.006905819333	0.04328332246	**0.8464001736**	**22.30525472**
FCN3D + ROI	40	0.01134652429	0.05662384959	0.777493871	19.77901635
Super SloMo	40	**0.01155048322**	**0.04815621729**	**0.8140604513**	**22.4334380**

Table 4.7: The above table shows the metrics obtained on the test set for the different experiments conducted. For each experiment, 2 different levels of corruption have been tested, 40% and a very high corruption of 75%. The above table also highlights how the behaviour of models varies across different experiments, thus further reaffirming the need for a more comprehensive test suite.

Model	**MSE**	**L1**	**SSIM**	**PSNR**
Viseme Corruption				
FCN3D	0.01460618444	0.0568077997	0.83858460187	18.52015423
BDLSTM	0.02084208424	0.0828854019	0.72696408065	17.28277912
FCN3D + ROI	0.00229573487	0.0268876856	**0.91065606555**	**26.54016680**
Super SloMo	0.00377557111	0.0265828257	**0.91293388146**	**25.26056400**
Word Level Corruption				
FCN3D	0.01476183945	0.0596284348	0.81098916828	18.45544786
BDLSTM	0.01841032799	0.0762009256	0.74111890852	18.06654904
FCN3D + ROI	0.00313893227	0.0303552080	0.88049633604	25.32792884
Super SloMo	0.00426348229	0.0304555968	**0.88005496244**	**24.59719328**
Multi Word Corruption				
FCN3D	0.01131343627	0.0555250485	0.80642498433	19.61113891
BDLSTM	0.01841032808	0.0762009258	0.74111890912	18.06654899
FCN3D + ROI	0.00538372894	0.0382826699	0.83086033181	23.13544090
Super SloMo	0.00524661909	0.0348772242	**0.84817349957**	**23.82953434**

Table 4.8: The above table shows similar metrics for the same models on the newly introduced datasets. It can be seen that simpler models trained with ROI embedding give results close to the baseline.

case of continuous corruption, it will require the start and the end frames to reconstruct the intermediate ones; hence, it is difficult to implement it in real-time while taking into account the random nature of the noise/corruption.

If we look at the results on viseme and word-level datasets (Table 4.6), it can be seen that FCN3D with ROI outperforms SuperSlomo in terms of SSIM in word-level corruption and PSNR for Viseme corruption, attaining a PSNR of 26.54 against 25.2605 by Super SloMo. This is due to the smaller duration of corruption in these datasets as compared to those in Table 4.4. This is also likely to be the practical corruption distribution for real-time speech video streaming, where networks like Super SloMo cannot function since they generate frames for only a third time t_3 step between t_1 and t_2 (one of which may not exist). The proposed models are not restricted by this limitation and, for the speech videos, capture a longer temporal context (not just the frames at timestamps t_1 and t_2) besides focusing on the mouth region using the ROI unit. As evident from Table 4.4 for random corruption, FCN3D+ROI gives a high PSNR value of 24.75, beating BDLSTM and coming in close with Super SloMo with a PSNR value of 28.26. At the same time, LSTM outperforms FCN3D+ROI for prefix and suffix corruptions however, with a smaller difference of 2.21. In contrast, Super SloMo outperforms both with a higher margin due to its ability to incorporate optical flow. This highlights a key limitation of such methods for speech video interpolation tasks due to their inability to capture context also reinforcing the need for a better test suite for a more faithful evaluation of speech video interpolation.

4.5 Closing Notes

In this paper, we demonstrate the necessity of a comprehensive linguistically-informed test suite that encompasses all the major aspects of speech in the task of speech video interpolation and reconstruction. We release six challenge datasets for this purpose. We also compare several different contemporary deep learning models for the different tasks proposed. We show the importance of incorporating visemic loss and provide a natural proxy to judge the visemic nature of reconstruction. In the future, we would like to cover more such linguistic aspects of speech. A parallel task would also be to take the audio model into account while reconstructing the corrupted and missing frames in video interpolation.

Chapter 5

SLI-pSp:Injecting Multi-Scale Spatial Layout in pSp

We propose SLI-pSp, a general-purpose Image-to-Image (I2I) translation model that encodes spatial layout information as well as style in the generator, using pSp as the base architecture. Previous methods like pSp have shown promising results by leveraging StyleGAN as a generator in various I2I tasks, but they seem to miss finer or underrepresented details in facial images like earrings and caps, and break down on complex datasets due to their solely global approach. To address these shortcomings, we propose a technique termed Spatial Layout Injection (SLI-pSp) that encodes spatial layout information in the input image in the StyleGAN generator along with the style. We do so without modifying the style vector injection in the generator through pSp's map2style network but rather by combining SLI with noise layers in the StyleGAN generator at multiple spatial scales. Such an approach helps preserve global aspects of image generation as well as enhance spatial layout details in the output. We experiment on several challenging datasets and across several I2I tasks that highlight the effectiveness of our approach over previous methods with respect to finer details in the generated image and overall visual quality.

Generative Adversarial Networks(GANs) [57] have revolutionized the field of generative modeling. A variety of GAN architectures have been proposed in the recent past that generate images with excellent visual fidelity and photorealism. Converting an image in a source domain to a target domain while preserving the core content and adapting the style according to the target domain is termed as Image to Image (I2I) translation [85]. Such I2I problems exist in various forms [169] and many solutions have been proposed in the following themes: semantic image synthesis [136, 171, 215, 217, 276], style transfer [108, 221], image inpainting [174, 211, 275], image super-resolution [264, 272], etc.

Consider the task of semantic image synthesis. A fundamental question is to understand how well we transform a semantic map into a realistic RGB image while preserving input semantics. Many previous approaches try to exploit semantic layout information in the generator by proposing encoder-decoder ar-

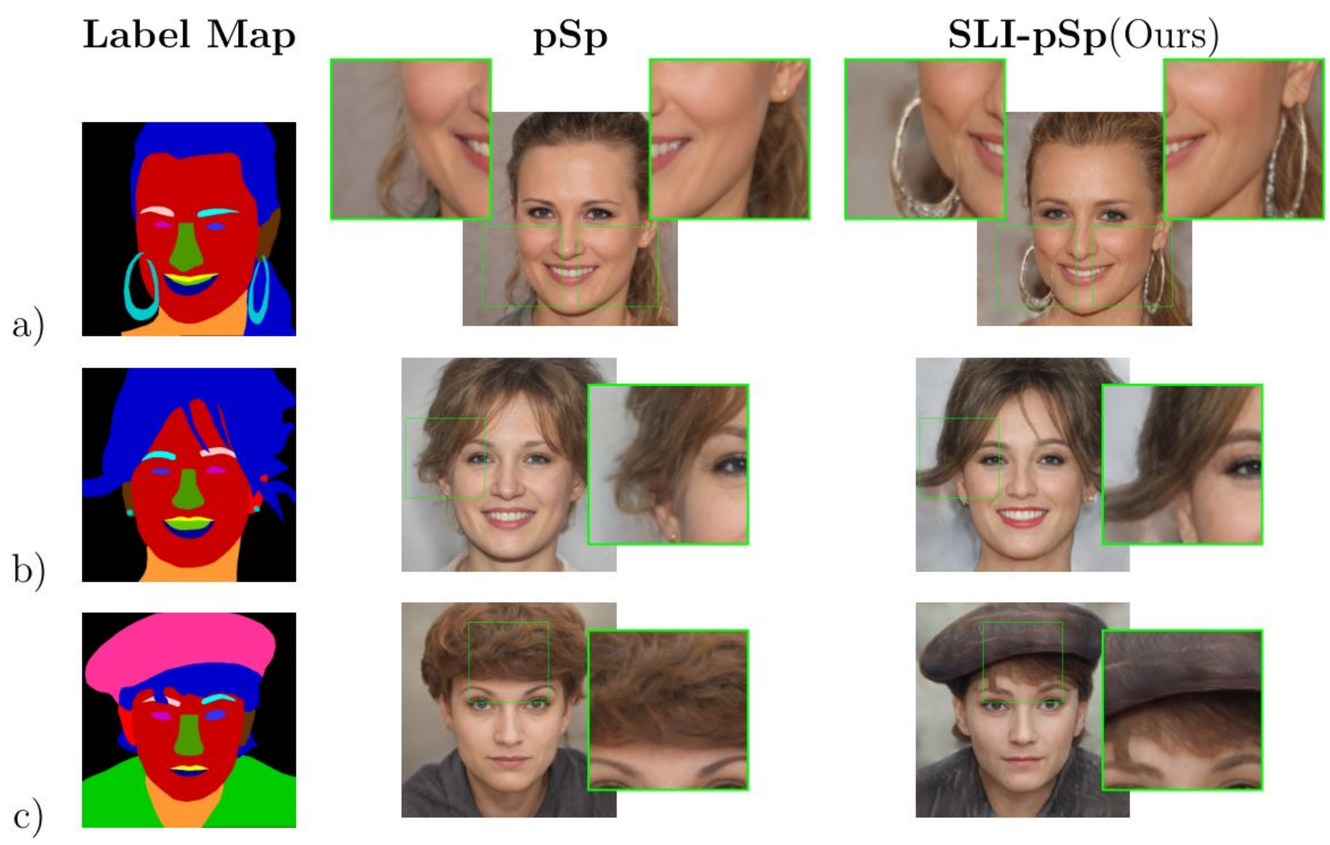


Figure 5.1: Our proposed **SLI-pSp** approach helps fix the lack of spatial context in the image generation process in pSp. By injecting spatial layout information in the decoder, we are able to synthesize finer and under-represented details (such as earrings, hats, and hairstyles) that are missed by pSp's global approach. We retain global properties of image generation even after our spatial layout injection (SLI), thereby making our method work much better across I2I tasks.

chitectures [85, 229]. A common pitfall in such methods seems to be that the network receives the semantic information once (in the form of input); therefore, it is difficult to maintain that information throughout the generation process via the decoder. To alleviate this problem, SPADE [171] proposes spatially-adaptive normalization that helps modulate the activations at various layers of the network via spatially adaptive, learnable affine transformations. However, this approach is limited in terms of representational capacity. We address the issue of representational capacity by having a separate style vector injection as well as a spatial layout injection in the generator.

Another key issue pertains to how well the discriminator architecture utilizes the semantic layouts. Previous methods, such as Pix2Pix [85] and SPADE [171], use a multi-scale PatchGAN discriminator that takes in a label map and an image as inputs and outputs a predicate (real or fake) as its decision. CC-FPSE [136] and OASIS [215] also propose variants of discriminators that give superior performance to previous methods. However, it is well known that GAN-based architectures suffer from training instabilities due to the adversarial (min-max) nature of the learning problem. While many improvements [157, 265] have been proposed, many architectures still require lots of tuning to train properly. We address this issue by adopting and extending the pSp [188] framework that allows the use of a pretrained StyleGAN generator as a proxy for the discriminator.

One of the most popular GAN architectures in use is StyleGAN [98] due to its

high-quality, state-of-the-art image generation capabilities. pSp [188] proposes an I2I framework using StyleGAN as a generator. They introduce a Feature Pyramid-based encoder that embeds input images into an intermediate latent space (separate from StyleGAN's latent space) and use pretraining of the StyleGAN generator in lieu of a discriminator. The key advantages of pSp include less training instability due to no discriminator, better style vectors injected into the StyleGAN generator due to their encoder and intermediate latent space, along with interesting properties of StyleGAN such as style mixing and multimodal synthesis. However, due to its global approach, it is not able to represent finer details in its generated images (Fig. 5.1). While the style gets propagated by its *map2style* network, the outputs lack the spatial layout consistency that is provided in the input and is often crucial in I2I tasks. As a result, pSp struggles with datasets with more complex details and higher inter-input image variance. The following are our key contributions in this work:

1. We propose a simple solution termed **Spatial Layput Injection (SLI)** that encodes spatial layout information present in the encoder and propagates it to the StyleGAN decoder. Since our aim is to encode style and structure, we do not replace the *map2style* network to have a spatial bottleneck (spatial size $> 1 \times 1$) but rather combine the multi-scale encoder feature maps with noise layers in the StyleGAN generator.

2. We demonstrate that our approach does not replace global consistency with local but rather enhances spatial layout details whilst preserving global aspects of image generation. (see Fig. 5.2).

3. With our proposed approach, we achieve state-of-the-art results on face data like CelebA and CelebAMask-HQ across a wide variety of I2I tasks. We also demonstrate the effectiveness of *SLI-pSp* on a more complex, custom dataset of building images.

4. Finally, we propose that SLI can be seen as a concept rather than a particular architectural design. We show that it is amenable to modifications by introducing Attentive SLI (ASLI-pSp) to achieve even better results. Such modifications can be made as seen fit, depending on memory and cost requirements.

5.1 Related Work

5.1.1 GANs

Generative Adversarial Networks (GANs) [57] work on the principle of a min-max game between a generator model and a discriminator model, each trying to defeat the other. GANs have paved the way for a variety of generative modelling

tasks across different modalities such as RGB images, depth maps, segmentation masks, etc. When it comes to facial data, StyleGAN [98] is considered the state-of-the-art. This is due to its unique, progressively growing structure, which helps in generating high-resolution images with great visual detail. Its mapping network helps disentangle factors of variation in the data, and noise layers in the synthesis block help with stochastic variations in outputs. While StyleGAN gives impressive results, there still exist some characteristic artifacts which StyleGAN2 [99] attempts to alleviate. One of the key issues is the artifacts introduced in generated images due to AdaIN, therefore normalization is proposed on the expected statistics of the incoming feature maps. StyleGAN2-ADA [96] proposes an adaptive discriminator augmentation scheme that stabilizes and makes StyleGAN2 training work well in low data regimes. Further, Karras et al. [97] demonstrate improvement over StyleGAN2 by interpreting all signals in the network as continuous and deriving small architectural changes to further improve FID of StyleGAN2. They aim at making the synthesis network equivariant w.r.t the continuous signal in order to ensure that the finer details and coarser details transform together using Fourier features.

Bartz et al. [14] use the noise inputs of the StyleGAN model to transfer content and color information in a fixed StyleGAN model. They train an encoder to reconstruct and perform denoising without re-training StyleGAN. However, instead of using stochastic noise, they force the model to rely only on latent code and only train the layers responsible for predicting stochastic noise inputs. Abdal et al. [1] propose a novel algorithm to embed a given image in the latent space of StyleGAN. They show results for different tasks such as morphing, style transfer, and expression transfer, and also provide insights into the latent space of StyleGAN. Park et al. [172] propose a Swapping Autoencoder model for performing image manipulation. Their method encodes the image into two independent components, structure and texture, and then combines them to form a realistic image. The swapping autoencoder consists of an encoder E and a generator G that form a mapping between the latent image and code, dividing the latent code into structure and texture, and enforcing swapping with other images. They also use a patch discriminator, which learns the co-occurrence statistics of image patches. Their model is based on Im2StyleGAN [1].

The ability to condition GANs on priors has led to exponential growth of work in conditional image synthesis, and more generally, image-to-image translation methods discussed further in the next section.

5.1.2 Conditional GANs & Image-to-Image Translation

Conditional GANs enable a more controlled image generation by conditioning the generator on some prior. Previous works have demonstrated the ability of conditional GANs across various modalities, such as 3DGAN [243], SegAN

[253], MedGAN [9] for medical image synthesis, text-to-image synthesis such as AttnGAN [251], StackGAN [266], StackGAN++ [267], ACGAN [167], to hand-written font generation with GlyphGAN [67].

Image-to-Image (I2I) translation methods aim to learn the mapping of an image from a source to a target domain. Isola et al. [85] first proposed the I2I method for conditioning the network on image and further introduced L_1 loss in the objective function. They demonstrated results on multiple tasks such as image $\rightarrow$ label, edges $\rightarrow$ image, day $\rightarrow$ night images, and map $\rightarrow$ aerial. The works of Wang et al. [230] extend this approach for high-resolution image generation and manipulation. Coming specifically to the task of semantic image synthesis, usually the semantic mask is provided to the generator via an encoder mapping [85, 217, 230]. However, it doesn't lead to great semantic coherence in the final image. Therefore, SPADE [171] method uses spatially adaptive normalization layers to modulate activations of the generator network. In CC-FPSE [136], the authors propose using a Feature Pyramid-based Discriminator to work at multiple scales, unlike a conventional PatchGAN discriminator. In OASIS [215], the authors propose using a semantic segmentation-based discriminator to get better mIoU scores for the generated images. They also get rid of a traditional encoder and instead enable multi-modal synthesis directly by sampling a 3D noise tensor at each layer of the model.

pSp [188], on the other hand, proposes to solve various I2I tasks with a common architecture, unlike previously mentioned methods. It uses an intermediate style space to embed input images and uses pretraining of a StyleGAN model as a proxy for the discriminator. It also uses a Feature Pyramid-based encoder to better extract style vectors for multiple scales. We identify the shortcomings of pSp's global approach to I2I tasks and propose Spatial Layout Injection (SLI) into the generator model to better preserve local structure information present in the input image. Such information is crucial for tasks such as semantic image synthesis, inpainting, super-resolution, etc. We do so without compromising the global approach of pSp that is provided by the style vectors being injected into the generator.

Alaluf et al. [6] propose a method for iterative refinement over the baseline pSp architecture. At every time step, they provide the network with the current input and the output obtained at the previous time step, while the initial output is initialized using the latent average.

5.1.3 Self-Attention Methods

Transformer architectures were initially proposed for machine translation tasks [224] in NLP, but these models have made their headway into computer vision in recent years. ViT [44] made a breakthrough with its superior performance on large-scale datasets like ImageNet. Other key architectures include DETR [25]

for object detection, VilBERT [140] for vision and language tasks, and CCNet [82] for semantic segmentation. BoTNet [212] proposes a hybrid model that uses both convolutions and self-attention. This is different from other models like DETR and CCNet, as they use self-attention outside their backbone, unlike BoTNet. We use BoTNet as a modification to the simple SLI-pSp model, and denote it by ASLI-pSp. Note that this modification is merely to portray how SLI can be seen as a concept and upgraded in its implementation depending on memory and cost requirements. We do not perform quantitative studies across attention models to compare the performance of our technique.

5.2 Approach

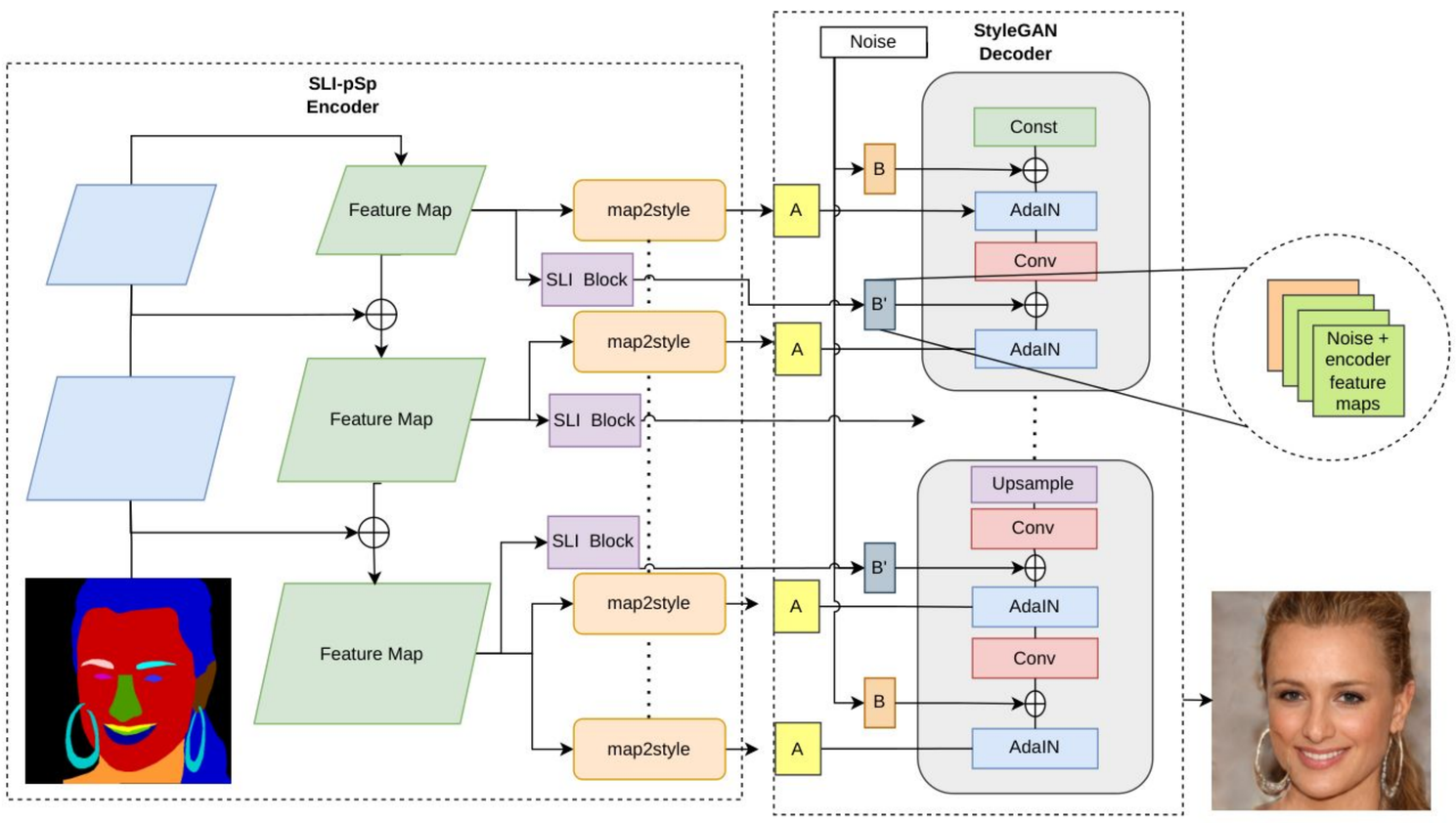


Figure 5.2: **SLI-pSp** framework. We extract feature maps using a feature pyramid network. These features are propagated via map2style blocks and injected into the StyleGAN generator. While this serves the global aspect of image generation, we also propagate these encoded features via Spatial Layout Injection (SLI) blocks to combine with the Noise Layers of StyleGAN. This helps in providing spatial context to the image synthesis process, as is required for various I2I tasks.

5.2.1 Overview

In this section, we present SLI-pSp, a model that introduces a spatial layout injection from the encoder to StyleGAN noise layers, thereby enabling better performance in image-to-image translation tasks. We use the pSp framework as the base architecture and utilize multi-scale features from the encoder to produce better local consistency in the generated images while retaining global properties. We empirically show that such feature injections help preserve semantic details

better while also enjoying the interesting properties like multi-modal synthesis that StyleGAN has to offer.

5.2.2 Baseline Method

We first briefly revisit the key components of StyleGAN and the pSp framework that form the backbone for our method.

StyleGAN [98] proposes a novel style-based generator architecture for unconditional image generation. A key idea is to transform the latent vector $\mathbf{z} \in \mathcal{Z}$ via a mapping network $\mathbf{f}$ into a vector $\mathbf{w}$ in the intermediate latent space $\mathcal{W}$. This aids the representation to be more disentangled with respect to the factors of variation in the network. $\mathbf{w}$ is then propagated via learned affine transformations to the synthesis network as styles $\mathbf{y}$ that control the Adaptive Instance Normalization (AdaIN) after convolution operations. Finally, to encourage stochastic variation in generated images, explicit single-channel Gaussian Noise is fed to each layer of the synthesis network.

pSp [188] proposes a Feature Pyramid Network-based encoder framework that embeds input images into an extended latent space $\mathcal{W}+$. The intermediate style representation offers many advantages, such as the ability to resample style vectors, thereby providing support for multi-modal synthesis. They also propose a new way to use a pre-trained StyleGAN generator model for generic image-to-image translation tasks. The style vectors are passed to the StyleGAN generator corresponding to their scales. A key advantage of this approach is that it does not require training a discriminator network.

5.2.3 Spatial Layout Injection

One of the drawbacks of pSp's global approach is that it seems to miss finer or underrepresented details in the face images, such as earrings or caps. On probing further, this approach broke down when tried on other datasets with more variation and complexity, such as Places2 [273]. This is due to a lot of high-frequency information and numerous objects in a scene that are insufficient to be captured without the transfer of input spatial layout information. pSp's *map2style* network downsamples the spatial resolution to 1×1, thereby hindering the propagation of input layout information from the encoder.

To remedy this, we propose Spatial Layout Injection (SLI) that encodes spatial layout information and propagates it to the pre-trained StyleGAN generator. One way to inject such information could be by modifying the $map2style$ network to create a spatial bottleneck and propagating style and encoded spatial layout to the generator. However, there is merit in having a global approach as is required by certain I2I tasks. Therefore, we embed the feature maps from the encoder by combining them with the noise layers in the generator without

changing the style injection via map2style in the network. Let the encoder feature maps be denoted as $\hat{E}_i$, where $i \in \{16, 32, 64\}$ corresponds to the spatial scales of the feature maps. These are generated using a feature pyramid over a ResNet backbone. Let the noise layers in the StyleGAN generator be represented by N_j, where $j \in \{4, 8, 16, ..., 1024\}$ represents spatial sizes. The combined spatial layout feature maps and noise added to the generator, B' can we written as

$$B' = concat(conv(\hat{E}_i), N_i) \quad i \in \{16, 32, 64\}, \tag{5.1}$$

where *conv* represents a convolution layer that reduces the number of channels from 512 to 256, and the *concat* operation concatenates along the channel dimension as spatial sizes are the same. Instead of using a single-channel noise, which is then broadcast and added to the output of the corresponding convolution, we use a 256-channel noise, which is then combined with a 256-channel *SLI*.

This ensures that style propagation via the *map2style* network remains unhindered whilst scene layout information is also injected into the generator. These feature maps are multi-scale, so it is possible to choose the scale and number of feature maps to inject into the generator. An added advantage of SLI is that it is input modality agnostic, i.e., the input could be a segmentation map, edge map, blurry image, etc., without any change required in the architecture. Moreover, this is done without compromising the StyleGAN properties, e.g. multi-modal synthesis via style-mixing, as shown in Appendix Figure B.2.

5.2.4 Variant - Attentive Spatial Layout Injection

To show the versatility of the SLI concept, we employ a variant of SLI with attention, namely Attentive SLI (ASLI-pSp). We use Bottleneck Transformers as introduced in [212] for our attention layer. This modification can be understood via the following equation

$$B' = concat(botnet(conv(\hat{E}_i)), N_i) \quad i \in \{16, 32, 64\}, \tag{5.2}$$

where $botnet(.)$ is the Bottleneck Transformer operation that takes in the output of the encoder feature maps after they are passed through the *conv* layer. The botnet-transformed feature maps are then similarly concatenated with the noise layers in the StyleGAN generator. We will see in later sections how such a modification can boost scores like FID.

5.2.5 Loss Functions

We use similar loss functions to those used in pSp, as it serves to make a fair comparison with our proposed approach. We also employ a weighted combination

of losses that are listed below. The pixel-wise reconstruction loss (or L_2 $Loss$) is defined as

$$\mathscr{L}_2(x) = ||x - SLIpSp(x)||_2.$$

We also use an LPIPS [270] loss, which helps preserve image quality. Here, F($\cdot$) denotes the perceptual feature extractor.

$$\mathscr{L}_{LPIPS}(x) = ||F(x) - F(SLIpSp(x))||_2.$$

The latent vector regularisation loss helps the encoder $\underline{\text{E}}$ to generate style vectors in the latent space closer to the average latent vector $\overline{w}$.

$$\mathscr{L}_{reg} = ||E(x) - \overline{w}||_2.$$

For facial images, preserving identity is crucial. Hence, an ID Loss is also employed when dealing with facial images. The loss measures the cosine similarity between the input and output image:

$$\mathscr{L}_{ID}(x) = 1 - < R(x), R(SLIpSp(x)) >,$$

where R(.) is a pretrained ArcFace Network [42]. Thus, the final loss equation can be expressed as

$$\mathscr{L}(x) = \lambda_{L_2}\mathscr{L}_2(x) + \lambda_{LPIPS}\mathscr{L}_{LPIPS}(x) + \lambda_{reg}\mathscr{L}_{reg}(x) + \lambda_{ID}\mathscr{L}_{ID}(x). \quad (5.3)$$

5.3 Experimental Setup

We demonstrate our approach across a wide variety of tasks and datasets to illustrate the benefits of combining style and spatial layout information in an I2I setting. Specifically, we pick the following tasks: Segmentation Map to Face (Seg2Face), Super Resolution, and Edges to RGB Image.

5.3.1 Datasets

- **CelebA-HQ** [95]: It contains about 30,000 high-resolution face images from the CelebA dataset. The train set consists of about 24,000 images. It is used in our Super Resolution task.
- **CelebAMask-HQ** [120]: It is a derivative of the CelebA-HQ dataset with the same train-test splits, but comes with segmentation masks. Each mask of an image is manually annotated, and the dataset contains 19 classes such as skin, nose, eyes, lip, hair, etc. We use this for the Seg2Face Task.

- **AFHQ-Dog** [33]: This dataset is a collection of around 5000 high-quality dog images (faces), of which about 500 images form the test set. We showcase super-resolution results on this dataset.

- **Places2-CustomBuildings**: We extract a few building categories from the Places2 dataset [273] for pretraining of the StyleGAN generator. These include building facades, courthouses, manufactured homes, office buildings, parking garages, outdoor and residential neighborhoods, and total to about 150,000 images. We select a subset of around 30,000 images from these as a train set (and ∼3700 as a test set) for evaluating the baseline pSp, and our methods SLI-pSp and ASLI-pSp. A custom dataset is chosen to showcase the effectiveness of SLI in the generated images. We use this dataset for the Edges to RGB image task.

5.3.2 Training Details

Similar to the pSp framework, we train our network using the ResNet-IR architecture. Further, for each of the datasets mentioned above, we train the StyleGAN separately and then use it as the decoder with the modified psp-Encoder for further training in a conditioned setting. The pSp-Encoder is trained from scratch, while the StyleGAN is further fine-tuned. Input image size is 256×256 for all tasks across different datasets. The learning rate for the experiment is 0.0001 with the Ranger optimizer. The loss coefficients used were the same as in pSp training, i.e $\lambda_{LPIPS} = 0.8, \lambda_{L_2} = 1.0, \lambda_{reg} = 0.005$. $\mathscr{L}_{ID}$ is not used for any task except CelebA-Super Resolution, which uses $\lambda_{ID} = 1.0$.

5.4 Results and Discussion

To evaluate the effectiveness of SLI, we compare our approach to pSp and a few other baselines across a variety of I2I tasks. We describe the experiment, its results, and quantitative evaluations to assess image diversity, quality, and correctness. We then discuss the key differences in our approach with respect to the results.

5.4.1 Segmentation Map to Face Image

Method: The generated face images are conditioned on corresponding segmentation maps provided during training and inference. During the evaluation, we do style-mixing at higher-level features, i.e. we combine the latent code of the input image (segmentation mask) with a randomly sampled latent vector. We evaluate our method against various baselines.

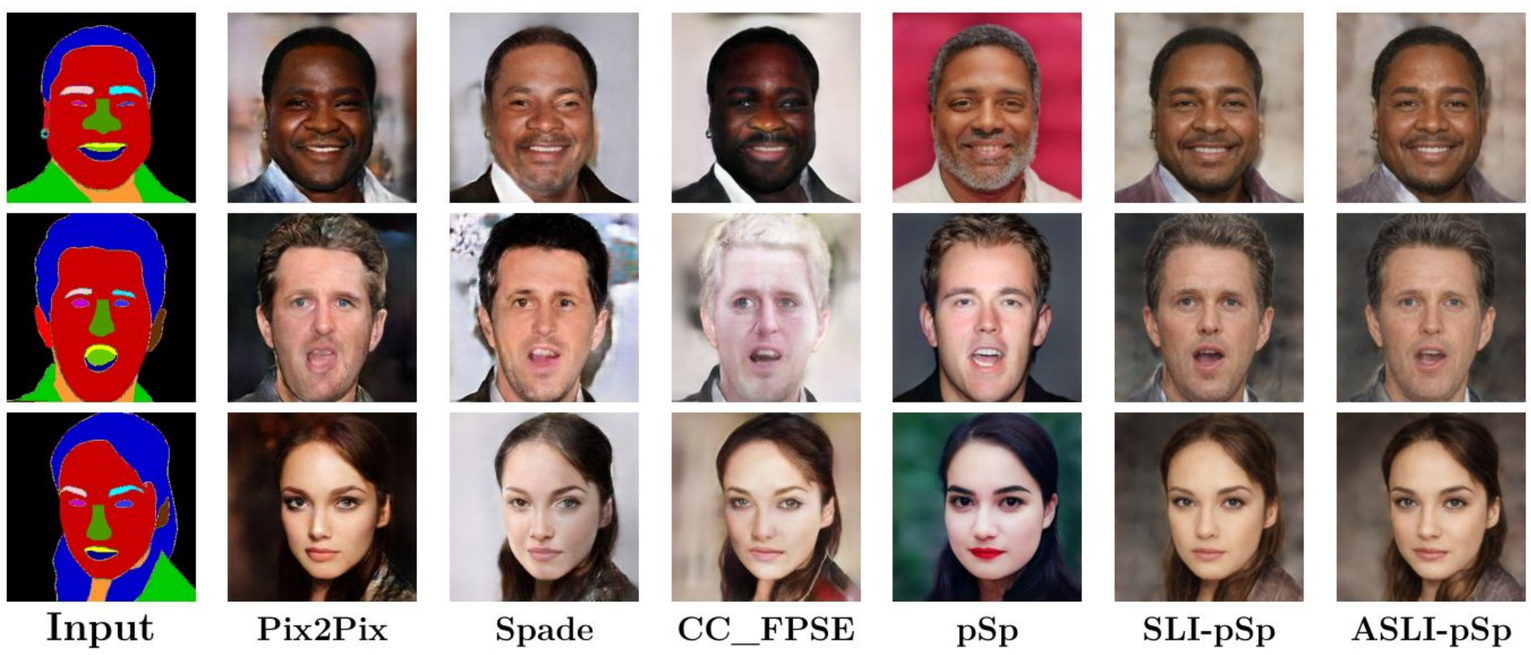


Figure 5.3: We show results on the Seg2Face task using the CelebAMask-HQ dataset and compare it to some previous works.

We evaluate SLI-pSp and ASLI-pSp on the CelebAMask-HQ dataset by synthesizing face images from segmentation maps. One of the biggest differences in the generated outputs is that SLI variants can synthesize smaller and underrepresented objects in the dataset, such as earrings, while maintaining image quality, as shown in Fig. 5.1. In addition to that, we observe a tighter correspondence of the generated images with the label map. This is especially evident in the case of varying hairstyles (See Fig. 5.1 (b), (c) and Fig. 5.3). We quantify this tighter correspondence by computing mean IoU scores across classes using the generated images for each method.

Table 5.1: Quantitative evaluation of our method and pSp on the test set.

Method	**LPIPS Loss ↓**	**FID ↓**	**mIoU ↑**
pSp	0.35	53.90	0.61
SLI-pSp	**0.31**	37.32	**0.81**
ASLI-pSp	0.32	**36.89**	**0.81**

Additionally, as shown in Fig. 5.1(c), pSp confuses the cap class with hair and generates a photorealistic but semantically incorrect image. In this case, both SLI variants are able to capture the correct semantic information. Note that not only does our method generate the correct cap structure while displaying the right texture, but it also displays realistic lighting along the contours where the shadow of the cap is visible.

To evaluate quantitatively, we employ various metrics such as LPIPS, FID, and mIoU. FID [69] is used as a proxy for realism and is sensitive to both quality and diversity. The SLI-variants perform significantly better than pSp, with a difference of greater than 16 points. The mean IoU scores are computed by evaluating the label map synthesized by our generated images against ground truth label maps using a pre-trained segmentation network. We outperform

pSp in this regard as well, and it strengthens our qualitative analysis about SLI variants being better with respect to underrepresented objects in the data.

5.4.2 Super-Resolution

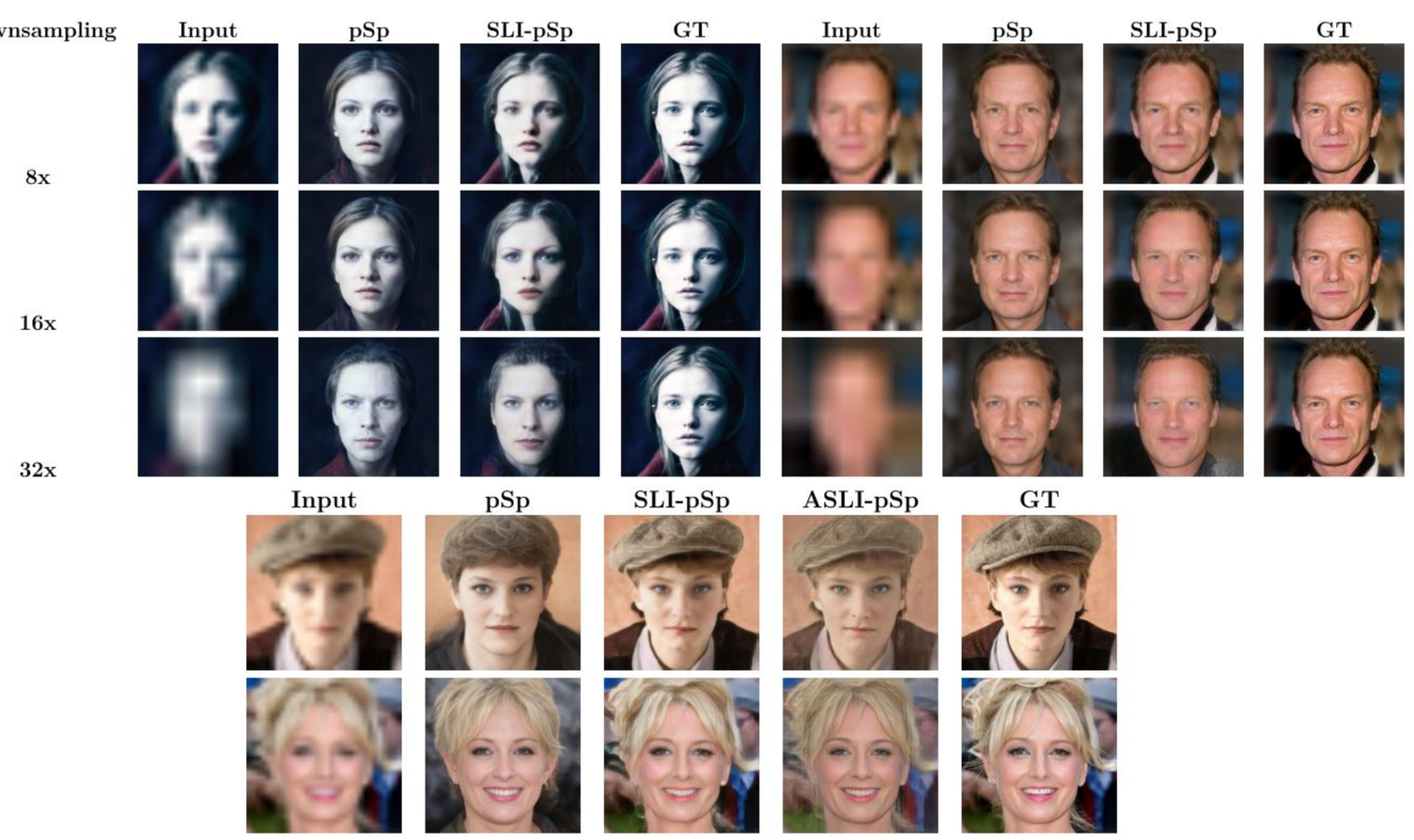


Figure 5.4: Superresolution Results. We observe a mild tradeoff between the preservation of semantics and overall image quality in both methods due to the introduction of locality bias.

Method: We follow the pSp setup to synthesize high-resolution images from their low-resolution counterparts. We downsample the high-resolution images at various scales (x2, x4, x8, x16, x32) using bi-cubic interpolation and use them as the input image.

We showcase our method's performance on the Super-resolution task. We choose 2 datasets: CelebA and AFHQ-Dog, on which we evaluate our method's performance.

In the case of CelebA, we notice that our method visually performs at par with pSp, but is better at preserving the spatial information present in the downsampled input. SLI variants are especially better at preserving the hairstyles and color scheme of the clothes present in the low-resolution image (See Fig. 5.4 (top)). On observing Fig. 5.4(bottom), we notice how well SLI variants can capture objects like caps that are omitted by pSp and misrepresented as hair. Similarly, we also notice the hairstyle is kept consistent in the upsampled images by SLI variants as opposed to pSp. The above claims are well supported quantitatively (Table 5.2) by substantial improvements in FID scores. LPIPS and L_2 losses also show a reduction on the test set, thereby signaling better performance using SLI. We do notice a slight reduction in the sharpness of the

Table 5.2: Quantitative Results for Super-resolution task on CelebA. We evaluate data with 8x and 16x downsampling. SLI variants show substantial improvements across various metrics.

Method	**LPIPS Loss** ↓	L_2 **Loss** ↓	**FID** ↓
pSp(8x)	0.23	0.06	31.35
SLI-pSp (8x)	0.12	**0.01**	10.21
ASLI-pSp (8x)	**0.10**	**0.01**	**9.90**
pSp(16x)	0.24	0.06	32.53
SLI-pSp (16x)	0.18	**0.02**	18.33
ASLI-pSp (16x)	**0.17**	**0.02**	**18.01**

image generated by our method that can be attributed to the introduction of local information injected by SLI and is something to improve for future work.

We also evaluate the task of super-resolution on the AFHQ-Dog dataset, which is different compared to the human faces dataset on which StyleGAN works quite well. We observe how pSp misses out on upsampling colors correctly, and that SLI-pSp does a better job at generating more accurate colors and lighting in the scene. We also notice in Fig. 5.5 (1st and 3rd row), how SLI variants are able to preserve the yellow part of the green grass. This can be argued to be representative of the style of the input image (and not merely structural), which our method is able to capture. We do notice a little washing out of colors in our ASLI variant that will be investigated in future work.

Overall, for tasks like super-resolution, we argue for approaches that encode spatial features accurately in the generation process, as it is imperative to have an accurate structure in the generated "upsampled" image. This is achieved to a great extent by our SLI-pSp method.

5.4.3 Edges to RGB Image

We explore our method on another task, which takes an edge image as input and generates an RGB image. After evaluating face data, which StyleGAN can handle relatively well, we wanted to stress-test our method on a complex dataset. Therefore, we compiled building images from various categories in the Places2 dataset, termed Places2-CustomBuildings. The variance among images is high as these buildings have been captured in varying lighting conditions, different poses, and comprise different types of structures like churches, skyscrapers, residential homes, etc. We then create edge images for the dataset using Canny edges and some post-processing to decrease noisy edges.

We observe that pSp doesn't perform well on this dataset at all and produces generic facade details, which seem to be common across different input edge

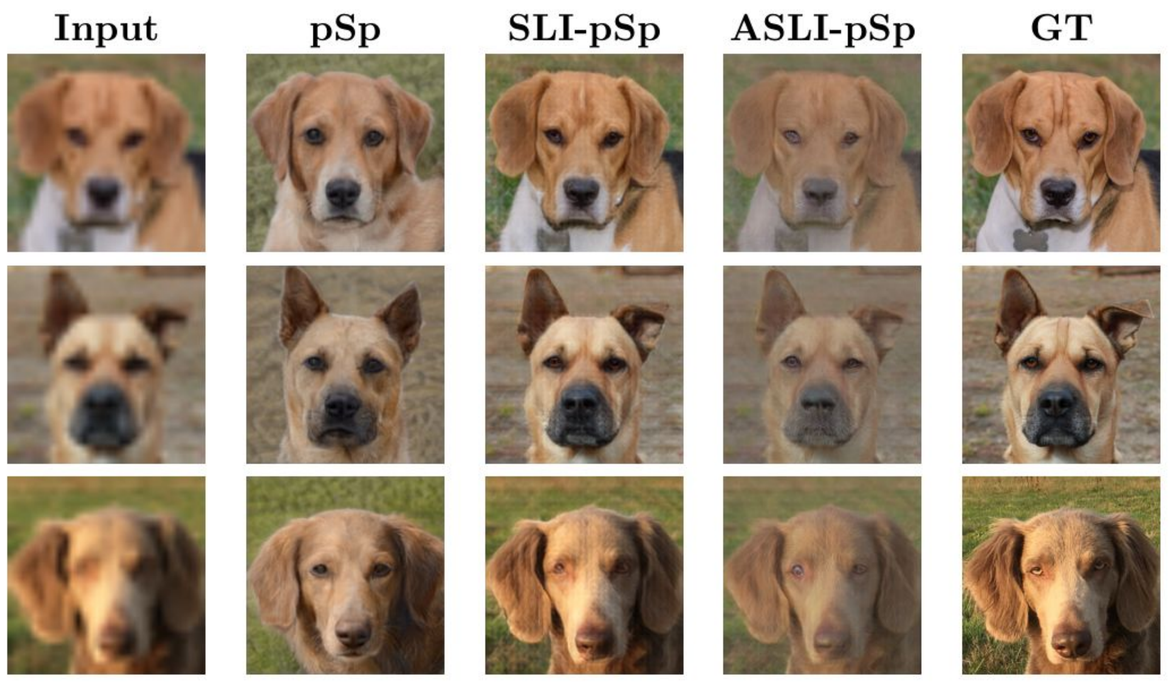


Figure 5.5: Results on AFHQ super-resolution. The first row shows results for incorrect color, the second and third rows show results for incorrect structure and colors.

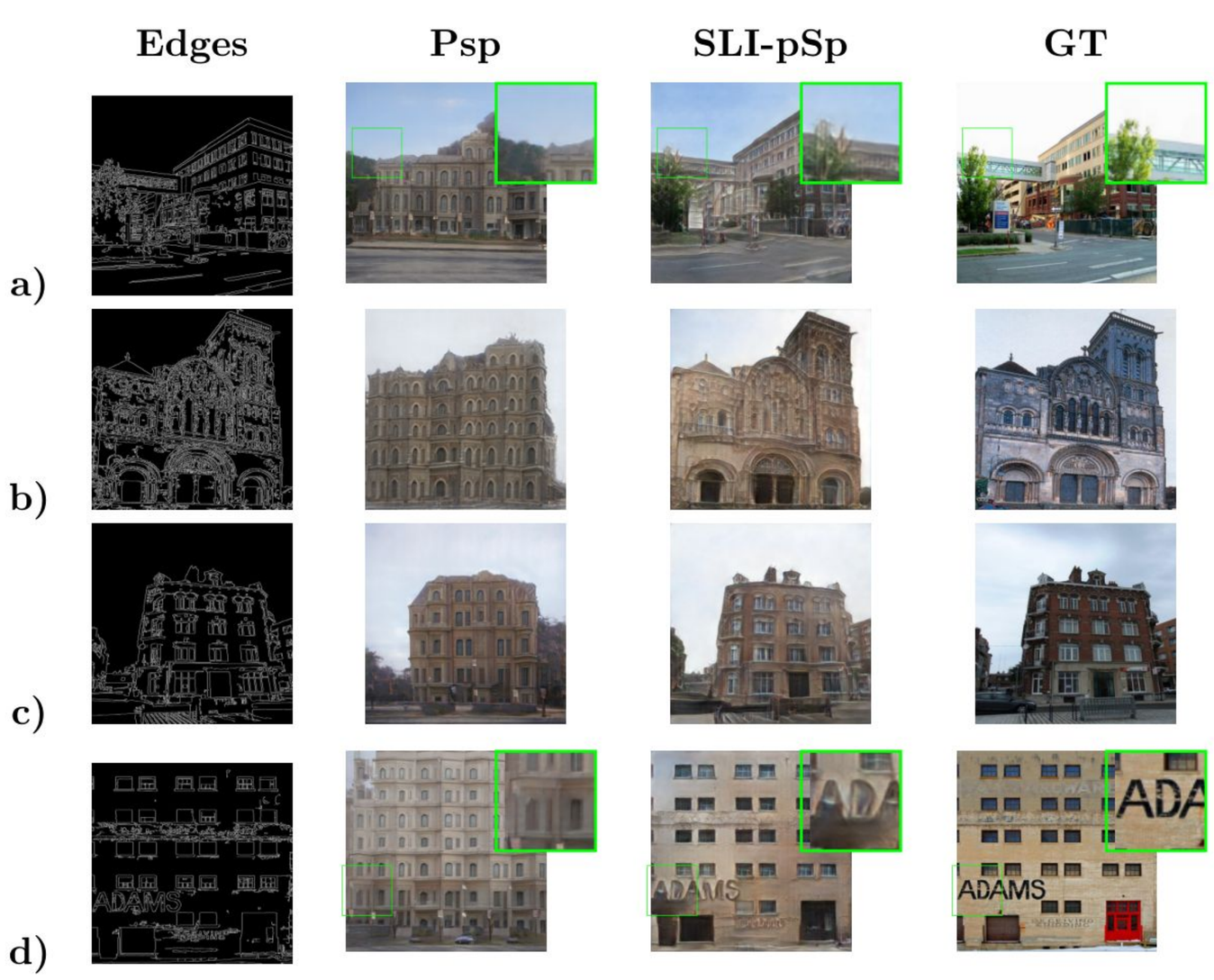


Figure 5.6: Results on Places2 dataset. Our method generates much reliable information due to feature injection.

Table 5.3: Quantitative Results for Edges to RGB image task on Places2-CustomBuildings dataset.

Method	**LPIPS Loss ↓**	**FID ↓**
pSp	0.48	236.733
SLI-pSp	0.32	**23.71**
ASLI-pSp	**0.31**	28.71

images, although it does very loosely follow the outline of the building and the sky. We attribute this to two primary factors: a) the lack of spatial information in the image synthesis process and b) the lack of semantic context at the edges. This can be verified by the very high FID score achieved by pSp on this task. The same set of edges can be used for representing a variety of different scene components, unlike face data, where one can visually distinguish between the edges that create different facial structures, such as nose, ears, or mouth, thereby making the problem highly ill-posed. On the other hand, we notice that SLI-pSp is able to capture spatial layout very well and is able to map the highly concentrated edges to semantically consistent building components that make sense globally (image-level). We see in Fig. 5.6(c), how our method can capture the individual window designs and even engravings in the buildings (Fig. 5.6(d)), unlike pSp. We also outperform pSp by quite a margin quantitatively (Table 5.3). While the generated image by our method lacks the visual quality that we got on facial images (primarily due to the complexity of the task and data), we put forward a strong case for SLI as an idea to be explored even further, along with the idea of using style vectors in the generation process, as done in pSp as demonstrated further in Appendix (B).

5.5 Closing Notes

In this work, we have identified the shortcomings of pSp, a generic Image-to-Image translation framework. We discover that due to the global approach of pSp and the lack of spatial layout context in the image generation process, it is not able to synthesize finer and less represented details in the dataset, and fails to work in more complex data settings. We propose a simple fix termed Spatial Layout Injection (SLI) that encodes the spatial layout information from the input image and propagates it to the StyleGAN decoder. While there is an advantage in having a global approach for certain problem settings, we argue that it is not sufficient and that we need both style and spatial information in the image synthesis process. Hence, SLI-pSp combines the multi-scale encoder feature maps with the noise layers in the StyleGAN generator without modifying the style propagation component. We demonstrate through our detailed experiments that we don't replace global context with local but rather amplify the spatial layout

details required for many I2I tasks. We showcase results and comparisons with pSp on various tasks such as segmentation map to face image, edges to RGB image, and super-resolution. We evaluate these tasks across various datasets and show, qualitatively and quantitatively, the importance of SLI as a concept in the I2I process. While SLI is done by a simple conv operation, we show how it can be upgraded to a different design choice by introducing ASLI-pSp, depending on the computational budget.

While we achieved good performance across tasks, there were minor sharpness reduction issues and slight artifacts in the case of super-resolution. This might require probing various configuration changes to the SLI Block or even modifying the encoder for better features to inject into the generator. While we achieved substantially better results in the edges $\rightarrow$ buildings task, notwithstanding its ill-posedness and complexity, there is much room for improvement, as the generated images lacked the finesse present in the facial images. This could be a potential future direction where the generator architecture might need to be modified, or maybe a discriminator would aid in better visual quality. We hope that the simple yet important concept that we explored will encourage the vision and graphics community to further conduct research in this direction.

Chapter 6

Curvy: A Parametric Cross-section based Point Cloud Generation

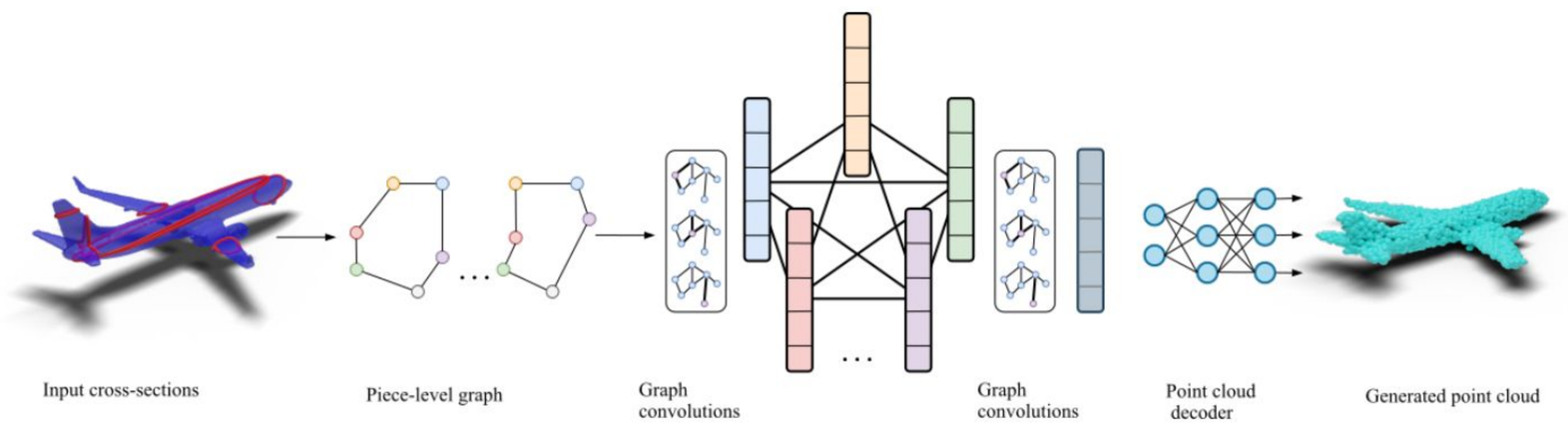


Figure 6.1: We present a novel graph neural network-based approach for generating point clouds from the parametric representation of cross-sections. The graph containing one-ring neighborhoods of pieces within a cross-section is passed through graph convolutions consisting of attention and further aggregation to obtain a high-level representation for every cross-section. A new fully connected graph is then used to obtain the embedding from which the point cloud is recovered.

Surface reconstruction from cross-sections is a well-explored problem. There is a rich literature on methods that demonstrate the generation of reliable surfaces from cross-sections with applications in medical volume reconstruction and engineering. Little work exists that provides insights into how complex objects could be generated using cross-sections with the help of deep learning methods that could provide an added advantage of capturing the semantic context associated with shapes. Deep learning-based methods can provide better generalizability qualities associated with unseen shapes of similar types. Unlike traditional methods of surface generation from incomplete point clouds, the problem of surface reconstruction from cross-sections brings unique challenges that we aim to address in this paper. Previous approaches for point cloud completion have focused on generating point clouds from images or representation learning using autoencoders. Previous methods focused on generating surfaces using

cross-sections and did not involve any learning based on the class of objects. In this work, we present a novel approach for generating point clouds using cross-sections of meshes with the help of generative modeling. We present unique challenges pertaining to the representation and generation in this problem setting. To the best of our knowledge, no prior work has tried to look at this problem through the lens of deep learning due to the nuances of the data and the challenges in representation. Most methods in classical literature cannot generalize based on object class and employ complex mathematical machinery to generate reliable surfaces. We present a simple, learnable approach to generate a large number of points from a small number of input cross-sections. We use a compact parametric polyline representation using adaptive splitting to represent the cross-sections and perform learning using a Graph Neural Network to reconstruct the underlying shape adaptively, reducing the dependence on the number of cross-sections provided. CAD sectioning is a key area where this has numerous applications. In real-world scenarios, reconstruction of point clouds or surfaces from cross-sections is needed for reverse engineering, legacy-part restoration, and areas where only sparse cross-sections are available. Several applications also exist in monument restoration, where capturing the artifact completely is not feasible.

In our approach, we change the input representation of the cross-sections to capture more salient information that is missed if points sampled from the surface are used. Unlike incomplete point clouds that contain dense information in most regions and incomplete information in some, cross-section curves have a highly non-uniform information distribution, thus calling for reconstruction methods that work with sparse anisotropic information. Our contributions in this work can be summarized as follows:

1. A parametric approach for learning surface reconstruction instead of relying on points or polylines for input in a deep learning-based setting.

2. A novel framework for generating a point cloud while adapting to the anisotropic and sparse nature of input cross-sections. We also present two attention mechanisms to focus on the local and global structure of the cross-sections and show their significance empirically.

3. A new dataset for parametric representation of cross-sections (source code available at https://github.com/graphics-research-group/curvy).

6.1 Related work

Surface reconstruction is a widely studied problem in computer graphics. As methods for representing 3D data change, so do the methods for shape reconstruction. The different methods for representing 3D data include a voxel-based

representation that gives information pertaining to points in a discrete grid, point clouds that contain the locations of information, and meshes that have added neighborhood information in the form of an adjacency matrix corresponding to the points. Newer implicit methods directly target surface generation by learning to produce the implicit field functions. We divide this section based on the representation of the output for different methods.

6.1.1 Pointcloud generation

Two primary approaches have been explored for point cloud reconstruction. Point clouds have been reconstructed using multi-view/single-view images and partial-point clouds.

A deep autoencoder network for reconstructing point clouds results in compact representations and can perform semantic operations, interpolations, and shape completion [2],[131]. These networks leverage 1-D and 2-D convolutional layers to extract latent representations for generating point clouds. Single image point cloud generation has also been performed hierarchically from low resolution by gradually upsampling the point cloud as explored in [47]. This multi-stage process uses EMD distance [47] and computes Chamfer distance for the later stages w.r.t. the ground truth dense point cloud. Another approach uses a multi-resolution tree-structured network to process point clouds for 3D shape understanding and generation [52]. Some newer methods also approach this problem from a local supervision perspective to understand the local geometry better [65]. Further, skip-attention has been shown to play an important role in tasks such as point cloud completion [237]. The architecture proposed consists primarily of three parts - a point cloud encoder, a decoder that generates the point cloud, and skip-attention layers that fuse relevant features from the encoder to the decoder at different resolutions. Reinforcement learning has also been explored with GANs trained for the point cloud generation. The agent is trained for predicting a good seed value for the adversarial reconstruction of incomplete point clouds [199]. The method uses an autoencoder trained on complete point clouds to generate the global feature vector (GFV) and a GAN that is trained to produce GFV. The pipeline uses GFV generated from an incomplete point cloud as a state and supplies it to an RL agent, which the GAN uses to generate GFV close to the GFV of a complete point cloud.

6.1.2 Surface reconstruction

One of the seminal works [150] proposes constructing 2D geometric shapes from 1D cross-sections. The method provides sampling conditions to guarantee the correct topology and closeness to the original shape for the Hausdorff distance. One of the early works [78] proposes a manifold mesh reconstruction method from

unorganized points with arbitrary topology. The method proposed defines a two-step process for reliably reconstructing the geometric shape from unorganized point cloud sampled from its surface.

Early works took inspiration from medical imaging problems, a two-step process for the reconstruction of a surface from cross-sections has been proposed by first computing the arrangement for the cross-section within each cell and then reconstructing an approximation of the object from its intersection with the cell boundary, and gluing the pieces back together yields the surface [19]. An algorithm for non-parallel cross-sections consisting of curve networks of arbitrary shape and topology has also been developed [134]. Several methods propose implicit field-based reconstruction. One such method utilizes sign agnostic learning for geometric shapes [10]. This method uses a deep learning-based approach that allows learning of implicit shape representations directly from unsigned raw data like point clouds and triangle soups. The proposed unsigned distance loss family possesses a plane reproduction property based on a suitable initialization of the network weights.

The surface reconstruction method has been performed with topological constraints [118]. The method relies on computing candidates for cell partitioning of ambient volume. The method is based on the calculation of a single surface patch per cell, so that the connected manifold surface of some topology is obtained. 3D surface reconstruction from unorganized planar cross-sections using a split-merge approach using Hermite mean-value interpolation for triangular meshes has also been used [203]. A divide-and-conquer optimization-based strategy can also be employed to perform topology-constrained reconstruction [278]. New methods like Orex [200] leverage deep learning for cross-section to surface generation.

6.2 Approach

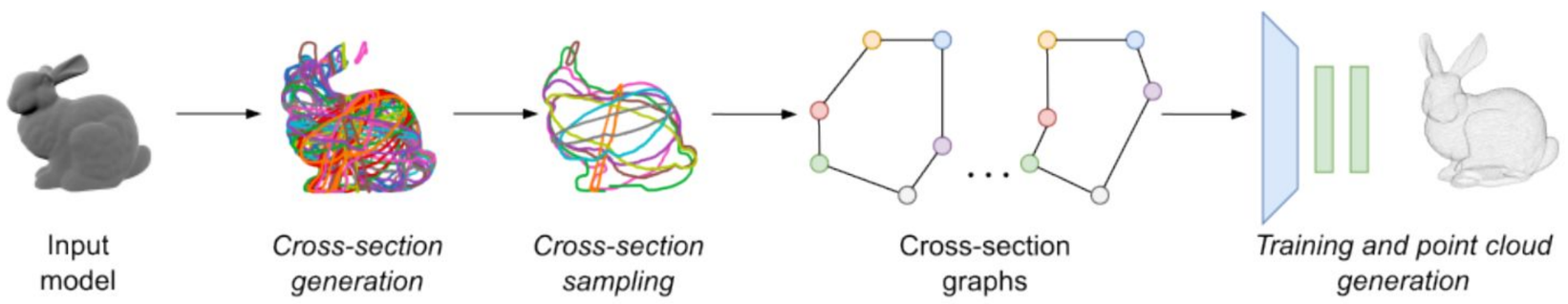


Figure 6.2: Overview of our reconstruction approach. Starting from a parametric representation, a network is trained to generate a surface point cloud.

In this work, we develop an approach for shape reconstruction from a set of unorganized cross-sections. We design a deep neural network that learns the overall structure of various shapes and generates a point cloud representing the original object.

Our approach can be defined as a three-step process. We first generate many cross-sections from 3D models and sample them to create input cross-sectional data. Then, surface points are sampled to generate a point cloud on which an autoencoder is trained to reconstruct the point cloud. In the final step, we use the encoded vector obtained from the autoencoder and train a Graph Neural Network on the parametric representation of input cross-sections to generate an embedding vector in a GAN-based setting to match the encoded vector for the same object. These embedding vectors can then be decoded to obtain the point cloud from the pre-trained autoencoder network.

The cross-sections may be obtained as points sampled along a shape's boundary that can further represent a polyline. However, for complex cross-sections, we would want a representation that optimally captures the curvature-related information. Instead of sampling points, we convert an entire cross-section curve into its parametric representation. This allows us to reduce any loss of information that may occur due to sampling and further helps reduce the memory requirements needed to represent a large number of points in the network. Let's assume the density of points ρ per unit length of a cross-section curve of length l. Depending on the sampling density ρ, the number of points in a curve can vary, and for better information capture, we need a high ρ value to capture the curvature accurately. We note that the parametric curve can be represented using a fixed number of coefficients from which any arbitrary density of points can be sampled. Our overall approach is shown in Figure 6.2.

In this section, we first revisit the parametric representation of curves and understand how we use the parameters of a curve to represent the cross-sections. We then discuss in detail how the problem can be approached in a neural network-based setting and try to answer the nuances associated with the unique problem setting.

6.2.1 Parametric curve fitting

Intersecting a surface mesh with cutting planes results in polygonal cross-sections. Handling the variable size of these polygons with possible holes is challenging for a neural network. A naïve solution is to fix the number of points and then increase (or reduce) the size of each cross-section. An ideal number would be the same as the number of points in the largest cross-section. However, this representation can be wasteful, especially for simple cross-sections with few points. Thus, we propose another solution to this problem where we parameterize each cross-section using piecewise p-degree polynomials. Additionally, continuity constraints can be imposed between pieces to ensure that the piecewise polynomials are connected and also to impose smoothness if required. Let c be an ordered set of points representing a cross-section, then we divide c into k parametric pieces $f_i(t)$ of degree p where the parameter $t \in [0, 1]$. For ensuring C_1 conti-

nuity we impose the constraints between successive pieces: $f_i(0) = f_{i-1}(1)$ and $f_i'(0) = f_{i-1}'(1)$. Fitting a parametric polynomial is a dual optimization problem [50] since we wish the curve to pass through given points and figure out the corresponding parameter values at the given points.

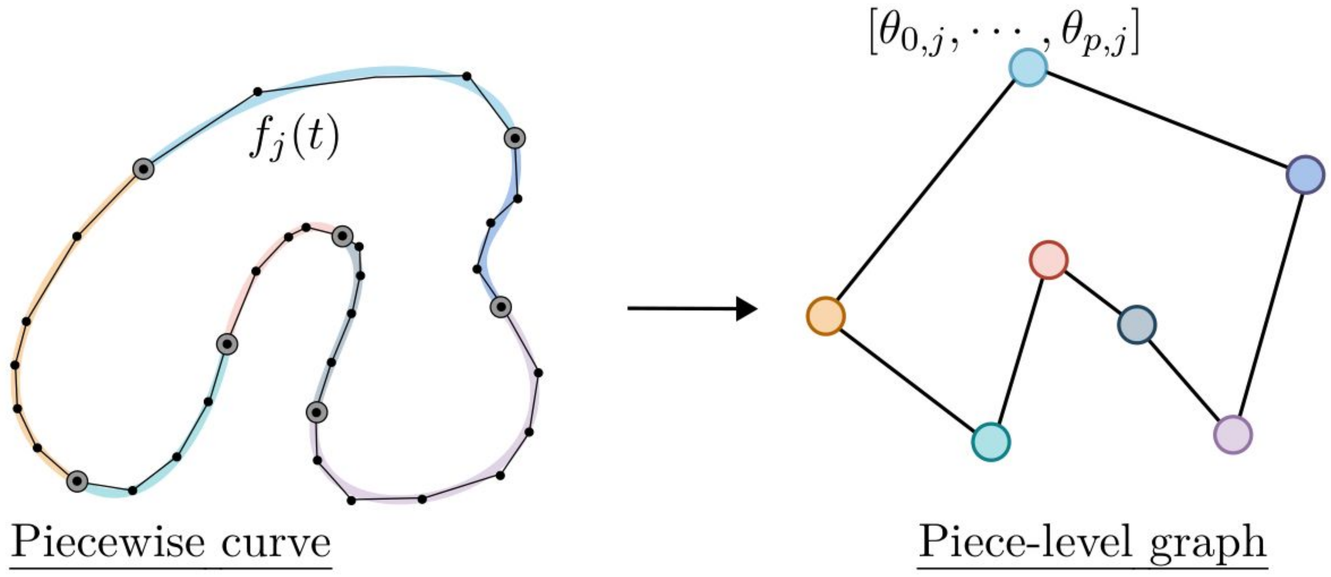


Figure 6.3: Converting a piecewise parametric representation of a cross-section to a graph. The nodes in the graph are matrices of coefficients of the parametric functions.

Each cross-section is divided into k pieces using an adaptive splitting scheme (see sec. 6.2.1). For the j^{th} piece with n_j points, we obtain the parameter values of the parametric function using the Chordial approximation [50] as,

$$t_{i,j} = t_{i-1,j} + \frac{||p_{i,j} - p_{i-1,j}||_2}{\sum_{i=2}^{n_j} ||p_{i,j} - p_{i-1,j}||_2},$$

where p_i for $i \in 1, 2, ...n_j$ are points on a piece, and $t_1 = 0$ such that for each piece $t_i \in [0, 1]$. A constrained least squares problem [21] can be solved to obtain unknown coefficients $\theta_{i,j}$ of the parametric function $f_j(t) = \sum_{i=0}^{p} \theta_{i,j} t^i$ of the j^{th} piece while imposing C_1 continuity between consecutive pieces. This results in the following optimization problem

$$\min \ ||\mathbf{A}\ \Theta - \mathbf{b}||_2^2, \quad \text{s.t.} \quad \mathbf{C}\ \Theta = 0,$$

where Θ is the matrix of size $(p+1)k \times 3$ of unknown coefficients

$$\Theta = [\theta_{0,1}, \theta_{1,1}, \cdots, \theta_{p,1} \ \cdots \ \theta_{0,k}, \theta_{1,k}, \cdots, \theta_{p,k}]^\intercal$$

$\mathbf{A}$ is a matrix of size $n \times (p+1)k$ with n being the total number of points in a

cross-section

$$\mathbf{A} = \begin{bmatrix} 1 & t_{1,1} & \cdots & t_{1,1}^p & 0 & & \cdots & 0 \\ 1 & t_{2,1} & \cdots & t_{2,1}^p & 0 & & \cdots & 0 \\ \vdots & & & & & & & \\ 1 & t_{n_1,1} & \cdots & t_{n_1,1}^p & 0 & & \cdots & 0 \\ \vdots & & & \ddots & & & & \\ 0 & & \cdots & & 0 & 1 & t_{1,k} & \cdots & t_{1,k}^p \\ 0 & & \cdots & & 0 & 1 & t_{2,k} & \cdots & t_{2,k}^p \\ \vdots & & & & & & & \\ 0 & & \cdots & & 0 & 1 & t_{n_k,k} & \cdots & t_{n_k,k}^p \end{bmatrix},$$

$$\mathbf{b} = \begin{bmatrix} p_{1,1} & p_{2,1} & \cdots & p_{n_1,1} & \cdots & p_{1,k} & p_{2,k} & p_{n_k,k} \end{bmatrix}^\intercal.$$

The constraint matrix $\mathbf{C}$ for imposing C_0 and C_1 continuity is of size $2k \times (p+1)k$, and submatrices $\mathbf{C}_a$ and $\mathbf{C}_b$ are of size $2 \times (p+1)$.

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_b & 0 & \cdots & 0 & \mathbf{C}_a \\ \mathbf{C}_a & \mathbf{C}_b & 0 & \cdots & 0 \\ & & \ddots & & \\ 0 & \cdots & 0 & \mathbf{C}_a & \mathbf{C}_b \end{bmatrix}, \text{where}$$

$$\mathbf{C}_a = \begin{bmatrix} -1 & -1 & -1 & \cdots & -1 \\ 0 & -1 & -2 & \cdots & -p \end{bmatrix}, \text{and } \mathbf{C}_b = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \end{bmatrix}.$$

The solution to Problem 6.2.1 can be determined by creating a Lagrange function and enforcing the Karush–Kuhn–Tucker (KKT) conditions [114]. The simplified expression obtained after enforcing the KKT conditions can be described as the following linear system of equations:

$$\begin{bmatrix} \mathbf{A}^\intercal\mathbf{A} & \mathbf{C}^\intercal \\ \mathbf{C} & 0 \end{bmatrix} \begin{bmatrix} \Theta \\ z \end{bmatrix} = \begin{bmatrix} \mathbf{A}^\intercal\mathbf{b} \\ 0 \end{bmatrix},$$

where z is the Lagrange multiplier, solving this system gives us the optimal parameters Θ^* for all the k pieces in a cross-section.

Adaptive Splitting

It is important to ensure that a simpler piece (such as a straight line) is represented by fewer points so that more points can be assigned to a piece with many sharp turns. Uniformly distributing points between all pieces does not ensure this; hence, we propose an adaptive splitting scheme for non-uniform distribution between pieces. We use the Douglas Peucker polyline simplification algorithm [46] for finding a set of endpoints to generate the pieces within the curve. The

Douglas Peucker algorithm is run for multiple iterations, and in each iteration, we reduce the threshold to increase the number of endpoints returned each time. If the final number of unique endpoints returned is more than k, we select the k points with maximum absolute angle, where the angle varies between -90 and $+90$. Once we have obtained k pieces, we fit piece-wise polynomials as discussed in section 6.2.1.

6.2.2 Training on parametric space

We take the ShapeNet dataset [26] and use the manifold meshes. The input cross-sections are generated using mesh-plane intersection provided by Trimesh [39], and we further convert these to a parametric representation. Further in the text, *cross-sections* shall refer to the parametric representation of cross-sections. We sample surface points from the meshes; thus, each set of cross-sections and point clouds forms the input and the corresponding ground truth for the network. In order to use parameters with a neural network, there are certain properties that the operations on the parametric representation must possess. Each piece of a cross-section is represented as a tensor in $\mathbb{R}^{6\times3}$ of coefficients of the parametric representation $f_j(t)$ of degree 5 ($p = 5$) in $\mathbb{R}^3$. Further, we propose that any operation performed on these coefficients must hold permutation invariance in both piece-level and cross-section-level operations.

Permutation Invariance and Neighborhoods

We represent the coefficients of the parametric representation as a vector for the neural network to act on. Thus, the cross-sections are represented as tensors containing the vectors for each parametric piece. Further, the cross-sections contain neighborhood information in the form of the adjacency of the pieces.

Therefore, the operations that we perform on the cross-sections must be permutation invariant since any combination of cross-sections represents the same object. Given a set of m parameterized cross-sections, where each cross-section is partitioned into k pieces with the coefficient matrix Θ_l of the parametric functions for the l^{th} piece, the full set of stacked coefficients for the entire set of cross-sections is represented as the tensor $\mathscr{C} = \left[\Theta_1, \Theta_2, \cdots, \Theta_m\right]^\intercal$ of size $m \times (p+1)k \times 3$.

Any permutation of rows of $\mathscr{C}$ still represents the same set of cross-sections (that is to say, the cross-sections can come in any order). Therefore, any operation performed on $\mathscr{C}$ should ideally yield the same result irrespective of the ordering of its rows. Similarly, at the level of each cross-section, we have k pieces spliced together, and any circular permutation of these pieces represents the same cross-section. Therefore, we need another level of permutation invariance for the representation at the level of each cross-section. Within a neural network,

representations are created using matrix multiplications, and different orders of the rows and columns of $\mathscr{C}$ would produce different results since,

$$W^\intercal \mathscr{C} \neq W^\intercal S'(\mathscr{C}),$$

where W is a weight matrix and S' is a shuffle operation. Therefore, we do away with this matrix-based representation. We create a graph-based representation using the piecewise parametric representation, which we explain further.

We note that each cross-section has some adjacency information since the pieces of a cross-section are arranged in linear order along the contour. In order to use the neighborhood properties, we propose a graph-based representation, where each node is represented as the matrix of coefficients of a piece of the parametric curve, and each edge denotes the adjacency. The graph-based representation allows our approach to take into account the desired permutation invariance while enabling us to use the additional adjacency information as needed. We further use a 2-level attention mechanism while changing the adjacency for piece-level and cross-section-level learning. Our final representation is thus in terms of the coefficients of the pieces, where the adjacency matrix stores the piece-level relations.

Learning Point Cloud Representation

Graph neural networks suffer from the problem of over-smoothing due to repetitive aggregation operations. In our experiments, we initially experimented with a learnable decoder that was trained jointly with the graph encoder; however, that did not yield very detailed results. Therefore, to circumvent this problem, we found that a pre-trained model is much more effective. We train a point cloud autoencoder trained on the ground truth point cloud and then use the encoder embedding from this as the ground truth embedding. Thus, the objective of the graph encoder is to learn a similar embedding from the cross-sections in order to produce a similar point cloud from the pre-trained decoder, as shown in Figure 6.4. Thus, the generator tries to produce an embedding that is similar to the embedding generated by the point cloud encoder for a given shape.

Cross-Section Attention

Attention mechanism [224] allows the network to focus on different features and enables a better learning setting for the network. Taking inspiration from this, we introduce attention at two levels in our network for learning shapes. We propose a cross-section attention mechanism, which we call global attention, and a piece-wise attention mechanism for focusing on local information. Each cross-section contains different amounts of information pertaining to the geometric shape of the object. Some cross-sections carry more salient features about the entire

object than others. Similarly, within a cross-section, some pieces contain more information pertaining to the local regions, such as regions of high curvature. In order to focus on such regions, we introduce local attention, which attends to each piece within a cross-section.

The global and local attention are computed using Graph Attention [225]. The normalized attention coefficient at the graph level can be expressed as

$$\alpha_{i,j} = softmax(e_{i,j}),$$

where $\alpha_{i,j}$ are the normalized attention coefficients for node i in the graph, $j \in \mathcal{N}_i$ where $\mathcal{N}_i$ is the neighbourhood of node i and $e_{i,j}$ is the attention coefficient. The attention coefficient is calculated using the same method as described in [225].

First, attention is computed over the pieces of each cross-section to compute the local attention. Then we aggregate the piece-level representations into a single vector to represent each cross-section node. Finally, we apply the cross-section level attention for which we create a new adjacency matrix representing a complete graph. Since at the cross-section level, there is no strict adjacency, representations for each cross-section must be learnable. We let the network pay attention to the complete graph, giving it complete flexibility to attend to any cross-section. We still need to maintain the graph-level representation at this stage since we still require permutation invariance at this stage.

In our implementation, in order to restrict the attention to piece-level and cross-section level, we explicitly pass the piece-level adjacency matrix during initial graph convolutions; this restricts the neighbourhood of the nodes to within cross-sections, after which we aggregate the piece-level information and replace the adjacency matrix with a complete graph adjacency. We use PyTorch Geometric [49] for this.

Adapting for Variable Cross-sections

Since the network takes the input in the form of parametric cross-sections, where each cross-section consists of piecewise C_1 parametric curves, the parameters of the network become fixed during training if MLPs are used, prohibiting any changes in the number of cross-sections or pieces provided. In order to adapt to the variable nature of our data, we are further motivated to use the graph-based representation by allowing piece-level aggregation and cross-section level aggregation, which allows for a variable number of cross-sections to be provided to the network. Furthermore, we cannot use convolutions directly in the parametric space because convolutions are not well defined on coefficient spaces. Since the parameters across pieces belong to the same cross-sections and different permutations of cross-sections can belong to the same object; thus, any permutations of cross-sections and pieces should yield the same object. Graph convolutions can

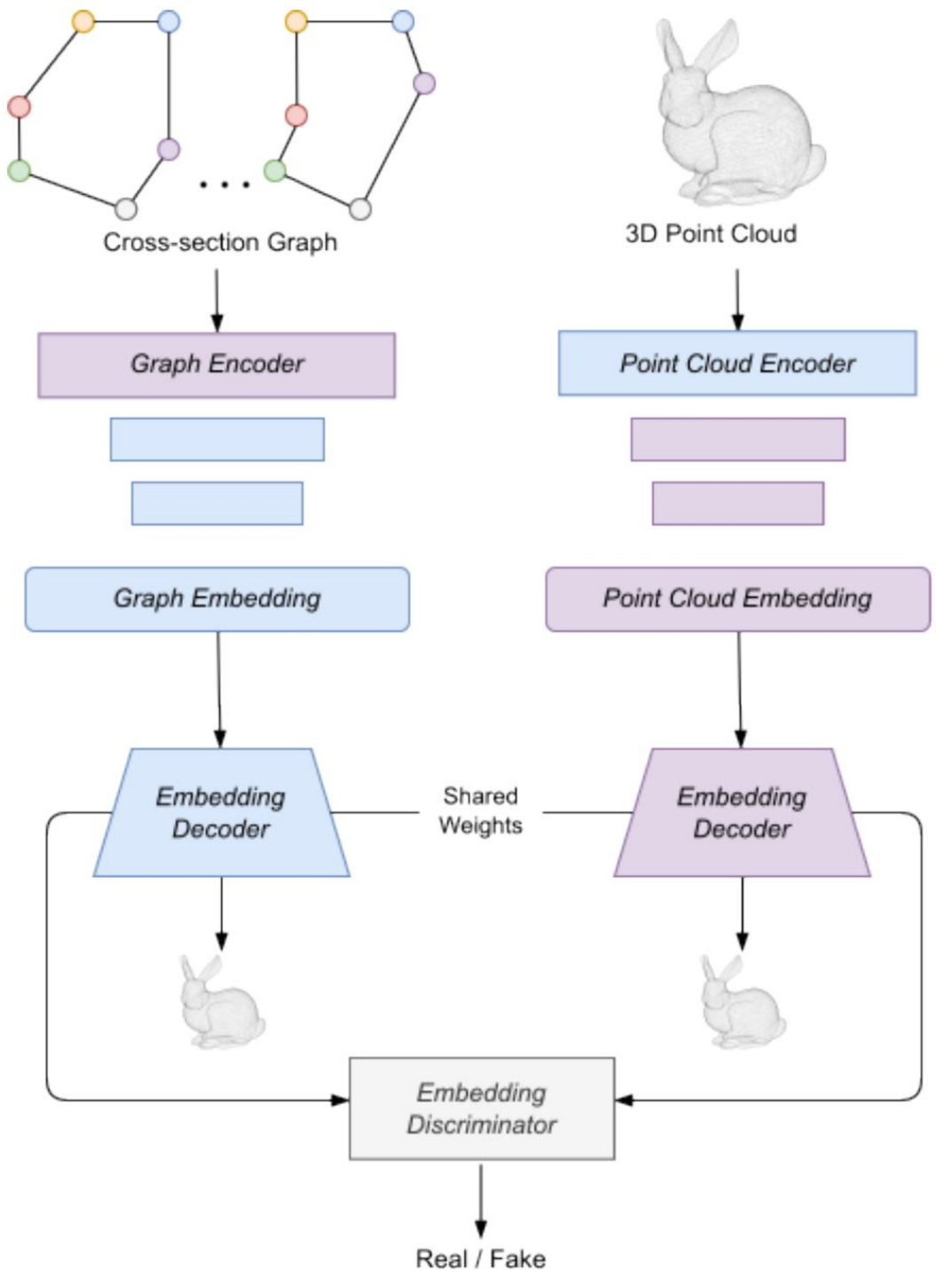


Figure 6.4: During training, the graph embedding decoder tries to generate an embedding that is similar to the point cloud embedding generated from the pre-trained encoder. This representation is then used by the decoder to generate the point cloud of a relevant shape.

potentially handle a variable number of nodes and allow for permutation invariance, giving the highest flexibility, making them most suitable for our problem setting.

We use graph convolutions in both the generator and discriminator. The discriminator is conditioned using the input graph parameters and predicts whether the generated embedding vector is real or fake. The input graph is converted to a graph-level embedding using successive graph convolutions [110] and aggregation. Then the embedding vector is concatenated with the generated embedding and passed to subsequent layers. While the generator consisted of SAGEConv [64] followed by DiffNorm [274] to prevent over-smoothing and allow for deeper networks and Graph Attention Convolutions [225] followed by aggregation and fully connected layers to generate graph embedding. In order to allow for stochasticity in the generated outputs, like in a general GAN setting, we append noise to the parameter vector of each piece.

Training Details

Given a pre-trained autoencoder with encoder En and decoder De and a GCN-based generator-discriminator pair $\{G, D\}$, we pass a ground truth point cloud P through the encoder to generate an embedding, $F_{gt} = En(P)$. For a set of parameterized cross-sections C obtained from the input point cloud, we create the piece-wise adjacency matrix for each cross-section and a cross-section adjacency matrix denoted by p_{adj} and c_{adj}, respectively. The generator is trained to generate an embedding using the cross-section set C and the two adjacency matrices for the point cloud. The generator loss is given by,

$$\begin{aligned}\mathscr{L}_G = &\log\left(1 - D\left(G\left(C, p_{adj}, c_{adj}\right), C, p_{adj}, c_{adj}\right)\right) + \\ &\mathscr{L}_{ch}\left(De\left(G\left(C, p_{adj}, c_{adj}\right)\right), De\left(F_{gt}\right)\right) + \\ &\mathscr{L}_{mse}\left(G\left(C, p_{adj}, c_{adj}\right), F_{gt}\right),\end{aligned}$$

where $\mathscr{L}_{ch}$ is the Chamfer loss between the point clouds generated using the embedding estimated by the generator and the embedding of the ground truth point cloud. $\mathscr{L}_{mse}$ is the mean squared error between the embedding estimated by the generator and the embedding of the ground truth point cloud. The discriminator loss can be formulated as,

$$\mathscr{L}_D = (1 - \log(D(G(C, p_{adj}, c_{adj}), C, p_{adj}, c_{adj}))) + \log(D(x, C, p_{adj}, c_{adj})),$$

where the discriminator is conditioned on the input cross-section graph. The generator and discriminator are trained adversarially as suggested in [57].

6.3 Results and Discussion

We perform tests on different classes of the ShapeNet dataset. We perform an experimental procedure similar to DeepSDF whereby owing to the vast dataset and restricted compute budget at the time, objects of each class are divided into train and test split; we divide the models for a given class into *known* shapes, i.e., shapes that were in the training set and testing set, referred to as *unknown* shapes.

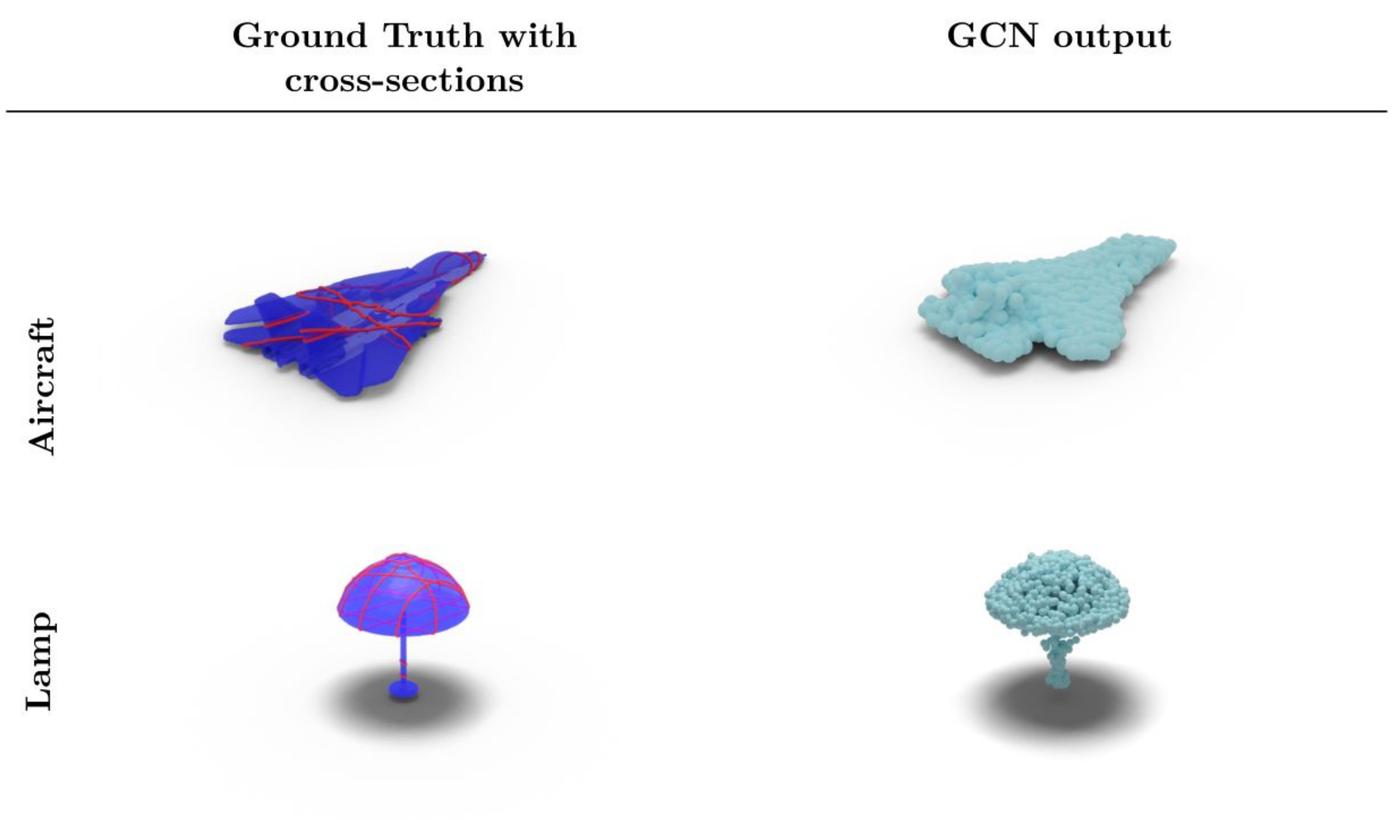


Figure 6.5: Results of the proposed model trained only on a single class of objects. Input to the network is the set of cross-sections (red) belonging to the ground truth mesh(blue).

In order to test our method, we first experimented with single-class reconstruction for two classes by training on each class separately. We perform training on the class of *airplane* and *chair*, and results are shown in Figure 6.5. We further extend the training in a multi-class setting as well. For the multiclass setting, we test on 4 classes - *airplane* (4K models), *chair* (6K models), *lamp* (2K models), and *sofa* (3K models). We do not perform any class balancing techniques and directly train on the ShapeNet dataset and observe the following results.

6.3.1 Cross-section dependence

The number of cross-sections provided as input has a vital control on the output of the generated point cloud surface, as can be seen from Table 6.1. We show some test outputs of the proposed model trained on four classes: Airplane, Chair, Lamp, and Sofa with a different number of input parameterized cross-sections in Figure 6.6. The first column displays the ground truth mesh used to sample the

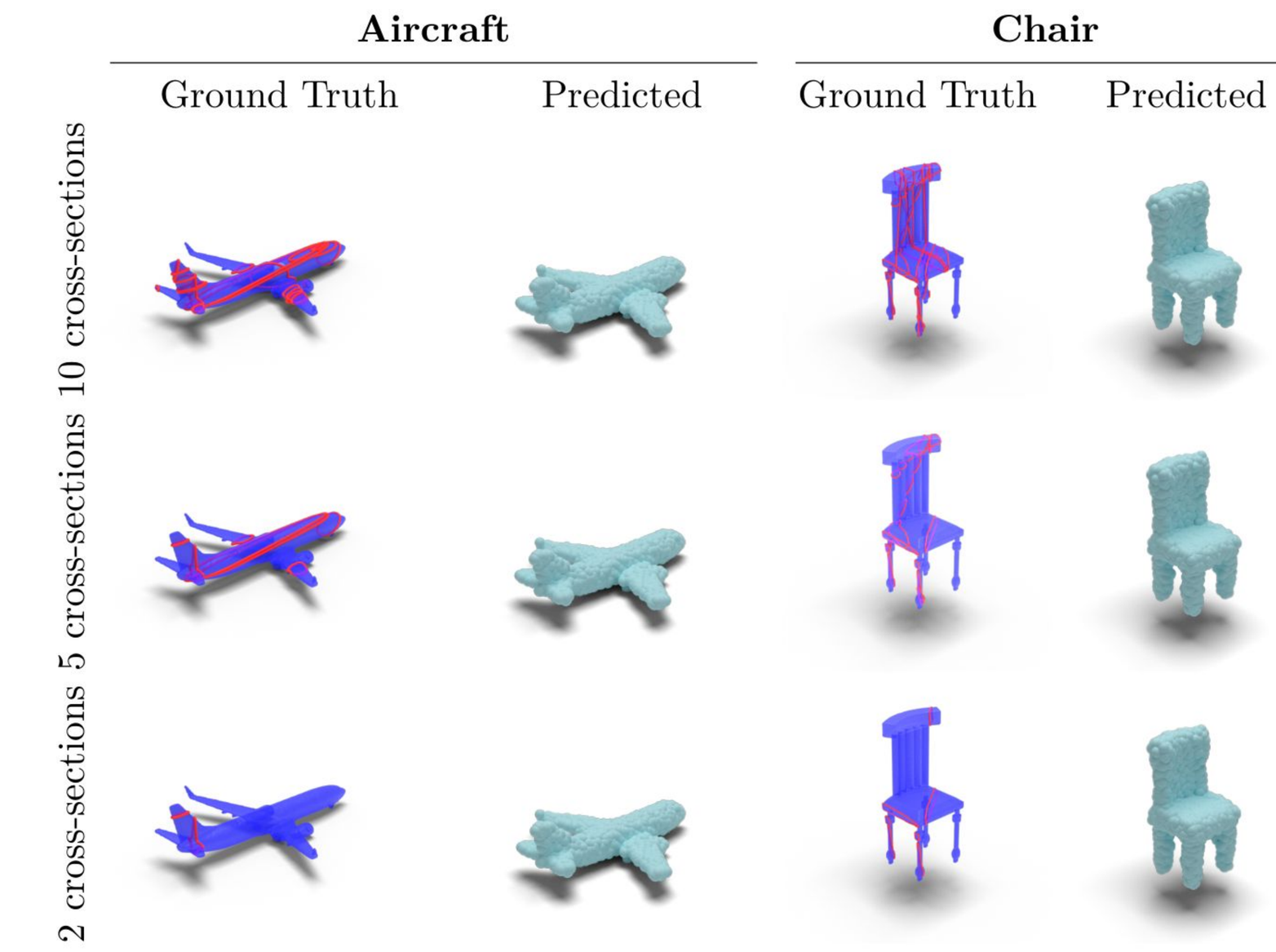
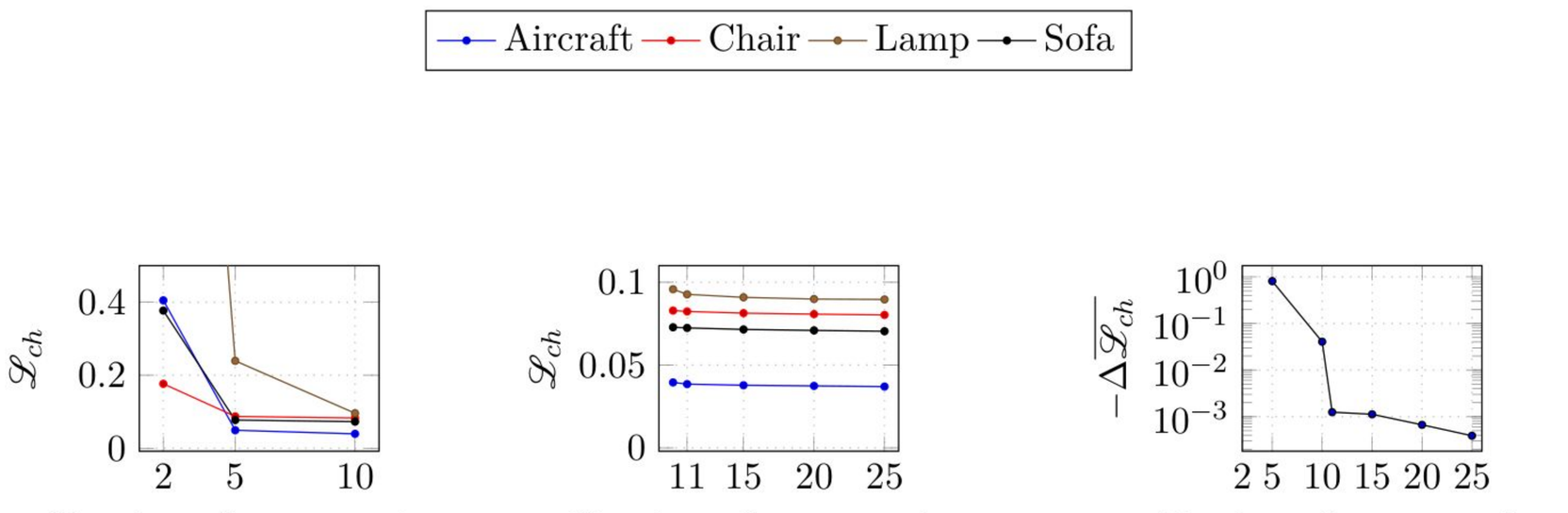


Figure 6.6: Comparison of reconstruction quality with an increasing number of cross-sections. Input to the network is the set of cross-sections (red) belonging to the ground truth mesh(blue).



(a) (b) (c)

Figure 6.7: Dependence of Chamfer loss $\mathscr{L}_{ch}$ on the number of cross-sections. (a) Undersampling conditions, (b) Oversampling conditions, (c) Negative change in mean Chamfer loss (log-scale).

ground truth point cloud with cross-sections in red; the second column shows the reconstruction using our method. We also analyze this variation in Chamfer distance between the generated and ground truth point surfaces as the number of input cross-sections increases/decreases in Figure 6.7(a), 6.7(b). We trained the network by randomly sampling 5-10 cross-sections out of 100 cross-sections, which were generated from the model in each batch, to make the training robust. To test the generalizability, we examine results in both undersampling, i.e., cases where the number of cross-sections is less than 5, and oversampling conditions, i.e., when more than 10 cross-sections are given. We start by slightly increasing the number of cross-sections by one, and then also test by regularly increasing by five cross-sections to see how the loss varies. In Figure 6.7(a), we see the case for undersampling. As expected, we observe that as the cross-sections are reduced, the amount of information supplied to the network reduces; hence, the reconstruction quality degrades. However, as in Figure 6.6, despite the sharp reduction in the number of cross-sections, the network still generates a reliable shape and is able to distinguish between parametric forms. In some cases, the failure of reconstruction is much higher depending on the number of samples of a particular shape of the object the network sees and the kind of cross-sections supplied. For example, in Figure 6.10, in an airplane object, the cross-sections do not contain sufficient information, leading to a completely different object being created, though it is noteworthy that the class of the object reconstructed does seem correct visually. We also note in Figure 6.7(a), that there is a sharp dip in Chamfer loss when the cross-sections are increased to 5, which is the minimum number of cross-sections on which the network is trained. However, further increasing the cross-sections to 10 does improve the reconstruction visually. In terms of empirical values, change is not very significant, and most changes occur in the order of $1e-3$, as can be seen in Figure 6.7(c), which shows the case for oversampling when cross-sections are increased in the range of $15-25$.

In case of oversampling 6.7(b), we again observe a sharper dip till 15 cross-sections. After 15 cross-sections, we stop observing much change in the Chamfer distance of the cross-sections; however, the Chamfer distance continues to go down as the information supplied to the network increases.

6.3.2 Attention

Further, with the help of graph attention [225], we can use the thus obtained attention weights to backtrack the relationship and relative importance that each cross-section plays for every other cross-section in the complete graph. We show the performance boost attained from attention mechanisms in Section 6.4, where we empirically show that both piece-level and cross-section-level attentions play an important role in the reconstruction.

# cross-sections	Per-class Chamfer distance				Mean
	Airplane	Chair	Lamp	Sofa	
2	0.4050	0.1765	2.7306	0.3770	0.9223
5	0.0493	0.0872	0.2394	0.0772	0.1133
10	0.0395	0.0829	0.0958	0.0728	0.0728
11	0.0385	0.0824	0.0927	0.0724	0.0715
15	0.0378	0.0813	0.0909	0.0715	0.0704
20	0.0374	0.0807	0.0898	0.0709	0.0697
25	0.0370	0.0803	0.0896	0.0704	0.0693

Table 6.1: Per-class Chamfer Distance corresponding to the variation in the number of cross-sections (results for both undersampled and oversampled cross-sections are shown for a model trained on all aforementioned classes.

6.3.3 Changes in Embedding

Since the graph embedding also varies as the number of cross-sections changes, we try to reason how the class information of the embedding varies as we increase the number of cross-sections. In order to do so, we compute the t-SNE [223] from the embedding vectors of 200 samples of each class in the multi-class setting and plot them to see how the cluster distance varies with cross-sections. From Figure 6.8, we observe that in the undersampling conditions, the overlap between the clusters increases, thus making it difficult for the network to predict the class of the object, whereas as the number of cross-sections is increased, the overlap of clusters reduces. However, we also note that some shapes exhibit a well-separated cluster even when the number of cross-sections is less, such as *airplane* in this case. We also note that classes *chair* (denoted by green) and *sofa* (denoted by grey), since these classes are similar in shape, show weaker disentanglement in the embedding space of cross-sections.

6.3.4 Edge Cases

We observe that in the case of a failure, the network reconstructs a simple object of the class. However, despite it being a failure case, the class of objects is still distinguishable by the network. Further, we also notice a deterioration in the samples containing holes, such as chairs and lamps. In Figure 6.10, we can see such samples; the reconstruction is not accurate in the case of airplanes. For example, in the case of a chair, the reconstruction does not accurately maintain the genus of the object for some samples.

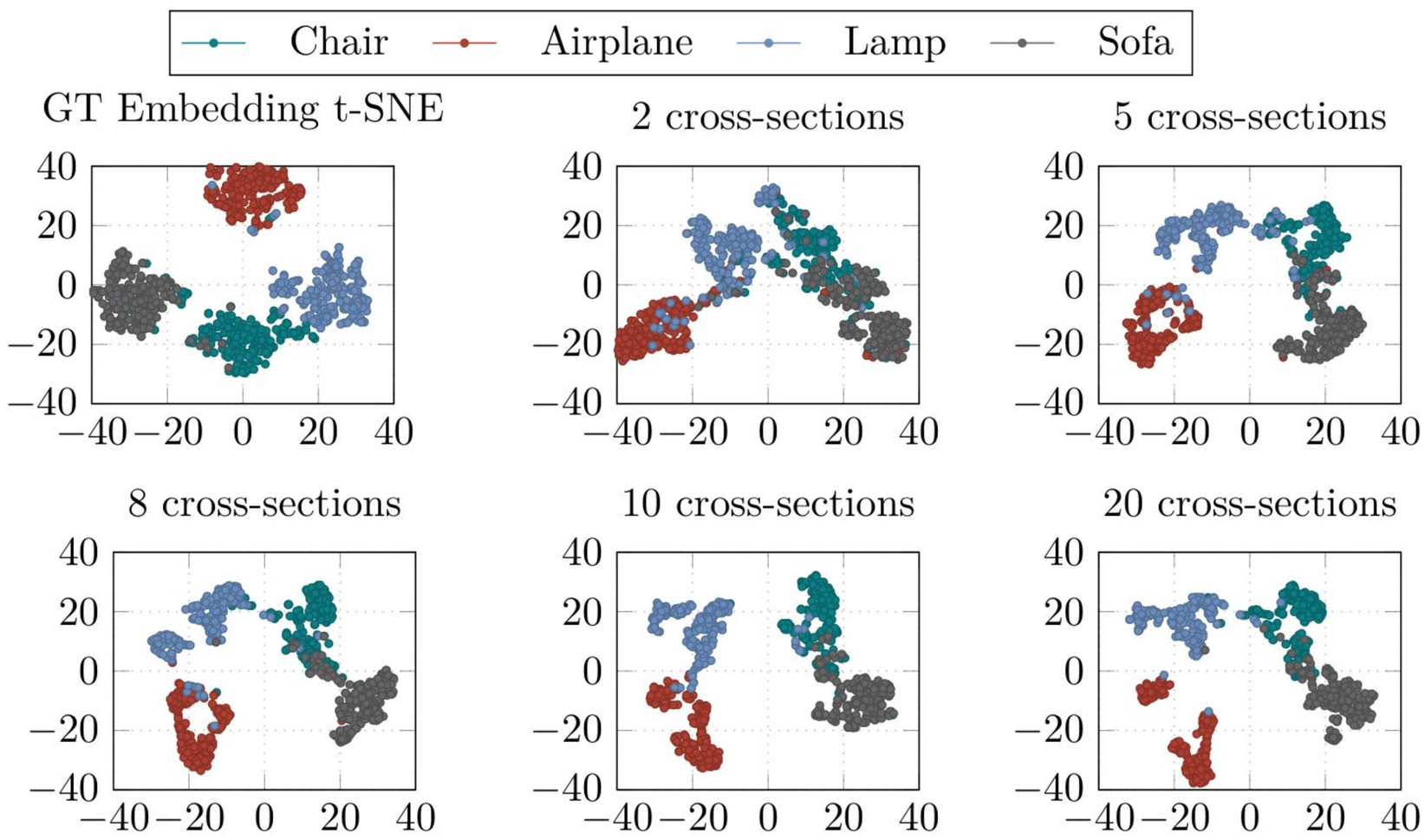


Figure 6.8: t-SNE plots of the embeddings obtained from the ground truth complete point cloud, followed by t-SNE plots obtained from the embeddings generated by GCN from the cross-sections.

6.4 Ablation Study

In order to test the impact of attention and connectivity in the network at different levels, we train multiple networks with different configurations and summarize the findings here. To identify the impact of attention, we evaluate different combinations of piece-wise attention and cross-section-level attention to see the change in network training. In order to test the changes due to connectivity, we change the connectivity of the pieces from a ring to a complete graph adjacency.

Approach	**Chamfer Loss (mean)**
Change in loss with different attention strategies	
Without Attention	9.02284977
Attention Cross-section level only	0.33693247
Attention Piece level only	1.10052492
Change in loss with different connectivity	
Complete Graph at attention layers	5.80743287
Complete Graph from first layer	0.86106642
Maintaining Ring Adjacency	9.57172529

Table 6.2: Ablation results with varying attention mechanisms and connectivity.

The networks were trained for 100 epochs with a slightly different configuration of the generator, increasing the number of generator parameters and making the network dense. We observe that the network has a reduced performance since

graph neural networks don't work well for dense configurations, and we further test different configurations of attention and connectivity on the same network to observe the impact of Attention layers for each experiment; the attention layer was replaced by the SAGEConv layer [64].

We compare the mean Chamfer loss obtained across the different classes for a different number of cross-sections (5, 10, 11, 15, 20, and 25) provided as input in Table 6.1. We observe that the attention module plays a key role in training and leads to much lower loss as compared to a network without any attention module. Further, we also observe that both cross-section and piece-level attention units play a crucial role (refer to Table 6.2).

We test the impact of piece-level adjacency at different levels of the network. We change the adjacency between the nodes in 3 different ways. Firstly, we use ring adjacency strictly for the nodes, and secondly, we change the adjacency at the higher levels. This is done with the intuition that the network should be able to view all the nodes during the attention operation, since nodes that are similar would share similar embeddings at higher levels, given that they share similar local neighborhoods. Thirdly, we provide a complete graph adjacency throughout all the levels and allow the network to deduce the relationships itself. Table 6.2 summarizes the results for different connectivity information that is fed to the network. We observe that despite providing the exact neighbor information in the form of ring adjacency, the network performs best for a complete graph adjacency; we suspect that this is due to better information capture since each neighbor of a node is no longer restricted to the adjacent nodes in the cross-section but can also accumulate information from nodes that are not directly linked allowing the network to get a better information gain thus reducing the impact of information bottleneck that graph neural networks suffer from as the depth of network increases.

6.4.1 Variance in Loss

Further in Table 6.3, we show the standard deviations for the losses across 20 different objects for varying cross-sections. We sample $n \in \{2, 5, 10\}$ cross-sections, randomly from the 100 cross-sections generated from the mesh as in our original experiment setup, and repeat this 40 times, thus obtaining 40 random samples for each object for a given n. We do this to measure the standard deviation for the losses computed for varying sampling of cross-sections. We first compute the standard deviation obtained across different samples for the same object for a given value of n. Then we compute the mean over the standard deviations obtained across different objects for the given n. This helps us to understand further the variations in loss across cross-sections for each object.

We observe in Table 6.3 that, similar to the reduction in mean Chamfer Losses in Table 6.1, the standard deviations vary across classes and across the number

No. of cross-sections	**Airplane**	**Lamp**	**Sofa**	**Chair**
2	0.09262	0.6465	0.0634	0.1131
5	0.01925	0.1060	0.0165	0.0243
10	0.00675	0.0253	0.0105	0.00985

Table 6.3: In the above table, we demonstrate a similar trend of reducing standard deviation in the chamfer losses obtained for varying cross-sections for 20 different objects sampled 40 times each.

of cross-sections. If the mean for a particular class is low, so is the variance in the loss since the model is able to faithfully reconstruct the shape given any given set of cross-sections. Further, as the cross-sections increase, the amount of information provided to the network increases; thus, the standard deviations reduce across the same object sampled multiple times.

6.5 Comparisons

This is a non-trivial task for many reconstruction methods, as they often struggle to generate missing structures from limited cross-sectional information. Consequently, our approach holds practical value, as it excels in accurately capturing the shape of objects in scenarios where cross-section data is sparse.

We compare our method against 4 methods: VIPSS [81] method for variational surface reconstruction from cross-sections, surface reconstruction from non-parallel curve networks [134], a state of the art deep learning based method P2P-Net [260] and the recent OReX [200] and show results in Figure 6.9. Most of these methods suffer from holes and instabilities for sparse cross-sections; therefore, to be fair, we sample more cross-sections in those cases. However, we restrict our method to 10 cross-sections. VIPSS, OReX, and Liu's method require careful sampling and sometimes tend to fail randomly for sparse cross-sections. We show the best-case results for these methods. VIPSS is very sensitive to λ and requires a large number ~ 80 of cross-sections for faithful reconstruction due to failure due to openness in cross-sections; however, we still notice artifacts.

We checked the reconstruction with the method proposed in [134]; however, the available implementation discards many cross-sections that lead to incorrect results. We show results for the cases where we did not observe this issue for a fair comparison. For OReX [200] as well, we observe that it performs really well when cross-sections are dense; however, it fails in the case of sparse cross-sections. Therefore, for some samples, we show results in cases where it performs reasonably well.

We further compare our method against a state-of-the-art deep learning-based method called P2P-Net. We modify P2P-Net and train it on points sam-

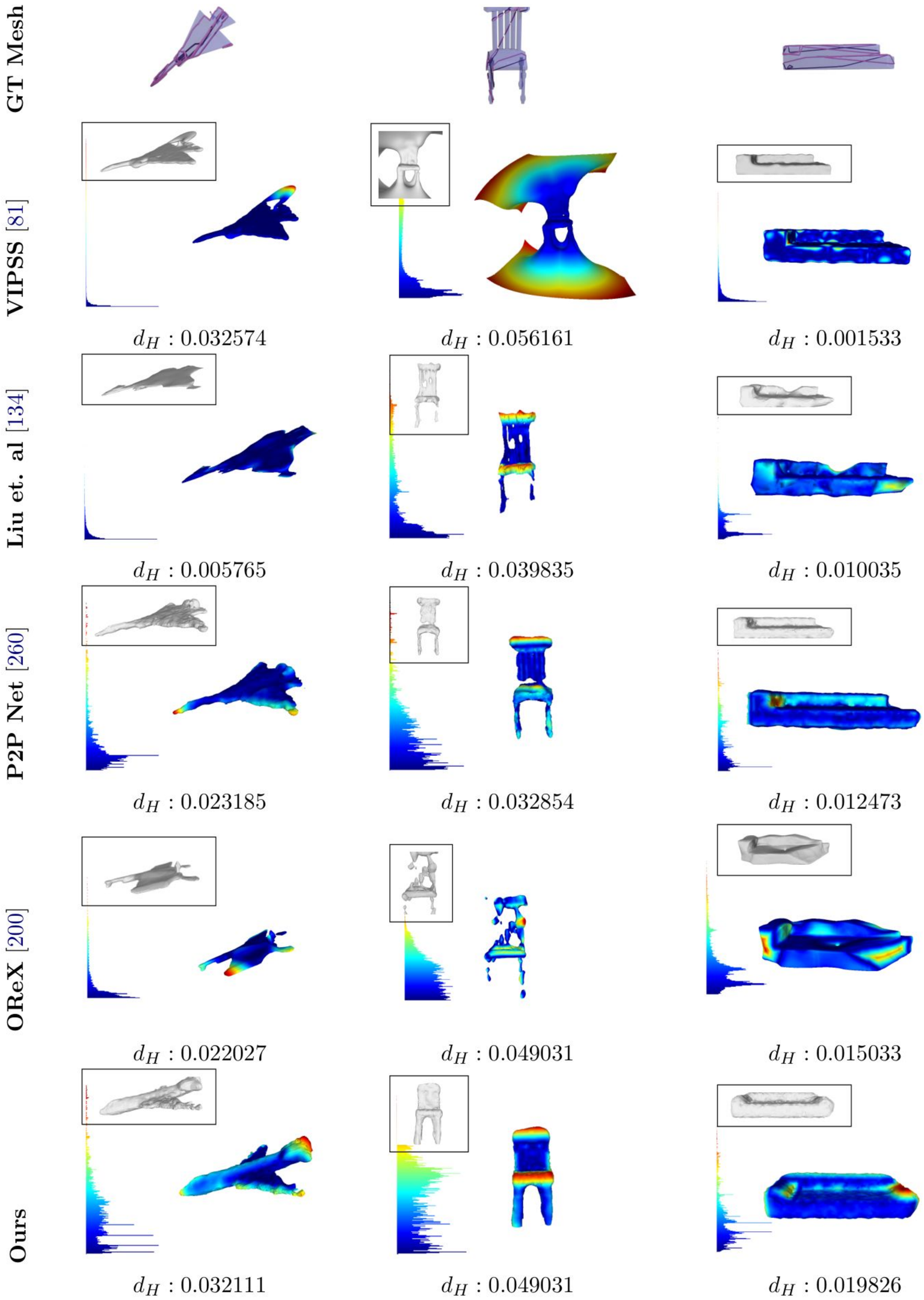


Figure 6.9: Comparison with the state-of-the-art methods. The inset shows point-wise surface error compared with the GT. The blue regions indicate areas of low error, while brighter areas indicate higher error.

pled from our cross-sections. We notice that in some cases, despite performing better in terms of metrics, there are still completion issues in several samples, such as the chair shown in Figure 6.9. Our method generates symmetric structures leading to higher loss value but better perception quality and semantically correct different structures, such as the right-hand rest of the sofa and a missing leg in the chair. This also highlights a weakness of our method pertaining to the lack of strict adherence to the cross-sections since our method relies on embedding decoded by the pre-trained decoder. However, we believe that this can be circumvented by better pre-training schemes since the performance of the pre-trained decoder forms the lower bound of the reconstruction error and can be swapped with any of the better-performing point cloud generators.

In order to visualize the error in reconstruction from our method and P2P-Net, we perform surface meshing of our resulting point cloud with Poisson Reconstruction [102], by computing normals from the ground truth mesh for the best-case scenario. For VIPSS, we also modified the method and provided normals from the GT mesh. We show similar histograms for the surfaces obtained from other methods. We also note the Hausdorff distance (d_H) obtained for different methods in Figure 6.9. We notice that during the generation of the point cloud, since our method does not have hard constraints for precise overlap with the input, the shift in the point cloud can lead to a relative rise in the Hausdorff distance, as can be seen in the case of the chair in Figure 6.9. However, it outperforms the other methods in both qualitative and quantitative comparisons in several cases.

6.6 Failure Cases

This is a non-trivial task for many reconstruction methods, as they often struggle to generate missing structures from limited cross-sectional information.

Ground Truth	Predicted	Ground Truth	Predicted

Figure 6.10: Failure cases resulting in incorrect shapes. Input to the network is the cross-sections (red) belonging to the ground truth mesh(white).

In some cases, the failure of reconstruction is much higher, depending on the number of samples of a particular shape of the object the network sees and the information in the cross-sections supplied. For example, in Figure 6.10, in an

airplane object, the cross-sections do not contain sufficient information, leading to a completely different object being created, though it must be noted that the class of the reconstructed object is correct. We observe that in the case of a failure, the network reconstructs a coarse object of the correct class.

Further, we also notice a deterioration in the samples containing holes, such as chairs and lamps. For example, in the case of the chair, the reconstruction does not accurately maintain the genus of the object for some samples.

6.7 Conclusion and Future Scope

With this work, we open a new direction for the exciting domain of cross-section-based reconstruction. We generate a new dataset that can be used for multiple tasks. The ability to use parametric cross-sections directly in a learning-based setting exempts the use of any sampling-based restrictions in deep learning-based methods. The complete information of the curve is encapsulated in the coefficients of the parametric representation. Further, we utilise GCNs at scale and demonstrate their effectiveness for parametric curves and the ability of the GCNs to capture neighborhood information, which helps deduce better relationships among the cross-sections, adding to the explainability. We show empirical evidence to analyze the changes in reconstruction, both in terms of the embedding space representation and point cloud reconstruction, to understand the changes with respect to the variation in the amount of information provided to the network. This builds a strong motivation and opens up the field to further research. We also encounter some edge cases and proposed improvements in making the method adapt better to genus > 0 objects. Another research direction could be better disentanglement of latent features, even when sparse cross-sections are supplied, an information-theoretic aspect of cross-section-based reconstruction in a deep learning-based setting, some of which we hope to cover in future works. To the best of our knowledge, no prior work looks at cross-section-based reconstruction from this perspective and opens up multiple avenues of research.

Chapter 7

RLQuad: Policy Gradient Optimization for Score Distillation based 3D Generation

3D generation has rapidly accelerated in the past decade owing to the progress in the field of generative modeling. Score Distillation Sampling (SDS) based rendering, combined with Gaussian splatting, has improved 3D asset generation to a great extent. We identify two key improvements that could further impact the generation in text-to-3D and image-to-3D settings. Firstly, we observe the recent work of Denoising Diffusion Policy Optimization (DDPO), which demonstrates that the diffusion process is compatible with policy gradient methods and has been shown to improve the 2D diffusion models using an aesthetic scoring function. We first show that this aesthetic scorer acts as a strong guide for a variety of SDS-based methods and demonstrates its effectiveness in text-to-3D synthesis. Further, we leverage the DDPO approach to improve the quality of the 3D rendering obtained from 2D diffusion models. Our approach, Denoising Diffusion Policy-based 3D Splat Optimization (DDPSO3D), employs the policy gradient-like method. It works in tandem with the aesthetic scoring model employs a pre-trained policy model as a critic and optimizes the splats directly. To the best of our knowledge, this is the first method that extends the policy gradient method to optimize the Gaussian splitting thus extending the policy gradient to optimize the input state. Our method relies on 3D score-based rendering and shows improvement across SDS-based methods such as DreamGaussian, which are currently driving research in text-to-3D synthesis. Secondly, we focus on a new splitting scheme that takes into account the image information with the help of an adaptive novel quadtree-based splat optimization, which helps to capture higher details and leads to a more complete structure in a sparse setting. We show its effectiveness for both text-2-3D and image-2-3D tasks.Our project page can be accessed via https://ddpo3d.github.io.

The intersection of deep learning and graphics has experienced significant advancements in recent years, unveiling novel possibilities for advancing our understanding of visual data [12, 13, 155, 262]. The combination of these two disciplines has given rise to innovative approaches that transcend traditional boundaries, unlocking new domains of exploration. One such frontier lies in bridging the gap between vision and language, a pursuit that holds the potential to transform how we interpret and engage with the visual world.

Recently, the advent of diffusion models [189, 196, 206] has played a pivotal role in extending the capabilities of this interdisciplinary field, enabling the seamless translation of textual descriptions into 2D representations. However, as we dive into the realm of three-dimensional (3D) reconstructions, a notable challenge emerges. Despite the development made in 2D translation, the transition from text to 3D remains a formidable task, primarily attributed to the scarcity of comprehensive 3D datasets. This scarcity not only hinders the training of robust models but also underscores the pressing need for innovative methodologies to overcome the inherent challenges in this dimensionally complex domain.

Diffusion models have demonstrated effective and high-quality generation of images, audio, video, and several other modalities. Beyond their initial training objectives, these models have surpassed their limitations and evolved into fundamental frameworks, extending their utility to a range of downstream tasks, including but not limited to image composition, inpainting, and image editing, as demonstrated in notable works like [100, 141, 159, 249].

Recently, pre-trained large diffusion models have been utilized for 3D shape generation techniques due to the advantages that they offer in terms of the vast knowledge base that they serve as, which can be distilled into neural radiance fields (NeRFs) for rendering [179, 184, 228]. Recent techniques such as score distillation sampling and variational score distillation have demonstrated how effectively these models can be leveraged to generate implicit 3D representations called NeRFs, allowing for fast, direct, high-quality 3D renderings from text prompt input [179, 228, 233] or single image [135, 149] and now even multi-image based generation [184].

DDPO [18] adopts a reinforcement learning framework to treat the diffusion process as a multi-step decision-making process, enhancing the image generation capabilities of pre-trained 2D diffusion models. In alignment with these advancements, our research endeavors to enhance the visual quality of 3D rendering techniques, specifically focusing on aesthetic improvements. We propose the integration of a DDPO-based policy gradient approach into the score distillation sampling process, aiming for better results in 3D rendering. This approach not only facilitates the refinement of aesthetic aspects but also allows for the incorporation of non-differentiable rewards, enabling diverse optimization strategies. Furthermore, we propose an additional Gaussian densification technique that takes into account the rendered image information and performs adaptive gen-

eration of Gaussians as compared to the original cloning-based generation. The main contributions of our work can be summarized below.

1. We introduce DDPSO3D, an adaptable framework designed for seamless integration with any SDS-based method and its derivatives in the field of 3D rendering.
2. We showcase the efficacy of DDPSO3D through both qualitative and quantitative assessments, showcasing improvements in CLIP scores. These enhancements are observed when our framework is integrated with existing SDS-based rendering methods, including DreamGaussian [218], GsGen [31], and Dreamfusion [179].
3. Propose a novel image-aware Gaussian splitting scheme using quadtree-based Gaussian generation which allows for adaptive densification strategy.

7.1 Related Work

Our work primarily relates to the 2D diffusion models and the subsequent derivative methods for generating 3D shapes.

7.1.1 NeRFs

One of the seminal works by Mildenhall [155] introduced novel view synthesis by using shallow MLPs and allowing them to overfit on a set of views conditioned on the respective camera poses. The input to such a network is a 5D point (position (x, y, z) and viewing direction (θ, ϕ)), and the output is an implicit field consisting of RGB color and density σ at any point. For rendering a view after the optimization, a volume raymarching process evaluates the MLP at various ray positions to compose an image. Several variants have evolved focusing on anti-aliasing, unbounded scenes demonstrated by Barron et al. [12, 13], reflections by Guo et al. [63], and stylization by Huang et al. [80]. NeRFs have been further extended to representations other than MLPs as well, such as voxel-based representations [111, 198], tensor-based representations such as Hexplane [24], K-planes [51], and TensorRF [27]. Recently, NeRFs have been used for generative modeling with diffusion-based rendering, as further discussed.

Gaussian splatting

Kerbl et al. [104] present a technique to enable fast high-resolution novel view synthesis using Gaussian Splatting, which has emerged as a faster alternative for NeRF-based approaches. The technique first generates a sparse point cloud from

provided multi-view images using Structure-from-Motion (SFM) and uses that for initializing 3D Gaussians that are differentiable and allow for easy projection to 2D splats, resulting in fast α-blending for real-time performance. These methods have also been extended beyond 3D owing to their speed and have been used for learning dynamic scene representation. Wu et al. [241] explore Gaussian splatting for dynamic scene rendering. They propose a pipeline in which they use the centers of the Gaussians and use the time step to query the multi-resolution voxel planes. Luiten et al. [142] have further performed 6-DOF tracking along with view synthesis. They allow for Gaussians to be transformed under persistent color, opacity, and size. While these methods learn from a set of images and generate novel views, there have been several works in the realm of generative modeling that have also been leveraged for the same.

7.1.2 Diffusion Models

Taking inspiration from non-equilibrium statistical physics, Sohl et al. [208] introduced a novel deep unsupervised learning approach aimed at capturing data distributions. Their method involves an initial disruption of the inherent structure in the data distribution through an iterative forward diffusion process. Subsequently, a deep learning model is trained to reconstruct the original distribution, thus facilitating the acquisition of generative modeling skills throughout the learning process. This methodology has been expanded to various modalities, encompassing images [43], videos [70, 193], audio [112, 156, 201], among others. Dhariwal et al. [43] showcased the efficacy of diffusion models, demonstrating significant enhancements in image generation. They achieved superior FID scores on ImageNet 512×512 by employing a modified UNet architecture as a generator and introducing classifier guidance, allowing the diffusion model to be conditioned on classifier gradients. The extension of this approach to high-resolution image generation without auxiliary classifier assistance was further advanced by Ho et al. [72]. In the medical domain, Kazerouni et al. [101] applied diffusion models, with subsequent extensions to conditional generation using text prompts. Moreover, these techniques have found application in image super-resolution [54, 124], inpainting [190], restoration [133], image translation [244], and further extended to image editing [195].

Image Generation from text

Diffusion models have played a pivotal role in advancing the field of generative AI, contributing to significant milestones. Notably, the seminal work by Rombach et al. [190] have revolutionized high-resolution text-to-image synthesis by employing diffusion within the latent space. Their innovative approach involves a two-step process, wherein an autoencoder is initially trained on a large dataset,

followed by training of a diffusion model on the latent vectors obtained from the encoder.

The evolution of prompt-based image generation owes much to diffusion models. Ramesh et al. [185] showcase the integration of contrastive models like CLIP [183] in text-conditioned image generation. The proposed two-stage model generates an image embedding from CLIP based on a text caption, followed by a decoder generating an image from the embedding. The shared embedding space of CLIP facilitates language-guided image manipulation. Experimental results demonstrate the superior performance of diffusion models compared to autoregressive models, particularly on DALL-E datasets [186]. Additionally, Gu et al. [59] showcase high-quality image generation using Vector Quantised-Variational AutoEncoder (VQVAE), offering an improved trade-off between quality and speed. Kang et al.[94] scale up GANs for extensive image datasets like LAION [202], delivering advantages such as high-speed generation, interpretable latent spaces, and high-resolution $4k$ generation.

Diffusion+NeRF Models for 3D Rendering

Numerous diffusion models have undergone training on extensive image datasets, such as LAION [202], containing over a billion meticulously curated training samples sourced from the web. These prompt-based models exhibit remarkable generalization capabilities. Leveraging the strengths of prompt-based diffusion models, researchers employ them to train Neural Radiance Fields (NeRF). Poole et al.[179] introduced Score Distillation Sampling that involves utilizing a pre-trained diffusion model as a frozen critic to guide the NeRF in image generation. To incorporate pose information, prompts are enriched with pose-centric phrases and keywords, enhancing the NeRF model's ability to distill accurate pose-dependent details.

Addressing limitations like blurring and the Janus problem, Wang et al. [228] acknowledge the distribution mismatch issue. They propose a Perturb and Average Scoring (PAAS) mechanism to mitigate this problem, computing scores on noisy images by perturbing them with noise and averaging the scores obtained from each perturbation.

Instead of relying on Score Distillation Sampling (SDS), Liu et al. [135] advocate fine-tuning a Stable Diffusion model on an input image and target camera pose. Training on the Objaverse dataset [40], they engage in single-image conditioned multi-view generation. Methods like DiffRF [161] explore the use of 3D UNets to learn diffusion-based NeRF-like rendering. Render Diffusion [7] endeavors to conduct 3D reconstruction, inpainting, and generation solely with monocular 2D supervision. The method employs diffusion to learn to render novel views conditioned on camera views directly. Similarly, [234] introduces a simple diffusion-based novel view rendering method for directly generating im-

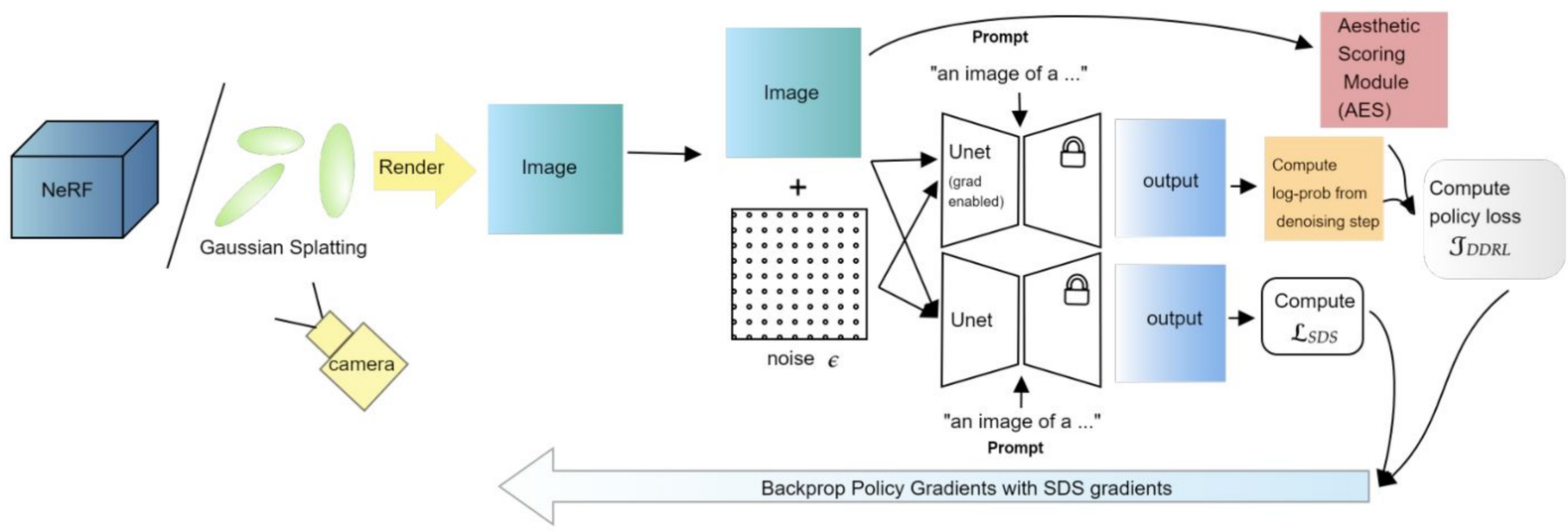


Figure 7.1: We demonstrate our approach in the above figure. We start with the SDS-based gradient calculation using a frozen UNet model calculated by performing the denoising operation on the image generated using the NeRF/Gaussian Splatting technique. Then, using the rendered image, we calculate the aesthetic score and perform another step of noising-denoising using the UNet as a policy network. We use the generated latents as the action and compute the policy gradient, which is then used to update the NeRF/Gaussian splat parameters to maximize the aesthetic score.

ages corresponding to novel viewpoints. Extensive results on the ShapeNet NMR dataset [26] validate their approach. Several extensions of Gaussian Splatting to SDS-based differentiable rendering [29], Tang et al. [218] accelerate text-to-3D and image-to-3D processes significantly.

Our approach provides an extension for DDPO to be used in conjunction with the aforementioned diffusion-based 3D generation, demonstrating higher-quality results.

7.2 Approach

We start by exploring the impact of the aesthetic scoring function in the 3D setting. Since it is computed using a neural network, it can be used directly in a differentiable setting. We observe that the aesthetic scoring function is vital in improving the quality of the diffusion models trained using DDPSO3D. We, therefore, first experiment with the impact of the aesthetic scoring function referred to as AES that takes in an input image x and provides an aesthetic score, $AES(x) : \mathbb{R}^{c\times h\times w} \mapsto \mathbb{R}$. This is first discussed by Schuhmann et al. [202], who implemented the scorer as a shallow MLP that was trained on annotated data for aesthetic score prediction using the CLIP embeddings of the images and later used in DDPO as a means to improve the diffusion model. We incorporate this term with diffusion loss to improve the aesthetic quality of the image rendered via the SDS method. Thus, the $AES(\cdot)$ model acts as a guide for the aesthetic quality and allows for a better generation quality. We optimize the typical Splat/NeRF parameters Θ in an SDS setup using the following objective

$$\Theta* = \arg\max_{\Theta} AES(x); x \sim p(x). \tag{7.1}$$

Thus, the complete training loss for a rendered sample x is,

$$\mathscr{L}_{total} = \mathscr{L}_{SDS} - \lambda AES(x).$$

where $\mathscr{L}_{SDS}$ is the SDS loss term, which we will further explain. However, this also requires investigation as to how to incorporate other rewards in the SDS generation. We derive inspiration from DDPO and realize that the same property of viewing the diffusion process from a reinforcement learning standpoint could offer advantages even for 3D generation. The seminal work by Black et al. [18] demonstrates effective ways for fine-tuning and training diffusion models. They view the diffusion process as an MDP and treat noise prediction by the network as the action taken by a policy network.

Thus, we aim to leverage the reinforcement learning framework for tuning the Score Distillation-based approaches for aesthetic scores that would yield higher-quality results. We use the method of Black et al. [18] for policy gradient estimation to get the Monte Carlo estimate of our policy loss $\mathscr{J}_{\text{DDRL}}$ as $\nabla_\theta \mathscr{J}_{\text{DDRL}}$ which we refer in our case as $\mathscr{J}_{\text{DDPSO3D}}$ and its gradient $\nabla_\theta \mathscr{J}_{\text{DDPSO3D}}$.

The pre-trained diffusion model takes the context $c \sim p(c)$ and induces a sample distribution $p_\theta(x_0|c)$, from the underlying context distribution for the training data $x_0 \sim p_\theta(x_0|c)$. This denoising diffusion RL (DDRL) objective is used to maximize the reward signal r defined on the samples and the context c for the MDP

$$\mathscr{J}_{\text{DDPSO3D}}(\theta) = \mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}),\ \mathbf{x}_0\sim p_\theta(\mathbf{x}_0|\mathbf{c})}\left[r(\mathbf{x}_0, \mathbf{c})\right]$$

The denoising diffusion process is considered as an MDP whereby the state is defined as a tuple of the context c, time t i.e. $\mathbf{s}_t \triangleq (\mathbf{c}, t, \mathbf{x}_t)$ and the latent x_t which are taken as input by the diffusion model which is represented as policy π and defined as $\pi(\mathbf{a}_t \mid \mathbf{s}_t) \triangleq p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c})$ and performs the action $\mathbf{a}_t \triangleq \mathbf{x}_{t-1}$ for the denoising operation. The state transition is given by P where $P(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t) \triangleq (\delta_{\mathbf{c}}, \delta_{t-1}, \delta_{\mathbf{x}_{t-1}})$ where δ_y represents the Dirac delta distribution with non-zero density solely at y. The reward $R(\mathbf{s}_t, \mathbf{a}_t)$ is considered as 0 for all timesteps except at the final denoising step $t = 0$ where it is given by $r(\mathbf{x}_0, \mathbf{c})$. Black and colleagues [18] employ a training regimen based on REINFORCE [158, 238] for fine-tuning, utilizing the gradients derived from:

$$\nabla_\theta \mathscr{J}_{\text{DDRL}} = \mathbb{E}\left[\sum_{t=0}^{T} \nabla_\theta \log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c})\ r(\mathbf{x}_0, \mathbf{c})\right].$$

We use the same REINFORCE-based training regimen as given in equation 7.2 and extend this policy gradient update to the generated image, as shown in Figure 7.1. This is akin to using the policy network as a critic for updating the state representation, much similar to a discriminator in a GAN-like setup. This aims at driving the state representation, which here forms the input to the policy

network in the form of the image rendered using splat to find an optimal state in which the policy would yield the highest rewards. To the best of our knowledge, this is the first attempt at using a policy network for finding the optimal state representation in this fashion. Most reinforcement learning problems involve finding the optimal policy for a given state; however, this is more suited to our problem due to the Markovian nature of the diffusion process and the ability to utilize these gradients to drive the state representation. This is different from DDRL for this very reason, since we do not update the policy network but rather use the pre-trained policy to derive the state.

We consider a setting similar to DreamFusion [179] that proposes 3D generation by considering differentiable image parameterization (DIP) where the generator g, which is a DIP parameterized by Θ produces an image $\mathbf{x} = g(\Theta)$ which is then optimized using the gradients calculated by the denoising process from the latents z_t obtained from the image $\mathbf{x}$ rendered at a timestep t conditioned on the prompt y

$$\nabla_{\Theta}\mathscr{L}_{\text{SDS}}(\phi, \mathbf{x} = g(\Theta)) \triangleq \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon}_{\phi}(\mathbf{z}_t; y, t) - \epsilon\right)\frac{\partial \mathbf{x}}{\partial \Theta}\right],$$

where $\hat{\epsilon}_{\phi}$ is the noise predicted from a network parameterized by ϕ; ϵ is the target noise and $w(t)$ is the time-dependent weight. This approach, termed Score Distillation Sampling (SDS), involves employing the denoising network, UNet, from the diffusion model as a critic for the generator. To accomplish this, we compute the gradients concerning the generator parameters Θ, acquiring the gradient term from the latents $\mathbf{z}$ derived from the rendered image $\mathbf{x}$ as in eq. 7.2.

$$\nabla_{\Theta}\mathscr{J}_{\text{DDPSO3D}}(\phi, \mathbf{x}_{t-1} = g(\Theta)) = \mathbb{E}\left[\sum_{t=0}^{T}\nabla_{\Theta}\log p_{\theta}(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \mathbf{c})\; r(\mathbf{z}_0, \mathbf{c}).\frac{\partial \mathbf{x}}{\partial \Theta}\right].$$

This gradient term is used to drive the generator g according to the additional reward functions used in conjunction with the generation. Instead of fine-tuning the Stable Diffusion backbone, we propagate the gradients back to the neural renderer to update the parameters with respect to the additional rewards. We refer to this method as DDPSO3D since it allows for the differentiable image parameterization to be updated using the additional reward-based updates. This can be intuitively thought of as state representation learning from a fixed policy since we do not update the Stable Diffusion model parameters. It can be explained as an attempt to find a state where the policy works successfully. This allows integration with Gaussian Splatting-based generation methods that, at each step, sample a random camera pose p around the object center and render the RGB image I^{p}_{RGB} and transparency I^{p}_{A} of the current view. For the Gaussian

splatting, the gradients are computed as

$$\nabla_\Theta \mathscr{L}_{\text{SDS(splatting)}} = \mathbb{E}_{t,p,\epsilon}\left[(\epsilon_\phi(I^p_{\text{RGB}}; t, e) - \epsilon)\frac{\partial I^p_{\text{RGB}}}{\partial \Theta}\right],$$

where e is the CLIP embedding of the input text description. Thus helping us find a sweet spot between speed and quality of generation. We combine the DDPO with the above gradient updates to yield the following update equations

- For SDS (using eq. 7.2 and 7.2)

$$\nabla_\Theta \mathscr{L}_{\text{total(SDS)}} = \nabla_\Theta \mathscr{L}_{\text{SDS}} + \nabla_\Theta \mathscr{L}_{\text{DDPSO3D}}, \quad \text{and}$$

- For splatting (using eq. 7.2 and 7.2)

$$\nabla_\Theta \mathscr{L}_{\text{total(splatting)}} = \nabla_\Theta \mathscr{L}_{\text{SDS(splatting)}} + \nabla_\Theta \mathscr{L}_{\text{DDPSO3D}}.$$

Our approach is compatible with both DreamFusion-based NeRF rendering and DreamGaussian-based Gaussian splatting, offering significant flexibility and enhancing the overall generation process. Since these methods rely on Stable Diffusion-based generation, we can integrate our proposed method relatively easily. However, since most of these methods rely on single-step updates during optimization, we currently restrict our reward computation to a one-step MDP, that is, the stage at which we start the noising process. We take the rendered image during the generation, perform the noising step in score distillation, and then do the denoising using the Stable Diffusion UNet. Through this, we compute the log probability similar to DDPO corresponding to the denoising step. However, instead of updating the parameters of UNet, we backpropagate the gradients back to the NeRF/Gaussian Splatting parameters and update them based on the policy loss.

The policy loss term provides an additional direction for the update to move in a direction that performs optimization for the reward obtained by the additional scoring function. In our experiments, we find that aesthetic score is highly supportive for 3D generation tasks and helps provide good feedback for the gradient updates.

Since our technique relies on policy gradient computed from Stable Diffusion, we can also use it with other variants, such as the recent 3D-SDS-based methods, such as GSGen [31], shown in the next section, thus showing the versatility of our proposed approach. This method, however, only governs how the Gaussians are optimized but does not govern how the Gaussians are generated. Keeping this in mind, we re-examine the densification strategy used in the next section.

7.2.1 Quad-Tree based Densification

Further, we re-examine the densification process. Currently, the densification process is governed using an adaptive densification process. The authors in [104] perform densification in over and under-construction cases where they observe large view-space positional gradients. They use a gradient threshold to perform the densification, whereby the densification is performed by cloning the Gaussians in the region of under-construction and moving in the direction of the positional gradient. For Gaussians with large variance, the Gaussians are split by dividing their scale using a factor of $\phi = 1.6$. However, we realize this method is not completely adaptive as there is a restriction in the number of Gaussians that get introduced using this strategy at any point in time. Therefore, we propose another adaptive densification strategy whereby the Gaussians that get introduced at the location depend on the amount of entropy in the region. We find that this leads to a more complete shape since it is no longer dependent on just the regions of high gradient but is rather derived from the image information itself. Thus, in order to capture the information in a particular region, we rely on an image quadtree-based strategy. We first compute the image quadtree of the image that is rendered. Using the quadtree leaves, we compute the quadtree centers, which are then back-projected using the depth estimated using the network. These back-projected points are then used as the new Gaussian centers.This leads to an adaptive generation of Gaussians without the need for explicit duplication, since the regions with high-frequency details contain a larger number of quads, thus leading to a higher density of Gaussians being generated automatically while ensuring the regions with lower entropy contain a lower number of Gaussians. We not only test this densification in text-to-3D but in image-to-3D settings as well. For pruning the Gaussians, we currently rely on the existing method, which uses gradients and opacity to determine the Gaussians that do not contribute to the shape. For both image-to-3D and text-to-3D, we show the effectiveness of this scheme by limiting the number of Gaussians to a much lower number $\sim 6k$ as discussed in the next section. We select this number since we observe that the impact on the reconstructed shape is most profound at lower numbers. Furthermore, we observe that the resulting meshes tend to be more *"natural"* since the number of non-manifold vertices goes significantly down using our densification.

7.3 Experiments, Results, and Discussion

We start by testing the AES term with several methods to understand its impact on SDS-based methods. In order to evaluate our method, we use CLIP score to check the semantic relatedness by measuring the cosine similarity between the CLIP prompt embedding and the rendered image embedding. During our experimentation, we observed that image-to-3D settings offer lower modifications with respect to the AES score due to a tighter adherence to the image provided,

Table 7.1: The following figure shows the results we obtained using our DDPSO3D method combined with Gaussian Splatting for text-to-3D generation. Our method further enhances the details and has better variation in the generated structures and lower artifacts. Using our method, we also observe better CLIP scores, where a larger CLIP Score suggests higher CLIP similarity between the rendered image and prompt.

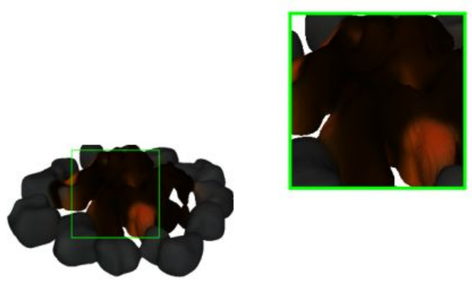
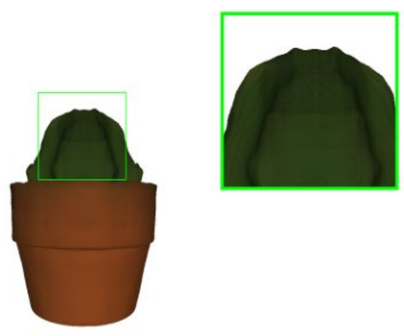
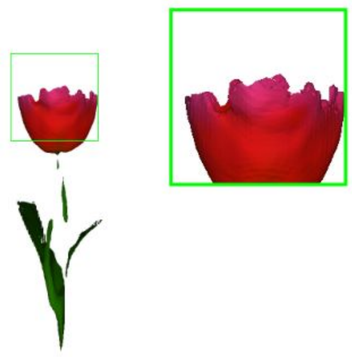
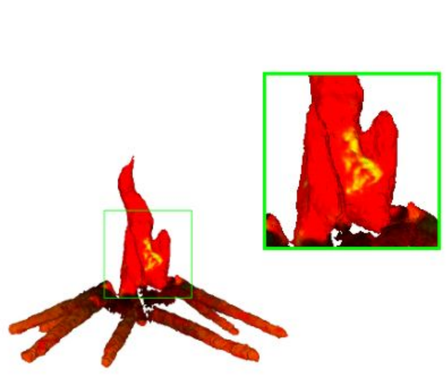
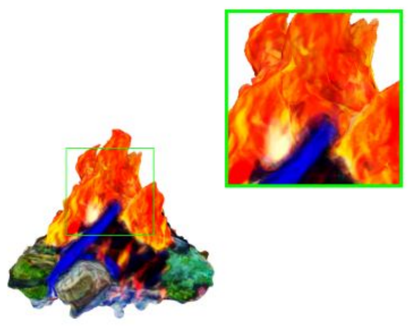
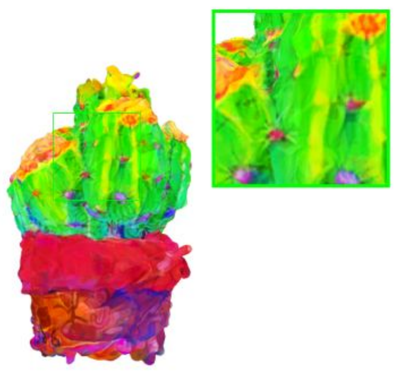
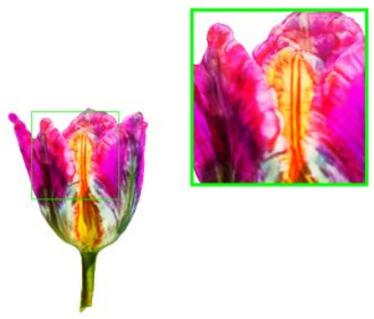
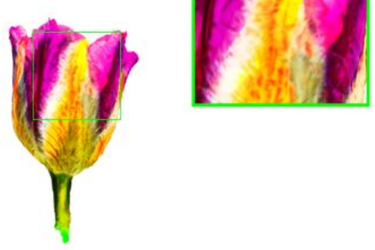
Method	"a campfire"	"a small saguaro cactus planted in a clay pot"	"a tulip"
Shap-E [92]			
DreamFusion [179]			
Dreamgaussian [218]			
Ours(AES)			
Ours(DDPSO3D)			

Table 7.2: CLIP scores and aesthetic scores for the models shown in 7.1. For each model, we generate 20 images around the object from the resulting mesh and compute the average scores, with higher scores indicating better performance.

Method	*"A campfire"*		*"A small saguaro cactus planted in a clay pot"*		*"A photo of a tulip"*	
	Aesthetic	CLIP (%)	Aesthetic	CLIP (%)	Aesthetic	CLIP (%)
Shap-E	4.51	28.09	4.17	23.94	4.45	23.28
DreamFusion	4.59	29.02	3.9	26.72	4.42	27.05
Dreamgaussian	4.88	30.69	5.02	27.42	4.91	29.51
Ours (AES)	4.85	30.46	4.98	27.25	**5.21**	**32.61**
Ours (DDPSO3D + AES)	**5.16**	**31.18**	**5.14**	**31.29**	5.07	31.58

while text-to-3D is a better setting for our approach since the generation can be guided a lot more flexibly due to the lack of ground truth in such a synthesis. Therefore, we perform experiments primarily in a text-to-3D setting with DreamGaussian due to its fast generation speed. However, we also demonstrate the effectiveness of our method with other methods, such as DreamFusion and GSGen, which also use SDS-based text-to-3D synthesis.

In the case of DreamFusion, we see noticeable improvement when combined with the aesthetic scorer; not only do we observe the improvement in the textures, but also an improvement in the geometry of the generated objects, as discussed in the next section.

With the usage of DDPSO3D, we again notice a substantial improvement in text-to-3D setting with DreamGaussian demonstrated in Table 7.1.

7.3.1 Impact of AES Term

Our findings emphasize the vital role of the aesthetic scorer in enhancing the visual quality of generated assets. Additionally, this scorer proves versatile, enabling its integration with policy gradients and facilitating improvements over the baseline DreamGaussian model.

In Figure 7.2, we can visually see the difference in the quality of the generation for both GSGen and Dreamfusion. In both the car generated using GSGen and the cupcake generated using Dreamfusion, we achieved higher clip scores of 35.16 & 31.19 respectively, as compared to the original scores of 32.01 & 29.57.

We notice a similar improvement in the case of DreamGaussian as well as evident from Figure 7.4 and Table 7.1. Our method consistently outperforms others in terms of both the aesthetic scores and CLIP scores, as shown in Table 7.2 with a considerable difference in scores.

We tested our method with text to 3D setting in DreamGaussian, whereby the policy gradient updates are used to update the Gaussian splats. The text-to-3D renderings of Gaussian splats are not on par with image-to-3D renderings

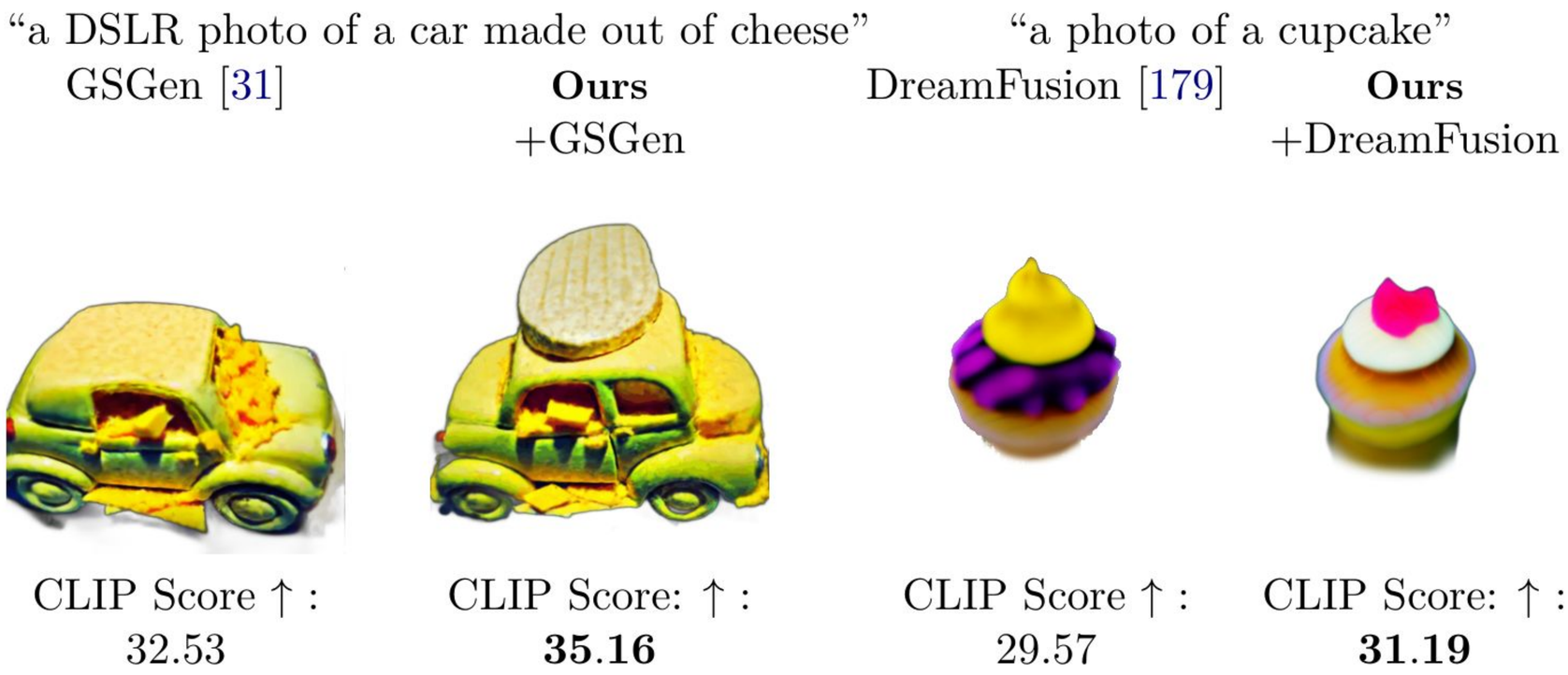


Figure 7.2: Assets generated in the same number of iterations using AES (right) based optimization and without AES (left) using GSGen [31].

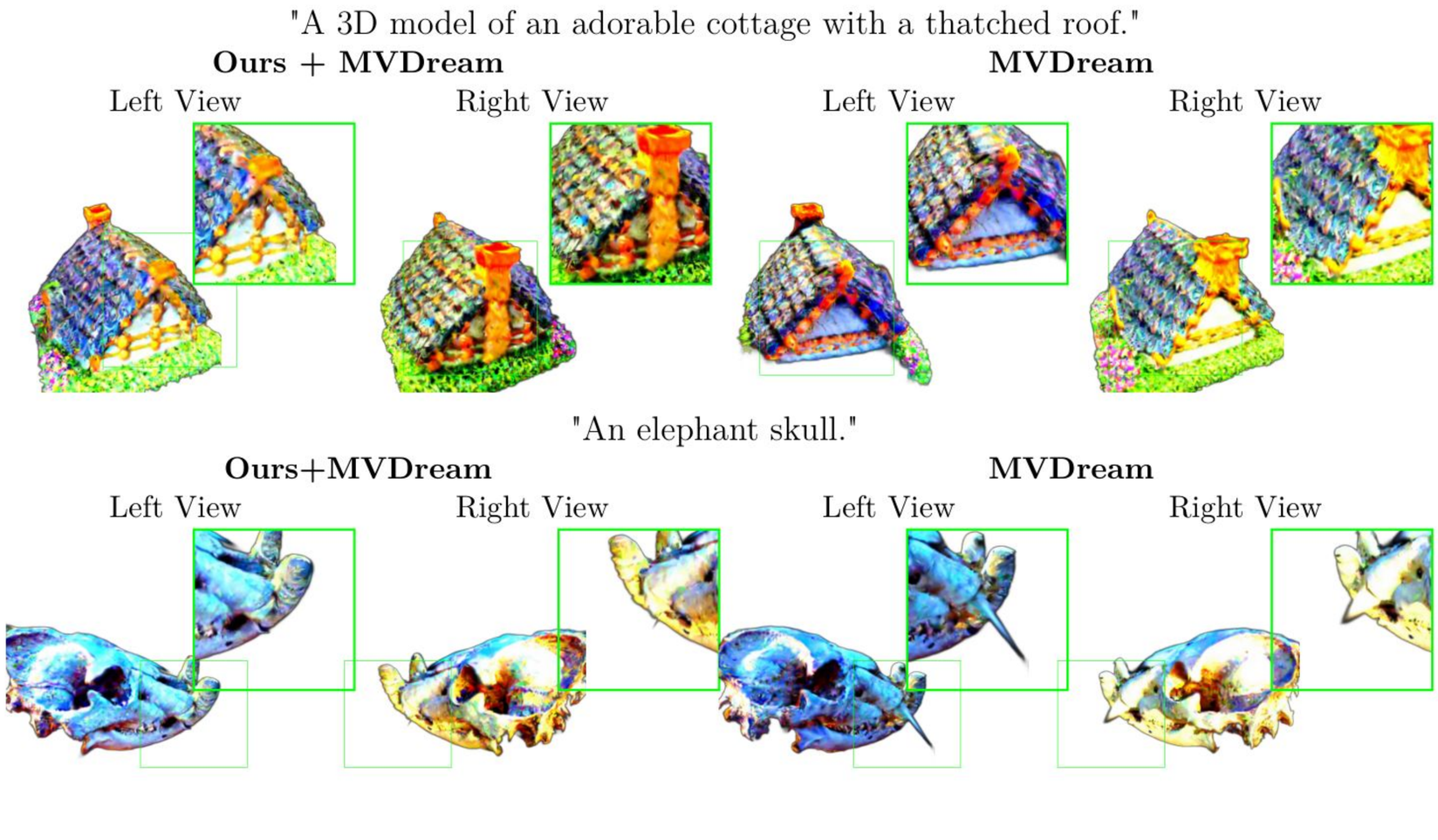


Figure 7.3: We show results with MVDream based guidance. Our method produces more detailed models (top row) and fewer artifacts(bottom row).

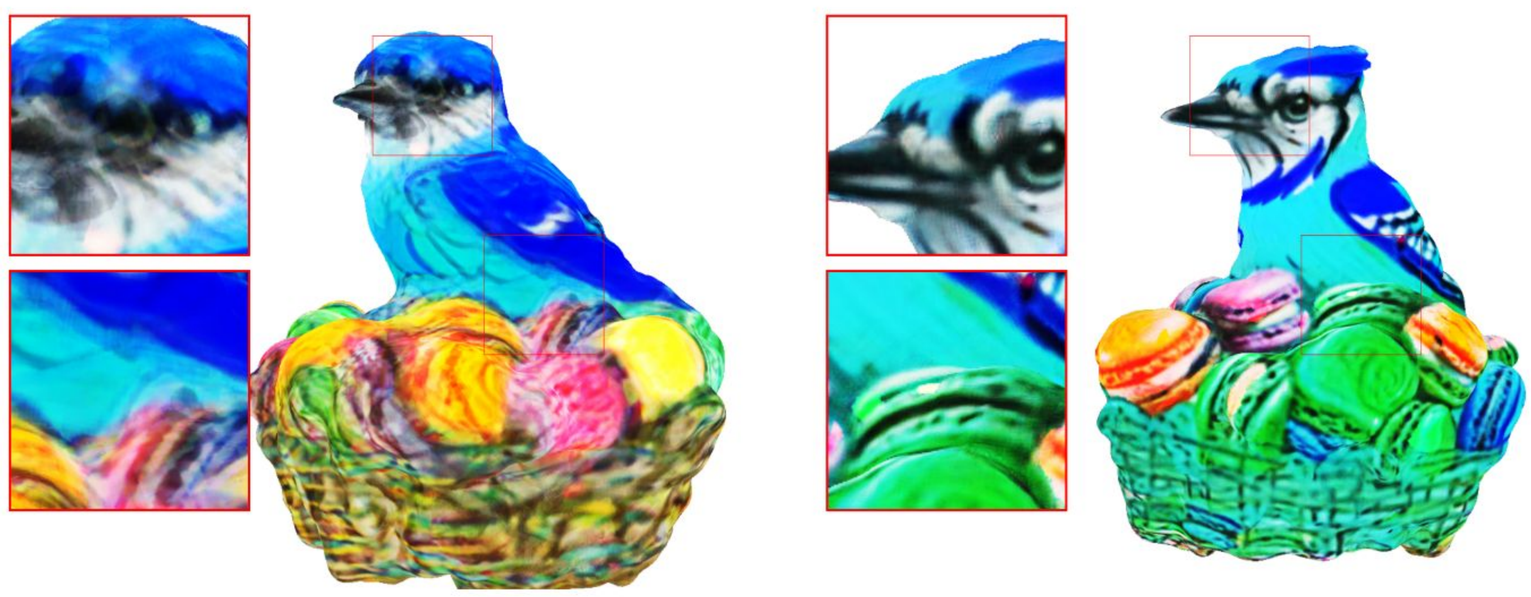


Figure 7.4: A zoom-in view of the differences in generation using our method (row 2) and the original DreamGaussian (row 1), for the prompt “A blue jay standing on a large basket of rainbow macarons”.

and tend to suffer from artifacts. We observe that DDPO-based optimization not only reduces these artifacts but also improves upon the structure of the generated mesh. We also observe an improvement in generated results under MVDream guidance as shown in Fig. 7.3. We see more detailed structures in the "cottage" and fewer artifacts in the "elephant skull", whereby the original method produced a sharp, tooth-like, unnatural structure.

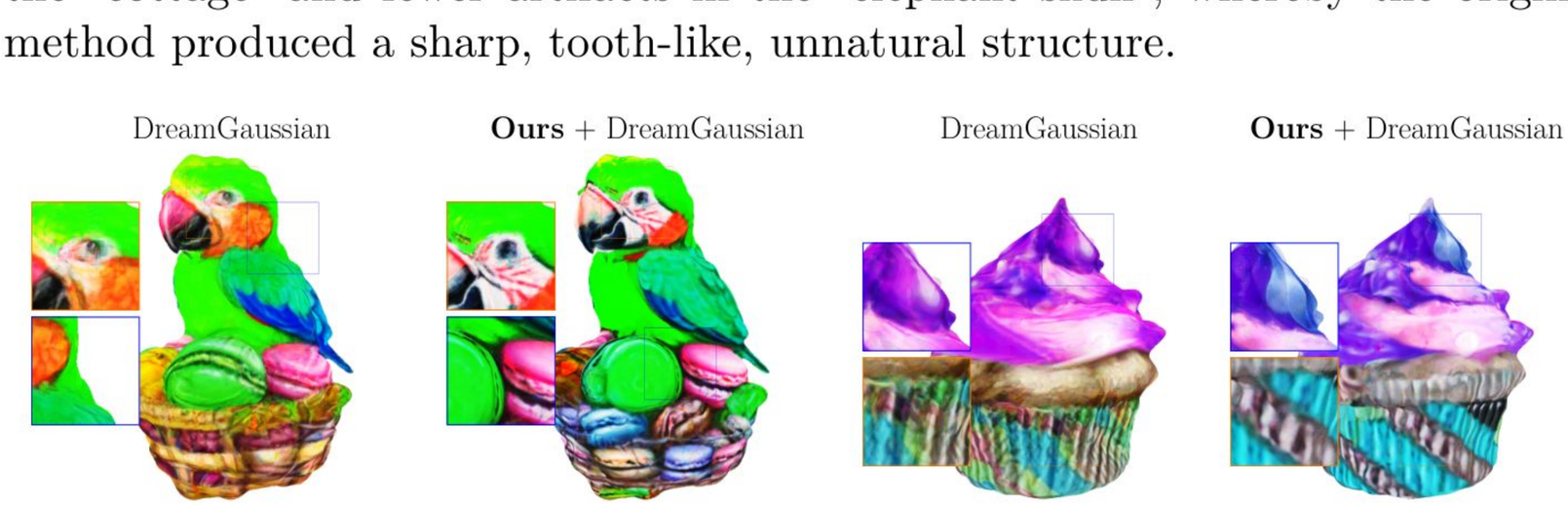


Figure 7.5: A zoom-in view of the differences in generation using our method and the original DreamGaussian in only stage two of the optimization process, for the prompts “a parrot sitting on a basket of macarons” and “a photo of a cupcake”.

DDPSO3D employs SDS in two stages: during the training of the Gaussian splat and the final stage of UV refinement. Our term assumes distinct roles in each scenario. In the initial stage, noticeable geometric improvements are evident, as illustrated in 7.4. Additionally, a significant enhancement in visual appearance is observed when our term is applied. To achieve this, we initially train the first stage following the original DreamGaussian method and introduce our term solely in the second stage. Remarkably, with $100 - 200$ iterations in stage two, our method surpasses existing approaches in terms of visual quality, as depicted in Figure 7.5.

7.3.2 QuadTree based Densification

We test the proposed densification strategy in both text-to-3D and image-to-3D. We observe similar gains in both tasks as well, and we also observe another advantage that our method offers. We observe that in the case of quadtree-based densification, our method produces more "natural" meshes, which can be attributed to a lower number of non-manifold vertices, as shown in Fig. 7.7.

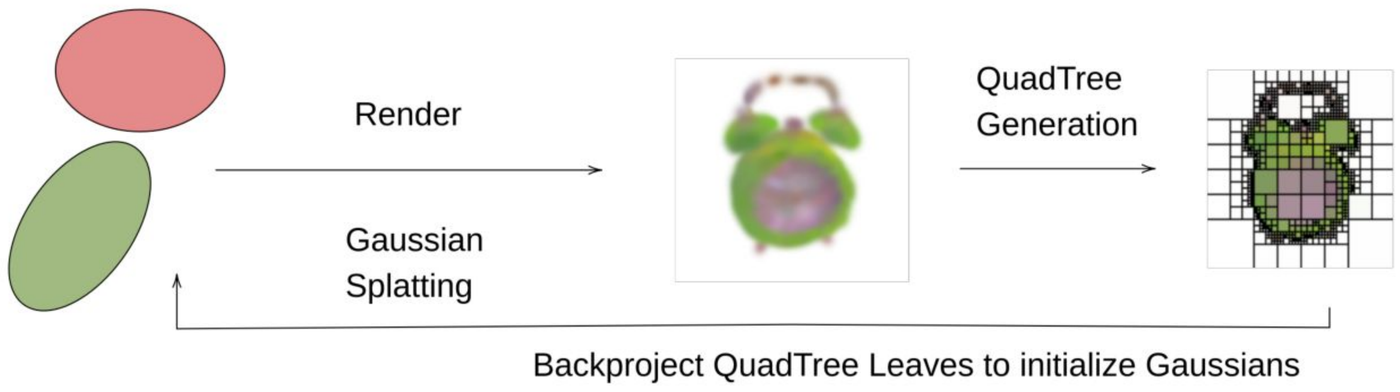


Figure 7.6: We generate the image using Gaussian splitting and then update the densification using the quadtree computation performed on the rendered view.

We further test in text-to-3D by limiting the number of Gaussians and carefully examining the nature of the resultant meshes. We show the results for two prompts in Fig. 7.10. We observe that the original scheme typically generates a very large number of Gaussians, typically in the range of $\sim 100k$, and we severely reduce the number of Gaussians to $\sim 6k$ to observe the behavior under constrained settings. The densification process is driven by gradients and relies only on cloning and splitting of existing Gaussians. We observe that the resultant shapes are not only incomplete, but the network focuses more on some sub-structures more than others, as can be seen in both the ice cream and cupcake models, where the original method on the left focuses more on the top leading of the object. This leads to much finer structures on the top as compared to the bottom in the generated mesh in Fig. 7.10. Whereas our method generates more complete shapes in such cases due to a better distribution of Gaussians, which leads to a more uniform generation. This is also evident in the image-to-3D task as shown in Fig. 7.8 and Fig. 7.9. Our method consistently outperforms the cloning-based densification scheme.

7.3.3 Limitations & Future Scope

We recognize a visible trade-off between runtime and quality arising from the inclusion of the additional policy gradient term. This is attributed to the increased computation of the reward function, a focus area for future improvements. Similar to many regularization terms, the RL term requires fine-tuning dependent on the magnitude of the SDS terms. Effective results were obtained in the range of $1e1$ to $5e2$, particularly when used in conjunction with AES only. The impact of weighting on generation is illustrated in Figure 7.11, where enhanced details

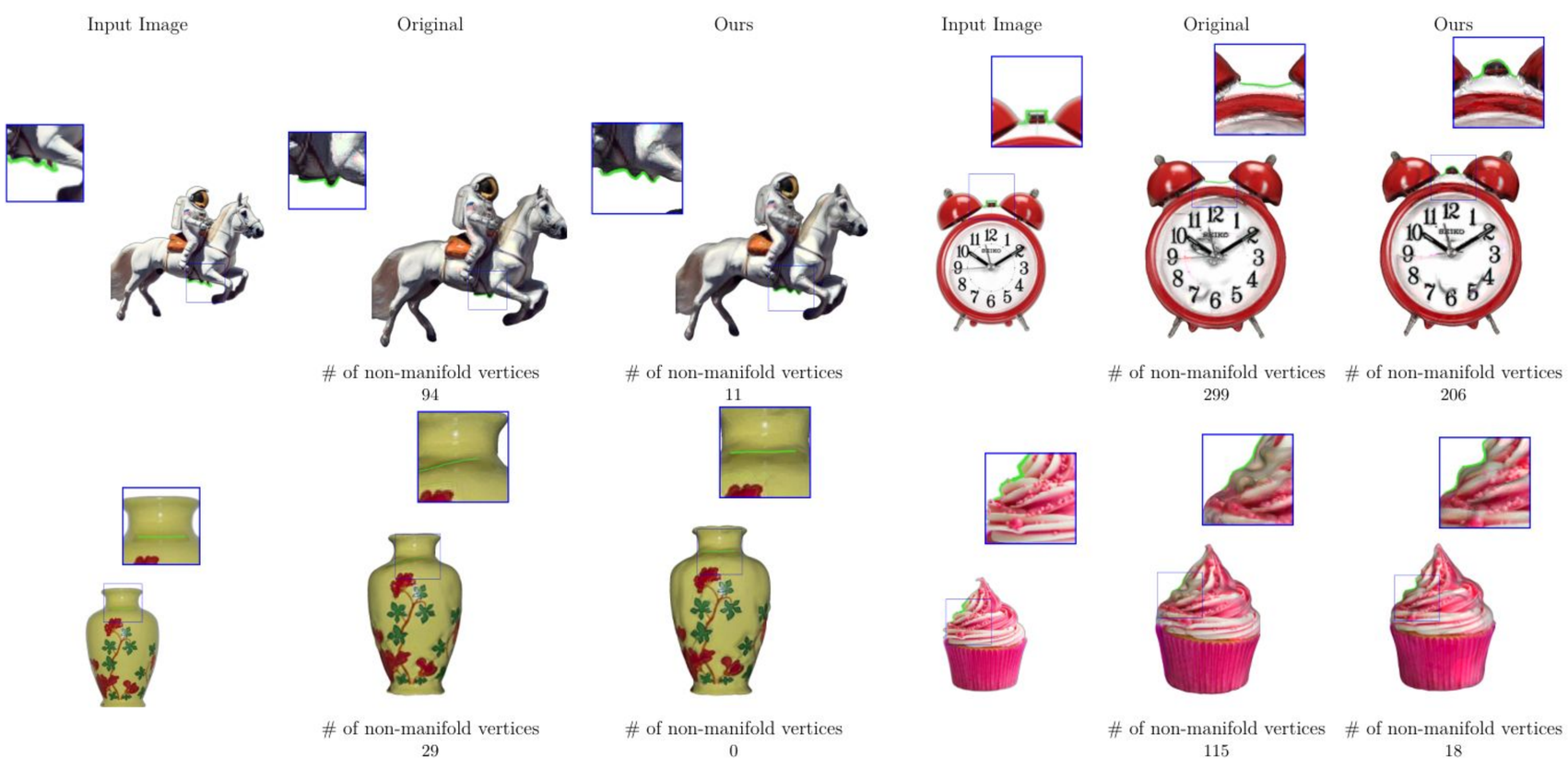


Figure 7.7: Input image(left), generated mesh using quadtree-based densification(center) and mesh generated using original densification strategy. Our method is able to capture finer details highlighted in green in the images. The structures under the body of the horse, the middle section of the alarm, the smoothness of the top of the vase, and the adherence to the boundary in the cupcake.

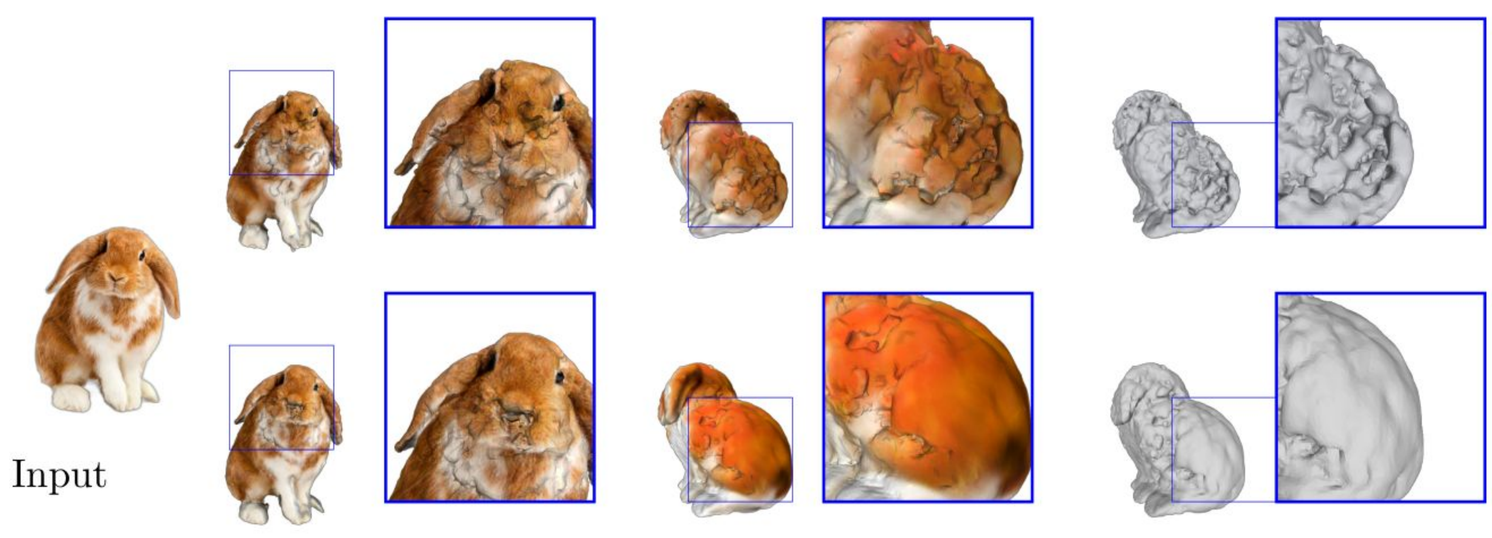


Figure 7.8: Image-to-3D comparison between dreamgaussian (first row) and DDPSO3D (ours, second row). We show the input image, the front (column 2), the back (column 3), and the generated mesh (last). Our method produces a smoother surface as evidenced by the generated mesh.

Figure 7.9: Under constrained settings where we limit the number of maximum Gaussians to be $\sim 6k$. We notice that our method produces more complete structures and smoother structures as compared to the original method for the image to 3D.

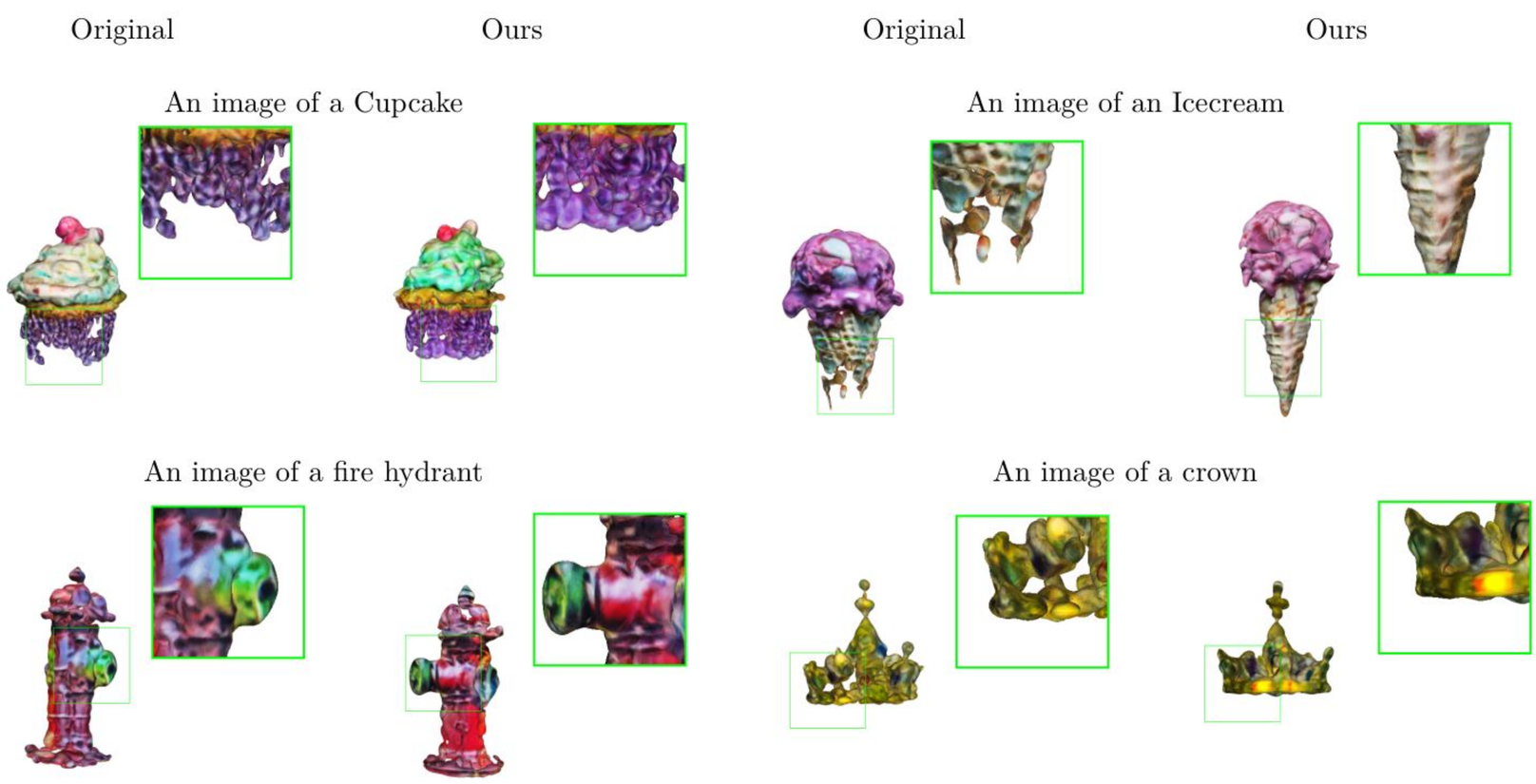


Figure 7.10: Under constrained settings where we limit the number of maximum Gaussians to be $\sim 6k$. We notice that our method produces more complete structures and lesser holes as compared to the original method.

are initially achieved but may diminish after a certain threshold (in this case, $5e2$).

A notable issue observed is over-colorization and color bleeding with an increased weight of the AES term, as depicted in Figure 7.12. However, despite challenges, AES-based guidance proves more stable compared to DDPSO3D, which relies on policy gradients that can be trickier to stabilize and requires hyperparameter tuning.

Another limitation is the lack of a good evaluation strategy. We observe that for many text-to-3D tasks, good evaluation methods are not available due to the lack of ground truth to be compared against. Therefore, we rely on the CLIP score and aesthetic score as a measure for semantic and quality quantification. While aesthetic scoring gives us an idea of how well would the rendered image be perceived by humans in general, the CLIP score talks about semantic relatedness. Despite this, we notice some ambiguity that arises from CLIP scoring. For the views rendered in Figure 7.4, we notice that the DreamGaussian result attains a higher value of CLIP score of 37.55 while our AES-only based approach achieves a score of 36.96, which is difficult to explain despite being clear differences in the bird structures, requiring further investigation. Though our method improves upon several other methods, we do not notice any improvement in the Janus problem, which we attribute to the lack of 3D awareness in the scoring mechanism, and leave the construction of a better scoring mechanism for the future. Further, since we use an aesthetic score, DDPSO3D without SDS is not sufficient for 3D generation, since the aesthetic score is not a sufficient signal to drive the entire reconstruction process and leads to failure in the absence of SDS.

In the case of a quad-tree-based densification strategy, we observe that the method at times does not perform well, leading to holes and incomplete structures in the case of Text-2-3D, which is due to the property of the quad-tree-based Gaussian densification to generate sparser points as compared to the original densification. Text-2-3D, unlike image-to-3D, requires a lot of denser Gaussians in the order of $\sim 200k$ while image-to-3D typically generates Gaussians in the order of tens of thousands. Apart from the densification process, text-to-3D is a lot noisier process due to SDS being performed on text-to-image guidance rather than image-to-3D, which uses Zero123/MVDream guidance that contains 3D support implicitly.

7.4 Conclusion

We currently primarily focus on Score distillation-based techniques. However, we hypothesize that this can further be extended to recent variational score distillation techniques such as ProlificDreamer [233], which also rely on Stable

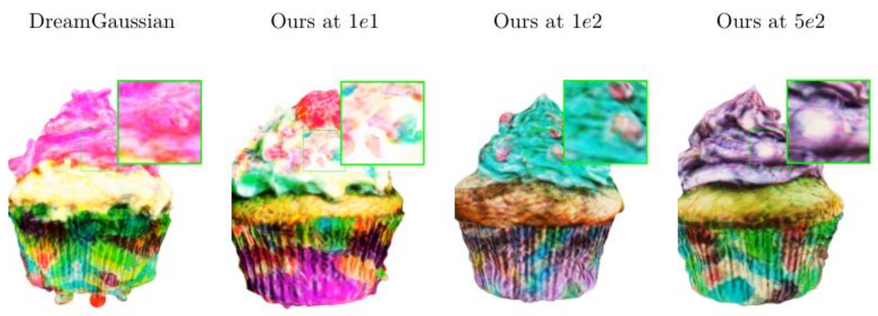


Figure 7.11: Changes in the appearance with increasing weightage of the AES term.

Figure 7.12: We observe color bleeding in some cases and some high-frequency artifacts.

Diffusion guidance. Our method demonstrates how policy gradient methods can be extended to score distillation-based techniques, allowing for better metrics and visual quality of results. Since it can be used with non-differentiable rewards, which creates an interesting future direction, we look forward to the vision and graphics community using it for new reward-based 3D asset generation and the creation of novel reward functions to support different tasks. We also show how a simple aesthetic score-based optimization can widely enhance the visual properties of the rendered object, not only impacting the textures but guiding changes in the geometry as well. Our aesthetic score-based technique shows improvement with many SDS-based techniques out of the box and can readily be used to steer the generation to a better result.

Chapter 8

MVGaussian: High-Fidelity text-to-3D Content Generation

Recent advancements in text-to-3D generation have opened new avenues for creating complex 3D content directly from textual descriptions. This capability is crucial as it provides a straightforward, intuitive means for creators across various industries like gaming, virtual reality, and filmmaking, enabling rapid prototyping and visualization without the need for advanced modeling software or specialized training.

In leveraging foundation models for image generation, recent works have used reconstruction methods like Neural Radiance Fields (NeRFs) [154] and 3D Gaussian Splatting (3DGS) [105] to make significant strides in the field. These models typically utilize Score Distillation Sampling (SDS) [179] to train the NeRF or Gaussian splatting methods, allowing for the generation of consistent 3D representations suitable for high-quality rendering and mesh extraction.

Recent approaches [32, 130, 218, 233, 259] have successfully generated 3D models, yet they face significant challenges that limit their practical applications. These challenges include the multi-face (or Janus) problem, in which models produce inconsistent appearances from different angles, lengthy training times, and a general lack of fine detail in the generated models. Furthermore, most methods suffer from issues related to the complexity of their components and hyperparameters. They require considerable computational resources and time to generate high-quality content, or they compromise on quality to achieve faster processing times due to the inherent trade-off between quality and speed.

To address these limitations, we propose a novel framework that enhances the text-to-3D content generation pipeline by integrating SDS with an efficient 3D Gaussian splatting representation. Our approach not only tackles the aforementioned issues but also significantly reduces the computational overhead and training time. Our contributions can be summarized as follows:

- We introduce a text-to-3D framework that combines 3D Gaussian Splatting with multi-view diffusion guidance and novel regularization terms, enabling more efficient optimization with reduced Janus artifacts.
- We propose a novel densification method by optimizing the placement and density of Gaussian elements that accelerate the generation process, reducing the overall training time to ~ 25 minutes.
- Through rigorous experiments, we demonstrate that our method not only matches but often surpasses the quality of existing SDS-based approaches with shorter training time.

For more visual results, please visit our project page at: mvgaussian.github.io.

8.1 Related work

Recent advancements in text-to-3D synthesis are built on the foundations established by text-to-image generation, 3D representations, and techniques for lifting 2D images to 3D models. This section reviews significant contributions in these areas, highlighting their methodologies and addressing their limitations.

8.1.1 Text-to-image generation

Earlier works in text-to-image generation leveraged GANs to map sentences to realistic images [123]. With the advent of diffusion models [71, 210], the field of text-to-image generation has advanced significantly, accelerating progress in content generation. Stable Diffusion [190] has demonstrated the effectiveness of diffusion over latent spaces for producing high-quality conditioned generations, particularly for text-to-image tasks.

Methods such as DALL-E [187] utilize text embeddings, such as CLIP [183], to jointly train text and image encoders and decoders. These models are trained on large-scale datasets, such as LAION [202], enabling zero-shot image generation. The method proposed by [162] explores two approaches, CLIP guidance and classifier-free guidance, and demonstrates that the latter is preferred in human evaluations.

Recent works have extended these approaches to multilingual image generation [257]. Additionally, text-to-image diffusion models have been further explored in image editing applications, such as Instruct Pix-to-Pix [22], leveraging advancements in image inversion techniques [53, 160]. Controlled image generation has also been a focus in methods like ControlNet [268], DreamBooth [195], and InteractDiffusion [74]. Beyond image generation, these models encode extensive semantic knowledge, making them effective for zero-shot classification tasks, as demonstrated by [36].

8.1.2 3D Representations

Recent advancements in 3D volumetric rendering have focused on using a shallow neural network that learns to represent complex scenes as Neural Radiance Fields (NeRF) [154]. This network predicts $RGB\sigma$ values at a given point as viewed from a certain direction. The optimization is performed using a ray-marching setup, which has proven effective for novel view synthesis even with sparse views. This approach has also been extended to temporal scenes [24, 154]. A vast body of work has rapidly emerged that builds upon these methods by exploring their various attributes. NeRFs have been investigated from different perspectives, including the use of sparse views [60], generating NeRFs from unknown camera parameters [132], and reconstructing refractive surfaces [63]. Additionally, they have been extended to be queried using language models [106] and have been employed for 4D or higher-dimensional representations [51].

Shifting from implicit to explicit representations, [105] introduced 3D Gaussians with a differentiable rasterization technique for faster and real-time rendering. This method optimizes Gaussian parameters like scale, rotation, opacity, and color, and includes gradient-based schemes for managing Gaussians in a scene. Due to its speed and efficiency, this approach has largely replaced NeRFs in many applications such as 3D content generation [32, 130, 137, 259], SLAM [103, 147, 178], and semantic scene understanding [175, 182].

8.1.3 Lifting to 3D

Building on previous methods, such as Dreamfusion [179], introduced a novel approach for generating 3D models by leveraging text-to-image models. Their method employs a pre-trained diffusion model to distill multi-view information into NeRF models. The core of this approach is the Score Distillation Sampling (SDS) technique, which optimizes the NeRF model while omitting the U-Net Jacobian term. However, due to the lack of 3D awareness in the Stable Diffusion model, this method suffers from limitations such as blurry renderings and the Janus problem.

To address these issues, the method Score Jacobian Chaining (SJC) [228] introduced voxel radiance fields, improving image quality while still facing challenges like multiview inconsistency and mode collapse. Prolific Dreamer [233] further enhanced visual quality, diversity, and robustness by introducing Variational Score Distillation. DreamGaussian [218] replaced the NeRF-based representation with 3D Gaussian Splatting (3DGS), enabling faster text-to-3D and image-to-3D generation. However, it still encounters issues such as the Janus problem [179], poor mesh quality, and a lack of fine details. To improve aesthetics, RL Dreams[146] introduced an aesthetic score function, leveraging reinforcement-based techniques in conjunction with SDS. Similarly, DreamReward [258] proposed a novel reward function based on consistency, user preferences, fidelity,

and alignment, using human-annotated data to train text-to-3D models. They optimize generation quality by combining a pre-trained multi-view score model with a diffusion model.

Beyond diffusion-based approaches, other 3D generation methods have been explored. For instance, Point-E [163] generates text-to-point clouds, while DMTet [204] employs a differentiable tetrahedral representation for 3D reconstruction. Several subsequent methods, including GSGen [32], LucidDreamer [130], and GaussianDreamer [259], incorporated Point-E for initialization. However, Point-E struggles to generalize to complex prompts, limiting its effectiveness. In contrast, Fantasia3D [29] introduced a unique approach that disentangles geometry and appearance for high-quality 3D generation, leveraging a DMTet-based hybrid surface representation [204].

Recent works draw ideas from multiple works discussed previously and find unique solutions. GaussianObject [254] creatively draws ideas from ControlNet [268] and allows for refinement of the generated Gaussian splatting-based rendering. Furthermore, to generate the 3D object from sparse views, the method relies on the visual hull instead of a conventional SfM pipeline. ESC3D [242] proposes a novel method for controllable text-to-3D generation. Their pipeline allows for richer prompt generation and also allows for additional priors for conditioning the generation. Hallo3D [227] focuses on hallucination alleviation. The authors propose the usage of a generation-detection-correction paradigm and leverage modern large multi-modal models for guiding the generation.

In parallel, recent advancements [75, 245] have explored large-scale training on comprehensive 3D datasets, such as Objaverse [40], to improve text-to-3D and image-to-3D models. While these feedforward methods achieve impressive speed and consistency, SDS-based pipelines remain attractive due to their ability to flexibly adapt to unseen prompts and leverage the rich priors of pre-trained diffusion models. However, SDS-based methods continue to face challenges in training efficiency, 3D consistency, and reconstruction quality. To address these limitations, we propose MVGaussian, which improves efficiency, quality, and robustness for text-to-3D generation using 3D Gaussian Splatting.

8.2 Method

Our proposed method leverages MVDream along with a novel densification and pruning scheme to reduce the Janus problem. The densification and pruning scheme utilizes multi-view guidance and leverages back-projected points from the estimated depth on the fly to optimize the Gaussians. We observe that surface alignment techniques, as demonstrated in SuGaR [61], are beneficial for improving surface fidelity and mesh extraction. In contrast, our approach is designed for SDS-based text-to-3D generation, where we introduce a **regularization term**

that allows for flattening the Gaussians during the learning process itself. This flattening facilitates more effective pruning and improves geometry consistency during optimization. We primarily rely on multi-view guidance to mitigate the Janus issue and ensure consistent 3D reconstruction across different viewpoints. Additionally, we refine the densification strategy and enforce surface proximity for the generated Gaussians.

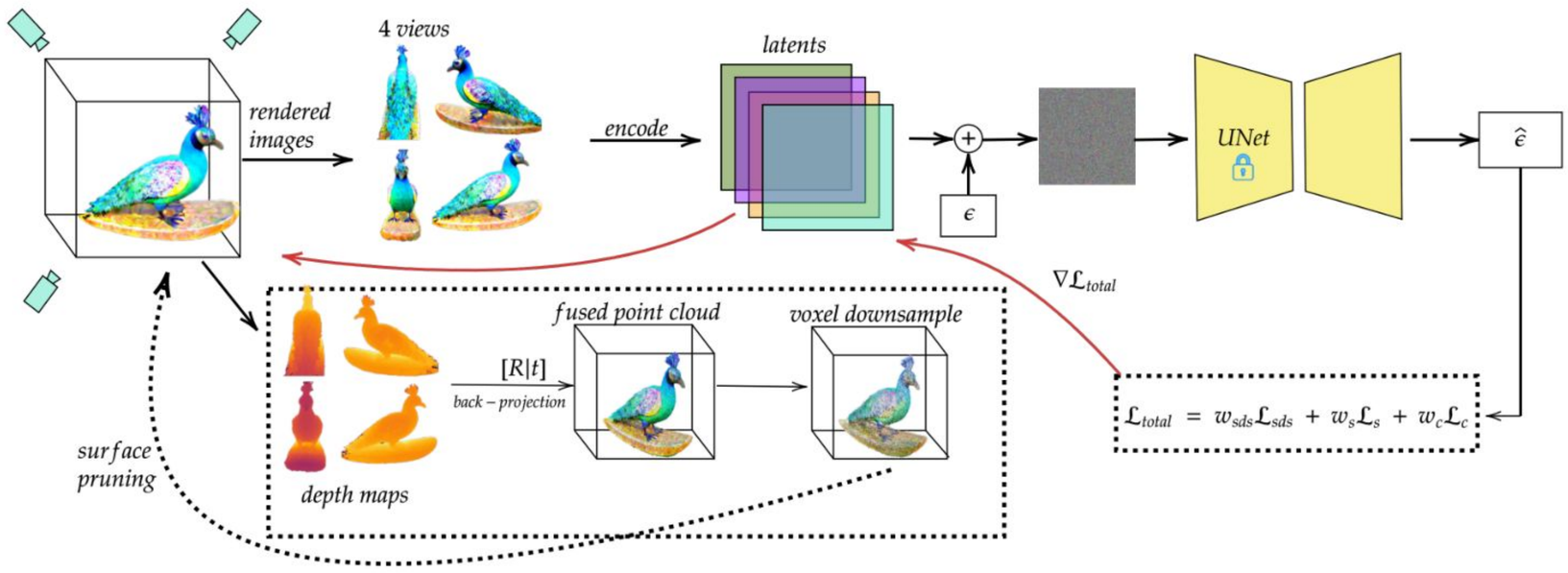


Figure 8.1: **Overview of our MVGaussian framework:** Our approach begins with the random initialization of Gaussians within a unit sphere, refined iteratively using an SDS-based optimization strategy. Gaussians are optimized near the true surface, moving toward the pseudo surface while pruning those farther away. Each iteration renders four views with random azimuth angles encoded into the latent space. Gaussian noise is added and subsequently denoised by a UNet to compute the losses. The resulting gradients update the Gaussians, forming a feedback loop that integrates fused point cloud data and voxel downsampling to improve accuracy.

The overall framework, as depicted in Figure 8.1, showcases how our approach integrates these components to produce more consistent and unified 3D reconstructions. This method not only addresses the shortcomings of prior techniques but also enhances the efficiency and quality of the generated 3D models.

8.2.1 Multi-view guidance for consistent 3D generation

SDS-based approaches for text-to-3D generation often suffer from the multi-face or the Janus problem [179]. This issue arises as the diffusion models are trained on 2D images and lack a true understanding of the 3D world. Consequently, while rendered images might appear plausible from different viewpoints, they often fail to represent a consistent and unified 3D object. Several strategies have been developed to address the Janus problem. Notably, Zero123 [135] and MVDream [205] have made significant strides by fine-tuning pre-trained diffusion models on 3D data. Zero123 predicts multi-view images conditioned on a reference image and camera position, while MVDream fine-tunes diffusion models to generate multi-view images from text inputs. Despite these advancements,

Figure 8.2: Extensive qualitative comparisons on object and scene prompts. We observe consistent improvement across prompts and baselines, demonstrating the effectiveness of our densification approach.

Figure 8.3: Additional qualitative comparisons on character and object prompts. Our method achieves visually coherent, high-fidelity 3D results and outperforms all the baselines.

Figure 8.4: Additional qualitative comparisons with several state-of-the-art methods.

these methods do not completely resolve the Janus problem, as the generated multi-view images often lack the exact consistency needed for unified 3D models since they lack precise symmetry and high-level detail correspondence across the generated views. However, they do provide reliable guidance for SDS-based approaches.

To address the Janus problem, we integrate the strengths of MVDream as the primary guidance mechanism within our framework. By adopting MVDream, we leverage its ability to generate multi-view images from textual inputs, thereby providing robust guidance for our 3D models. As shown in Figure 8.1, we render four views around the current object at each training step. We then map these multi-view images to the latent space and perform the noising and denoising steps. Similar to Dreamfusion, we adopt classifier-free guidance (CFG) proposed in Classifier-free diffusion guidance [73] to enhance the quality of generated 3D models. CFG adjusts the score function to favor regions with a higher ratio of conditional to unconditional density, using a guidance scale parameter ω.

8.2.2 Gaussian Alignment for Optimal Geometry

We propose a novel regularization term that is computationally efficient and facilitates on-the-fly optimization of Gaussians. SuGaR [61] has explored aligning Gaussians to the surface by minimizing a Signed Distance Function (SDF)-based regularization term $\mathscr{R}$, which requires precomputing the SDF before appearance modeling, as shown in Fantasia3D [30]. However, such methods introduce additional computational overhead and are not well-suited for score distillation strategies. Additionally, existing approaches often apply regularization as a post-processing step rather than integrating it into the optimization process.

In contrast, our method directly optimizes Gaussian alignment during training, improving efficiency and adaptability.

Suppose we have a true surface of the 3D scene described by a given text prompt; we want the Gaussians to lie on the surface to capture fine details and intricate geometries, resulting in high-fidelity reconstructions. For any point $x \in \mathbb{R}^3$ on the surface we can find the Gaussian g^* that has the most significant influence on the appearance of x:

$$g^* = \arg\max_g \left[\sigma_g \exp\left(-\frac{1}{2}(x-\mu_g)^T \Sigma_g^{-1}(x-\mu_g) \right) \right]. \tag{8.1}$$

Ideally, we want the center of g^* to be close to the surface point x, i.e., $\mu_g^* \approx x$, so that the exponent approaches zero:

$$T = (x-\mu_{g^*})^T \Sigma_{g^*}^{-1}(x-\mu_{g^*}) \to 0. \tag{8.2}$$

Minimizing this exponent term encourages the Gaussians to be close to the surface. When a Gaussian lies on the surface, it should be flattened to accurately represent the geometry of the 3D object. Therefore, one of the three scales of the Gaussian g^* should be close to 0. We can express Σ_{g^*} in terms of its eigenvalues and eigenvectors as

$$\Sigma_{g^*} = U \Lambda U^T,$$

where U is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues.

$$U = [\mathbf{v}_1\ \mathbf{v}_2\ \mathbf{v}_3]; \quad \Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}. \tag{8.3}$$

The inverse of the covariance matrix Σ_{g^*} is given by

$$\Sigma_{g^*}^{-1} = U \Lambda^{-1} U^T = \begin{pmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{pmatrix} \begin{pmatrix} \frac{1}{\lambda_1} & 0 & 0 \\ 0 & \frac{1}{\lambda_2} & 0 \\ 0 & 0 & \frac{1}{\lambda_3} \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \mathbf{v}_3^T \end{pmatrix}. \tag{8.4}$$

Substituting into the exponent term T in Eq. (8.2), we get

$$T = \sum_i \frac{1}{\lambda_i}(x-\mu_g)^T \mathbf{v}_i \mathbf{v}_i^T (x-\mu_g), \tag{8.5}$$

where $\mathbf{v}_i$ are the eigenvectors in U.

Let $j \in \{1,2,3\}$ be the direction of the smallest scale of a Gaussian, whose corresponding eigenvalue λ_j and the scale s_j is also minimal. Consequently,

when the Gaussian is nearly flat (i.e., s_j is small thus λ_j is small), the exponent is dominated by this direction and can be approximated by the term

$$T \approx \frac{1}{\lambda_j}(x-\mu_g)^T \mathbf{v}_j \mathbf{v}_j^T (x-\mu_g). \tag{8.6}$$

Since we minimize the projection of the points along the direction of the smallest axis, we force the Gaussians to reorient themselves via rotation and scale to minimize the projection on this axis. Additionally, to ensure that Gaussians are flattened when they align with the surface, we directly penalize the smallest scale s_g of each Gaussian. This loss term enforces that at least one of the scales collapses towards zero when a Gaussian lies on the surface. Thus, the final loss function for flattening Gaussians while ensuring they stay close to the surface is given by:

$$\mathscr{L}_s = \sum_g \left[\frac{1}{\lambda_g}(x-\mu_g)^T \mathbf{v}_g \mathbf{v}_g^T (x-\mu_g) + |s_g| \right]. \tag{8.7}$$

where λ_g, $\mathbf{v}_g$ are the eigenvalue, eigenvector corresponding to the smallest scale s_g of the Gaussian g, x is sampled from on the pseudo surface constructed from the rendered depth maps.

To enforce smoothness in both the geometry (depth) and appearance (color), we introduce the **smoothness loss**, which penalizes abrupt changes in depth and color while preserving object boundaries. The loss leverages image gradients to weight the smoothness penalty adaptively, ensuring strong regularization in homogeneous regions and reduced regularization near edges.

Given depth maps $D \in \mathbb{R}^{B\times H\times W}$ and corresponding RGB images $I \in \mathbb{R}^{B\times 3\times H\times W}$, the smoothness loss is defined as

$$\mathscr{L}_c = \sum_{i=1}^{B} \sum_{u,v} \left(w_u(u,v) \cdot \|\nabla_u D(u,v)\| + w_v(u,v) \cdot \|\nabla_v D(u,v)\| \right), \tag{8.8}$$

with

$$w_v(u,v) = \exp\left(-\|\nabla_u I(u,v)\|\right), \quad w_v(u,v) = \exp\left(-\|\nabla_v I(u,v)\|\right), \tag{8.7}$$

where $\nabla_u I(u,v)$ and $\nabla_v I(u,v)$ are the horizontal and vertical gradients of the RGB image, averaged across the color channels:

$$\nabla_u I(u,v) = \frac{1}{3}\sum_{c=1}^{3} |I_c(u+1,v) - I_c(u,v)|, \quad \nabla_v I(u,v) = \frac{1}{3}\sum_{c=1}^{3} |I_c(u,v+1) - I_c(u,v)|. \tag{8.9}$$

The depth gradients $\nabla_u D(u,v)$ and $\nabla_v D(u,v)$ measure the horizontal and vertical changes in depth values, respectively.

The smoothness loss couples color and geometry by using image gradients as proxies for scene boundaries. By weighting depth gradients inversely proportional to image gradients, the model enforces smoothness in regions with uniform color while preserving sharp transitions near edges. This adaptive weighting reduces the penalty on depth variations in high-gradient areas, ensuring that fine details in both geometry and appearance are retained.

The final loss is a weighted sum of the individual losses, balancing their contributions. The overall objective is defined as

$$\mathscr{L}_{total} = w_{sds}\mathscr{L}_{sds} + w_s\mathscr{L}_s + w_c\mathscr{L}_c. \tag{8.10}$$

In our experiments, we set the weighting parameters to prioritize the contributions of the smoothness and regularization terms relative to the SDS loss. Specifically, we use $w_{\text{sds}} = 1$, as the magnitude of the SDS loss is significantly higher than the other terms, ensuring its influence remains balanced without overshadowing other contributions. The weights for the smoothness loss and regularization loss are set to $w_{\text{s}} = w_{\text{c}} = 200$ to enforce strong geometric and appearance constraints that enhance surface fidelity and color consistency. These settings were chosen empirically to achieve high-quality reconstructions while maintaining efficient convergence.

8.2.3 Surface densification and pruning

In this section, we re-examine the densification approach used in 3DGS and discuss our strategy to overcome the limitations of the existing methods. Naive 3D Gaussian splatting methods densify the Gaussians based on the gradient of the Gaussian centers and the scales of the Gaussians. While this approach is straightforward, it presents several significant drawbacks. One of the primary challenges lies in defining an appropriate threshold value for the gradient. If the threshold value is set too high, fewer Gaussians are added to the scene, leading to a lack of detail in the reconstructed model. Conversely, if the threshold value is set too low, the number of Gaussians increases significantly. This not only hinders the learning speed but also impedes the convergence of the model due to the excessive computational load.

We propose an intuitive method that utilizes the rendered image and depth to backproject the rendered pixels to the world using camera parameters. This allows us to progressively reconstruct the surface of the 3D model. We densify the Gaussians that are close to the surface, allowing the model to gradually reconstruct the missing parts and speed up the training time due to the significantly reduced number of Gaussians to update. Mathematically, we define the backprojection of a pixel p with depth d and camera parameters K (intrinsic matrix) and $[R|t]$ (extrinsic matrix) as follows:

$$P = R^{-1}(K^{-1}p'd - t) \tag{8.11}$$

where p' is the homogeneous coordinate of p. Let $\{P_i\}$ be the set of all backprojected points. We then define the distance D_g of a Gaussian g from the surface as the Euclidean distance between the Gaussian center μ_g and the closest backprojected point:

$$D_g = \min_{P_i} \|\mu_g - P_i\| \tag{8.12}$$

We prune Gaussians for which D_g exceeds an empirical threshold $\epsilon = 0.02$. This approach allows us to significantly reduce the number of Gaussians and improve the efficiency and quality of the final 3D reconstruction.

Prompt	**Human evaluation scores**				**No. of Gaussians (in millions)**	
	GaussianDreamer	**GSGen**	**LucidDreamer**	**Ours**	**Naive**	**Ours**
A blue jay sitting on a willow basket of macarons	3.23	2.89	3.07	**4.65**	16.2	**1.1**
A flying dragon, highly detailed, realistic, majestic	2.51	2.12	3.45	**4.81**	24.5	**1.2**
An armored green-skin orc warrior riding a vicious hog	3.12	2.94	3.21	**4.56**	22.7	**1.3**
A forbidden castle high up in the mountains	3.24	2.46	2.87	**4.45**	24.2	**1.2**
A peacock standing on a surfing board, highly detailed, majestic	2.02	3.56	2.95	**4.64**	18.8	**1.1**
Jack Sparrow wearing sunglasses, head, photorealistic, 8k, HD, raw	4.01	3.21	4.15	**4.45**	16.7	**0.9**
Medieval soldier with shield and sword, fantasy, game, character, highly detailed, photorealistic, 4K, HD	3.5	2.57	3.64	**4.32**	17.3	**1.2**
A 3D model of an adorable cottage with a thatched roof	3.45	1.89	3.42	**3.89**	16.9	**1.2**

Table 8.1: Comparison of different methods based on the provided prompts. The table includes human evaluation scores for each method, along with the number of Gaussians (in millions) utilized by both the naive and our approach. The highest scores in each row are highlighted in bold.

8.3 Experiments and Results

We generate a variety of outputs using a diverse set of prompts and observe that our method significantly outperforms other 3DGS-based approaches within the same time constraints. Our approach not only achieves a higher level of detail but also exhibits fewer artifacts compared to existing methods.

In this work, we focus on SDS-based approaches that utilize the 3DGS representation. Existing NeRF-based approaches have been thoroughly studied and compared in previous work [32, 130, 259], demonstrating that they suffer from

slow convergence and inferior quality compared to recent Gaussian splatting-based methods. Figure 8.5 illustrates that our method outperforms ProlificDreamer, one of the state-of-the-art NeRF-based approaches, in terms of visual quality, geometric accuracy, and computational efficiency.

We primarily compare our method against several state-of-the-art 3DGS-based techniques, including GaussianDreamer [259], GSGen [32], and LucidDreamer [130]. As shown in Figures 8.2 and 8.3, our method produces brighter colors and sharper structures, achieving a photorealistic appearance. Our findings indicate that methods such as GSGen and LucidDreamer, which rely heavily on Point-e [163] initialization, struggle to produce high-quality results if the initial point cloud is suboptimal. For instance, GSGen still exhibits the Janus problem, particularly evident in the ***Jack Sparrow*** model, while LucidDreamer produces extraneous arms holding the surfboard in the ***Peacock*** model. Even in more recent methods such as GCS [129], we continue to observe Janus artifacts (e.g., in the ***Jack Sparrow*** example), along with weaker prompt alignment, such as the missing basket in the ***Blue Jay*** prompt.

In contrast, our method excels at generating photorealistic results with a higher level of detail compared to other approaches and better prompt adherence. We observe that Point-e initialization often leads to missing structures, such as the absence of the hog in the ***Green Orc*** model and the lack of mountains in the ***Castle*** model generated by GSGen. As shown in Figures 8.2 and 8.3, our method consistently uses approximately 1M Gaussians to render higher-quality details, while other methods require roughly $15 - 20$ times more Gaussians to achieve similar results, as shown in Figure 8.4. Furthermore, in most cases, our method achieves a higher CLIP score.

Our training process involves $10,000$ steps, with the densification process starting at $1,000$ steps and occurring every 200 iterations on an NVIDIA A100 GPU. The code is written in PyTorch, and the entire process takes approximately 25 minutes per prompt.

8.4 User Study

To further evaluate the performance of our method, we conducted a user survey with 42 participants, as shown in Figure 8.6. Each participant was asked to rate eight outputs generated by different text-to-3D models on a scale from 1 to 5 (higher is better). As shown in Table 8.1, the average human evaluation scores demonstrate that our method significantly outperforms other methods by a wide margin. Additionally, participants were asked to assess the models based on visual quality, geometry, and prompt alignment of the generated content. As illustrated in Figure 8.6, our method achieved superior scores across all metrics, with average ratings of 4.45 for visual quality, 4.65 for geometry, and 4.78

for prompt alignment. In contrast, the next best-performing method, LucidDreamer, received average scores of 3.11, 2.88, and 2.45, respectively. These human evaluation results underscore our method's ability to produce more accurate and aesthetically pleasing 3D models, highlighting its effectiveness in overcoming the limitations of existing approaches. Further, we demonstrate the effectiveness of our proposed method via ablations in the Appendix in Figure C.5.

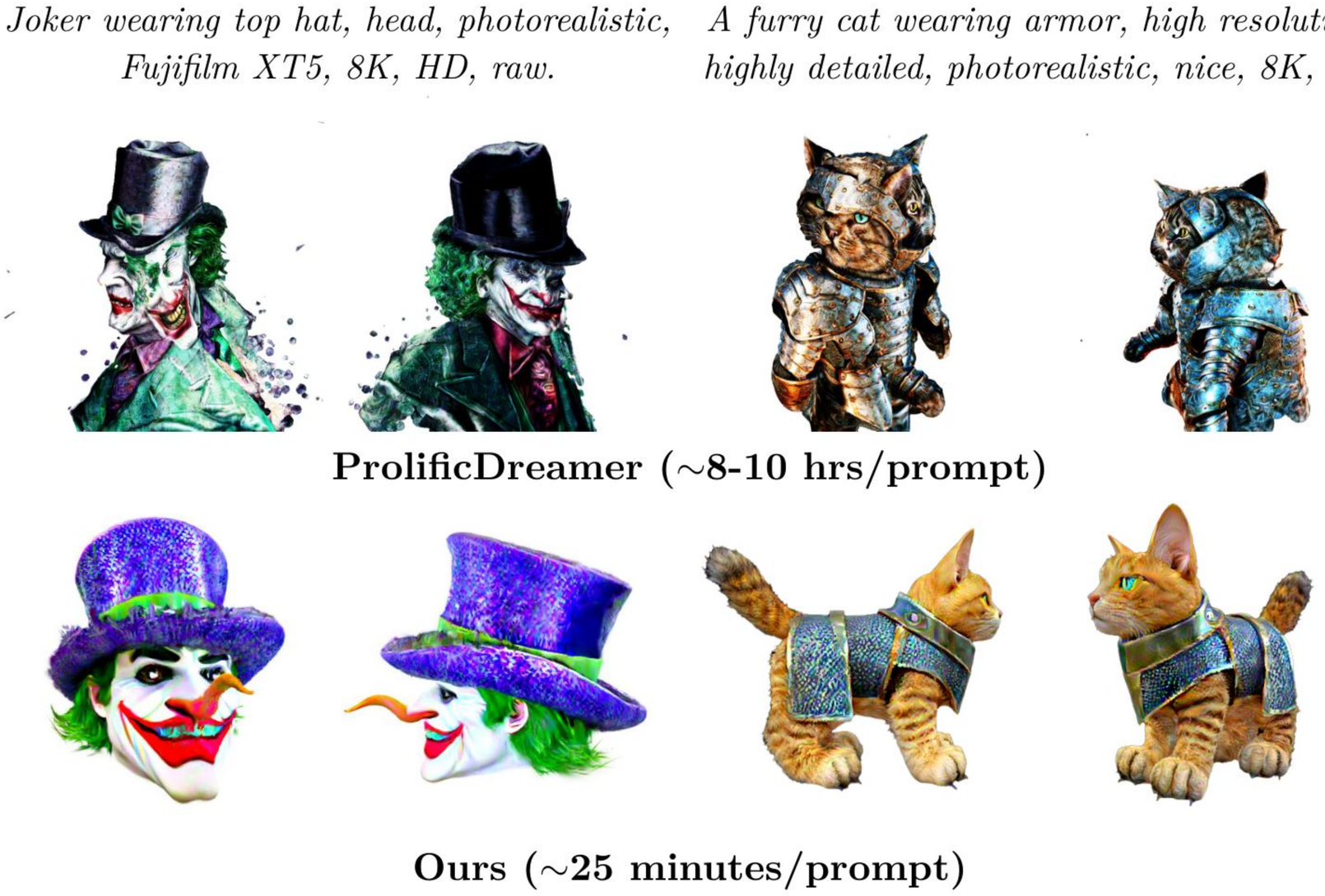


Figure 8.5: Comparison of ProlificDreamer (top, 8–10 hrs/prompt) and our 3DGS-based method (bottom, ∼25 mins/prompt). Our method produces more coherent, photorealistic results with fewer artifacts.

8.5 Limitations

While our method generates high-quality results in a relatively short time, it has several limitations. The failure cases shown in Figure 8.7 highlight issues specific to our approach, particularly the occurrence of spiky Gaussian artifacts. These artifacts, noticeable in models such as the ***Plate of cookies*** and the ***Iron Man***, may result from the flatness regularization used in our method. This regularization could inadvertently introduce sharp spikes on the model surfaces, disrupting their smoothness.

Additionally, the ***Warrior on a horse*** and the ***Plate of cookies*** exhibit unrealistic textures that detract from their fidelity. For instance, in the ***Warrior on a horse***, the horse's body is covered with unnatural floral or abstract

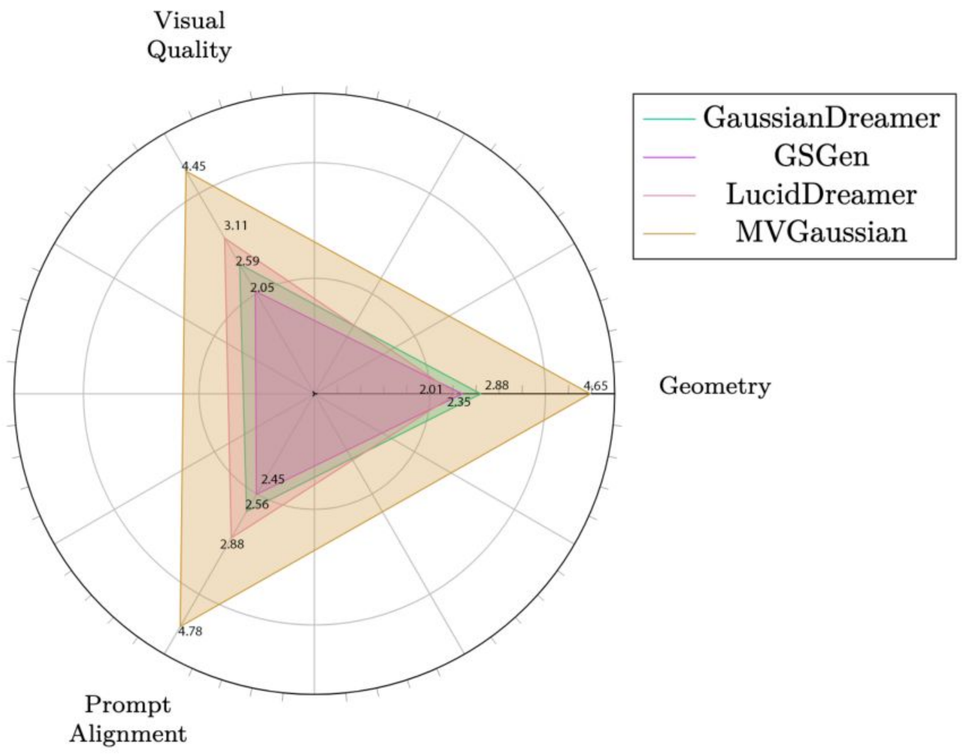


Figure 8.6: Evaluation of various aspects of the generated 3D content across different text-to-3D models based on human assessments.

patterns that do not resemble realistic fur or skin. While visually interesting, these patterns disrupt the semantic alignment with the prompt, making the output appear more like a surreal painting than a 3D render. Similarly, in the ***Plate of cookies***, the plate is rendered with an irregular, multicolored texture instead of the expected smooth white ceramic appearance. This inconsistency may stem from an over-reliance on guidance that prioritizes artistic styles over photorealism.

These issues could be mitigated with longer training, allowing the model to better converge and reduce such artifacts. Despite these occasional imperfections, our method performs better across a wider range of prompts compared to other approaches, demonstrating its robustness and effectiveness in generating high-quality 3D content.

Figure 8.7: Failure cases usually contain spike-like artifacts or irregular texture.

8.6 Conclusion

We present an intuitive and elegant method for high-quality text-to-3D renderings using depth maps without external supervision. Our approach employs the back-projection of screen space points to 3D for filtering Gaussians and leverages multi-view diffusion guidance along with surface alignment to achieve superior results. This technique not only produces higher-quality renderings in significantly less time but also demonstrates robustness across diverse text prompts. Our method generates highly detailed renderings using Gaussian splatting in under half an hour, striking an optimal balance between quality and speed, unlike other methods. This establishes a rapid, SDS-based high-quality rendering scheme.

8.7 Ethical concern and societal impact

Generative models for 3D content synthesis raise a number of ethical considerations, which apply to our method as well. Because MVGaussian uses MVDream [205] as a multi-view diffusion prior, it inherits any biases or limitations present in MVDream and its underlying models, such as Stable Diffusion [189]. As these diffusion models are trained on large-scale web data, they may reflect undesirable biases or produce outputs that are inappropriate or unlicensed. It is important to carefully consider the text prompts and model outputs, and to implement appropriate content moderation to avoid generating harmful or unethical media. We encourage responsible use of this method and adherence to ethical standards in its deployment.

Chapter 9

Conclusion

We set out to explore the exciting field of 3D reconstruction and synthesis, which has numerous applications across domains.

With advancements in science and technology, 3D methods are now at the forefront and are increasingly applicable to everyday tasks. Reconstruction and synthesis play a vital role across different modalities of 3D data that we explore in this dissertation. 3D generation has found its applications across a wide variety of modalities, from volumes to meshes. The applications are innumerable. Within the past few years, the field has seen rapid progress with the advent of deep learning-based techniques, which have significantly advanced the field and are solving longstanding problems while also introducing new challenges. As part of this thesis, we set out to solve some of these challenges. As we dive into various complexities across the field of reconstruction, we identify new directions that need further exploration.

This thesis examines the problem holistically, exploring different representations and the interlinked challenges. Throughout the thesis, we tackle the underlying difficulties associated with both geometric and radiometric sparsity, attributed to the nature of data representation.

We begin by exploring the medical domain and analyzing 3D reconstruction problems pertaining to voxel data. Our proposed method combines the classical Poisson Reconstruction and modern GAN-based methods, introducing a novel style transfer-based pipeline as an alternative to traditional transfer functions for realistic volume visualization. Although the pipeline is capable of adapting to newer tissue intensities, additional exploration is required for critical medical diagnostic tasks, such as tumor detection, to ensure more accurate and reliable colorization. Style transfer methods, though generally effective, are prone to artifacts, particularly in high-frequency details that are difficult to control. To address this, we currently restrict colorization in the LAB color space, which requires further investigation, especially with tighter coupling between the generated structures and the ground truth. After working with voxel-based data, we shifted focus to temporal data of linguistic videos, proposing linguistically informed frame interpolation. This problem was particularly challenging since it

required handling both geometric and radiometric sparsity while ensuring linguistic consistency. Our novel loss function leverages the ROI of the mouth region for faithful visemic reconstruction, making the method highly flexible and modular. The method allows for easy integration with other deep learning-based methods that focus on the visemic or similar aspects of video reconstruction. Nevertheless, we observed that while the ROI loss improves the visemic accuracy of the generated images, the 3D convolution-based model is limited to shorter sequences of videos. For longer sequences, modern state-space methods could be explored within this video generation setting using the proposed ROI loss. Addressing the above challenges with generative methods prompted us to re-examine fundamental limitations of the state-of-the-art generative methods. With the advent of StyleGANs, we explore these fundamental limitations of the underlying techniques and propose a simple yet efficient method to improve the underrepresented and high-frequency details in image-conditioned StyleGAN reconstructions. This opens up several possible future directions, particularly in enhancing feature injection and achieving finer control over the generation process. However, current methods still lack interpretability and robust disentangled control over semantic features, highlighting the need for further investigation.

Moving on to other 3D modalities, we address geometric sparsity, particularly focusing on cross-section-based point cloud synthesis using a GCN-based approach, as well as geometric and radiometric sparsity in text-based 3D rendering using SDS-based methods. Methods such as GNN suffer from over-smoothing, which limits the depth of network architectures. These challenges make it harder to use SDF-based methods, which have historically relied on latent representations unique to each sample of a class [170]. Scaling the training of such methods is challenging due to their graph-based nature, which necessitates larger adjacency matrices as the number of cross-sections increases. The field has further evolved rapidly with the advent of diffusion models that have demonstrated superior performance in the field of image generative modeling. Our proposed method integrates reinforcement learning within an SDS-based setting, revealing a lack of good 3D-specific aesthetic scoring functions since most existing scoring functions are trained on 2D images. This presents a promising research direction for the development of more 3D-centric scoring methods that could be used in tandem with REINFORCE-based methods such as the one proposed by us. Additionally, other scoring mechanisms can be integrated into our framework to target various aspects of the generated mesh or rendering. We also note that standard surface extraction techniques, such as marching cubes, used by these methods inherently hinder the usage of any mesh-based quality metrics that otherwise would be compatible with our optimization. A key bottleneck of such methods lies in the slow mesh extraction process. Therefore, exploring near-real-time extraction strategies could perhaps boost the capability and further improve the flexibility of such methods.

Thus, we address multiple reconstruction and synthesis problems, balancing

the depth of individual challenges with the breadth of modality coverage, including volumetric, video, image, point cloud, and text. Our overall methodology tackles reconstruction challenges across different priors and conditioning scenarios. We begin with a set of simpler problems where the input is closely aligned to the target modality, making structural consistency easier to achieve. This includes volumetric colorization using MRI and performing video reconstruction using sparse frames. We then transition into more complex settings, such as generating a point cloud from parametric curves, whereby the conditioning is based on an input representation that is farther away from the target modality. Further, we move to a trickier problem set that is learning the mapping of segmentation to images, and finally, tackling the highly challenging multi-modal reconstruction problem of text-to-3D, handling an even stronger case of geometric and radiometric sparsity. Our work contributes to the dynamic field of 3D generation by combining classical methods with modern data-driven techniques, opening new avenues for future research.

Appendices

Appendix A

Multi-Modal Medical Image Colorization

In the following sections, we discuss the additional training details and results on several new volumes.

A.1 Preventing unwanted textures in modality conversion using GANs

Stabilizing GANs in 3D is not easy, and often GANs can add unwanted textures or artifacts. When working with medical data, these artifacts can affect the perceptual quality of the final colorized volume, and this must be avoided. In our modality conversion step, we do not learn a direct mapping from MRI to cryosection (cryo) volume and instead use Poisson reconstruction to generate a Poisson gray volume, which helps the GAN learn a mapping to the modality of cryo volume while maintaining the geometry of the structures in MRI.

During the training phase of GANs, the generator tries to maximize the structural similarity (SSIM) to ensure that the geometric structures of the MRI are preserved. To ensure that the network has converged, we plot a graph of the SSIM loss versus the number of epochs. In Figure A.1, it can be seen that the SSIM loss is close to -1.95. This implies that the SSIM score is close to 0.95 since the SSIM loss is given by: $\mathscr{L}_{SSIM} = -(1 + SSIM)$.

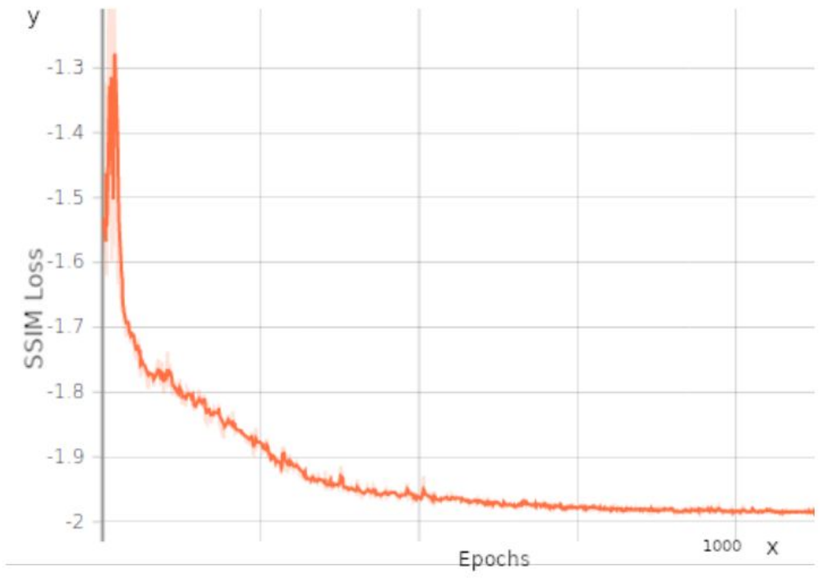


Figure A.1: Convergence of training loss.

A.2 Testing on Unknown Volumes

We test our proposed approach on sample volumes taken from the Max Planck Institute–Leipzig(MPI) Mind-Brain-Body dataset [11] and from the MIDAS dataset [23]. The two volumes have the mouth part cropped out to protect the identity of the patient. Testing on these volumes enables us to check how the GAN performs on more recent MRI volumes since it has been trained on a much older MRI dataset. We stylize the two volumes using one of the style images to show a direct comparison in perception. The results are shown in Figure A.2. The colors obtained on the volumes are still comparable to those of the style images with some minor gamma adjustment on the output volumes. Further, in Figure A.3 we show comparison with transfer function-based volume rendering. Our method improves perception and helps in making different parts more distinguishable while retaining a photorealistic appearance.

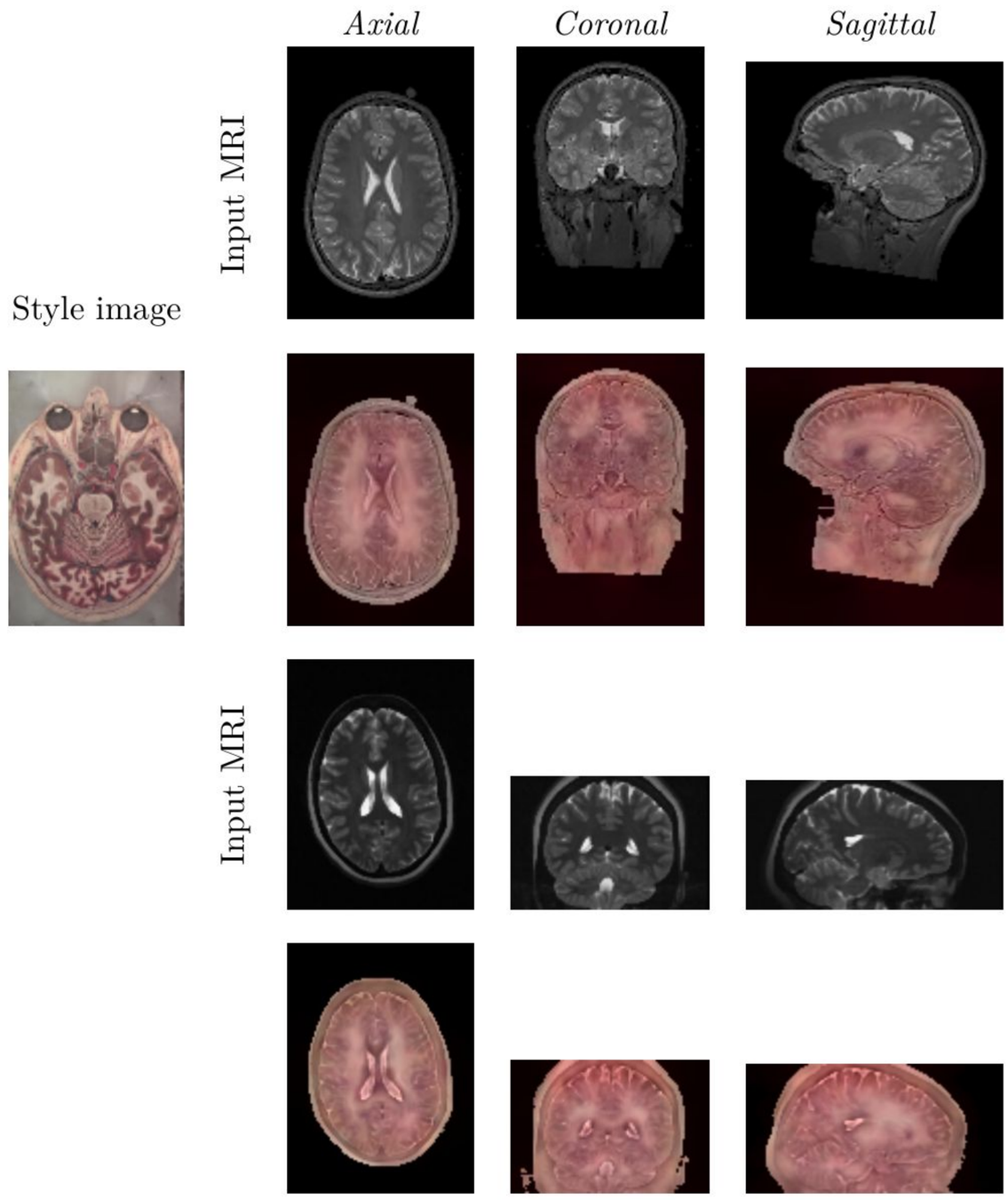


Figure A.2: Colorization result with a cryo style image on an MRI volume taken from the MIDAS dataset (row I and II) and the MPI dataset (row III and IV).

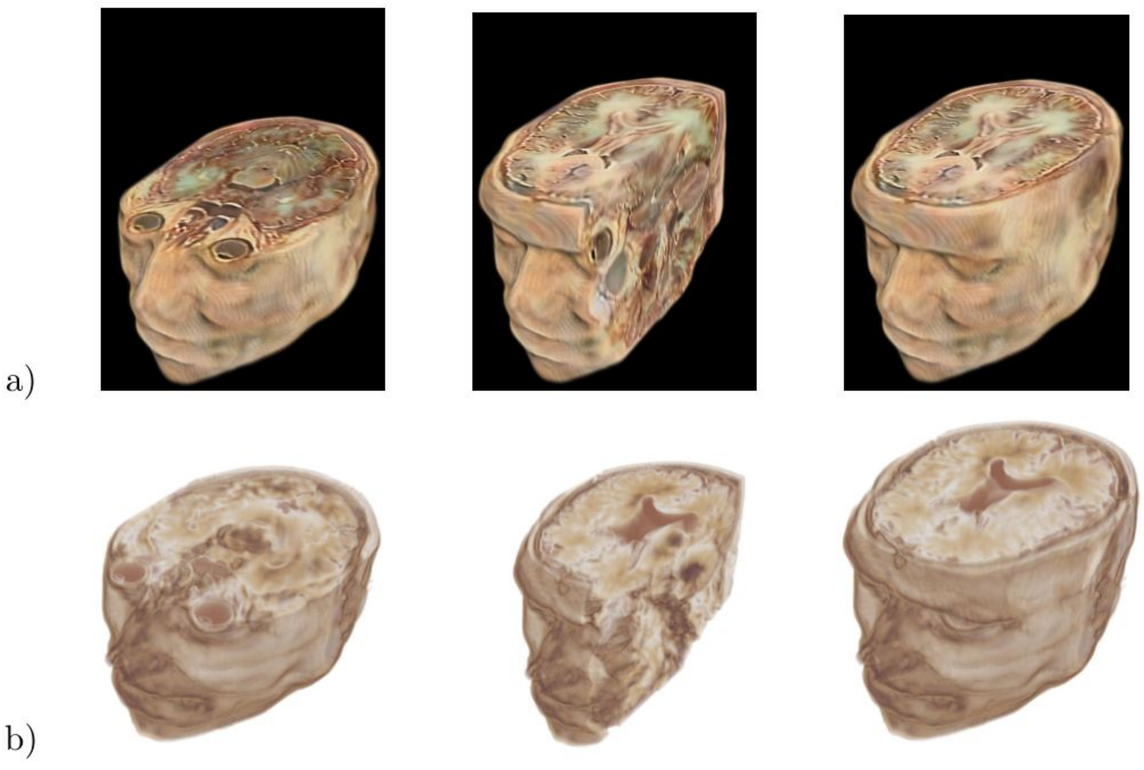


Figure A.3: a) images rendered by our method, b) images rendered using the standard transfer function. The rendering was performed using Inviwo [214].

Appendix B

SLI-pSp:Injecting Multi-Scale Spatial Layout in pSp

In the following sections, we show the additional results for both SLI-psp and ASLI across different datasets. In Figure B.1, we show additional samples for the Places2 dataset and demonstrate how our method is able to capture high-frequency details faithfully. We further demonstrate the variation in the results that our method can produce in Figure B.2 on the Seg2Face task and show a variety of generations. We observe higher variations across skin tones and hair colors in our method. Whereas in Figures B.3 & B.4 we show the results on the Seg2Face task, super-resolution in Figures B.5 & B.6 for 8× and 16× for the SLI and in Figures B.7 & B.8 for 8× and 16× for the ASLI followed by results on the AFHQ dataset in B.9.

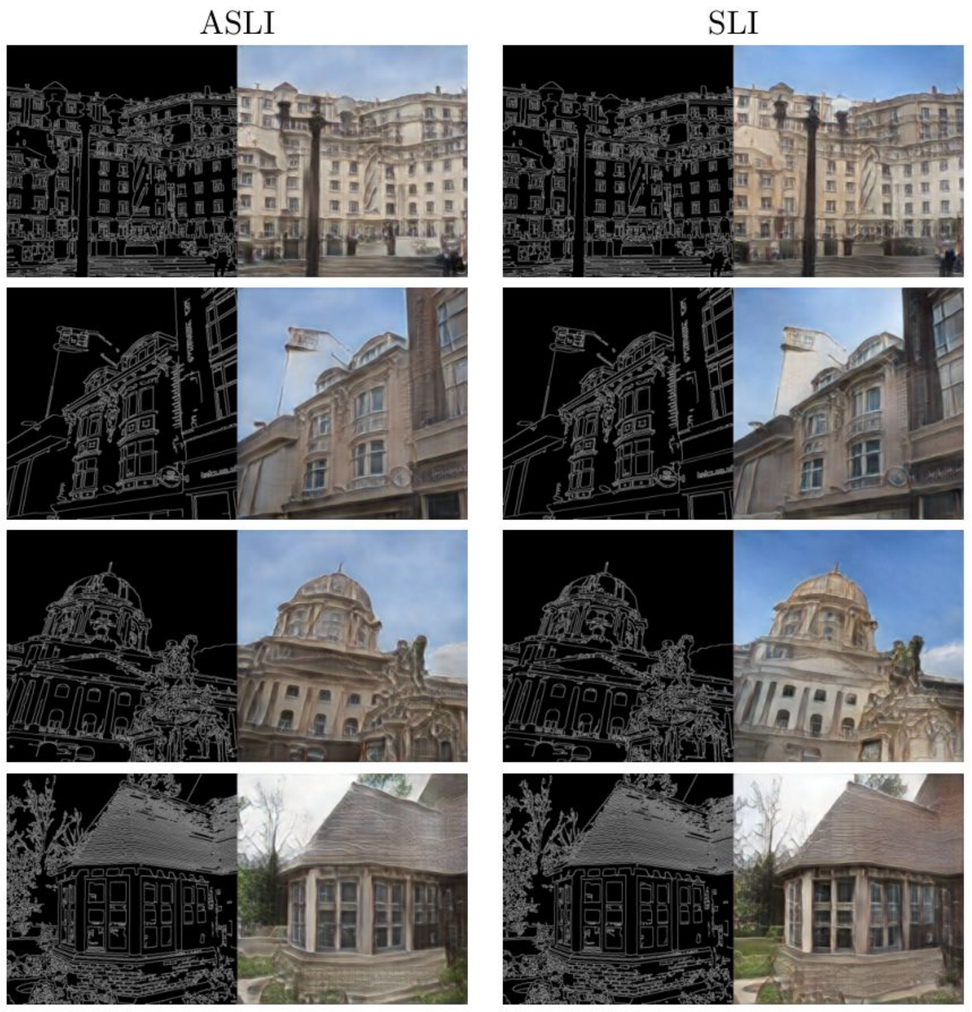


Figure B.1: Additional Results generated by ASLI(left) and SLI-pSp (right) on Places2-CustomBuildings dataset.

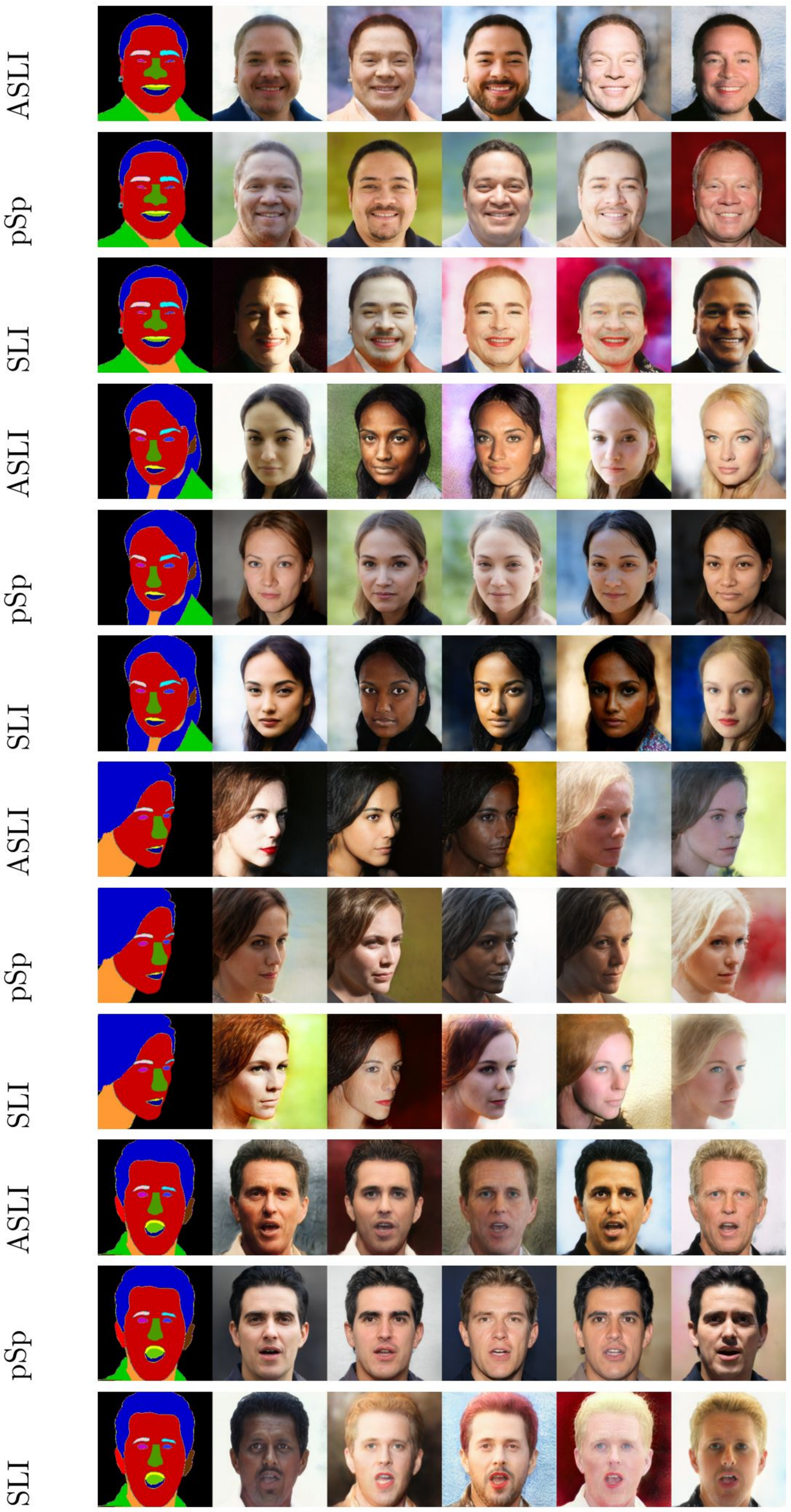


Figure B.2: Multi-Modal Synthesis Results on Seg2Face task.

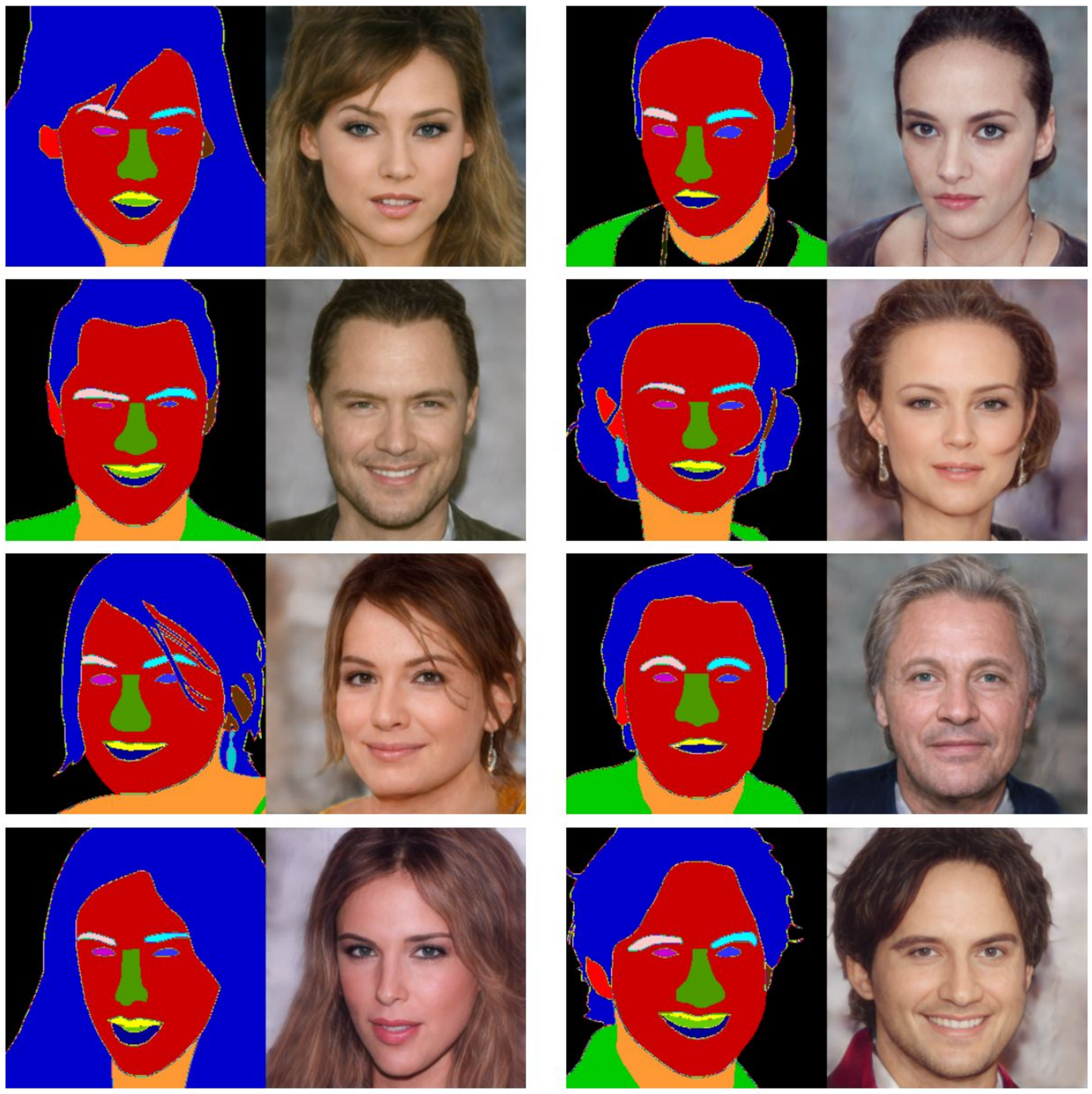


Figure B.3: Additional Results generated by SLI-pSp on Seg2Face task.

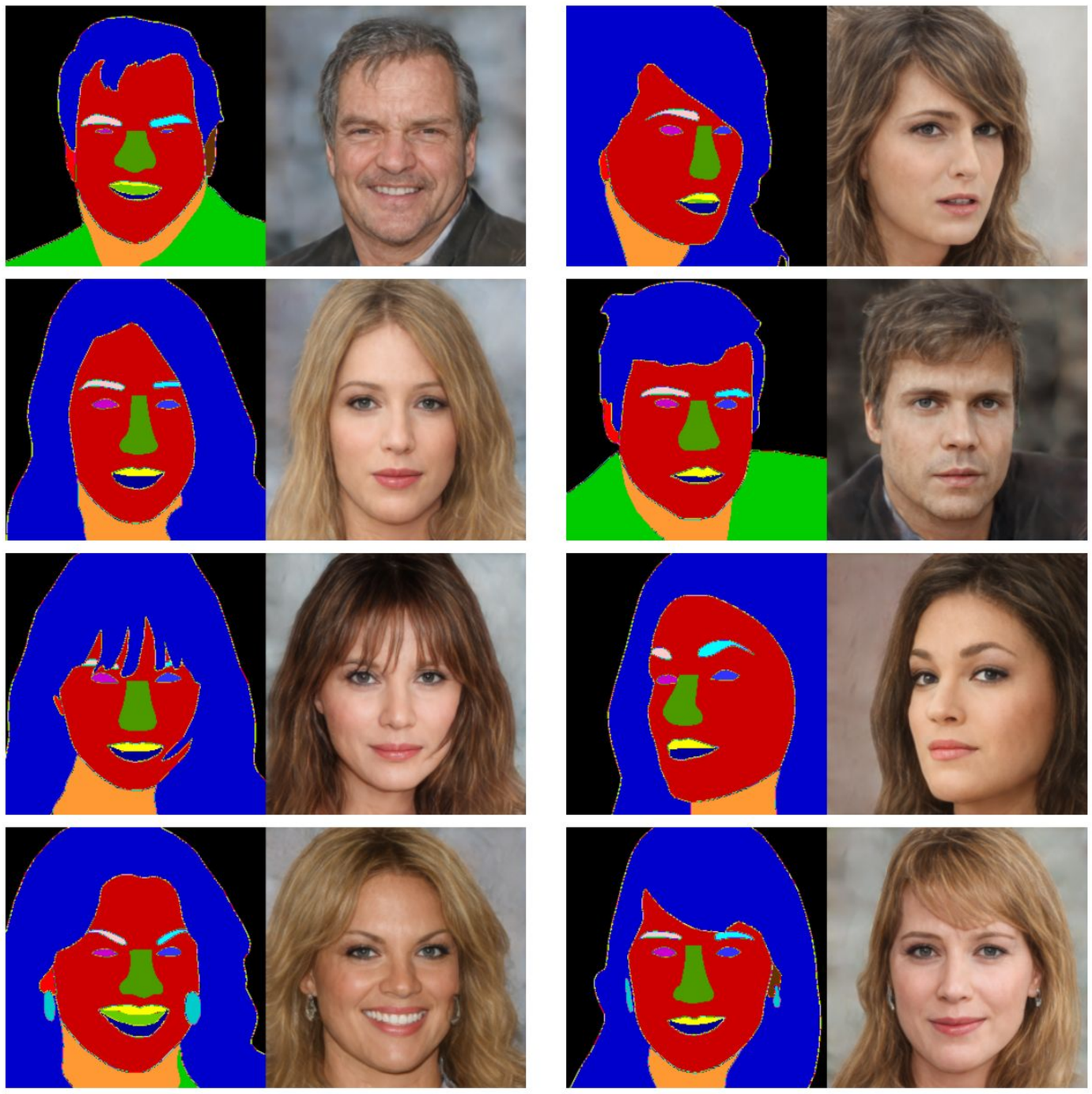


Figure B.4: Additional Results generated by ASLI-pSp on Seg2Face task.

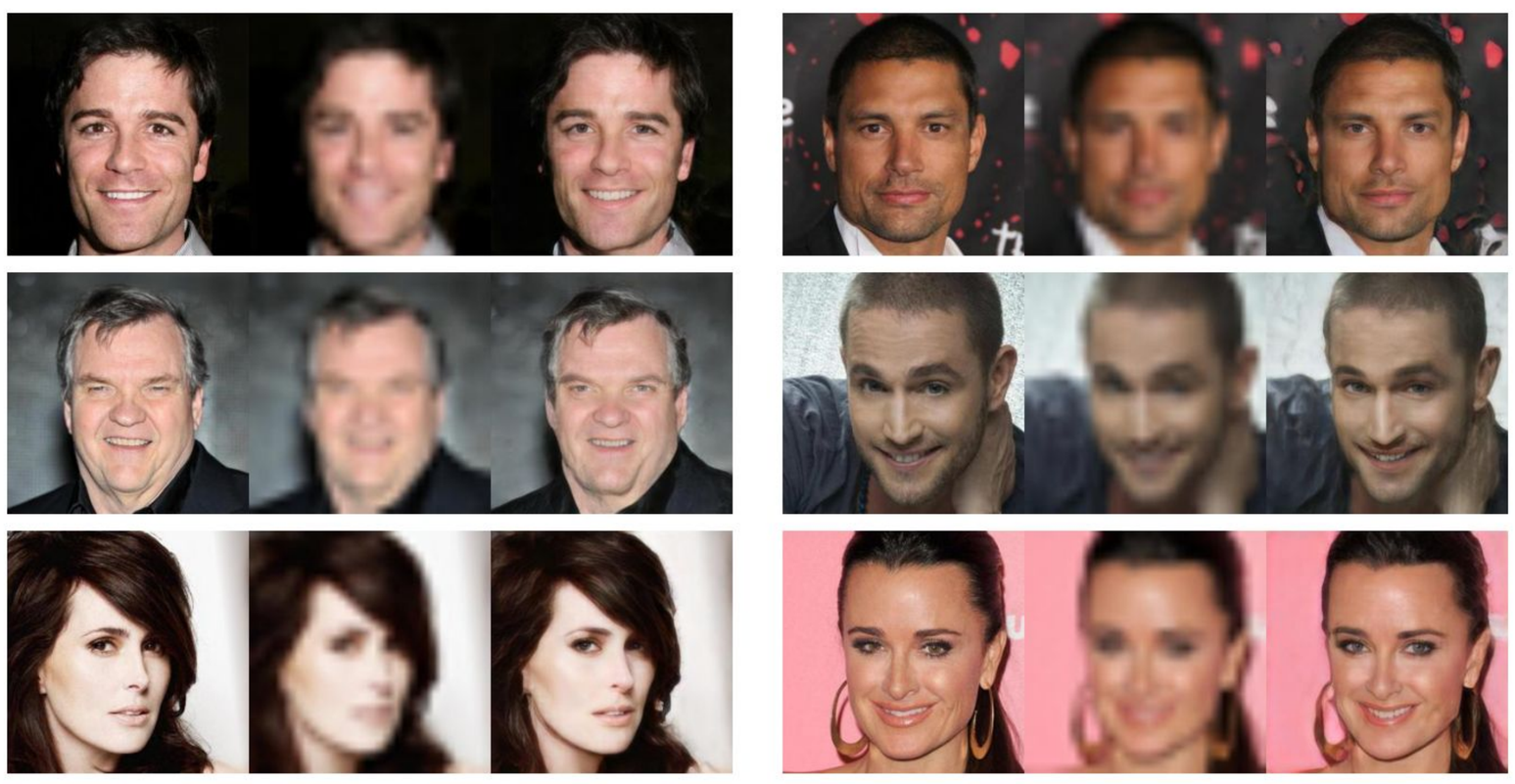


Figure B.5: Additional Results generated by SLI-pSp on Super resolution task (8x down-sampling) on CelebA.

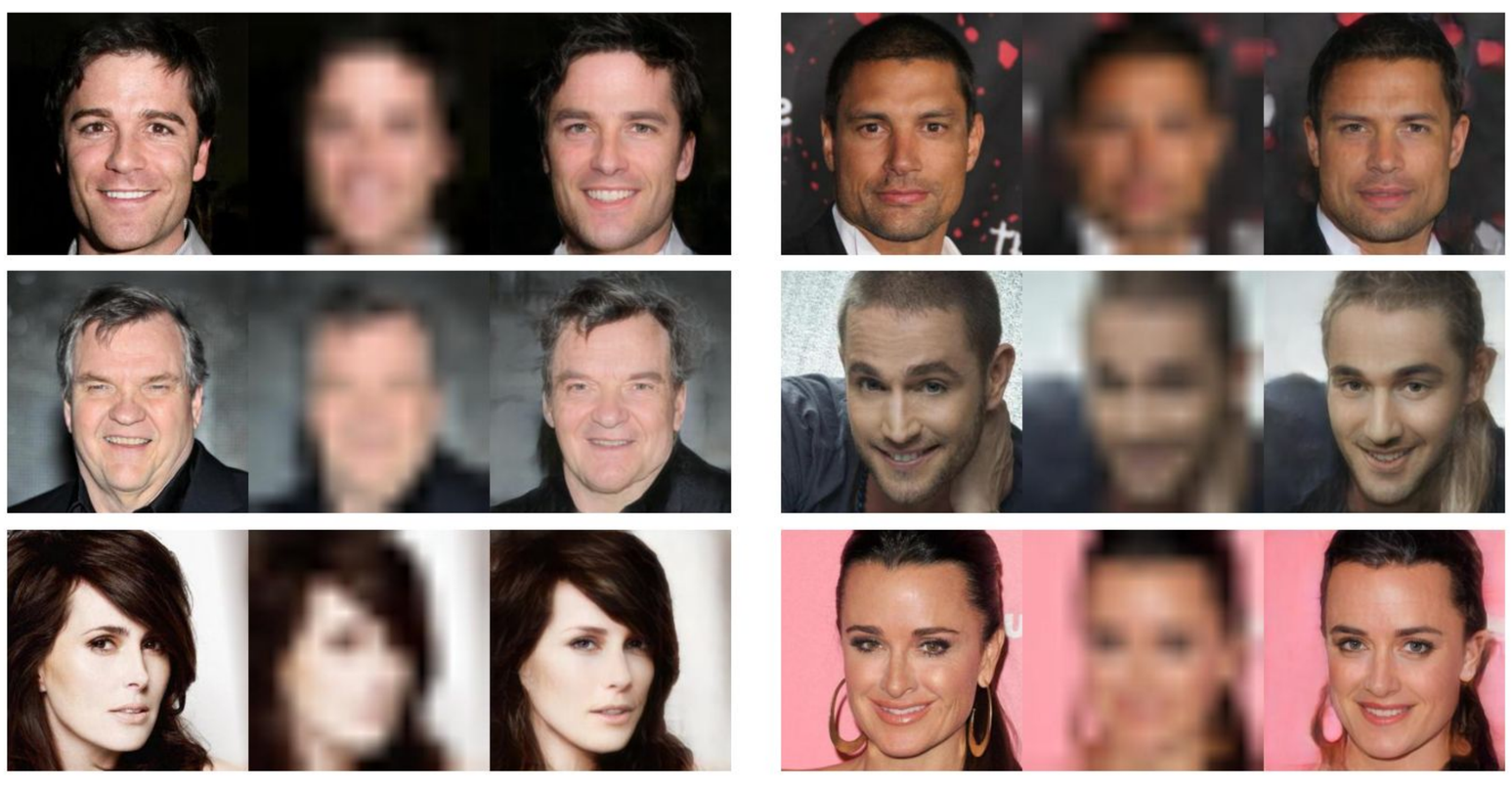


Figure B.6: Additional Results generated by SLI-pSp on Super resolution task (16x downsampling) on CelebA.

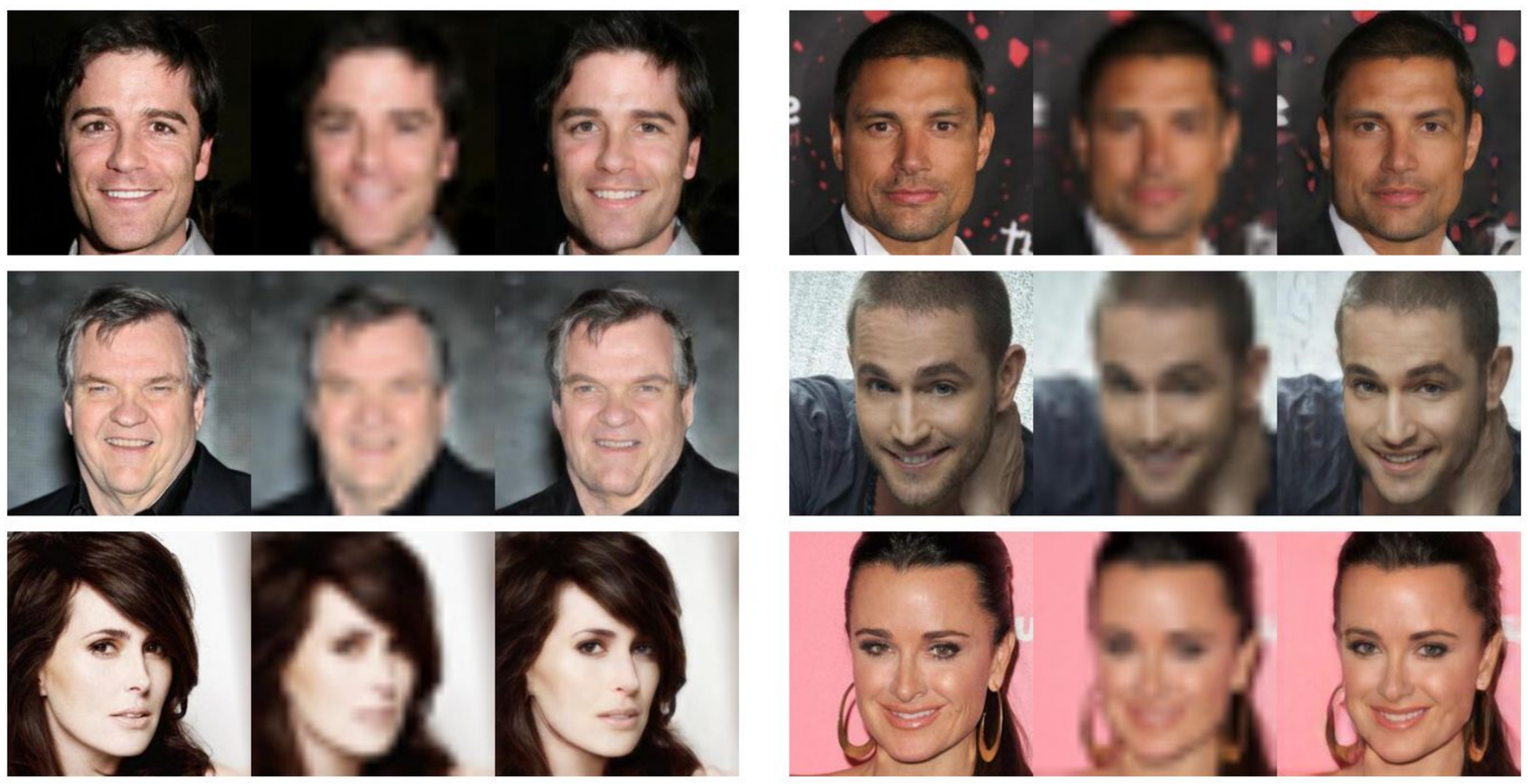


Figure B.7: Additional Results generated by ASLI-pSp on Super resolution task (8x downsampling) on CelebA.

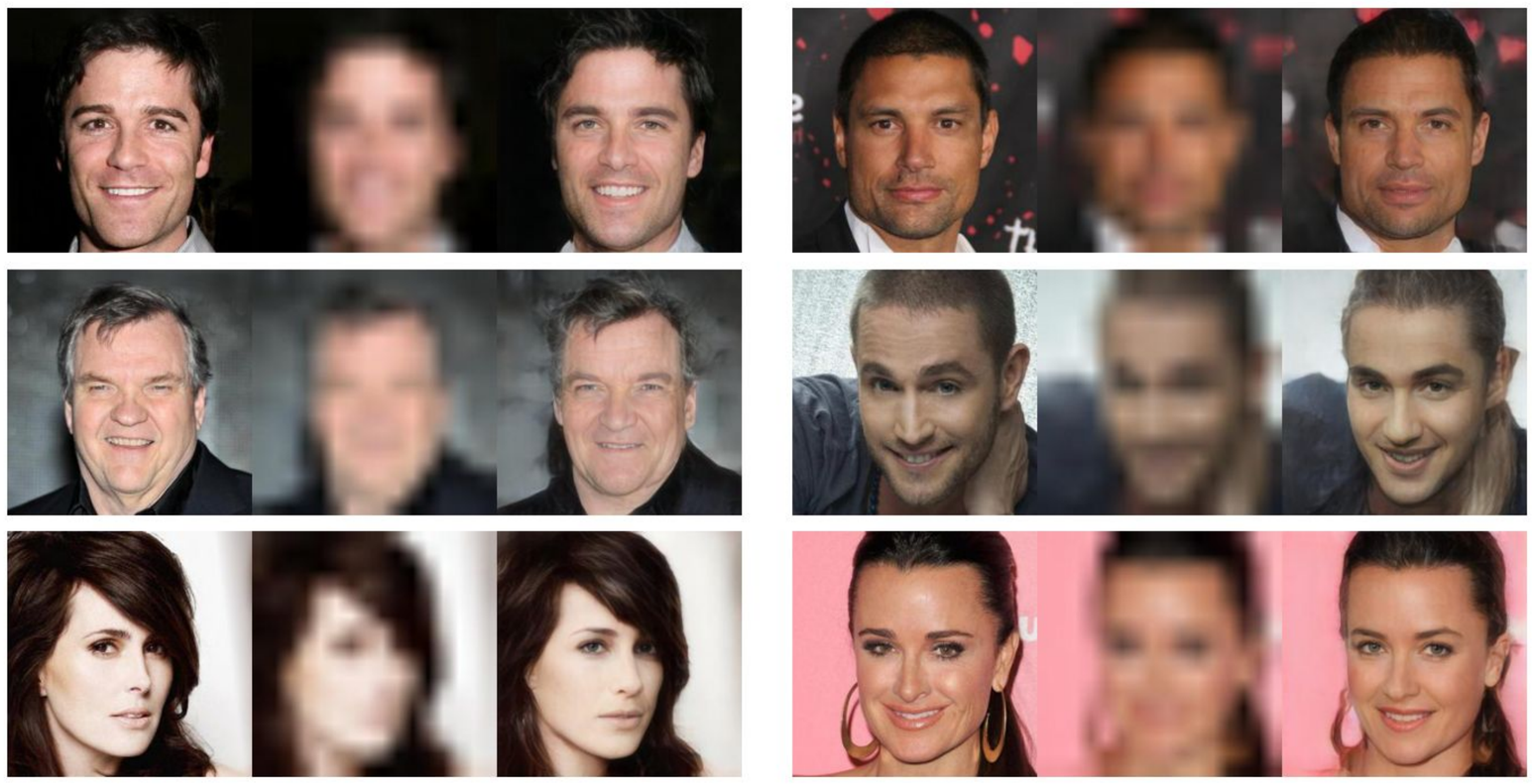


Figure B.8: Additional Results generated by ASLI-pSp on Super resolution task (16x downsampling) on CelebA.

Figure B.9: Additional Results generated by SLI-pSp for Super-resolution task (8x downsampling) on AFHQ-Dog Dataset.

Appendix C

MVGaussian: High-Fidelity text-to-3D Content Generation

C.1 Quantitative Evaluation

Prompt	CLIP score				
	GaussianDreamer	GSGen	LucidDreamer	GCS	Ours
A blue jay sitting on a willow basket of macarons	0.28	0.27	0.33	0.32	**0.34**
A flying dragon, highly detailed, realistic, majestic	0.30	0.30	0.30	0.28	**0.32**
An armored green-skin orc warrior riding a vicious hog	0.29	**0.31**	0.29	0.29	**0.31**
A forbidden castle high up in the mountains	**0.32**	0.25	0.27	0.25	0.30
A peacock standing on a surfing board, highly detailed, majestic	0.31	0.29	0.29	0.30	**0.33**
Jack Sparrow wearing sunglasses, head, photorealistic, 8k, HD, raw	0.27	**0.33**	0.29	0.29	0.31
Medieval soldier with shield and sword, fantasy, game, character, highly detailed, photorealistic, 4K, HD	0.25	0.29	0.26	0.28	**0.31**
A 3D model of an adorable cottage with a thatched roof	0.32	0.31	**0.34**	**0.34**	0.31

Table C.1: Comparison of different methods based on the given prompt. The CLIP score is computed for 15 views generated from the 3D model of each method, then averaged. We also compute the number of Gaussians for the naive method, which does not use the surface densification proposed by our method.

In Table C.1, we compare the performance of our method against GaussianDreamer [259], GSGEN [32], LucidDreamer [130] and GCS [129] using the CLIP score, averaged over 15 views generated from 3D models for each prompt shown in Figures 8.2 and 8.3. Our method consistently achieves the highest or near-highest scores across various prompts, such as ***Blue jay***, ***Peacock***, and ***Dragon***, indicating superior text-image alignment. However, despite the superior visual quality, the CLIP scores do not show a significant improvement for all prompts, and in some cases (***Castle***, ***Jack Sparrow***, ***Cottage***), our CLIP scores are

even lower. This discrepancy arises because the CLIP score, while useful for measuring 2D image-text alignment, is not a reliable metric for evaluating the performance of text-to-3D models. The CLIP score does not fully capture the fidelity, coherence, or geometric accuracy of the 3D models across different views, leading to potential underestimation of the quality improvements introduced by our method, e.g., the Jacksparrow model, despite having a Janus problem in GSGen scores higher than ours. Therefore, additional metrics beyond the CLIP score, such as human evaluation in the form of user studies, as we conducted, are necessary to thoroughly assess the overall quality of text-to-3D model generation.

C.2 Ablation studies

Ablation on $\mathscr{L}_s$. The ablation study presented in Figure C.1 examines the impact of the surface regularization loss $\mathscr{L}_s$ on the quality and consistency of 3D reconstructions. The top row displays results generated without $\mathscr{L}_s$, while the bottom row shows outputs with $\mathscr{L}_s$ applied. The comparison highlights that models utilizing the $\mathscr{L}_s$ loss produce more refined and visually compelling results characterized by clearer details and reduced artifacts. For instance, in the ***Wine glass*** example, the model with regularization achieves smoother surfaces and more consistent transparency effects. Likewise, the ***Crown*** with regularization displays a more polished and realistic appearance, with additional details and enhanced structural elements. The ***Cottage*** and ***Jack Sparrow*** examples also benefit from regularization, showing sharper details and more accurate textures, leading to a more realistic and appealing visual representation.

Figure C.2 provides a qualitative comparison of **surface reconstruction** quality with and without the $\mathscr{L}_s$ regularization loss. The addition of $\mathscr{L}_s$ results in smoother and more consistent surfaces, as seen in the normal maps on the right side of each example. In the ***Cottage*** example, the surface reconstructed without $\mathscr{L}_s$ exhibits noisy and uneven geometry, whereas the version with $\mathscr{L}_s$ yields a cleaner, well-aligned surface. Similarly, in the ***Jack Sparrow*** example, the hat and facial details show noticeable geometric noise and irregularities without $\mathscr{L}_s$, which are substantially reduced when the regularization is applied.

Ablation on s_g term. To further analyze the contribution of the s_g term of $\mathscr{L}_s$, we conduct an ablation study as shown in Figure C.3. This experiment isolates the effect of explicitly encouraging the flattening of Gaussians along the surface normal direction. Without the s_g term, the reconstructed shapes tend to exhibit thicker, more blob-like geometry, which blurs or oversmooths fine structures. In contrast, applying the s_g term results in sharper representations of thin and delicate features. For example, in the ***Fluffy dog*** example, the fur is better captured with clear texture and volume when the flattening term is used, making it more aligned with the *fluffy* aspect of the prompt. The ***Peacock*** tail

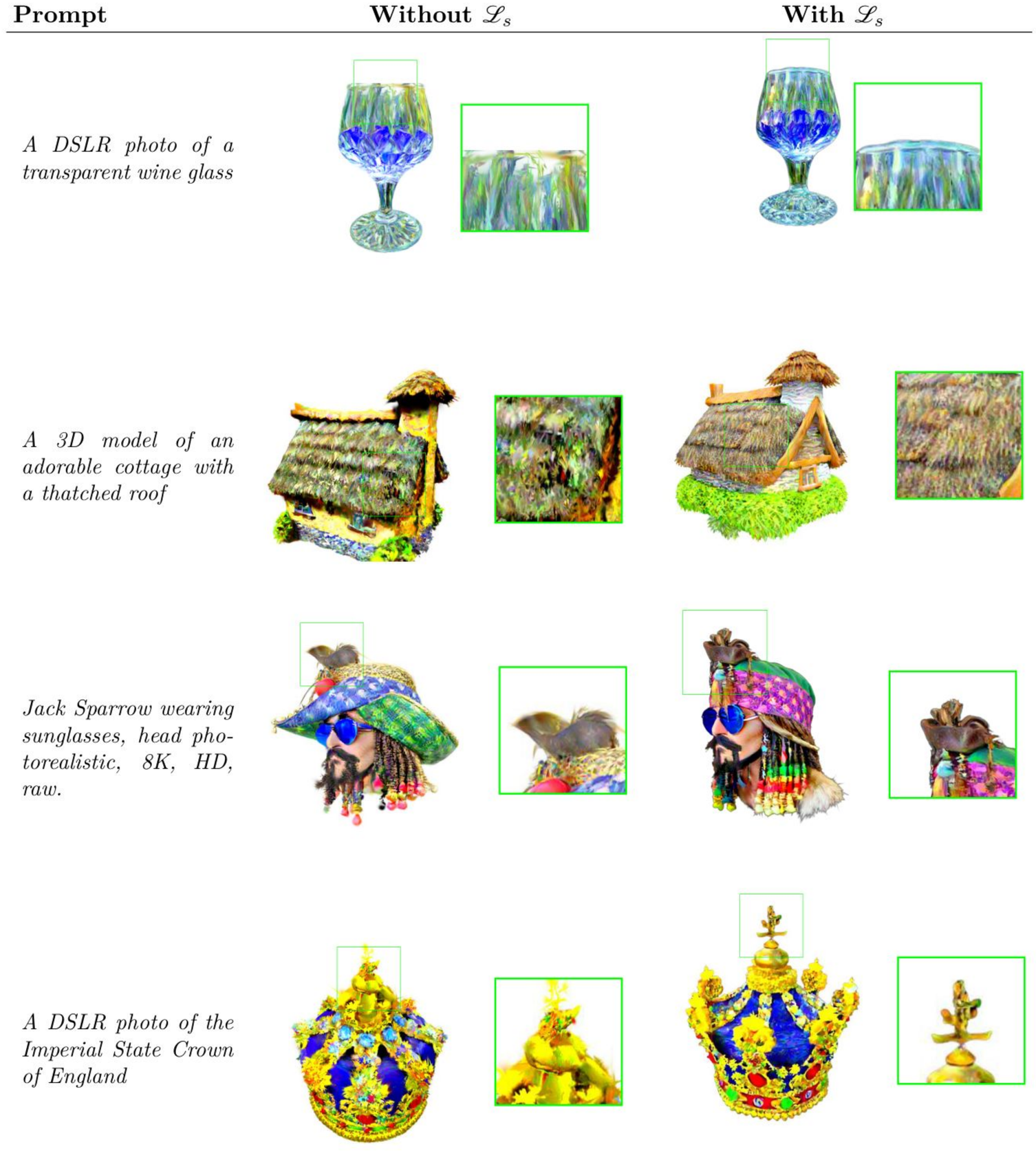


Figure C.1: Ablation study on the surface and flattening loss. We observe smoother, better textured surfaces with the $\mathscr{L}_s$ term in the loss. Each row shows the same prompt rendered without (Sample 1) and with (Sample 2) the regularization loss.

also exhibits more refined feather structures, rather than a clumped appearance. Similarly, in the ***Quill and ink*** example, the quill becomes thinner and more naturally shaped with the flattening regularization.

Ablation on $\mathscr{L}_c$. Figure C.4 presents an ablation study analyzing the effect of the smoothness regularization loss $\mathscr{L}_c$ on color consistency and saturation in 3D reconstructions. The top row shows results without $\mathscr{L}_c$, while the bottom row displays results with $\mathscr{L}_c$ applied. Without the smoothness constraint, the models exhibit noticeable color artifacts, including oversaturated and unnatural textures, as seen in the ***Elephant skull***, where the surface appears overly blue and unevenly colored. Similarly, in the ***Astronaut riding a horse***, the textures are overly contrasted, leading to unrealistic color transitions. In contrast, incorporating $\mathscr{L}_c$ effectively smooths color distributions, producing more natural and visually coherent results. For example, the snail on the leaf demonstrates improved color consistency, reducing abrupt shifts in hue while maintaining fine details. These results indicate that the smoothness loss $\mathscr{L}_c$ is essential for mitigating excessive color saturation and ensuring high-quality, realistic 3D reconstructions.

Further, in Figure C.5, we illustrate a comparison with the incorporation of 2DGS [77] and demonstrate that incorporation of our proposed regularization terms leads to sharper renderings and improved geometric structures, exhibiting both smoother and more detailed features.

Cottage without $\mathscr{L}_s$

Cottage with $\mathscr{L}_s$

Jack Sparrow without $\mathscr{L}_s$

Jack Sparrow with $\mathscr{L}_s$

Figure C.2: Qualitative comparison of surface reconstruction on two examples with and without the regularization loss $\mathscr{L}_s$.

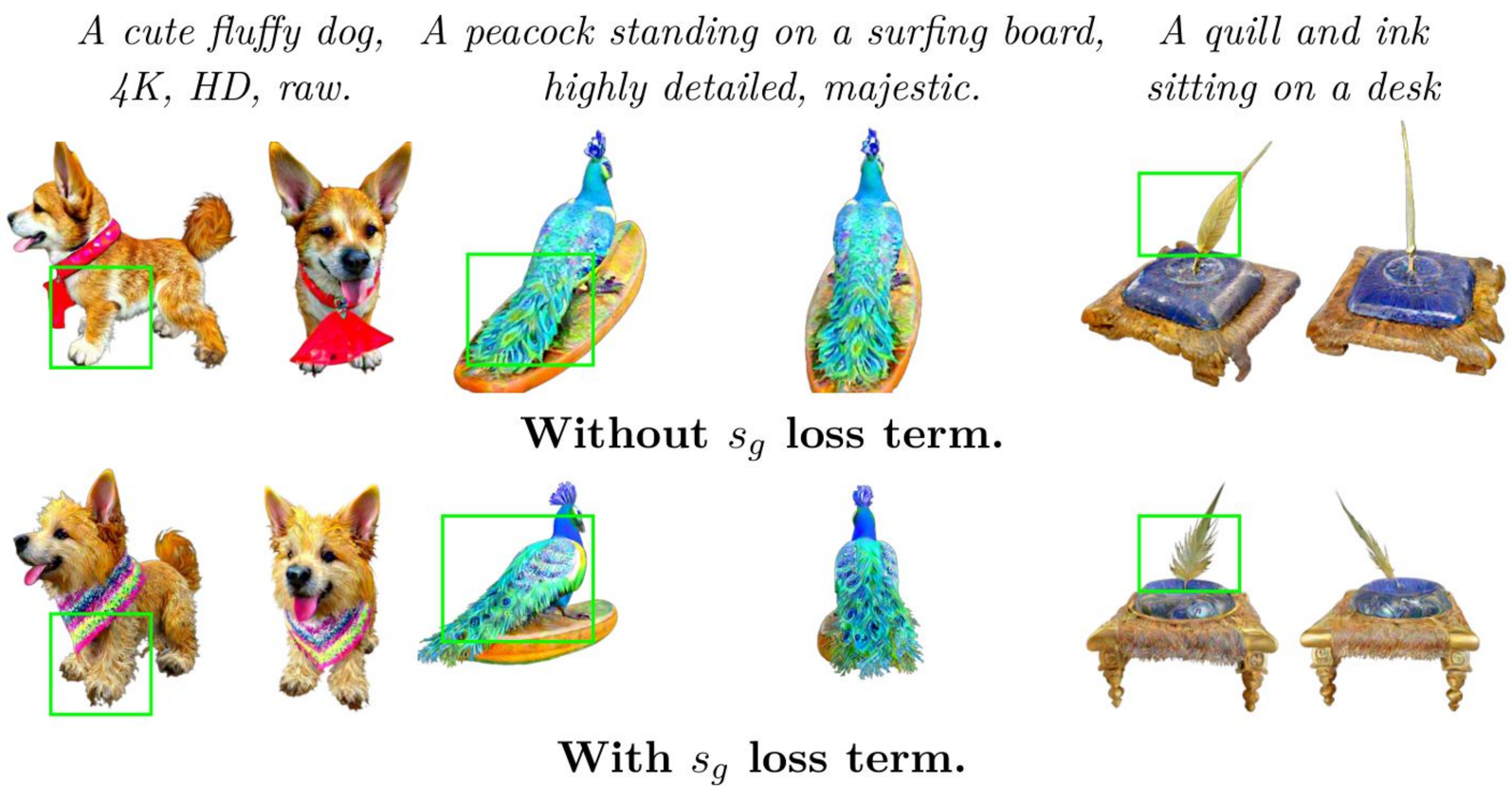


Figure C.3: Ablation study on the s_g loss term.

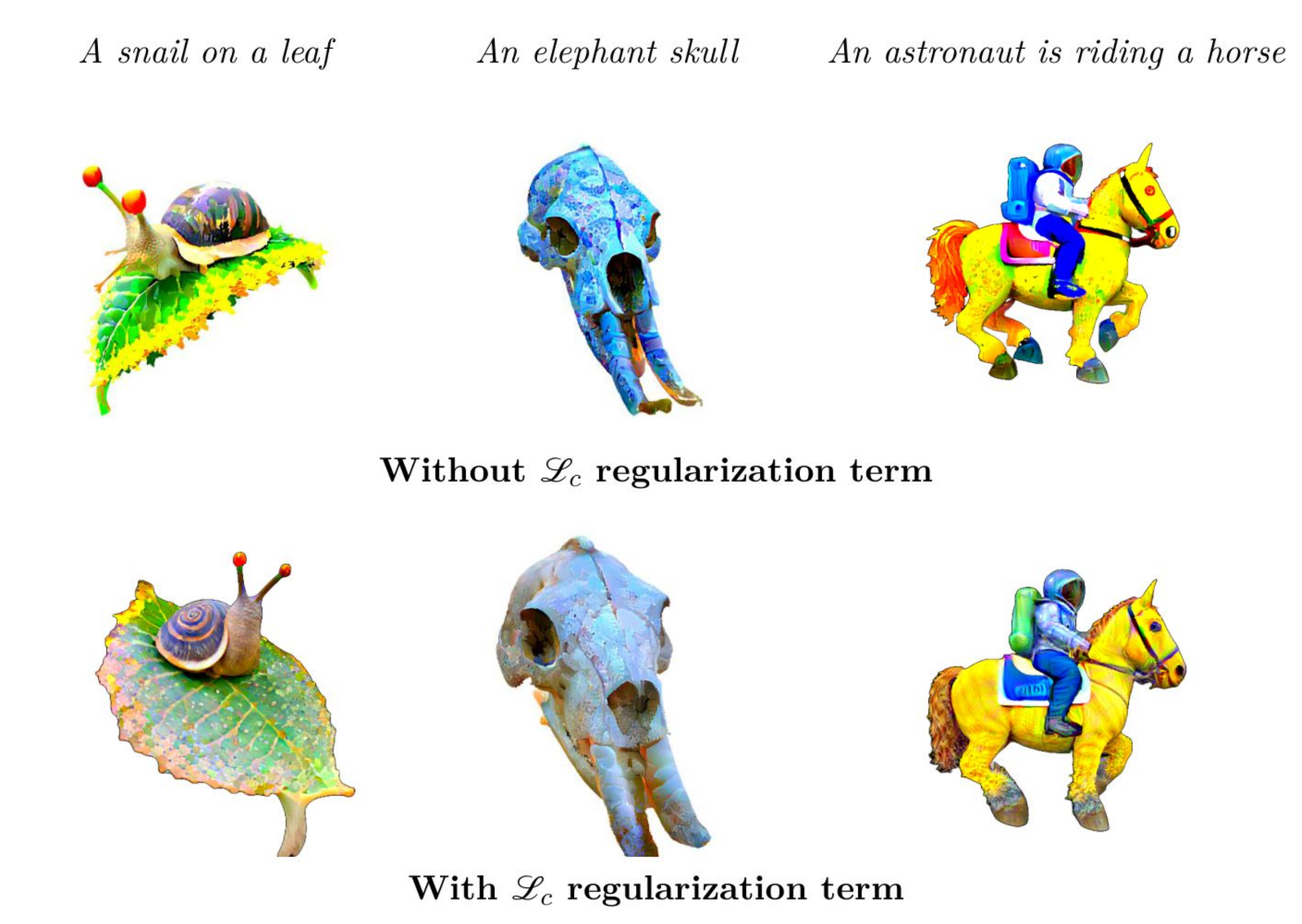


Figure C.4: Ablation study on the color and depth smoothness loss.

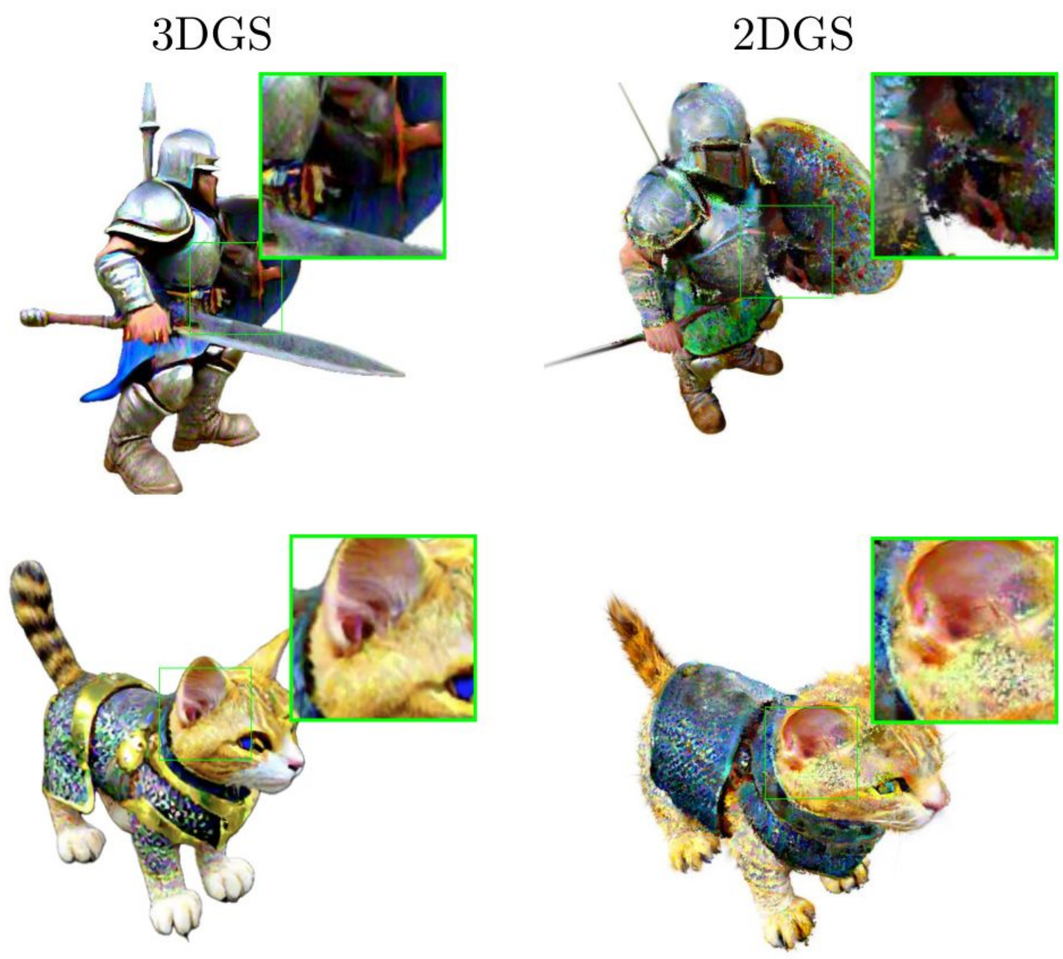


Figure C.5: Artifacts observed when densification is based on 2D Gaussian splatting (2DGS) compared to 3D Gaussian splatting (3DGS). The 2DGS method results in noticeable artifacts, particularly around regions like the hilt of the weapon and the tail of the cat, leading to a loss of details.

Pruning ablation. To evaluate the effectiveness of our densification and pruning strategy, we compare it against two baselines: (1) no densification, and (2) the standard densification approach from the original 3D Gaussian Splatting (3DGS) paper by [105]. Table C.2 summarizes the results in terms of the number of Gaussians and training time. Without any densification and pruning, the method cannot reach satisfactory quality, regardless of the initial number of Gaussians. The standard 3DGS densification produces a much larger number of Gaussians (up to 13–20 million), resulting in significantly longer training times (∼1–1.5 hours). In contrast, our method maintains a compact set of Gaussians (∼1M) and converges much faster (∼25 minutes).

Figure C.6 shows qualitative results on the prompt *A woolly mammoth.* Without densification, the generated shape remains blurry and lacks detail even with more initial Gaussians. The standard 3DGS densification improves the quality, but at the cost of a much higher number of Gaussians and slower training. Our method achieves comparable or better quality with fewer Gaussians and faster convergence.

Method	**No. gaussians init**	**Final no. gaussians**	**Time to converge**
	50k	50k	Failed
No densification	100k	100k	Failed
	1M	1M	Failed
3DGS densification	5k	∼13–20M	∼1–1.5 hrs
Ours	5k	∼1M	∼25 mins

Table C.2: Comparison of densification methods.

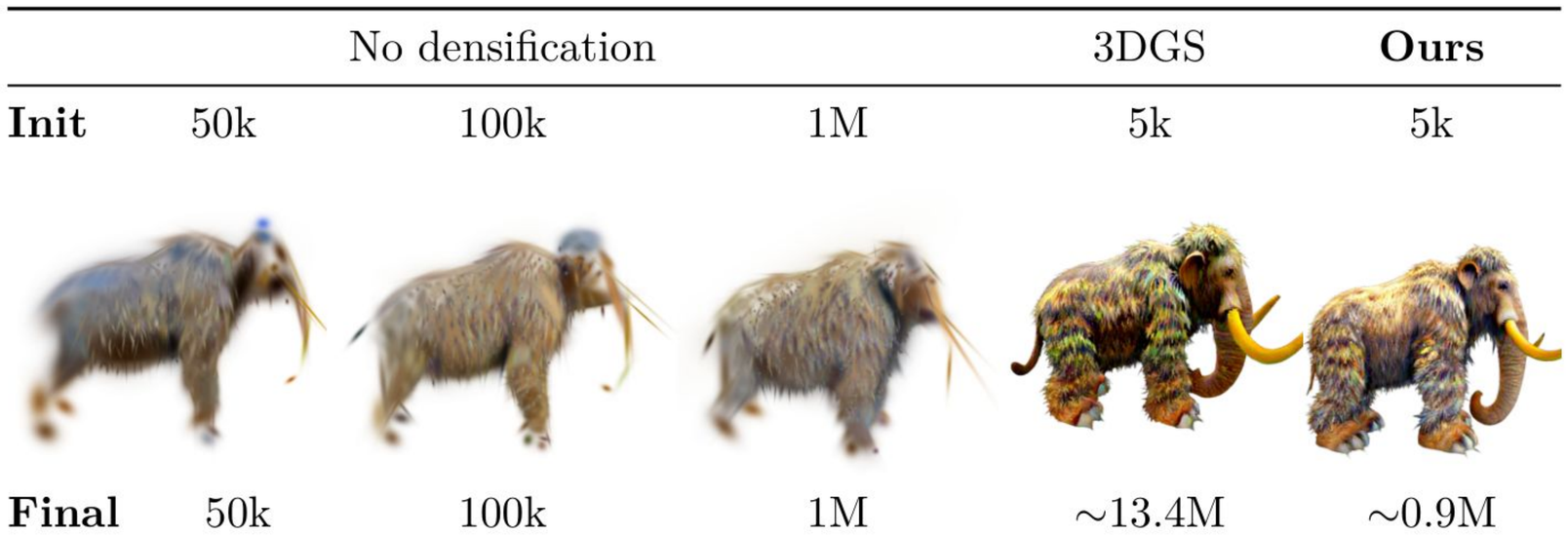


Figure C.6: Qualitative comparison of densification strategies on the prompt ***A woolly mammoth***. Our method achieves higher visual quality with fewer Gaussians and faster training time, compared to no densification or standard 3DGS densification.

C.3 More Qualitative Comparisons

Figure C.7 presents additional qualitative comparisons of our method, MVGaussian, with GaussianDreamer, LucidDreamer, and GSGEN across various prompts.

For the ***Michelangelo dog statue***, our model accurately captures both the style and the cellphone, while GaussianDreamer and GSGEN miss the cellphone, and LucidDreamer suffers from a multi-face Janus issue. In the ***Steampunk airplane*** prompt, our method integrates the steampunk aesthetic effectively, unlike other methods that produce fighter jets without the steampunk elements. For the ***Opulent couch*** prompt, GaussianDreamer, LucidDreamer, and MVGaussian produce detailed, prompt-aligned models, whereas GSGEN's output is of lower quality. In the ***Hatsune Miku robot*** prompt, our method captures the anime aesthetics, avoiding the distortions seen in other methods. Finally, in the ***Flamethrower*** prompt, MVGaussian, along with GaussianDreamer and LucidDreamer, produces detailed and cohesive models, while GSGEN's result lacks artistic detail and quality.

Overall, MVGaussian outperforms other methods in producing high-quality, detailed, and prompt-aligned 3D models.

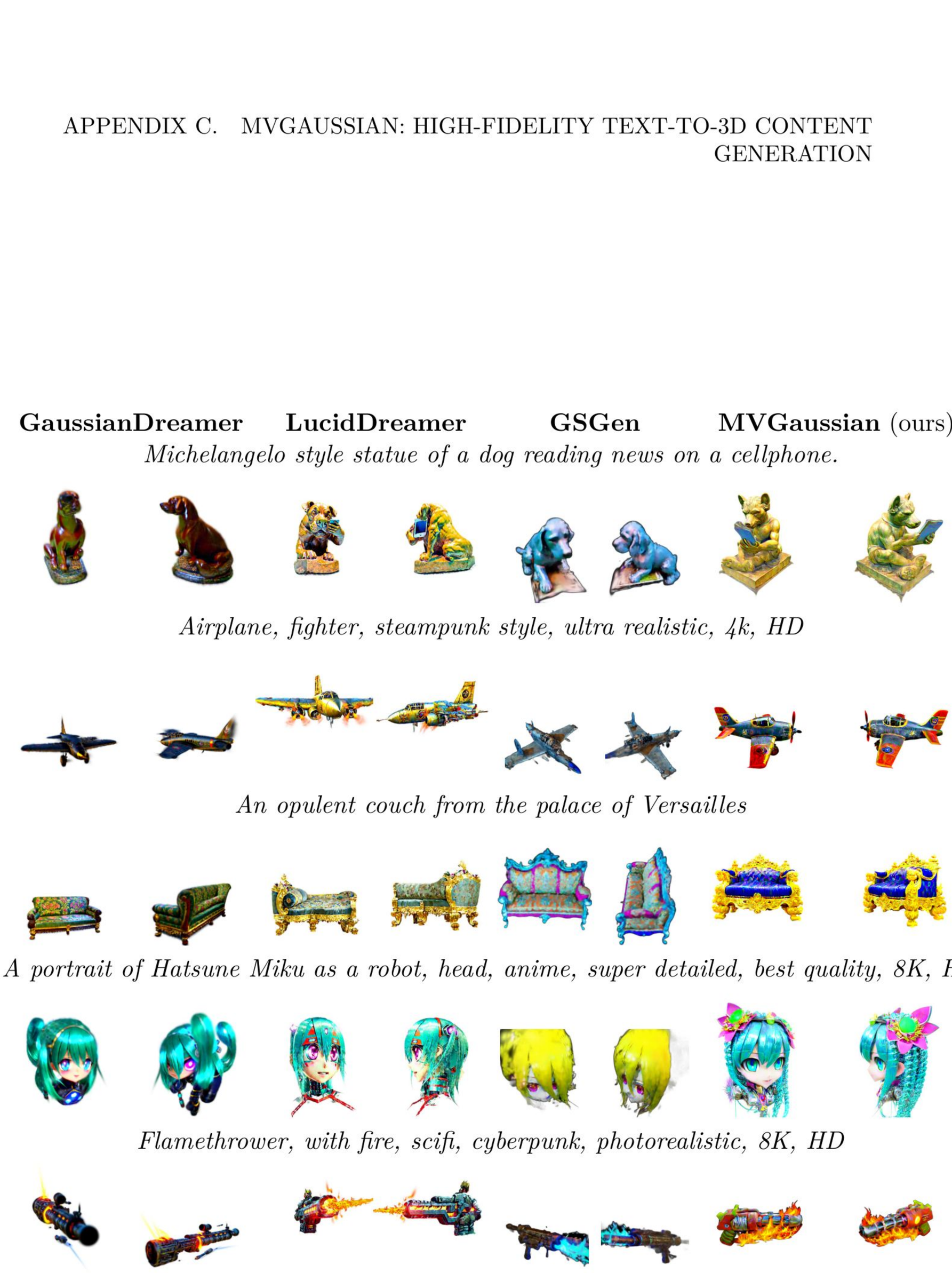


Figure C.7: Additional comparison results with existing methods.

Comparison with MVDream: Since our method leverages the same multi-view diffusion guidance as MVDream [205], we include a direct comparison to better illustrate the effects of our optimization framework. Figure C.8 presents qualitative results on several prompts. MVDream requires approximately one hour of training per prompt, yet we observe several limitations in the generated results. For the prompt ***Furry cat***, MVDream produces an irregular, overly long, and distorted tail that disrupts the structure of the cat. In the ***Hatsune Miku robot*** prompt, the output lacks any robotic elements and resembles a standard character portrait. In the ***Blue jay*** example, MVDream omits the basket entirely, placing the bird on a set of macarons instead. For the ***Michaelangelo dog statue*** prompt, MVDream generates a statue of a human figure rather than a dog. Finally, for the ***Castle*** prompt, MVDream fails to produce a plausible 3D structure, generating instead a flat cliffside scene. In contrast, our method produces more complete and geometrically consistent 3D results that more closely align with the given prompts.

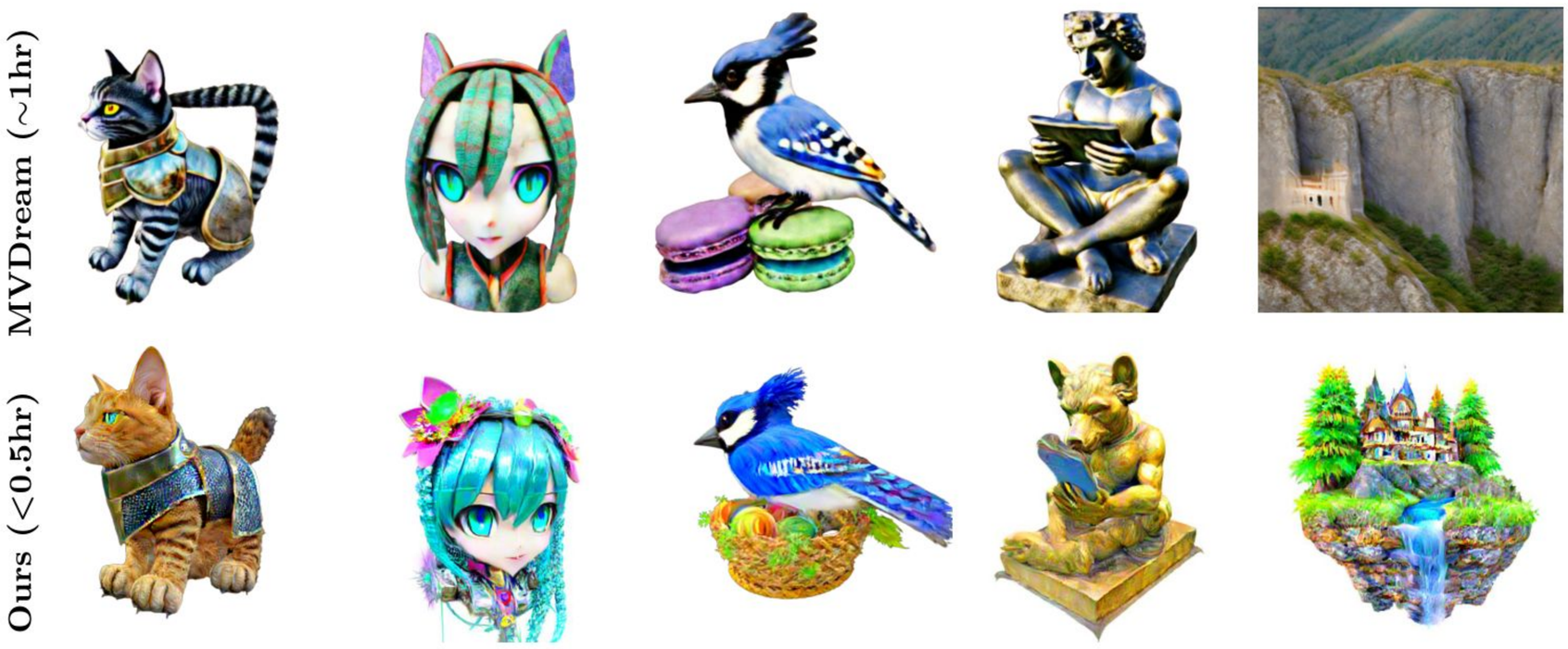


Figure C.8: Qualitative comparison with MVDream [205] on five prompts: ***Furry cat***, ***Hatsune Miku robot***, ***Blue jay***, ***Michaelangelo dog statue***, and ***Castle***.

C.4 Implementation Details

3D Gaussian Splatting (3DGS): Similar to the original 3DGS implementation [105], we primarily retain the initial learning rates for positions, color, opacity, scaling, and rotation, with adjustments to clip these values to prevent excessively small rates that could hinder convergence. The learning rate ranges are $[1.6 \times 10^{-4}, 1.6 \times 10^{-6}]$ for position, $[3 \times 10^{-3}, 2.5 \times 10^{-3}]$ for color, $[0.1, 0.05]$ for opacity, $[5 \times 10^{-3}, 1 \times 10^{-3}]$ for scaling, and $[1 \times 10^{-3}, 2 \times 10^{-4}]$ for rotation. We initialize 5000 Gaussians and train for 10000 iterations, initiating the densification and pruning process after 1000 iterations and repeating it every 200 iterations. We avoid starting densification too early or too frequently, as this

can lead to the creation of redundant Gaussians that complicate optimization. Densification and pruning are halted after 8000 iterations, allowing the final 2000 iterations to focus on optimizing the existing Gaussians.

Densification is performed by cloning or splitting Gaussians with accumulated gradients greater than 0.05, while pruning removes Gaussians with opacity below 0.05. Additionally, we apply a surface pruning technique to eliminate redundant Gaussians, retaining only those near the surface. Algorithm 1 provides pseudo code for our pruning algorithm. For each surface point, we identify the 5 nearest neighbors from the set of Gaussian centers, preserving these close-to-surface Gaussians while pruning those farther away. We also offer an option to retain a percentage p of the remaining Gaussians by calculating the distances from Gaussian centers to the surface and pruning those with distances exceeding the threshold determined by the p percentile.

Procedure 1 Surface Point Extraction and Pruning

function GET_SURFACE_POINTS($\mathscr{G}$)
  $\mathscr{P} \leftarrow \emptyset$ *// Initialize list of surface points*
  $(\theta, \phi) \leftarrow$ sample_camera_positions() *// Sample azimuth and elevation angles*
  for each (θ_i, ϕ_i) in (θ, ϕ) **do**
    $\mathbf{C} \leftarrow$ get_camera_pose(θ_i, ϕ_i) *// Compute camera pose*
    $\mathscr{I}, \mathscr{D} \leftarrow$ render_views_from_3dgs$(\mathbf{C}, \mathscr{G})$ *// Render RGB and depth images*
    $\mathbf{R}, \mathbf{T} \leftarrow$ get_cam_parameters() *// Retrieve camera intrinsics/extrinsics*
    $\mathscr{P}_i \leftarrow$ project_image2world$(\mathscr{I}, \mathscr{D}, \mathbf{R}, \mathbf{T})$ *// Back-project image points to world space*
    $\mathscr{P}_i \leftarrow$ remove_low_density_points$(\mathscr{P}_i)$ *// Remove unreliable points*
  end for
  $\mathscr{P} \leftarrow$ add_points$(\mathscr{P}_i)$ *// Aggregate extracted surface points*
  return $\mathscr{P}$
end function
function SURFACE_PRUNING($\mathscr{G}, p = 0$)
  $\mathscr{P} \leftarrow$ get_surface_points$(\mathscr{G})$ *// Extract surface points*
  $\mathscr{C} \leftarrow$ get_gaussian_centers$(\mathscr{G})$ *// Retrieve Gaussian centers*
  if $p > 0$ **then** *// Use percentile-based pruning if $p > 0$*
    $\mathscr{D} \leftarrow$ compute_knn_distances$(\mathscr{C}, \mathscr{P}, k = 1)$ *// Compute nearest surface distances*
    $\tau \leftarrow$ compute_quantile$(\mathscr{D}, p)$ *// Set distance threshold at p percentile*
    $\mathscr{M} \leftarrow \mathscr{D} > \tau$ *// Mark Gaussians exceeding threshold for pruning*
  else
    $\mathscr{I} \leftarrow$ compute_knn_indices$(\mathscr{P}, \mathscr{C}, k = 5)$ *// Find nearest Gaussian indices*
    $\mathscr{M} \leftarrow$ compute_pruning_mask$(\mathscr{I})$ *// Determine pruning mask*
  end if
  $\mathscr{G}' \leftarrow$ prune_gaussians$(\mathscr{C}, \mathscr{M})$ *// Remove pruned Gaussians*
  return $\mathscr{G}'$ *// Return remaining Gaussians*
end function

Multi-view Guidance: We utilize the pretrained model (*sd-v2.1-base-*

4view) provided by MVDream [205], which is based on the diffusion checkpoint at *stabilityai/stable-diffusion-2-1-base*. To adapt MVDream's guidance, we generate four views that are linearly distributed around the object at a random elevation. These four views are then used to compute the SDS loss as introduced in DreamFusion [179]. We also use the following negative prompt to guide the generation model: *"shadow, oversaturated, low quality, unrealistic, ugly, bad anatomy, blurry, pixelated, obscure, unnatural colors, poor lighting, dull, unclear, cropped, lowres, low quality, artifacts, duplicate, morbid, mutilated, poorly drawn face, deformed, dehydrated, bad proportions."*

The overall algorithm of our approach is outlined in Algorithm 2. It involves iteratively optimizing Gaussian parameters with integrated densification and pruning. The process includes rendering multiple views, computing losses, and refining the model by selectively densifying and pruning existing Gaussians. **Hardware Setup:** Our experiments were run on a system equipped with an NVIDIA A100 Tensor Core GPU with 80 GB of VRAM, supported by two AMD EPYC 7543 32-Core Processors. The system supports a total of 128 CPUs.

Procedure 2 Overall Optimization Process

function OPTIMIZE_GAUSSIANS($\mathscr{G}, N$)
    Initialize optimizers for Gaussian parameters and loss weights
    for each $i \in \{1, \ldots, N\}$ **do**
        $\Theta \leftarrow$ sample_camera_positions(4) *// Randomize four camera positions*
        $\mathscr{B} \leftarrow$ randomize_background_color()
        $\mathscr{I}, \mathscr{D} \leftarrow$ render_views_from_3dgs($\Theta, \mathscr{G}, \mathscr{B}$)
        $\mathscr{L}_{\text{SDS}} \leftarrow$ compute_sds_loss($\mathscr{I}, \mathscr{D}$)
        $\mathscr{L}_{\text{reg}} \leftarrow$ compute_regularization_losses($\mathscr{G}$)
        $\mathscr{L} \leftarrow$ compute_final_weighted_loss($\mathscr{L}_{\text{SDS}}, \mathscr{L}_{\text{reg}}$)
        perform_backpropagation($\mathscr{L}$)
        if check_densification_requirement() **then**
            $\mathscr{P} \leftarrow$ get_surface_points($\mathscr{G}$)
            $\mathscr{G} \leftarrow$ surface_pruning($\mathscr{G}$)
            if compute_gaussian_distance($\mathscr{G}, \mathscr{P}$) $> \tau$ **then**
                remove_gaussians($\mathscr{G}$)
            end if
            if check_opacity_reset_requirement() **then**
                reset_opacity($\mathscr{G}$)
            end if
        end if
        perform_optimization_step($\mathscr{G}$)
    end for
    return $\mathscr{G}$
end function

C.5 Parameter analysis

We evaluate the sensitivity of w_s and w_c by running each configuration on 8 prompts listed in Table 8.1. The loss weights are relative, so we fix $w_{\text{sds}} = 1$ and vary only w_s and w_c to understand their effects. We report four metrics: **CLIP score** (range 0-1), which measures alignment between the generated image and the text prompt; **ImageReward** [250], a continuous score estimating image quality and relevance; **ChatGPT-4o** score, where we prompt ChatGPT-4o to rate the image from 1 to 10 based on how well the generated object matches the input prompt in terms of object shape, geometry, texture, and overall quality; and **Human preference**, based on pairwise comparisons. For all metrics, higher scores indicate better quality. As shown in Tables C.3 and C.4, extreme values of w_s or w_c lead to lower quality and alignment, while moderate values ($w_s = 200$, $w_c = 200$) often yield the better results. We also observe that while automatic metrics provide useful guidance, they do not always align with human judgment, underscoring the importance of incorporating human evaluation in tuning these parameters.

w_s	**CLIP**	**ImageReward**	**ChatGPT 4o**	**Human preference**
1	0.21	-0.15	5.86	✗
50	0.18	0.16	5.92	✗
100	**0.26**	0.32	6.54	✗
200	0.25	**0.33**	**7.83**	✓
400	0.22	0.11	6.44	✗

Table C.3: Sensitivity analysis of surface regularization weight w_s.

w_c	**CLIP**	**ImageReward**	**ChatGPT 4o**	**Human preference**
1	0.20	0.10	6.21	✗
50	0.27	0.15	7.34	✗
100	0.28	**0.35**	**7.50**	✗
200	**0.31**	**0.35**	7.45	✓
400	0.21	-0.05	5.33	✗

Table C.4: Sensitivity analysis of the smoothness regularization weight w_c.

Figure C.9 illustrates the effect of the pruning threshold τ, which controls the maximum distance allowed between a Gaussian and the estimated pseudo-surface. As shown with the ***Bichon frise*** prompt, a smaller τ leads to more aggressive pruning. For example, at $\tau = 0.001$, only $\sim$0.7M Gaussians remain, but the model loses many fine details and the overall surface quality degrades. In contrast, a large τ such as 0.05 retains more Gaussians ($\sim$2M), but results in slower optimization and diminishing returns in visual quality. We find that

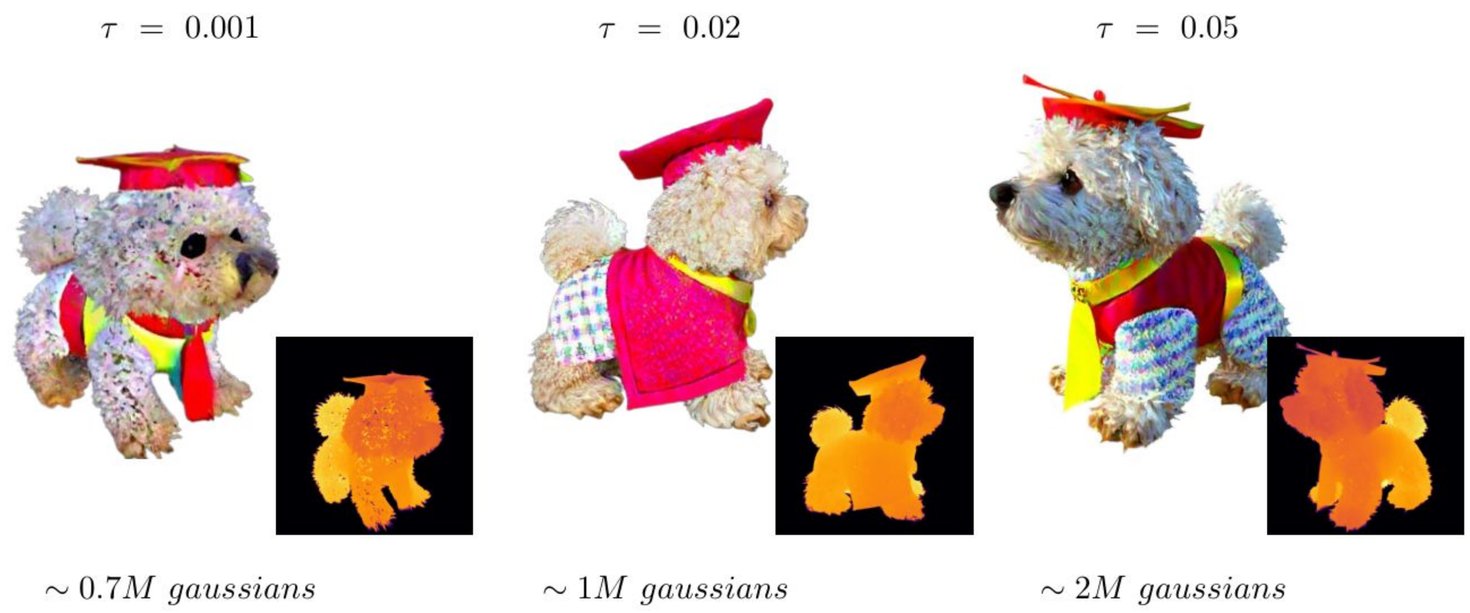


Figure C.9: Effect of pruning threshold τ on the prompt ***"a bichon frise wearing academic regalia, 8K, HD, raw."***.

$\tau = 0.02$ provides a good balance between quality and efficiency, yielding compact Gaussian sets (∼1M) while maintaining good surface fidelity and fast training.

C.6 Additional Results

Figure C.10 showcases additional results generated by our model using a generic prompt template such as ***"a DSLR photo of a ..."***. The diverse range of objects includes a castle, a cyborg, a marble bust of Captain America, a dragon, Gandalf, and so on. These examples demonstrate the model's capability to create highly detailed and visually coherent 3D representations across various subjects, from complex fantasy characters to everyday objects, illustrating the robustness and versatility of our approach.

Figure C.10: Additional results generated using generic prompt template of "a DSLR photo of a ...".

Bibliography

[1] R. Abdal, Y. Qin, and P. Wonka, Image2StyleGAN: How to embed images into the StyleGAN latent space?, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 4432–4441.

[2] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, Learning representations and generative models for 3D point clouds, in International conference on machine learning, PMLR, 2018, pp. 40–49.

[3] M. J. Ackerman, The visible human project, Proceedings of the IEEE, 86 (1998), pp. 504–511.

[4] Adobe, Ai asset statistics: About adobe. https://www.adobe.com/about-adobe.htmlhttps://www.adobe.com/about-adobe.html, 2025. Accessed: 2025-08-11.

[5] T. Afouras, J. S. Chung, and A. Zisserman, LRS3-TED: a large-scale dataset for visual speech recognition, arXiv preprint arXiv:1809.00496, (2018).

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

[7] T. Anciukevičius, Z. Xu, M. Fisher, P. Henderson, H. Bilen, N. J. Mitra, and P. Guerrero, Renderdiffusion: Image diffusion for 3D reconstruction, inpainting and generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 12608–12618.

[8] I. Anina, Z. Zhou, G. Zhao, and M. Pietikäinen, Ouluvs2: A multi-view audiovisual database for non-rigid mouth motion analysis, in 2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), vol. 1, IEEE, 2015, pp. 1–5.

[9] K. Armanious, C. Jiang, M. Fischer, T. Küstner, T. Hepp, K. Nikolaou, S. Gatidis, and B. Yang, Medgan: Medical image translation using GANs, Computerized Medical Imaging and Graphics, 79 (2020), p. 101684.

[10] M. Atzmon and Y. Lipman, Sal: Sign agnostic learning of shapes from raw data, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2565–2574.

[11] A. Babayan, M. Erbey, D. Kumral, J. D. Reinelt, A. M. Reiter, J. Röbbig, H. L. Schaare, M. Uhlig, A. Anwander, P.-L. Bazin, et al., A mind-brain-body dataset of MRI, EEG, cognition, emotion, and peripheral physiology in young and old adults, Scientific data, 6 (2019), p. 180308.

[12] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 5855–5864.

[13] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, Zip-NeRF: Anti-aliased grid-based neural radiance fields, arXiv preprint arXiv:2304.06706, (2023).

[14] C. Bartz, J. Bethge, H. Yang, and C. Meinel, One model to reconstruct them all: A novel way to use the stochastic noise in StyleGAN, arXiv preprint arXiv:2010.11113, (2020).

[15] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, Blazeface: Sub-millisecond neural face detection on mobile GPUs, 2019.

[16] P. Bezier, Mathematical and practical possibilities of unisurf, in Computer aided geometric design, Elsevier, 1974, pp. 127–152.

[17] C. M. Bishop and N. M. Nasrabadi, Pattern recognition and machine learning, vol. 4, Springer, 2006.

[18] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine, Training diffusion models with reinforcement learning, arXiv preprint arXiv:2305.13301, (2023).

[19] J.-D. Boissonnat and P. Memari, Shape reconstruction from unorganized cross-sections, in Symposium on geometry processing, Citeseer, 2007, pp. 89–98.

[20] H. Bowles, K. Mitchell, R. W. Sumner, J. Moore, and M. Gross, Iterative image warping, in Computer graphics forum, vol. 31, Wiley Online Library, 2012, pp. 237–246.

[21] S. Boyd and L. Vandenberghe, Convex optimization, Cambridge university press, 2004.

[22] T. Brooks, A. Holynski, and A. A. Efros, Instructpix2pix: Learning to follow image editing instructions, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 18392–18402.

[23] E. Bullitt, J. Smith, and W. Lin, Designed database of mr brain images of healthy volunteers, MIDAS Insight-Journal, (2011).

[24] A. Cao and J. Johnson, Hexplane: A fast representation for dynamic scenes, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 130–141.

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, End-to-end object detection with transformers, in European Conference on Computer Vision, Springer, 2020, pp. 213–229.

[26] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, J. X. Su, Hao, et al., Shapenet: An information-rich 3D model repository, arXiv preprint arXiv:1512.03012, (2015).

[27] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, TensoRF: Tensorial radiance fields, in European Conference on Computer Vision (ECCV), 2022.

[28] L. Chen, Z. Li, R. K Maddox, Z. Duan, and C. Xu, Lip movements generation at a glance, in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 520–535.

[29] R. Chen, Y. Chen, N. Jiao, and K. Jia, Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 22246–22256.

[30] ——, Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation, in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2023.

[31] Z. Chen, F. Wang, Y. Wang, and H. Liu, Text-to-3D using gaussian splatting, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 21401–21412.

[32] Z. Chen, F. Wang, Y. Wang, and H. Liu, Text-to-3d using gaussian splatting, 2024.

[33] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, Stargan v2: Diverse image synthesis for multiple domains, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 8188–8197.

[34] J. S. Chung, A. Jamaludin, and A. Zisserman, You said that?, arXiv preprint arXiv:1705.02966, (2017).

[35] J. S. Chung and A. Zisserman, Lip reading in the wild, in Asian Conference on Computer Vision, Springer, 2016, pp. 87–103.

[36] K. Clark and P. Jaini, Text-to-image diffusion models are zero shot classifiers, Advances in Neural Information Processing Systems, 36 (2024).

[37] B. Curless and M. Levoy, A volumetric method for building complex models from range images, in Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996, pp. 303–312.

[38] S. U. Dar, M. Yurt, L. Karacan, A. Erdem, E. Erdem, and T. Cukur, Image synthesis in multi-contrast mri with conditional generative adversarial networks, IEEE transactions on medical imaging, 38 (2019), pp. 2375–2388.

[39] Dawson-Haggerty, trimesh, 2019.

[40] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi, Objaverse: A universe of annotated 3d objects, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 13142–13153.

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database, in CVPR09, 2009.

[42] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, Arcface: Additive angular margin loss for deep face recognition, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4690–4699.

[43] P. Dhariwal and A. Nichol, Diffusion models beat gans on image synthesis, Advances in neural information processing systems, 34 (2021), pp. 8780–8794.

[44] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929, (2020).

[45] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, Flownet: Learning optical flow with convolutional networks, in Proceedings of the IEEE international conference on computer vision, 2015, pp. 2758–2766.

[46] D. H. Douglas and T. K. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, Cartographica: the international journal for geographic information and geovisualization, 10 (1973), pp. 112–122.

[47] H. Fan, H. Su, and L. J. Guibas, A point set generation network for 3D object reconstruction from a single image, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 605–613.

[48] O. D. Faugeras and S. Maybank, Motion from point matches: multiplicity of solutions, International Journal of Computer Vision, 4 (1990), pp. 225–246.

[49] M. Fey and J. E. Lenssen, Fast graph representation learning with PyTorch Geometric, in ICLR Workshop on Representation Learning on Graphs and Manifolds, 2019.

[50] M. S. Floater and T. Surazhsky, Parameterization for curve interpolation, in Studies in Computational Mathematics, vol. 12, Elsevier, 2006, pp. 39–54.

[51] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, K-planes: Explicit radiance fields in space, time, and appearance, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 12479–12488.

[52] M. Gadelha, R. Wang, and S. Maji, Multiresolution tree networks for 3D point cloud processing, in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 103–118.

[53] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-Or, An image is worth one word: Personalizing text-to-image generation using textual inversion, arXiv preprint arXiv:2208.01618, (2022).

[54] S. Gao, X. Liu, B. Zeng, S. Xu, Y. Li, X. Luo, J. Liu, X. Zhen, and B. Zhang, Implicit diffusion models for continuous super-resolution, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 10021–10030.

[55] L. A. Gatys, A. S. Ecker, and M. Bethge, Image style transfer using convolutional neural networks, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2414–2423.

[56] L. Gondara, Medical image denoising using convolutional denoising autoencoders, in 2016 IEEE 16th international conference on data mining workshops (ICDMW), IEEE, 2016, pp. 241–246.

[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in Advances in neural information processing systems, 2014, pp. 2672–2680.

[58] ——, Generative adversarial nets, in Advances in Neural Information Processing Systems, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds., vol. 27, Curran Associates, Inc., 2014.

[59] S. Gu, D. Chen, J. Bao, F. Wen, B. Zhang, D. Chen, L. Yuan, and B. Guo, Vector quantized diffusion model for text-to-image synthesis, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10696–10706.

[60] Guangcong, Z. Chen, C. C. Loy, and Z. Liu, Sparsenerf: Distilling depth ranking for few-shot novel view synthesis, IEEE/CVF International Conference on Computer Vision (ICCV), (2023).

[61] A. Guédon and V. Lepetit, Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering, CVPR, (2024).

[62] J. T. Guibas, T. S. Virdi, and P. S. Li, Synthetic medical images from dual generative adversarial networks, arXiv preprint arXiv:1709.01872, (2017).

[63] Y.-C. Guo, D. Kang, L. Bao, Y. He, and S.-H. Zhang, Nerfren: Neural radiance fields with reflections, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2022, pp. 18409–18418.

[64] W. L. Hamilton, R. Ying, and J. Leskovec, Inductive representation learning on large graphs, in Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 1025–1035.

[65] Z. Han, X. Wang, Y.-S. Liu, and M. Zwicker, Multi-angle point cloud-vae: Unsupervised feature learning for 3D point clouds from multiple angles by joint self-reconstruction and half-to-half prediction, in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE, 2019, pp. 10441–10450.

[66] R. I. Hartley, In defense of the eight-point algorithm, IEEE Transactions on pattern analysis and machine intelligence, 19 (1997), pp. 580–593.

[67] H. Hayashi, K. Abe, and S. Uchida, Glyphgan: Style-consistent font generation based on generative adversarial networks, Knowledge-Based Systems, 186 (2019), p. 104927.

[68] M. He, D. Chen, J. Liao, P. V. Sander, and L. Yuan, Deep exemplar-based colorization, ACM Transactions on Graphics (TOG), 37 (2018), pp. 1–16.

[69] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, GANs trained by a two time-scale update rule converge to a local nash equilibrium, Advances in neural information processing systems, 30 (2017).

[70] J. Ho, W. Chan, C. Saharia, J. Whang, R. Gao, A. Gritsenko, D. P. Kingma, B. Poole, M. Norouzi, D. J. Fleet, et al., Imagen video: High definition video generation with diffusion models, arXiv preprint arXiv:2210.02303, (2022).

[71] J. Ho, A. Jain, and P. Abbeel, Denoising diffusion probabilistic models, Advances in neural information processing systems, 33 (2020), pp. 6840–6851.

[72] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans, Cascaded diffusion models for high fidelity image generation, The Journal of Machine Learning Research, 23 (2022), pp. 2249–2281.

[73] J. Ho and T. Salimans, Classifier-free diffusion guidance, in NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications, 2021.

[74] J. T. Hoe, X. Jiang, C. S. Chan, Y.-P. Tan, and W. Hu, Interactdiffusion: Interaction control in text-to-image diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 6180–6189.

[75] Y. Hong, K. Zhang, J. Gu, S. Bi, Y. Zhou, D. Liu, F. Liu, K. Sunkavalli, T. Bui, and H. Tan, Lrm: Large reconstruction model for single image to 3d, arXiv preprint arXiv:2311.04400, (2023).

[76] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, Surface reconstruction from unorganized points, in Proceedings of the 19th annual conference on computer graphics and interactive techniques, 1992, pp. 71–78.

[77] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, 2D gaussian splatting for geometrically accurate radiance fields, in ACM SIGGRAPH 2024 Conference Papers, 2024, pp. 1–11.

[78] J. Huang and C. Menq, Combinatorial manifold mesh reconstruction and optimization from unorganized points with arbitrary topology, Computer-Aided Design, 34 (2002), pp. 149–165.

[79] X. Huang and S. Belongie, Arbitrary style transfer in real-time with adaptive instance normalization, in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1501–1510.

[80] Y.-H. Huang, Y. He, Y.-J. Yuan, Y.-K. Lai, and L. Gao, StylizedNerf: consistent 3D scene stylization as stylized nerf via 2D-3D mutual learning, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18342–18352.

[81] Z. Huang, N. Carr, and T. Ju, Variational implicit point set surfaces, ACM Transactions on Graphics (TOG), 38 (2019), pp. 1–13.

[82] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, Ccnet: Criss-cross attention for semantic segmentation, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 603–612.

[83] P. J. Huber, Robust estimation of a location parameter, in Breakthroughs in statistics, Springer, 1992, pp. 492–518.

[84] S. Iizuka, E. Simo-Serra, and H. Ishikawa, Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification, ACM Transactions on Graphics (TOG), 35 (2016), p. 110.

[85] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, Image-to-image translation with conditional adversarial networks, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.

[86] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, Spatial transformer networks, in Advances in neural information processing systems, 2015, pp. 2017–2025.

[87] O. Jamriška, Š. Sochorová, O. Texler, M. Lukáč, J. Fišer, J. Lu, E. Shechtman, and D. Sỳkora, Stylizing video by example, ACM Transactions on Graphics (TOG), 38 (2019), pp. 1–11.

[88] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, Super slomo: High quality estimation of multiple intermediate frames for video interpolation, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9000–9008.

[89] ——, Super slomo: High quality estimation of multiple intermediate frames for video interpolation, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9000–9008.

[90] M. Jirik, V. Lukes, M. Svobodova, and M. Zelezny, Image segmentation in medical imaging via graph-cuts., 2013.

[91] J. Johnson, A. Alahi, and L. Fei-Fei, Perceptual losses for real-time style transfer and super-resolution, in European conference on computer vision, Springer, 2016, pp. 694–711.

[92] H. Jun and A. Nichol, Shap-e: Generating conditional 3d implicit functions, arXiv preprint arXiv:2305.02463, (2023).

[93] B. Kainz, M. Steinberger, W. Wein, M. Kuklisova-Murgasova, C. Malamateniou, K. Keraudren, T. Torsney-Weir, M. Rutherford, P. Aljabar, and D. R. Hajnal, Joseph V, Fast volume reconstruction from motion corrupted stacks of 2D slices, IEEE transactions on medical imaging, 34 (2015), pp. 1901–1913.

[94] M. Kang, J.-Y. Zhu, R. Zhang, J. Park, E. Shechtman, S. Paris, and T. Park, Scaling up gans for text-to-image synthesis, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[95] T. Karras, T. Aila, S. Laine, and J. Lehtinen, Progressive growing of gans for improved quality, stability, and variation, arXiv preprint arXiv:1710.10196, (2017).

[96] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, Training generative adversarial networks with limited data, in Proc. NeurIPS, 2020.

[97] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, Alias-free generative adversarial networks, Advances in Neural Information Processing Systems, 34 (2021).

[98] T. Karras, S. Laine, and T. Aila, A style-based generator architecture for generative adversarial networks, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4401–4410.

[99] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, Analyzing and improving the image quality of StyleGAN, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 8110–8119.

[100] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani, Imagic: Text-based real image editing with diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2023, pp. 6007–6017.

[101] A. Kazerouni, E. K. Aghdam, M. Heidari, R. Azad, M. Fayyaz, I. Hacihaliloglu, and D. Merhof, Diffusion models for medical image analysis: A comprehensive survey, arXiv preprint arXiv:2211.07804, (2022).

[102] M. Kazhdan, M. Bolitho, and H. Hoppe, Poisson surface reconstruction, in Eurographics, 2006.

[103] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, Splatam: Splat track & map 3d gaussians for dense rgb-d slam, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2024, pp. 21357–21366.

[104] B. Kerbl, G. Kopanas, T. Leimkuhler, and G. Drettakis, 3d gaussian splatting for real-time radiance field rendering, ACM Transactions on Graphics, 42 (2023).

[105] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, 3d gaussian splatting for real-time radiance field rendering, ACM Transactions on Graphics, 42 (2023).

[106] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, Lerf: Language embedded radiance fields, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 19729–19739.

[107] J. Kim, J. K. Lee, and K. M. Lee, Accurate image super-resolution using very deep convolutional networks, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1646–1654.

[108] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, Learning to discover cross-domain relations with generative adversarial networks, in International Conference on Machine Learning, PMLR, 2017, pp. 1857–1865.

[109] D. P. Kingma and M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114, (2013).

[110] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907, (2016).

[111] N. Kondo, Y. Ikeda, A. Tagliasacchi, Y. Matsuo, Y. Ochiai, and S. Gu, Vaxnerf: Revisiting the classic for voxel-accelerated neural radiance field, ArXiv, abs/2111.13112 (2021).

[112] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, Diffwave: A versatile diffusion model for audio synthesis, arXiv preprint arXiv:2009.09761, (2020).

[113] P. KR, R. Mukhopadhyay, J. Philip, A. Jha, V. Namboodiri, and C. Jawahar, Towards automatic face-to-face translation, in Proceedings of the 27th ACM international conference on multimedia, 2019, pp. 1428–1436.

[114] H. W. Kuhn and A. W. Tucker, Nonlinear programming, in Traces and emergence of nonlinear programming, Springer, 2014, pp. 247–258.

[115] R. Kumar, J. Sotelo, K. Kumar, A. de Brébisson, and Y. Bengio, Obamanet: Photo-realistic lip-sync from text, arXiv preprint arXiv:1801.01442, (2017).

[116] Y. Kumar, D. Sahrawat, S. Maheshwari, D. Mahata, A. Stent, Y. Yin, R. R. Shah, and R. Zimmermann, Harnessing gans for zero-shot learning of new classes in visual speech recognition., in AAAI, 2020, pp. 2645–2652.

[117] D. Lahat, T. Adali, and C. Jutten, Multimodal data fusion: an overview of methods, challenges, and prospects, Proceedings of the IEEE, 103 (2015), pp. 1449–1477.

[118] R. Lazar, N. Dym, Y. Kushinsky, Z. Huang, T. Ju, and Y. Lipman, Robust optimization for topological surface reconstruction, ACM Transactions on Graphics (TOG), 37 (2018), pp. 1–10.

[119] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, Photo-realistic single image super-resolution using a generative adversarial network, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4681–4690.

[120] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, MaskGAN: Towards diverse and interactive facial image manipulation, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[121] A. Levin, D. Lischinski, and Y. Weiss, Colorization using optimization, in ACM transactions on graphics (tog), vol. 23, ACM, 2004, pp. 689–694.

[122] B. Li, Y.-K. Lai, M. John, and P. L. Rosin, Automatic example-based image colorization using location-aware cross-scale matching, IEEE Transactions on Image Processing, 28 (2019), pp. 4606–4619.

[123] B. Li, X. Qi, T. Lukasiewicz, and P. Torr, Controllable text-to-image generation, Advances in neural information processing systems, 32 (2019).

[124] H. Li, Y. Yang, M. Chang, S. Chen, H. Feng, Z. Xu, Q. Li, and Y. Chen, Srdiff: Single image super-resolution with diffusion probabilistic models, Neurocomputing, 479 (2022), pp. 47–59.

[125] H. Li, Y. Yuan, and Q. Wang, Video frame interpolation via residue refinement, in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2020, pp. 2613–2617.

[126] X. Li, S. Liu, J. Kautz, and M.-H. Yang, Learning linear transformations for fast image and video style transfer, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3809–3817.

[127] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, Universal style transfer via feature transforms, in Advances in neural information processing systems, 2017, pp. 386–396.

[128] Y. Li, M.-Y. Liu, X. Li, M.-H. Yang, and J. Kautz, A closed-form solution to photorealistic image stylization, in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 453–468.

[129] Z. Li, M. Hu, Q. Zheng, and X. Jiang, Connecting consistency distillation to score distillation for text-to-3d generation, in Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XLIII, Berlin, Heidelberg, 2024, Springer-Verlag, p. 274–291.

[130] Y. Liang, X. Yang, J. Lin, H. Li, X. Xu, and Y. Chen, Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching, 2023.

[131] C.-H. Lin, C. Kong, and S. Lucey, Learning efficient point cloud generation for dense 3D object reconstruction, in proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018.

[132] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, Barf: Bundle-adjusting neural radiance fields, in Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 5741–5751.

[133] H. Liu, J. Xing, M. Xie, C. Li, and T.-T. Wong, Improved diffusion-based image colorization via piggybacked models, arXiv preprint arXiv:2304.11105, (2023).

[134] L. Liu, C. Bajaj, J. O. Deasy, D. A. Low, and T. Ju, Surface reconstruction from non-parallel curve networks, in Computer Graphics Forum, vol. 27, Wiley Online Library, 2008, pp. 155–163.

[135] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, Zero-1-to-3: Zero-shot one image to 3d object, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 9298–9309.

[136] X. Liu, G. Yin, J. Shao, X. Wang, and H. Li, Learning to predict layout-to-image conditional convolutions for semantic image synthesis, arXiv preprint arXiv:1910.06809, (2019).

[137] X. Liu, X. Zhan, J. Tang, Y. Shan, G. Zeng, D. Lin, X. Liu, and Z. Liu, Humangaussian: Text-driven 3d human generation with gaussian splatting, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2024, pp. 6646–6657.

[138] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, Video frame synthesis using deep voxel flow, in 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 4473–4481.

[139] P. Ljung, J. Krüger, E. Groller, M. Hadwiger, C. D. Hansen, and A. Ynnerman, State of the art in transfer functions for direct volume rendering, in Computer Graphics Forum, vol. 35, Wiley Online Library, 2016, pp. 669–691.

[140] J. Lu, D. Batra, D. Parikh, and S. Lee, Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks, arXiv preprint arXiv:1908.02265, (2019).

[141] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. V. Gool, Repaint: Inpainting using denoising diffusion probabilistic models, CoRR, abs/2201.09865 (2022).

[142] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan, Dynamic 3D gaussians: Tracking by persistent dynamic view synthesis, arXiv preprint arXiv:2308.09713, (2023).

[143] D. Mahajan, F.-C. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur, Moving gradients: a path-based method for plausible image interpolation, ACM Transactions on Graphics (TOG), 28 (2009), pp. 1–11.

[144] S. Markidis, S. W. Der Chien, E. Laure, I. B. Peng, and J. S. Vetter, Nvidia tensor core programmability, performance & precision, in 2018 IEEE international parallel and distributed processing symposium workshops (IPDPSW), IEEE, 2018, pp. 522–531.

[145] D. W. Massaro and J. A. Simpson, Speech perception by ear and eye: A paradigm for psychological inquiry, Psychology Press, 2014.

[146] A. N. Mathur, P. Pham, A. Bera, and O. Sharma, Rl dreams: Policy gradient optimization for score distillation based 3d generation, arXiv preprint arXiv:2312.04806, (2023).

[147] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, Gaussian splatting slam, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2024, pp. 18039–18048.

[148] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, Montreal forced aligner: Trainable text-speech alignment using kaldi., in Interspeech, vol. 2017, 2017, pp. 498–502.

[149] L. Melas-Kyriazi, I. Laina, C. Rupprecht, and A. Vedaldi, Realfusion: 360deg reconstruction of any object from a single image, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 8446–8455.

[150] P. Memari and J.-D. Boissonnat, Provably good 2D shape reconstruction from unorganized cross-sections, in Computer Graphics Forum, vol. 27, Wiley Online Library, 2008, pp. 1403–1410.

[151] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, Phase-based frame interpolation for video, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1410–1418.

[152] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, Phase-based frame interpolation for video, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1410–1418.

[153] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, Distributed representations of words and phrases and their compositionality, in Advances in neural information processing systems, 2013, pp. 3111–3119.

[154] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, Nerf: Representing scenes as neural radiance fields for view synthesis, CoRR, abs/2003.08934 (2020).

[155] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, Nerf: Representing scenes as neural radiance fields for view synthesis, Communications of the ACM, 65 (2021), pp. 99–106.

[156] G. Mittal, J. Engel, C. Hawthorne, and I. Simon, Symbolic music generation with diffusion models, arXiv preprint arXiv:2103.16091, (2021).

[157] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, Spectral normalization for generative adversarial networks, arXiv preprint arXiv:1802.05957, (2018).

[158] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih, Monte carlo gradient estimation in machine learning, The Journal of Machine Learning Research, 21 (2020), pp. 5183–5244.

[159] R. Mokady, A. Hertz, K. Aberman, Y. Pritch, and D. Cohen-Or, Null-text inversion for editing real images using guided diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2023, pp. 6038–6047.

[160] ——, Null-text inversion for editing real images using guided diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 6038–6047.

[161] N. Müller, Y. Siddiqui, L. Porzi, S. R. Bulo, P. Kontschieder, and M. Niessner, Diffrf: Rendering-guided 3D radiance field diffusion, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 4328–4338.

[162] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, Glide: Towards photorealistic image generation and editing with text-guided diffusion models, arXiv preprint arXiv:2112.10741, (2021).

[163] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen, Point-e: A system for generating 3D point clouds from complex prompts, arXiv preprint arXiv:2212.08751, (2022).

[164] D. Nie, Q. Ma, L. Ma, and S. Xiao, Optimization based grayscale image colorization, Pattern recognition letters, 28 (2007), pp. 1445–1451.

[165] D. Nie, R. Trullo, J. Lian, C. Petitjean, S. Ruan, Q. Wang, and D. Shen, Medical image synthesis with context-aware generative adversarial networks, in International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2017, pp. 417–425.

[166] D. Nie, R. Trullo, J. Lian, L. Wang, C. Petitjean, S. Ruan, Q. Wang, and D. Shen, Medical image synthesis with deep convolutional adversarial networks, IEEE Transactions on Biomedical Engineering, 65 (2018), pp. 2720–2730.

[167] A. Odena, C. Olah, and J. Shlens, Conditional image synthesis with auxiliary classifier gans, in International conference on machine learning, PMLR, 2017, pp. 2642–2651.

[168] L. N. Olson and J. B. Schroder, PyAMG: Algebraic multigrid solvers in Python v4.0, 2018. Release 4.0.

[169] Y. Pang, J. Lin, T. Qin, and Z. Chen, Image-to-image translation: Methods and applications, arXiv preprint arXiv:2101.08629, (2021).

[170] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, DeepSDF: Learning continuous signed distance functions for shape representation, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[171] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, Semantic image synthesis with spatially-adaptive normalization, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2337–2346.

[172] T. Park, J. Zhu, O. Wang, J. Lu, E. Shechtman, A. A. Efros, and R. Zhang, Swapping autoencoder for deep image manipulation, CoRR, abs/2007.00653 (2020).

[173] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in Advances in Neural Information Processing Systems 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., Curran Associates, Inc., 2019, pp. 8026–8037.

[174] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, Context encoders: Feature learning by inpainting, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2536–2544.

[175] Q. Peng, B. Planche, Z. Gao, M. Zheng, A. Choudhuri, T. Chen, C. Chen, and Z. Wu, 3d vision-language gaussian splatting, in The Thirteenth International Conference on Learning Representations, 2025.

[176] P. Pérez, M. Gangnet, and A. Blake, Poisson image editing, in ACM SIGGRAPH 2003 Papers, 2003, pp. 313–318.

[177] K. Perlin, An image synthesizer, ACM Siggraph Computer Graphics, 19 (1985), pp. 287–296.

[178] P. Pham, D. Conover, and A. Bera, Flashslam: Accelerated rgb-d slam for real-time 3d scene reconstruction with gaussian splatting, arXiv preprint arXiv:2412.00682, (2024).

[179] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, Dreamfusion: Text-to-3d using 2d diffusion, arXiv, (2022).

[180] K. Prajwal, R. Mukhopadhyay, V. P. Namboodiri, and C. Jawahar, A lip sync expert is all you need for speech to lip generation in the wild, in Proceedings of the 28th ACM international conference on multimedia, 2020, pp. 484–492.

[181] P. Prusinkiewicz, Graphical applications of l-systems, in Proceedings of graphics interface, vol. 86, 1986, pp. 247–253.

[182] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister, Langsplat: 3d language gaussian splatting, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2024, pp. 20051–20060.

[183] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in International conference on machine learning, PMLR, 2021, pp. 8748–8763.

[184] A. Raj, S. Kaza, B. Poole, M. Niemeyer, B. Mildenhall, N. Ruiz, S. Zada, K. Aberman, M. Rubenstein, J. Barron, Y. Li, and V. Jampani, DreamBooth3D: Subject-driven Text-to-3D generation, ICCV, (2023).

[185] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, Hierarchical text-conditional image generation with clip latents, arXiv preprint arXiv:2204.06125, 1 (2022), p. 3.

[186] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, Zero-shot text-to-image generation, in Proceedings of the 38th International Conference on Machine Learning, M. Meila and T. Zhang, eds., vol. 139 of Proceedings of Machine Learning Research, PMLR, 18–24 Jul 2021, pp. 8821–8831.

[187] ——, Zero-shot text-to-image generation, in International conference on machine learning, Pmlr, 2021, pp. 8821–8831.

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

[189] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, High-resolution image synthesis with latent diffusion models, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2022, pp. 10684–10695.

[190] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, High-resolution image synthesis with latent diffusion models, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 10684–10695.

[191] L. A. Rowe, K. D. Mayer-Patel, B. C. Smith, and K. Liu, Mpeg video in software: Representation, transmission, and playback, in High-Speed Networking and Multimedia Computing, vol. 2188, International Society for Optics and Photonics, 1994, pp. 134–144.

[192] G. Rozenberg and A. Salomaa, The mathematical theory of L systems, vol. 90, Academic press, 1980.

[193] L. Ruan, Y. Ma, H. Yang, H. He, B. Liu, J. Fu, N. J. Yuan, Q. Jin, and B. Guo, Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 10219–10228.

[194] L. I. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, Physica D: nonlinear phenomena, 60 (1992), pp. 259–268.

[195] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 22500–22510.

[196] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi, Photorealistic text-to-image diffusion models with deep language understanding, in Advances in Neural Information Processing Systems, vol. 35, 2022, pp. 36479–36494.

[197] R. Salakhutdinov and H. Larochelle, Efficient learning of deep boltzmann machines, in Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 693–700.

[198] Sara Fridovich-Keil and Alex Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, Plenoxels: Radiance fields without neural networks, in CVPR, 2022.

[199] M. Sarmad, H. J. Lee, and Y. M. Kim, RL-GAN-Net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 5898–5907.

[200] H. Sawdayee, A. Vaxman, and A. H. Bermano, Orex: Object reconstruction from planar cross-sections using neural fields, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 20854–20862.

[201] F. Schneider, Archisound: Audio generation with diffusion, arXiv preprint arXiv:2301.13267, (2023).

[202] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, et al., Laion-5b: An open large-scale dataset for training next generation image-text models, Advances in Neural Information Processing Systems, 35 (2022), pp. 25278–25294.

[203] O. Sharma and N. Agarwal, Signed distance based 3d surface reconstruction from unorganized planar cross-sections, Computers & Graphics, 62 (2017), pp. 67–76.

[204] T. Shen, J. Gao, K. Yin, M.-Y. Liu, and S. Fidler, Deep marching tetrahedra: a hybrid representation for high-resolution 3D shape synthesis, Advances in Neural Information Processing Systems, 34 (2021), pp. 6087–6101.

[205] Y. Shi, P. Wang, J. Ye, L. Mai, K. Li, and X. Yang, MVDream: Multi-view diffusion for 3d generation, in The Twelfth International Conference on Learning Representations, 2024.

[206] Z. Shi, X. Zhou, X. Qiu, and X. Zhu, Improving image captioning with better use of captions, CoRR, abs/2006.11807 (2020).

[207] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, (2014).

[208] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics, in International conference on machine learning, PMLR, 2015, pp. 2256–2265.

[209] O. Solon, The future of fake news: Don't believe everything you read, see or hear, The Guardian, 26 (2017).

[210] J. Song, C. Meng, and S. Ermon, Denoising diffusion implicit models, arXiv preprint arXiv:2010.02502, (2020).

[211] Y. Song, C. Yang, Z. Lin, X. Liu, Q. Huang, H. Li, and C.-C. J. Kuo, Contextual-based image inpainting: Infer, match, and translate, in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3–19.

[212] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, Bottleneck transformers for visual recognition, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 16519–16529.

[213] P. Sturm and B. Triggs, A factorization based algorithm for multi-image projective structure and motion, in European conference on computer vision, Springer, 1996, pp. 709–720.

[214] E. Sunden, P. Steneteg, S. Kottravel, D. Jonsson, R. Englund, M. Falk, and T. Ropinski, Inviwo-an extensible, multi-purpose visualization framework, in 2015 IEEE Scientific Visualization Conference (SciVis), IEEE, 2015, pp. 163–164.

[215] V. Sushko, E. Schönfeld, D. Zhang, J. Gall, B. Schiele, and A. Khoreva, You only need adversarial supervision for semantic image synthesis, arXiv preprint arXiv:2012.04781, (2020).

[216] I. E. Sutherland, Sketch pad a man-machine graphical communication system, in Proceedings of the SHARE design automation workshop, 1964, pp. 6–329.

[217] H. Tang, D. Xu, Y. Yan, P. H. Torr, and N. Sebe, Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 7870–7879.

[218] J. Tang, J. Ren, H. Zhou, Z. Liu, and G. Zeng, Dreamgaussian: Generative gaussian splatting for efficient 3d content creation, in The Twelfth International Conference on Learning Representations, 2024.

[219] S. Taraglio and S. Chiesa, Evolutionary approach to epipolar geometry estimation, Advances in Theory and Application s of Stereo Vision, (2011), pp. 1–16.

[220] C. Tomasi and T. Kanade, Shape and motion from image streams: a factorization method., Proceedings of the National Academy of Sciences, 90 (1993), pp. 9795–9802.

[221] M. Tomei, M. Cornia, L. Baraldi, and R. Cucchiara, Art2Real: Unfolding the reality of artworks via semantically-aware image-to-image translation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 5849–5859.

[222] Z. Tu and X. Bai, Auto-context and its application to high-level vision tasks and 3D brain image segmentation, IEEE transactions on pattern analysis and machine intelligence, 32 (2009), pp. 1744–1757.

[223] L. Van der Maaten and G. Hinton, Visualizing data using t-SNE., Journal of machine learning research, 9 (2008).

[224] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, in Advances in neural information processing systems, 2017, pp. 5998–6008.

[225] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, Graph attention networks, arXiv preprint arXiv:1710.10903, (2017).

[226] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in Proceedings of the 25th international conference on Machine learning, 2008, pp. 1096–1103.

[227] H. Wang, J. Cao, J. Liu, X. Zhou, H. Huang, and R. He, Hallo3d: Multi-modal hallucination detection and mitigation for consistent 3d content generation, Advances in Neural Information Processing Systems, 37 (2024), pp. 118883–118906.

[228] H. Wang, X. Du, J. Li, R. A. Yeh, and G. Shakhnarovich, Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 12619–12629.

[229] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, High-resolution image synthesis and semantic manipulation with conditional gans, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8798–8807.

[230] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, High-resolution image synthesis and semantic manipulation with conditional GANs, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[231] Y.-S. Wang, H.-C. Lin, O. Sorkine, and T.-Y. Lee, Motion-based video retargeting with optimized crop-and-warp, in ACM SIGGRAPH 2010 papers, 2010, pp. 1–9.

[232] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE transactions on image processing, 13 (2004), pp. 600–612.

[233] Z. Wang, C. Lu, Y. Wang, F. Bao, C. Li, H. Su, and J. Zhu, Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation, in Thirty-seventh Conference on Neural Information Processing Systems, 2023.

[234] D. Watson, W. Chan, R. Martin-Brualla, J. Ho, A. Tagliasacchi, and M. Norouzi, Novel view synthesis with diffusion models, arXiv preprint arXiv:2210.04628, (2022).

[235] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, Deepflow: Large displacement optical flow with deep matching, in Proceedings of the IEEE international conference on computer vision, 2013, pp. 1385–1392.

[236] T. Welsh, M. Ashikhmin, and K. Mueller, Transferring color to greyscale images, in ACM Transactions on Graphics (TOG), vol. 21, ACM, 2002, pp. 277–280.

[237] X. Wen, T. Li, Z. Han, and Y.-S. Liu, Point cloud completion by skip-attention network with hierarchical folding, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1939–1948.

[238] R. J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Reinforcement learning, (1992), pp. 5–32.

[239] G. Wolberg, Digital image warping, vol. 10662, IEEE computer society press Los Alamitos, CA, 1990.

[240] J. M. Wolterink, A. M. Dinkla, M. H. Savenije, P. R. Seevinck, C. A. van den Berg, and I. Išgum, Deep MR to CT synthesis using unpaired data, in International workshop on simulation and synthesis in medical imaging, Springer, 2017, pp. 14–23.

[241] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang, 4d gaussian splatting for real-time dynamic scene rendering, arXiv preprint arXiv:2310.08528, (2023).

[242] H. Wu, L. Yao, J. Mei, Y. Huang, Y. Xu, J. You, and Y. Liu, Esct3d: Efficient and selectively controllable text-driven 3d content generation with gaussian splatting, Proceedings of the ACM on Computer Graphics and Interactive Techniques, 8 (2025), pp. 1–20.

[243] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling, CoRR, abs/1610.07584 (2016).

[244] B. Xia, Y. Zhang, S. Wang, Y. Wang, X. Wu, Y. Tian, W. Yang, R. Timotfe, and L. Van Gool, Diffi2i: Efficient diffusion model for image-to-image translation, arXiv preprint arXiv:2308.13767, (2023).

[245] J. Xiang, Z. Lv, S. Xu, Y. Deng, R. Wang, B. Zhang, D. Chen, X. Tong, and J. Yang, Structured 3d latents for scalable and versatile 3d generation, arXiv preprint arXiv:2412.01506, (2024).

[246] ——, Structured 3d latents for scalable and versatile 3d generation, in Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 21469–21480.

[247] X. Xiang, Y. Tian, Y. Zhang, Y. Fu, J. P. Allebach, and C. Xu, Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020, pp. 3370–3379.

[248] C. Xiao, C. Han, Z. Zhang, J. Qin, T.-T. Wong, G. Han, and S. He, Example-based colourization via dense encoding pyramids, in Computer Graphics Forum, vol. 39, Wiley Online Library, 2020, pp. 20–33.

[249] S. Xie, Z. Zhang, Z. Lin, T. Hinz, and K. Zhang, Smartbrush: Text and shape guided object inpainting with diffusion model, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2023, pp. 22428–22437.

[250] J. Xu, X. Liu, Y. Wu, Y. Tong, Q. Li, M. Ding, J. Tang, and Y. Dong, Imagereward: learning and evaluating human preferences for text-to-image generation, in Proceedings of the 37th International Conference on Neural Information Processing Systems, 2023, pp. 15903–15935.

[251] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, Attngan: Fine-grained text to image generation with attentional generative adversarial networks, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 1316–1324.

[252] Z. Xu, T. Wang, F. Fang, Y. Sheng, and G. Zhang, Stylization-based architecture for fast deep exemplar colorization, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[253] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, Segan: Adversarial network with multi-scale l_1 loss for medical image segmentation, CoRR, abs/1706.01805 (2017).

[254] C. Yang, S. Li, J. Fang, R. Liang, L. Xie, X. Zhang, W. Shen, and Q. Tian, Gaussianobject: Just taking four images to get a high-quality 3d object with gaussian splatting, CoRR, (2024).

[255] S. Yang, Y. Zhang, D. Feng, M. Yang, C. Wang, J. Xiao, K. Long, S. Shan, and X. Chen, Lrw-1000: A naturally-distributed large-scale

benchmark for lip reading in the wild, in 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), IEEE, 2019, pp. 1–8.

[256] Z. Yang, L. Liu, and Q. Huang, Learning generative neural networks for 3d colorization, in AAAI, 2018.

[257] F. Ye, G. Liu, X. Wu, and L. Wu, Altdiffusion: A multilingual text-to-image diffusion model, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, 2024, pp. 6648–6656.

[258] J. Ye, F. Liu, Q. Li, Z. Wang, Y. Wang, X. Wang, Y. Duan, and J. Zhu, Dreamreward: Text-to-3d generation with human preference, in European Conference on Computer Vision, Springer, 2025, pp. 259–276.

[259] T. Yi, J. Fang, J. Wang, G. Wu, L. Xie, X. Zhang, W. Liu, Q. Tian, and X. Wang, Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models, in CVPR, 2024.

[260] K. Yin, H. Huang, D. Cohen-Or, and H. Zhang, P2P-Net: Bidirectional point displacement net for shape transform, ACM Transactions on Graphics (TOG), 37 (2018), pp. 1–13.

[261] C. You, Q. Yang, H. Shan, L. Gjesteby, G. Li, S. Ju, Z. Zhang, Z. Zhao, Y. Zhang, W. Cong, and G. WANG, Structurally-sensitive multi-scale deep neural network for low-dose CT denoising, IEEE Access, 6 (2018), pp. 41839–41855.

[262] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, PixelNeRF: Neural radiance fields from one or few images, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4578–4587.

[263] D. Yu and L. Deng, Automatic Speech Recognition., Springer, 2016.

[264] Y. Yuan, S. Liu, J. Zhang, Y. Zhang, C. Dong, and L. Lin, Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 701–710.

[265] D. Zhang and A. Khoreva, Progressive augmentation of GANs, in Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019, pp. 6249–6259.

[266] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks, in Proceedings of the IEEE international conference on computer vision, 2017, pp. 5907–5915.

[267] ——, Stackgan++: Realistic image synthesis with stacked generative adversarial networks, IEEE transactions on pattern analysis and machine intelligence, 41 (2018), pp. 1947–1962.

[268] L. Zhang, A. Rao, and M. Agrawala, Adding conditional control to text-to-image diffusion models, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 3836–3847.

[269] R. Zhang, P. Isola, and A. A. Efros, Colorful image colorization, in European conference on computer vision, Springer, 2016, pp. 649–666.

[270] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 586–595.

[271] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros, Real-time user-guided image colorization with learned deep priors, arXiv preprint arXiv:1705.02999, (2017).

[272] Y. Zhang, S. Liu, C. Dong, X. Zhang, and Y. Yuan, Multiple cycle-in-cycle generative adversarial networks for unsupervised image super-resolution, IEEE transactions on Image Processing, 29 (2019), pp. 1101–1112.

[273] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, Places: A 10 million image database for scene recognition, IEEE transactions on pattern analysis and machine intelligence, 40 (2017), pp. 1452–1464.

[274] K. Zhou, X. Huang, Y. Li, D. Zha, R. Chen, and X. Hu, Towards deeper graph neural networks with differentiable group normalization, arXiv preprint arXiv:2006.06972, (2020).

[275] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, Multimodal image-to-image translation by enforcing bi-cycle consistency, in Advances in neural information processing systems, 2017, pp. 465–476.

[276] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, Sean: Image synthesis with semantic region-adaptive normalization, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5104–5113.

[277] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, High-quality video view interpolation using a layered representation, ACM transactions on graphics (TOG), 23 (2004), pp. 600–608.

[278] M. Zou, M. Holloway, N. Carr, and T. Ju, Topology-constrained surface reconstruction from cross-sections, ACM Transactions on Graphics (TOG), 34 (2015), pp. 1–10.