SKED: Sketch-guided Text-based 3D Editing

by

Aryan Mikaeili

B.Sc., Sharif University of Technology, 2021

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Summer 2024

Declaration of Committee

Name:	**Aryan Mikaeili**
Degree:	**Master of Science**
Thesis title:	**SKED: Sketch-guided Text-based 3D Editing**
Committee:	**Chair:** Saba Alimadadi Assistant Professor, Computing Science
	Ali Mahdavi-Amiri Supervisor Assistant Professor, Computing Science
	Daniel Cohen-Or Committee Member Adjunct Professor, Computing Science
	Manolis Savva Examiner Assistant Professor, Computing Science

Abstract

With the recent advent of Text-to-3D models based on Text-to-image diffusion models, they are increasingly being integrated into Computer Graphics pipelines, facilitating creative 3D design tasks in unrestricted domains. Nevertheless, simple textual prompting is often insufficient for interactive and local manipulation of 3D content. Incorporating user-guided sketches offers a more intuitive control for editing and manipulation. However, since state-of-the-art Text-to-3D models rely on gradients generated by rendering arbitrary views for optimization, integrating sketches in these pipelines is not straightforward. This thesis presents SKED, a method for editing 3D objects represented as Neural Radiance Fields (NeRFs) conditioned on a text prompt and two sketches from different views. We ensure semantic alignment of the edit by using a pretrained text-to-image model as guidance. Additionally, our framework uses novel loss functions to make the edit adhere to the provided sketches. We demonstrate the effectiveness of our method through various qualitative and quantitative experiments.

Keywords: Text-to-3D modeling, Neural Radiance Fields, Diffusion models, Sketch-based editing

Dedication

To Nooshin and Mohammad, my beloved parents, whose courage in starting anew taught me that embarking on new journeys is not to be feared but embraced.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Ali Mahdavi-Amiri, for his unwavering support and guidance throughout my master's studies. I am also deeply thankful to Dr. Daniel Cohen-Or, whose vast experience and insightful guidance have been instrumental in shaping my development as a researcher. Additionally, I extend my thanks to Or Perel, whose depth of knowledge and infectious enthusiasm continue to inspire me. I am particularly grateful to my friend and labmate, Mehdi, with whom I embarked on this research journey. His support and shared insights have been invaluable. I have learned immensely from him, and this work would not have been possible without his contributions. Finally, I want to thank everyone at the GrUVi lab. Being part of such a remarkable team has been a great privilege, and I consider myself fortunate to have had this opportunity.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis is based on the ICCV 2023 paper titled "SKED: Sketch-guided Text-based 3D Editing." I would like to express my sincere gratitude to my co-authors and collaborators on this paper: Dr. Ali Mahdavi-Amiri, Dr. Daniel Cohen-Or, Or Perel, and Mehdi Safaee.

In this chapter, we establish the context and motivation for our research. We then clearly define the problem, discussing its significance and the challenges it presents in the field. Following this, we outline our approach to addressing these challenges and highlight the contributions our study aims to make.

1.1 Motivation

Creativity finds expression through various mediums, transforming ideas from the mind into reality. Recent advancements in neural generative art have streamlined these mediums, often reducing them to a simple text prompt—users can now create content by merely describing their visions. While this development significantly democratizes art, enabling both amateur and professional artists to easily express their ideas, it also restricts user control by limiting them to coarse and simplistic descriptions and interactive handles. In this thesis, we propose a method to regain some of this control, allowing neural networks to augment rather than replace human creativity, thereby enhancing the user's ability to shape the generative process.

1.2 Research problem

Recently, significant progress has been made in the field of text-conditioned image generation [4, 68, 64, 66]. These advancements are largely due to harnessing the power of diffusion models [76, 23, 77, 78] trained on large-scale text-image datasets [72]. These developments have transformed computer vision, enabling the creation of diverse and remarkable visual content from mere text prompts.

The power and versatility of text-to-image models have expanded the scope of image editing methods. Many of these methods approach editing by modifying the input text prompt [31, 22, 53,

10, 21]. While this approach is simple, it often fails to meet the user's need for flexible and fine-grained control over the editing process. Conversely, sketch-guided editing provides a more intuitive and detailed level of control, enabling users to precisely define their desired outcomes through their drawing skills [94, 85, 92, 35].

Although sketch-guided text-driven image generation and manipulation are well-studied, The intriguing problem of bringing this framework into the realm of 3D editing remains fairly unsolved. In this thesis, we aim to take a step towards solving this problem.

1.3 Significance and challenges

Designing a text-to-3D framework presents several complex challenges, each requiring careful consideration to address effectively. A fundamental question is the choice of 3D representation, as there are various options, including point clouds, polygonal meshes, and implicit models such as Neural Radiance Fields (NeRFs). While there is extensive research focusing on modeling 3D objects as meshes [48, 65] and point clouds [56], each approach has its own limitations. Mesh-based models, for instance, primarily allow for color editing and minor deformations. This is because maintaining the integrity of a high-quality mesh throughout extensive transformations is challenging. On the other hand, although point clouds offer a flexible structure for 3D modeling, they pose major difficulties in rendering realistic images.

Secondly, the lack of text-3D datasets comparable in scale to their text-image counterparts presents a considerable challenge. In response, DreamFusion [62] and various subsequent studies have leveraged pretrained text-to-image models by distilling their knowledge of the visual world to create 3D objects. These methods typically operate by rendering multiple views of an object represented by a NeRF and utilizing the diffusion model as a score function estimator. This process optimizes the object within the probability distribution function learned by the diffusion model, aiming to produce an outcome that is plausible based on the input text prompt. While these techniques help to address the issue of data scarcity, incorporating additional controls, such as sketches, poses further complications. It is impractical for users to provide sketches from every possible view. Therefore, we aim to design a framework that requires users to provide only a minimal number of sketches, potentially as few as two or three.

Figure 1.1: Overview of our Sketch-guided, Text-based 3D Editing Framework. Starting with a NeRF representation of a 3D object (e.g., a cat), our framework integrates input from sketches of two views and a text prompt describing the desired modifications. The output is an edited NeRF where the changes align semantically with the text prompt and adhere to the multiview sketches.

1.4 Method and contributions

We present SKED, A **Sk**etch guided 3D **Ed**iting technique. Our framework modifies a 3D object represented as a Neural Radiance Field (NeRF), using a combination of a text prompt and a minimum of two multiview sketches. Achieving consistency with the constraints imposed by the text prompt and sketches is challenging due to potential semantic mismatches between the text and the sketches. Moreover, the assumption of free-form hand-drawn sketches can result in a lack of multiview consistency. To address these challenges, we divide the problem into two manageable subtasks: the first utilizes geometric reasoning to determine the location and boundaries of the edits based on the sketches, while the second leverages the semantic knowledge from a pretrained text-to-image model to refine the geometry and add semantic texture to the edited regions. We demonstrate the versatility of our method through various results, showing its capability to add objects, accessories, and artifacts or to replace parts of existing objects. An overview of our framework can be observed in Figure 1.1. Our approach is validated through both qualitative and quantitative assessments, including ablation studies that highlight the contributions of different components of our framework.

1.5 Structure of the thesis

This thesis is organized into six chapters. Chapter 2 explores related works and the state-of-the-art in sketch-based 3D modeling, diffusion models, and neural fields. Chapter 3 provides a summary of essential preliminaries and definitions related to diffusion models, Score Distillation Sampling (SDS), and volumetric rendering. In chapter 4, we provide a detailed explanation of our method. Chapter 5 presents extensive qualitative and quantitative evaluations of our method, including ablation studies of various components. Finally, Chapter 6 discusses the limitations of our work and proposes several directions for future research.

Chapter 2

Related Works

2.1 Sketch-based modeling

From the early Renaissance through the emergence of computer animation in the late 1920s, artists have consistently focused on creating believable representations of the 3D world. The highly acclaimed art guidebook, The Illusion of Life [26], emphasizes the importance of "weight, depth, and balance" in three-dimensional drawings. With the advent of digital art, these fundamental principles have continued to underpin the systems and frameworks used in artistic content creation [37]. Traditional sketch-based modeling systems usually work by inflating and stitching user-drawn lines to 3D meshes [91]. Beginning with Teddy [24], many works have focused on converting scribbles and 2D contours into intermediate forms, such as closed polylines [28] or implicit functions [27, 81, 1, 71, 8]. Since transforming 2D sketches into 3D representations poses an under-constrained challenge, various approaches have been developed to address this issue. Some methods integrate additional priors, correlating the inflation thickness with the chordal axis depth of curved shapes [24], while others incorporate extra user annotations [71, 28, 19, 58] or utilize predefined models, such as reference figures for human bodies [83]. Departing from traditional systems, recent works have introduced data-driven approaches and power of deep neural networks for the task of sketch-to-3D [44, 39, 15, 9].

Our work differs from past research by utilizing a pretrained text-to-image model as a semantic prior. This way, we bypass the need for arduous tuning of inflation parameters and collecting large 3D datasets.

2.2 Diffusion models

Diffusion models [23, 76, 77, 78] are state-of-the-art generative models for image synthesis, capable of generating diverse and high-quality images. Particularly in recent years, text-to-image models have emerged [66, 64, 4, 68] that utilize large-scale text-image datasets [72] to integrate text embeddings into diffusion model training, enabling text-driven image generation. These advancements have significantly transformed the field of visual content creation, enabling a variety of tasks through

enhanced image generation and editing capabilities [10, 22, 53, 31, 46]. Text alone is insufficient for many applications as it only provides coarse control over the generated content. Because of this, some recent work has focused on introducing extra spatial control such as semantic masks [3], localized object level control [61], and local pattern transfer [67]. There is another line of work that enables general conditioning on spatial data such as depth, normal and segmentation maps by adding extra modules to the pretrained diffusion model and training on a paired dataset [94, 54].

Related to our work, Voynov et al. [85] developed a differentiable edge detector that incorporates an edge loss at each diffusion step, facilitating sketch-conditioned image generation. Cheng et al. [13] introduced a method that enhances sketch-based image generation by differentiating between sketches, strokes, and various levels of realism. More recently, Parmar et al. [60] advanced this area by proposing a cycle consistency-based training approach for SDXL-Turbo [70], which supports sketch-based image generation and other image translation tasks in a single diffusion step.

Diffusion models are also used for 3D content generation by training directly on point clouds [56, 45], feature maps of neural fields [74, 2], or directly learning the parameters of an MLP representing the implicit representation of a shape [17]. While showing potential for high-quality 3D content creation, these works are hard to scale due to their dependency on large-scale 3D data.

2.3 Neural fields

Neural Radiance Fields (NeRFs) [49] have recently emerged as promising 3D representations because of their compressed representation and versatility in various applications, such as novel view synthesis and editing. These models represent 3D scenes and objects through implicit functions that are represented through deep neural networks. Since their introduction, there have been various efforts to increase these models' reconstruction quality [5, 6, 7, 84, 88]. Additionally, some research works have tried to increase rendering speed and make these frameworks more compact by combining them with explicit grid representations [69, 11, 12] or by taking the volumetric rendering approach of NeRF and combining it with primitive rasterization [32]. The pivotal work of Mueller et al. [55] proposes a compact hash-based grid approach that allows for NeRF optimization in a few seconds. This work paves the way for interactive research directions in neural fields. Recent works have introduced interactive editing of NeRFs through latent manipulation [57, 11, 59], intermediate representations [14, 43], local editing through segmentation masks [50, 34, 40, 36], and text-based stylization and inpainting [90, 48, 86, 87, 18, 51].

Neural radiance fields are a suitable representation for 3D content manipulation [38] since, contrary to meshes, they do not depend on topological properties such as genus and watertightness [52, 18, 48]. Early works on NeRF editing and generation with large vision language models such as CLIP [63] attempted to align arbitrary rendered views with a given text prompt [87, 25]. However, without a geometric prior, CLIP fails to produce multiview-consistent 3D assets. Dream3D [93] uses shape embeddings and language prior from a text-to-image diffusion model to generate 3D shapes. However, this model requires large 3D datasets to generalize.

DreamFusion [62] bypasses the need for a 3D dataset by using the power of a pretrained text-to-image diffusion model as a prior. Their framework utilizes a diffusion model as a score function estimator to optimize a NeRF based on an input text prompt. They render arbitrary views of an object represented by a NeRF and gradually optimize the parameters of the NeRF with gradients coming from the diffusion model to align the NeRF with the input text prompt. Magic3D [41] further improves the DreamFusion framework by utilizing a two-stage optimization process where the second stage optimizes a mesh extracted from the NeRF. Both of these frameworks show the potential ability of global 3D editing with simple prompt changes. Concurrently, Latent-NeRF [47] proposes to generate a NeRF able to directly render latent maps of a latent diffusion model for faster training and rendering.

Our framework builds on a simplified version of DreamFusion [62] to combine the generative prior of a text-to-image model with traditional sketch-based constraints. Similar to DreamFusion [62], our pipeline works in a zero-shot manner, requiring no text-3D datasets.

Concurrent with our research, Vox-E [73] has introduced a grid mixing method for NeRF editing that utilizes cross-attention maps from the attention modules of a text-to-image model. Similarly, Instruct-NeRF2NeRF [20] facilitates NeRF editing through a training set updating procedure. Our work, however, distinctively offers precise sketch-based control over the editing region, setting it apart from Vox-E. Additionally, unlike Instruct-NeRF2NeRF, our approach supports localized sketch-based editing, providing a more targeted modification capability.

Chapter 3

Preliminaries and Background

This chapter provides the necessary background on Neural Radiance Fields (NeRFs), Diffusion models, and Score Distillation Sampling (SDS).

3.1 Neural Radiance Fields (NeRFs)

Neural Radiance Fields (NeRFs) [49] represent a class of implicit 3D representations primarily utilized for novel view synthesis. These methods represent a scene as being composed of light-emitting particles, which allows them to bypass the computationally expensive ray-triangle intersections [80]. Instead, they employ volumetric rendering as their backbone, enabling the rendering of scenes from arbitrary viewpoints. NeRFs are coordinate-based frameworks that take a point in the 3D space as input and output the density σ and a view dependent RGB color c of that point. Formally, a NeRF comprises of two functions. $f_\sigma(x)$, which maps 3D points to a single real number which is the density of that point, and $f_c(x, d)$, which takes the coordinates of a 3D point and the viewing direction and outputs the RGB color of that points. f_σ and f_c can be MLPs [49], 3D grids [69], or a mixture of both [55]. Given these two functions, the expected color of a pixel represented by a ray $r(t) = o+td$ can be calculated using the volumetric rendering equation:

$$C(r) = \int_{t_n}^{t_f} T(t) f_\sigma(r(t)) f_c(r(t), d) dt, \tag{3.1}$$

where

$$T(t) = \exp\left(-\int_{t_n}^{t} f_\sigma(r(s)) ds\right) \tag{3.2}$$

is the transmittance function.

In this thesis, we propose a framework that applies sketch-based text-guided editing, to 3D objects represented by NeRFs.

3.2 Diffusion models

Diffusion models [76, 23] belong to a class of generative models that sample from a target data distribution, p_{data} through a sequential process. This involves reversing a forward process that incrementally transforms samples from the target distribution into a standard Gaussian distribution by progressively adding noise. The initial transformation to the Gaussian distribution is known as the forward process, while the reverse transformation back to the target distribution is termed the backward process. More precisely, the intermediate noisy data at timestep t is calculated by:

$$x_t = \alpha_t x + \sigma_t \epsilon, \tag{3.3}$$

where x is a sample from the target distribution provided by the training data, ϵ is a standard Gaussian noise, and α_t and σ_t are the parameters of the diffusion noise scheduler.

The key part of a diffusion model is a neural network ϵ_θ usually parameterized as a UNet [23] that is trained to predict clean, denoised data $\hat{x}$ from noisy data x_t. This network is trained with the loss

$$\mathcal{L}_{\text{diffusion}} = \mathbb{E}_{t\sim U[1,T],\boldsymbol{\epsilon}_t\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[w(t)\left\|\boldsymbol{\epsilon}_\theta\left(\mathbf{z}_t; t, c\right) - \boldsymbol{\epsilon}_t\right\|_2^2\right]. \tag{3.4}$$

In this equation, c is a conditional input such as text [4, 68, 64] and $w(t)$ is a weighting function usually set to 1 [23, 16]. In simpler terms, the diffusion training process involves sampling data from the training set, adding noise to this data where the noise timestep t is sampled from a uniform distribution, and training the UNet ϵ_θ to predict the noise that was added.

After training, multiple schedulers [23, 77, 42, 29] can be used to sample from the target distribution. Starting from a random sample from the standard Gaussian distribution x_T, these methods iteratively denoise the input. In each timestep, t, these methods take as input x_t, and according to the specific scheduler, they predict a less noisy data x_{t-1} until they finally reach a clean sample from the target distribution $x_0 \sim p_{data}$. Additionally, these methods usually use classifier-free guidance to enhance the quality and diversity of the samples. This guidance involves linearly combining the noise predicted with the input condition, $\epsilon_\theta(x_t, c)$ and without the input condition, $\epsilon_\theta(x_t, \emptyset)$, where $\emptyset$ denotes null condition. The final noise prediction $\hat{\epsilon}_\theta$ is calculated by:

$$\hat{\epsilon}_\theta = \omega\epsilon_\theta(x_t, c) + (1-\omega)\epsilon_\theta(x_t, \emptyset), \tag{3.5}$$

where ω is a hyperparameter determining the effect of the conditional input.

Latent Diffusion Models: Latent Diffusion Models (LDMs) [66] are a type of image diffusion models that operate within a latent space rather than directly sampling high-resolution images. These models comprise two main components: firstly, a variational autoencoder, which includes an image encoder $\mathcal{E}(x)$ and a decoder $\mathcal{D}(z)$. This autoencoder is pretrained on the image training dataset. The second component is the diffusion UNet ϵ_θ, which is trained to generate plausible samples directly in the latent space created by the autoencoder. During inference, the latents produced

by the diffusion process are inputted into $\mathcal{D}(z)$ to generate high-resolution images. A key advantage of LDMs is their ability to produce high-resolution images using fewer memory and time resources.

3.3 Score Distillation Sampling (SDS)

Diffusion models excel at learning distributions where enough data is available. In many domains, such as text-to-3D, data is notably scarce, complicating the training process. Additionally, there is often a desire to condition the generation process on extra conditions, such as sketches or edges, but the available data may be insufficient for these requirements. In such cases, Score Distillation Sampling (SDS) [62] proves invaluable. It utilizes a pretrained diffusion model as a prior, allowing for the optimization of parameters within a differentiable, parameterized image framework.

More precisely, let g_ϕ be a parameterized image representation. e.g. a differentiable 3D renderer with parameters ϕ. It is notable that NeRFs are such a representation with ϕ being the parameters of the coordinate-based MLP or the parameters of a feature grid. SDS optimizes the parameters ϕ by gradients that minimize the diffusion loss. If in equation 3.4 we assume z_t is a noised version of g_ϕ, taking the gradient with respect to ϕ and using the chain rule, will give us:

$$\nabla_\phi \mathcal{L}_{\text{diffusion}} = \mathbb{E}_{t,\epsilon}[\omega(t)(\hat{\boldsymbol{\epsilon}}_\theta(z_t; t, c) - \boldsymbol{\epsilon}_t)\frac{\partial \hat{\epsilon}_\theta(g_\phi^t; t, c)}{\partial z_t}\frac{\partial g_\phi}{\partial \phi}]. \tag{3.6}$$

In this equation the constant $2\frac{\partial z_t}{\partial g_\theta}$ is absorbed into the constant $\omega(t)$. A Naive approach to generation would be to simply optimize the parameters of g_θ with gradient descent and gradients coming from this loss function. However, DreamFusion [62] empirically shows that the diffusion UNet jacobian $\frac{\partial \hat{\epsilon}_\theta(g_\phi^t; t, c)}{\partial z_t}$ is difficult to optimize and is very expensive to compute. Hence, they propose to omit this term to get to the final SDS loss gradient, which is:

$$\nabla_\phi \mathcal{L}_{\text{SDS}} = \mathbb{E}_{t,\epsilon}[\omega(t)(\hat{\boldsymbol{\epsilon}}_\theta(z_t; t, c) - \boldsymbol{\epsilon}_t)\frac{\partial g_\theta}{\partial \theta}]. \tag{3.7}$$

DreamFusion [62] further formally proves that optimizing a parameterized image representation with this loss is equivalent to minimizing the KL divergence between a family of Gaussians around g_ϕ and the diffusion conditional likelihood $p(z_t; t, c)$.

The algorithm for DreamFusion is fairly simple. In each iteration of optimization, g_ϕ is utilized to render an image. Noise is then added to this image with respect to a random timestep t, and then this noisy render will be inputted into the UNet to get the predicted noise $\hat{\epsilon}_\theta$. After that, the gradient of the SDS loss as defined in equation 3.7 will be used to optimize g_ϕ.

In summary, SDS is an alternative to the original ancestral sampling of diffusion models that allows for extra conditioning on images or generation of parameterized image representations such as NeRFs. In Fig. 3.1, you can see the difference between ancestral sampling and SDS.

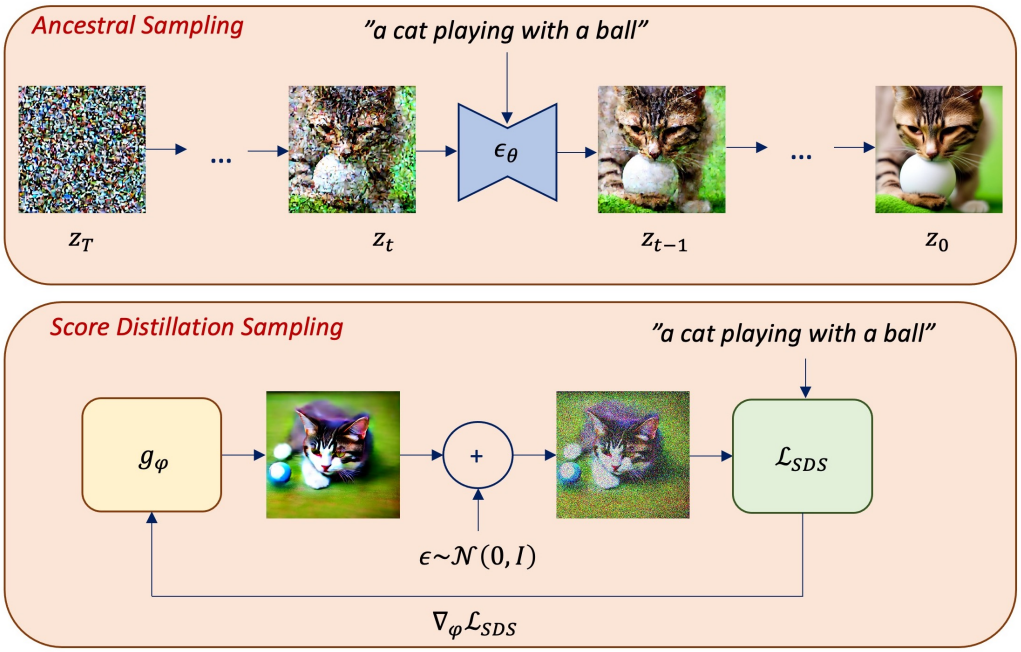

Figure 3.1: Score Distillation Sampling compared to the ancestral sampling used in text-to-image models. g_ϕ is a parameterized image representation such as a NeRF, or an identity map. Gradients computed with $\mathcal{L}_{SDS}$ are used to optimize the parameters of g_ϕ.

Chapter 4

Method

In this chapter, we introduce our framework, SKED, which enables the editing of 3D objects represented as NeRFs, utilizing a text prompt and a few multiview sketches. Our approach comprises two parts that work in parallel. Firstly, a coarse 3D geometry is constructed from the multiview sketches, which, while requiring further refinement, serves as a guide for subsequent optimization. During optimization, two loss functions are employed to ensure that edits closely align with the provided sketches. Secondly, we implement Score Distillation Sampling (SDS) [62] with a text-to-image latent diffusion model to generate fine-grained and realistic edits based on the text prompt given to the framework.

The two loss functions employed in our framework are fundamental in ensuring adherence to the principles of effective local editing. The first principle is the preservation of the original object. To achieve this, we introduce the **preservation** loss, which minimizes the difference between the geometry and appearance of the edited output compared to the base object outside the sketch regions. The second principle is alignment for the given constraints, which, in our case, are the multiview sketches provided as input. To address this, we implement a **silhouette** loss, ensuring that any added mass conforms accurately to the outlines defined by the sketches.

4.1 Method overview

As illustrated in Figure 4.1, our framework begins with a base Neural Radiance Field model, $F_o : (\mathbf{p}, \hat{\mathbf{r}}; \theta) \rightarrow (\mathbf{c}_o, \sigma_o)$. This model works by mapping 3D coordinates and unit rays to corresponding color and volumetric density outputs. F_o is used in volumetric rendering, enabling the rendering of arbitrary views of a base 3D object. This base object may be derived from multiview images [55] or through a text-to-3D generation pipeline [62]. Furthermore, F_o allows for the rendering of multiple views of the 3D object, over which sketches can be drawn to define the outlines of the desired edits. Let $C_{i=1}^{N}$ be renderings of F_o from N different views, onto which we have drawn sketches. We refer to these renderings as ***sketch canvases*** and their respective viewpoints as ***sketch views***. The sketches can be delineated by masks or a closed curve that outlines the border of the edit region. In both instances, the sketches are preprocessed into masks $M_{i=1}^{N}$, where each $M_i = \{m_1, m_2, ..., m_S\}$

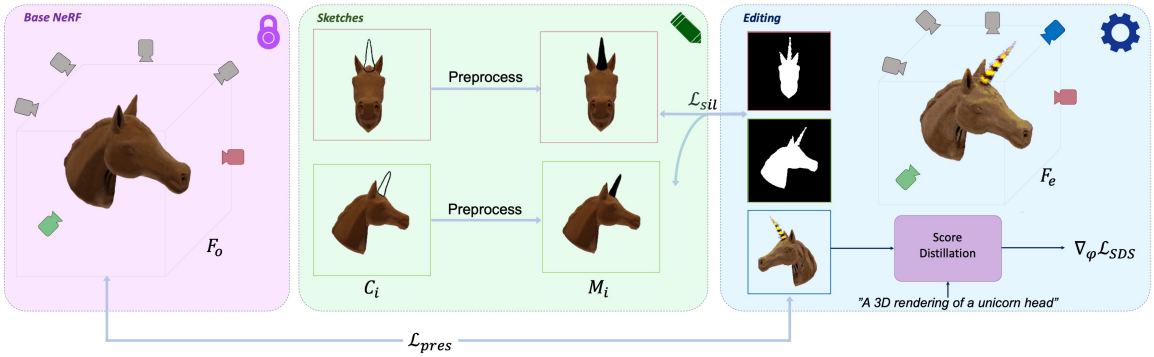

Figure 4.1: An overview of our framework. We use F_o to render at least two views of the base object and then sketch over them (C_i). The input to the editing framework are these sketches preprocessed to masks (M_i) and a text prompt. Similar to DreamFusion [62], in each iteration of optimization, we render F_e from a random view around the object and calculate the gradient of the SDS loss to optimize F_e. This ensures that the edit aligns with the text prompt semantically. Additionally, we apply $\mathcal{L}_{pres}$ to constrain the density and color of F_e to be similar to F_o away from the sketch regions. Finally, we also constraint the object mask renderings of F_e to occupy the sketch region by computing a loss $\mathcal{L}_{sil}$.

represents a set of pixels within the edit region. Additionally, our framework takes a text prompt T as input, which defines the semantics of the edit. We start our optimization process by making a copy of the base NeRF F_o, calling the copy F_e, which at the start of optimization is equal to F_o.

We utilize the Instant-NGP [55] architecture for our NeRF model, chosen for its rapid rendering capabilities and memory efficiency. This framework employs a density grid that identifies and tracks empty regions to optimize rendering efficiency and avoid sampling from empty spaces. During raymarching, the model skips these empty areas. The grid is periodically pruned throughout training. In our editing process, if the density grid from F_o is used without change, it would initially prevent sampling from the edit regions that intersect with the empty regions of F_o. This can disrupt correct gradient flow and force our framework to depend on random alterations of the occupancy grid.

To mitigate this issue, we calculate the bounding boxes of the sketch masks M_i, extrude them, and intersect them within $\mathbb{R}^3$. This process defines a coarse edit region in 3D space where we activate the density grid of F_e. Additionally, we define a warm-up period at the start of optimization, during which we avoid pruning the density grid. This step helps the model stabilize the edit region and prevents it from mistakenly culling it as empty space.

At each iteration of the optimization, we sample a random viewpoint on a sphere around the 3D object and use F_e to render the object from that view. Then, we use the gradients of the SDS loss to push F_e to high-density regions of the conditional probability distribution function $p(F_e; T)$, i.e. a NeRF of a 3D object which is both realistic and aligns with the input text prompt T. However, as expected and shown in our ablation studies in chapter 5, using text guidance alone would drastically change the base object outside the sketch region, and the edit would not adhere to the given sketches.

Therefore, through applying two novel loss functions, we constraint the editing process in a way that the base object is unchanged and the edits align with the given sketches.

4.2 Preservation loss

One of the principles of an effective 3D editing framework is that the geometry and color of the base object stay unchanged during editing. To encourage this, we employ a loss function referred to as ***preservation loss*** $\mathcal{L}_{pres}$. At each iteration of the optimization process, we sample a random view on a sphere around the object and use F_e to render that view. Our key idea is that when rendering, for each point sampled for raymarching, we decide whether that point's density and color should be changed. For this, we define and calculate a distance $D(\mathbf{p}_i)$ between the multiview sketches and point $\mathbf{p}_i$. Intuitively, the point's density and color should be changed if $D(\mathbf{p}_i)$ is large and should remain unchanged otherwise. We define $D(\mathbf{p}_i)$ by first projecting $\mathbf{p}_i$ on each of the sketch canvases and calculating a per-view 2D distance $d(\mathbf{p}_i)$ as:

$$d_j(\mathbf{p}_i) = \min_k || \left\lfloor \Pi(\mathbf{p}_i, C_j) + \frac{1}{2} \right\rfloor - m_k||^2, \tag{4.1}$$

where $\Pi(\mathbf{p}_i, C_j)$ is the projection of the 3D point $\mathbf{p}_i$ to the sketch canvas C_j rounded to the nearest integer (Fig. 4.2). This equation calculates the minimum distance between the projection of the sampled 3D coordinate $\mathbf{p}_i$ to the pixels of the sketch masks. After calculating $d_j(\mathbf{p}_i)$ for every sketch view, we calculate the multiview distance as:

$$D(\mathbf{p}_i) = \frac{1}{N} \sum_{j=1}^{N} d_j(\mathbf{p}_i). \tag{4.2}$$

The reason we use the mean of the per-view distances is that it relaxes the constraint in the case that the multiview sketches are not fully aligned. Additionally, it is smoother and easier to optimize than using other aggregators such as maximum value.

After computing the multiview sketch distance for all the sampled points during raymarching $\{p_i\}_{i=1}^{K}$, we can compute $\mathcal{L}_{pres}$:

$$\mathcal{L}_{\text{pres}} = \frac{1}{K} \sum_{i=1}^{K} w_i [CE(\alpha_e, \overline{\alpha_o}) + \lambda_c \overline{\alpha_o} ||\mathbf{c}_e - \mathbf{c}_o||^2], \tag{4.3}$$

where $\overline{\alpha_o}$ is the occupancy of the base object from F_o, computed by thresholding the density, and α_e is the alpha value of $\mathbf{p}_i$ derived by $\alpha_e = 1 - exp(-\sigma_e \delta)$, such that σ_e is the density $F_e(\mathbf{p}, \hat{\mathbf{r}}; \phi)$ and δ is the step between two consecutive points sampled along a ray. CE denotes the cross entropy loss. Furthermore, $\mathbf{c}_o$ and $\mathbf{c}_e$ are the color values of the base object and edited object derived from F_o and F_e, respectively, and λ_c is a hyperparameter specifying the importance of color preservation in the editing process. We limit color preservation to points which are occupied by the base object. The density and color preservation constraints are modulated by the weight w_i, which is calculated

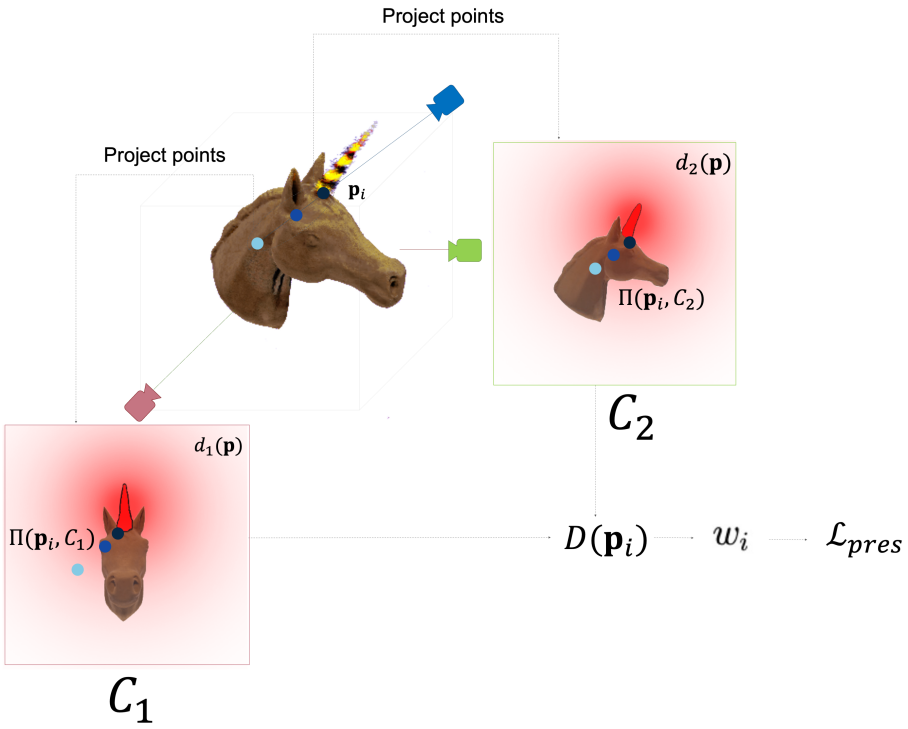

Figure 4.2: During raymarching, sampled points $\mathbf{p}_i$s are projected onto the sketch canvases C_j to determine $\Pi(\mathbf{p}_i, C_j)$. Subsequently, the minimum distance $d_i(\mathbf{p}_i)$ between the projected points and the sketch pixels is calculated. The red color in C_1 and C_2 illustrates the distance functions d_1 and d_2 in the image space, respectively. These per-view distances are averaged to compute the point-sketch distance $D(\mathbf{p}_i)$. This distance $D(\mathbf{p}_i)$ is then utilized to calculate per-point weights, which modulate the preservation loss $\mathcal{L}_{pres}$ in our framework.

by:

$$w_i = 1 - \exp(-\frac{D(\mathbf{p}_i)^2}{2\beta^2}). \tag{4.4}$$

Intuitively, this equation classifies sampled points into three categories. First, points where $D(\mathbf{p}_i) = 0$, which means that the per-view sketch distance for all sketch views is equal to zero, and the point falls into the sketch region and can be changed. Second, points where $D(\mathbf{p}_i)$ is a very small value, which means that the point probably lies in the sketch region in at least one of the sketch views. This could be caused by misalignments in the multiview sketches. Third points where $D(\mathbf{p}_i)$ is large and w_i is large, which are points that should be conserved. Additionally, the sensitivity of w_i to $D(\mathbf{p}_i)$ is controlled with the hyperparameter β. small values of β tighten the constraint so that only sketch regions are modified, and larger values of β impose a more relaxed constraint.

4.3 Silhouette loss

In addition to ensuring that the base object's original content is preserved, we also have to force the framework's edits to respect the input sketches. We do this by rendering the object masks of all the

sketch views and enforcing that in the sketch regions; the object masks are occupied. In more detail, we minimize the following loss, which we call the ***silhouette loss*** ($\mathcal{L}_{sil}$).

$$\mathcal{L}_{\text{sil}} = \frac{1}{H.W.N} \sum_{j=1}^{N} \sum_{i=1}^{H.W} -\mathbb{I}_{M_j}(\mathbf{x}_i) \log C_j^{\alpha}(\mathbf{x}_i). \tag{4.5}$$

In this equation, H and W are the dimensions of the rendered masks, $\mathbb{I}_{M_j}(\mathbf{x}_i)$ is an indicator function which is 1 if pixel $\mathbf{x}_i$ is in the sketch region and 0 otherwise. $C_j^{\alpha}(\mathbf{x}_i)$ is the alpha object mask rendered with $F_e(\mathbf{p}, \hat{\mathbf{r}}; \phi)$ from each sketch view.

4.4 Optimization

Our final objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{SDS}} + \lambda_{\text{pres}}\mathcal{L}_{\text{pres}} + \lambda_{\text{sil}}\mathcal{L}_{\text{sil}} + \lambda_{\text{sp}}\mathcal{L}_{\text{sp}}, \tag{4.6}$$

where $\lambda_{pres}, \lambda_{sil}$ and λ_{sp} are the weights of each separate loss term in our objective. Building on prior work [62, 41, 25], we incorporate an additional sparsity loss, $\mathcal{L}_{sp}$, which promotes coherent output and assists in preventing the generation of floaters and artifacts. This loss is defined as:

$$\mathcal{L}_{sp} = -\frac{1}{K} \sum_{i=1}^{K} \hat{\omega}_i \log(\hat{\omega}_i) + (1 - \hat{\omega}_i) \log(1 - \hat{\omega}_i), \tag{4.7}$$

where $\hat{\omega}_i$ is the accumulated alpha value along a ray. This loss encourages each ray's opacity to be either 1 or 0, preventing cloudy artifacts.

4.5 Implementation

Our implementation is based on the Stable-DreamFusion [82] GitHub repository, which is a simple implementation of DreamFusion [62]. We use StableDiffusion v1.4 [66] as the pretrained text-to-image model. Although, our model is compatible with any other text-to-image diffusion model. For most of our experiments, we set λ_{pres}, λ_{sil}, λ_{sp} and λ_c to 5×10^{-6}, 1, 5×10^{-4} and 5. In our ablation studies in chapter 5 we show the effect of changing these values. Following DreamFusion [62], we set the guidance scale of classifier-free guidance to 100. We sample noise timesteps in each iteration of optimization uniformly in the range of $[20, 980]$. The warmup period for the occupancy grid pruning is set to 1000. We sample the camera poses in each iteration on a sphere around the object. We choose the radius of the sphere according to the size of the base object, and we sample elevation and azimuth in a way that ensures that the sketch regions are always visible. For memory efficiency in implementing equation 4.3 we employ a KD-Tree for efficient point querying. We use the ADAM [33] optimizer with a learning rate of 0.005 and a learning rate scheduler that exponen-

tially decays the learning rate by 0.1 at the end of the optimization. We run our algorithm for $10,000$ iterations, which takes around 30-40 minutes on a single NVIDIA RTX 3090 GPU.

Chapter 5

Experiments

In this chapter, we conduct extensive qualitative and quantitative experiments and ablation studies to validate our framework and design choices. In our experiments, we use 3D assets which are freely available and assets that are generated with text-to-3D models [62].

5.1 Qualitative results

In Fig. 5.1, we demonstrate several edits performed using our framework. Our approach successfully adheres to the coarse geometry outlined by the multiview sketches while semantically blending the edit region with the base object. Notably, the multiview sketches do not need to be precise or tightly drawn. By increasing the complexity of the contour curves, users can guide the framework to generate more complex shapes. In Fig.5.1, we illustrate this flexibility with an example where using the same text prompt but providing two completely different sets of sketches, our framework successfully performs localized edits (cherry on top of a sundae).

Furthermore, in Fig. 5.2, we demonstrate the capability of our framework to perform various types of edits. We are able to perform both additive edits (e.g. adding a crown on a teddy bear or adding whipped cream on a stack of pancakes) and replacement edits where a part of the base object is completely replaced according to user input (e.g. tree to cactus or white long skirt).

A natural application of our framework is progressive editing, where additional details and ornaments are sequentially added to an object. Fig. 5.3 illustrates a two-step editing process in which a red tie is first added to the cat, followed by a chef's hat.

In Fig. 5.4, we demonstrate that we can control the strength of the preservation constraint by changing the hyperparameter β in equation 4.4. It is evident that increasing β would relax the preservation constraint and results in more edits outside the sketch regions.

Our edits are applied to objects represented as NeRF models. These objects can reconstructed from multiview images using Instant-NGP [55] or generated with DreamFusion [62] with a diffusion model identical to the model we use for editing. The renderings of the NeRF model are assumed to be in the distribution of the pretrained diffusion model. In Fig. 5.1 and Fig. 5.2, we demonstrate that our method is capable of editing both reconstructed objects (Cat, Anime girl, Horse, Ficus)

Figure 5.1: Examples of SKED used for editing various objects reconstructed from multiview images (Anime girl) or generated with DreamFusion [62] (Teddy, Cupcake, Sundae, Plant). All examples were edited using two multiview sketches and a text prompt as input.

and generated objects (Stack of pancakes, Teddy bear, Cupcake, Sundae, Plant). Additionally, in Fig. 5.5, we illustrate the geometry of the edits by rendering normal maps and depth maps. Our model is able to construct plausible geometry.

Figure 5.2: Our method supports multiple types of edit. We can both perform additive edits where a part is added to the base object (Cream on top of pancakes, adding a crown on a Teddy bear) and replacement edits where a part of an object is overwritten according to the user input (Ficus replaced with cactus, Girl wearing long skirt).

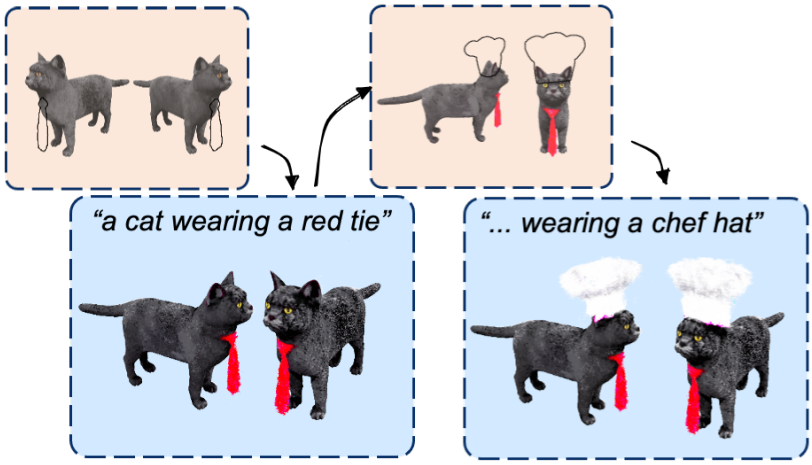

Figure 5.3: Progressive editing. We first add a red tie to the cat and then add a chef hat.

Figure 5.4: Sensitivity control. By changing the parameter β, we are able to control the strength of the preservation constraint. Increasing β would result in a more relaxed constraint where regions outside the sketched areas are edited, and we can observe a softer blending of the edited part with the base object. Overlay of the sketches is demonstrated on the edited output.

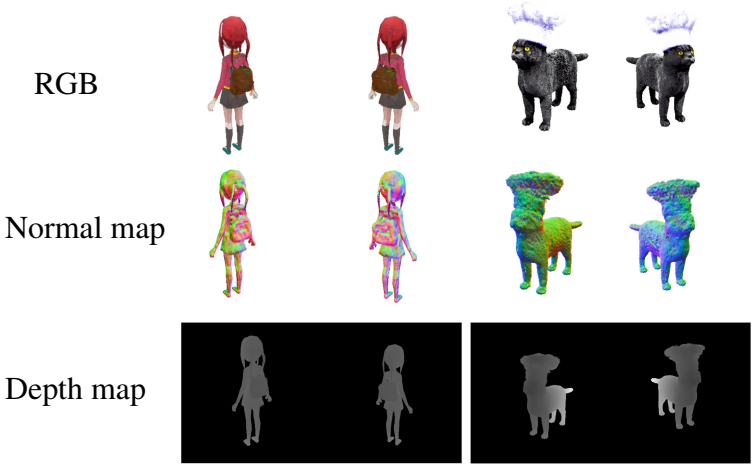

Figure 5.5: The normal and depth maps of the edits generated by our method.

Figure 5.6: Examples from the modified version of the sketch shape pipeline of Latent-NeRF [47].

5.2 Comparison

To the best of our knowledge, this is the first framework that enables sketch-guided and text-based 3D editing. Due to the novelty of our approach, direct comparisons with existing methods are not feasible. However, we have adapted the sketch-shape generation pipeline from Latent-NeRF [47] for use as a comparative baseline. Initially, we start with a base NeRF of the 3D asset intended for editing. We then intersect the bounding boxes of the multiview sketches to create a coarse geometry and determine the edit's location as a mesh. Finally, we apply the generation process from Latent-NeRF [47]. We must note that we avoid using the direct intersection of sketches because the view inconsistency of the multiview sketches causes the optimization process not to converge.

In Fig. 5.6, it is evident that although the sketch region is occupied and the edits align semantically with the input text prompts, the input multiview sketches are not adequately respected, and the base object is excessively altered. We also note that our framework typically converges in 30-40 minutes, whereas Latent-NeRF [47] usually requires 60-70 minutes, making our framework approximately twice as fast.

5.3 Quantitative results

To our knowledge, we present the first framework that enables sketch-guided and text-based 3D editing. Without an existing benchmark for systematic comparison, we establish a series of quantitative tests to evaluate our framework's ability to preserve the base object while generating edits that adhere to the input sketches and text prompt. We conduct our experiments on a set of five different representative examples. Each example includes a 3D object in the form of a NeRF [55], a text prompt, and a pair of multiview sketches.

5.3.1 Base model fidelity

We measure the model fidelity by calculating the PSNR between renderings of F_o and F_e from the sketch views and outside the sketch regions. As indicated in Table 5.1, our framework consistently maintains the base object more effectively than the Text-Only approach. Furthermore, we show the critical and effective role of our $\mathcal{L}_{pres}$ in preserving the base object. Finally, we highlight the superiority of our framework compared to the modified version of Latent-NeRF [47]. However, PSNR alone is not a comprehensive measure for an effective editing framework. It is important to note that if the framework does not alter the base object at all, it results in an artificially high PSNR.

Additionally, we assess the perceptual fidelity to the base object by computing the LPIPS distance between the base and edited objects. The results, as shown in Table 5.2, indicate that our framework generally maintains the perceptual integrity of the base object more effectively than competing baselines.

Table 5.1: Fidelity to the base field. We calculate **PSNR** ↑ between renderings of F_e and F_o from the sketch views and outside the sketch region. SKED *(no-preserve)* refers to a variant of our method which doesn't apply $\mathcal{L}_{pres}$. Text-Only refers to a public re-implementation of DreamFusion[62]. Latent-NeRF [47] is a modified version of their sketch-shape generation pipeline.

Method	Cat +*chef hat*		Cupcake +*candle*		Horse +*horn*		Sundae +*cherry*		Plant +*flower*		Mean
	View 1	View 2	View 1	View 2	View 1	View 2	View 1	View 2	View 1	View 2	
SKED	**31.05**	**34.13**	**23.73**	**25.98**	**32.45**	**31.46**	**26.47**	**25.99**	**21.71**	**22.31**	**27.53**
SKED *(no-preserve)*	15.58	16.59	20.12	19.47	18.02	16.52	17.39	17.44	10.16	10.12	16.14
Text-Only	15.63	16.78	17.38	17.15	16.69	15.09	20.57	20.74	13.75	12.68	16.65
Latent-NeRF [47]	21.15	22.62	21.99	21.20	17.00	15.97	16.07	15.47	17.66	16.78	18.59

Table 5.2: Perceptual fidelity to the base field. We measure the **Perceptual Image Patch Similarity (LPIPS** ↓**)** between renderings of F_e and F_o from the sketch views outside the sketch regions. We use VGG [75] as the base network for calculating LPIPS. SKED *(no-preserv)* refers to a variant of our method which doesn't apply $\mathcal{L}_{pres}$. Text-Only refers to a public re-implementation of DreamFusion [62]. Latent-NeRF [47] is a modified version of their sketch-shape generation pipeline.

Method	Cat +*chef hat*		Cupcake +*candle*		Horse +*horn*		Sundae +*cherry*		Plant +*flower*		Mean
	A	B	A	B	A	B	A	B	A	B	
SKED	**0.070**	**0.069**	0.069	**0.061**	**0.028**	**0.032**	0.086	0.094	0.158	0.128	**0.079**
SKED *(no-preserv)*	0.290	0.250	0.091	0.093	0.089	0.098	0.169	0.154	0.291	0.309	0.183
Text-Only	0.150	0.137	0.076	0.076	0.115	0.134	**0.081**	**0.079**	0.170	0.180	0.120
Latent-NeRF [47]	0.102	0.101	**0.066**	0.065	0.081	0.100	0.139	0.141	**0.108**	**0.113**	0.101

Table 5.3: Sketch alignment score. We measure whether the sketch region is filled with generated mass by a metric we refer to as Intersection-over-Sketch (**IoS** ↑). The IoS is calculated by comparing the sketch regions' area and the generated edit's alpha mask. Refer to Section 5.3 for more details. The SKED *(no-silh)* variant, which runs with $\mathcal{L}_{pres}$ and without $\mathcal{L}_{sil}$ avoids generating content in the sketch region (see also Fig. 5.7)

Method	Cat +*chef hat*		Cupcake +*candle*		Horse +*horn*		Sundae +*cherry*		Plant +*flower*		Mean
	View 1	View 2	View 1	View 2	View 1	View 2	View 1	View 2	View 1	View 2	
SKED	**0.9384**	**0.9689**	**0.8364**	**0.8875**	**0.6423**	**0.5363**	**0.7817**	**0.9096**	**0.9388**	**0.7801**	**0.8220**
SKED *(no-silh)*	0.0196	0.0176	0.0263	0.0209	0.0090	0.0077	0.0541	0.0506	0.0024	0.0028	0.0211

5.3.2 Sketch adherence

To evaluate how closely our framework adheres to the input multiview sketches, we have developed a new metric called Intersection-over-Sketch (IoS). This metric calculates the ratio of the sketch regions that are filled with the generated mass. The IoS is computed as follows:

$$IoS = \sum_{i=1}^{N} \frac{|M_i \cap C_i^{\alpha}|}{|M_i|}, \tag{5.1}$$

where M_i represents the area of the sketch region in sketch view i, and C_i^{α} denotes the thresholded alpha object mask rendered from the same view using F_e with threshold value α. To ensure that our measurement is not sensitive to the thresholding value α we average the IoS with nine α values in the range $[0, 1]$.

As shown in Table 5.3, our $\mathcal{L}_{sil}$ effectively ensures that the sketch region is filled in accordance with the input multiview sketches. However, it is important to note that this metric does not assess the semantic accuracy of the edit relative to the input text prompt. For instance, if a method were to uniformly fill the entire sketch region with a single color, it would still yield high IoS values.

5.3.3 Semantic alignment

To measure whether our framework generates semantically correct edits according to the input text prompt, we use CLIP-similarity [63] of renderings of the edited object from F_e and the text prompt. More precisely, we render the object from forty randomly sampled views for each object and average the CLIP-similarity of these images with the text prompt. As can be seen in Table 5.4, although our framework does not directly optimize for CLIP-similarity, the results are comparable with the Text-Only framework. We attribute the lower scores of our method in some examples to the additional constraints we optimize for, such as sketches, which prevent the model from generating artificially meaningful results. In contrast, the Text-Only approach often produces artifacts that, while semantically close to the input text prompt, lack realism. (See qualitative ablation results for Text-Only in Fig. 5.7).

Table 5.4: Semantic alignment score. We measure the **CLIP-similarity** [63] ↑ of renderings of each method's output from random views and the input text prompt. Text-Only refers to a public re-implementation of [62]. Compared to the Text-Only framework which alters the whole base object to increase image-text similarity, SKED preserves the base model while respecting the sketch constraints and maintaining semantic similarity with the text prompt.

Method	Cat (+hat)	Cupcake (+candle)	Horse (+horn)	Sundae (+cherry)	Plant (+flower)	Mean
Ours	0.2336 ± 1.1e-3	**0.2849 ± 4.2e-3**	**0.2943 ± 4.8e-3**	0.2635 ± 2.4e-3	**0.2933 ± 4.0e-3**	0.2739
Text-Only	**0.2744 ± 4.4e-3**	0.2818 ± 6.2e-3	0.2928 ± 8.4e-3	**0.2674 ± 4.9e-3**	0.2865 ± 3.3e-3	**0.2806**

5.4 Ablation Studies

We utilize two loss terms in our framework to achieve localized sketch-based editing: $\mathcal{L}_{pres}$, which preserves the original content of the object, and $\mathcal{L}_{sil}$, which generates edits based on the input multiview sketches. We conduct ablation studies on these two loss functions and present visual examples in Fig. 5.7. Text-Only refers to directly applying DreamFusion[62] to a base NeRF. While the output aligns with the given text prompts, the geometry and color of the base object change drastically. Similarly, when only $\mathcal{L}_{sil}$ is applied, the sketch region changes according to the text prompt, but the base object also undergoes significant alterations. In contrast, applying only $\mathcal{L}_{pres}$ preserves the base object but leaves the sketch regions unchanged. Only by using both loss functions together does the framework ensure that the sketch region is edited according to the input text prompt and multiview sketches while preserving the base object.

In Fig. 5.8, we investigate the impact of the diffusion model type used as our semantic prior. Prior research [82] has indicated that using open-source text-to-image models like Stable Diffusion [66] in text-to-3D pipelines often results in lower quality 3D objects compared to commercial, larger models such as Imagen [68]. This disparity is due to open-source models being less responsive to directional prompts and generally less powerful overall. In our work, we observe a similar trend. When utilizing a more advanced text-to-image model, DeepFloyd [79], we can frequently achieve successful edits where Stable Diffusion fails. As illustrated in Fig. 5.8, Stable Diffusion cannot generate a gift box on the table, while DeepFloyd, serving as the generative prior, successfully creates a plausible gift box.

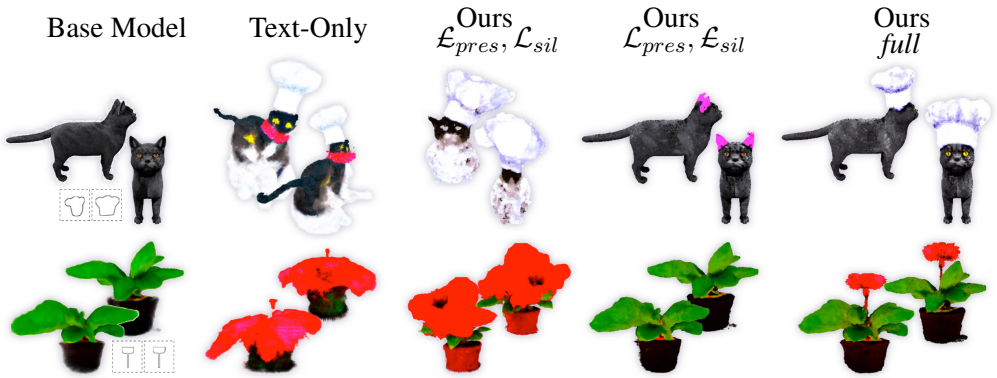

Figure 5.7: Ablation study. The effect of our loss functions is shown in two examples. The text prompts used for these edits are *"A cat wearing a chef hat"* and *"A red flower stem rising from a potted plant"*. All settings for experiments are identical except the use of the two proposed losses $\mathcal{L}_{pres}$ and $\mathcal{L}_{sil}$. Text-Only is a reimplementation of DreamFusion [82].

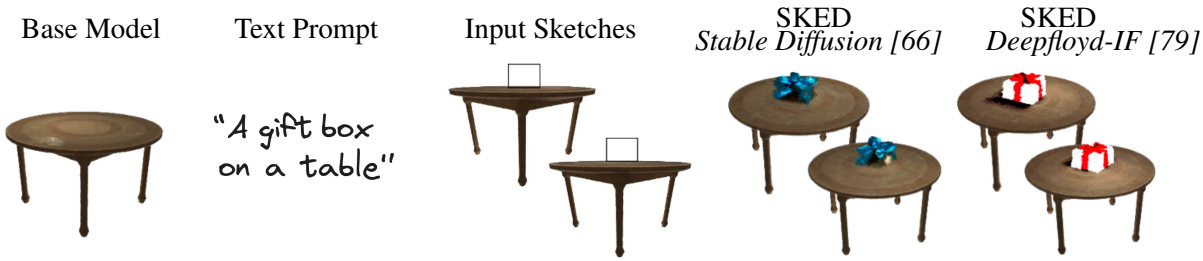

Figure 5.8: Effect of the diffusion Model backbone. Any text-to-image diffusion model can be used in our framework

Chapter 6

Conclusion and Limitations

In this thesis, we introduced SKED, a framework for editing 3D objects represented as Neural Radiance Fields (NeRFs), utilizing both text prompts and multiview sketches. We established that semantic, sketch-based 3D editing of NeRFs is complex, requiring the problem to be split into two distinct parts. The first part creates a coarse geometry from the input sketches and ensures the edits align with these sketches. The second part focuses on producing realistic edits that comply with the text prompt by leveraging the capabilities of a pretrained text-to-image diffusion model as a semantic prior.

To guide and constrain the editing process effectively, we developed two novel loss functions. The first, the preservation loss, ensures that the density and color fields of the NeRF outside the sketched regions remain unchanged. The second, the silhouette loss, guarantees that the edits occur within the sketched regions and that these areas are adequately filled. Our evaluations demonstrate that both loss functions are essential, providing the necessary constraints for successful sketch-based 3D editing.

However, our framework, SKED, inherits limitations from the text-to-3D generation pipeline of DreamFusion [62]. Firstly, SKED involves a lengthy optimization process that prevents its use in real-time applications. Secondly, it encounters issues common to SDS-based 3D generation, namely over-saturated colors and the multi-face problem. Over-saturated colors result from the necessary use of a large classifier-free guidance scale for score distillation sampling. Recent developments in SDS loss variants [89, 30], have shown potential in mitigating this issue, and their integration may benefit our framework. The multi-face issue, illustrated in Fig.6.1, arises due to biases in the pre-trained text-to-image diffusion models. These models often develop a preference for common poses found in training data—for example, more images of cats from the front than the back—leading to a lack of directional understanding. Consequently, 3D assets generated may be semantically coherent but structurally unrealistic, such as a cat with multiple faces or a structurally incorrect Santa hat (see Fig.6.1).

There are numerous possibilities for future research building on our work. One intriguing direction is using sketch scribbles to introduce motion into generated 3D assets. By interpreting dynamic sketches, this approach could facilitate the creation of animated sequences or interactive 3D models

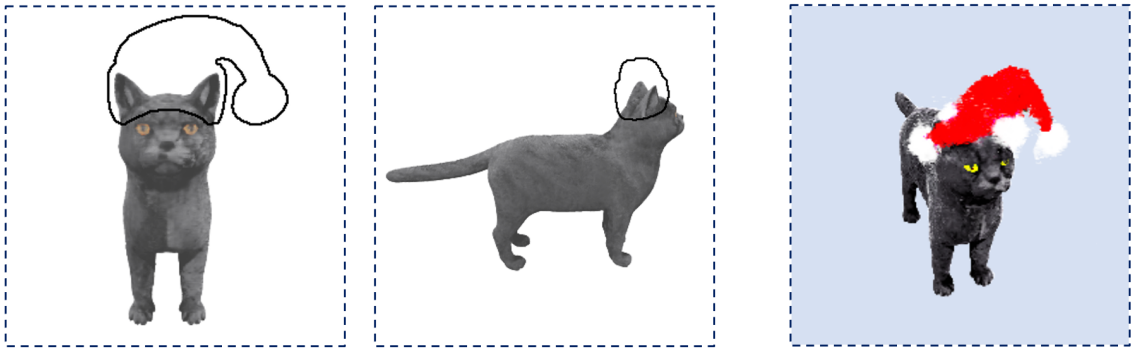

Figure 6.1: An example of the multi-face problem in our framework. Because the diffusion model lacks the directional understanding of a *"Santa hat"*, it generates an edit which, while semantically correct, is not structurally realistic.

that respond to user input, enriching the application of NeRFs in virtual and augmented reality environments. Another potential avenue involves adapting our sketch-based loss functions to edit other visual domains, such as videos. This could lead to innovative techniques for video post-production, where editors can make substantial modifications through intuitive sketch inputs. Finally, our research can be extended for multiview or single-view sketch-based 3D generation.

Bibliography

[1] A. Alexe, V. Gaildrat, and L. Barthe. Interactive modelling from sketches using spherical implicit functions. In *Proceedings of the 3rd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, AFRIGRAPH '04, page 25–34, New York, NY, USA, 2004. Association for Computing Machinery.

[2] Titas Anciukevicius, Zexiang Xu, Matthew Fisher, Paul Henderson, Hakan Bilen, Niloy J. Mitra, and Paul Guerrero. RenderDiffusion: Image diffusion for 3D reconstruction, inpainting and generation. *arXiv*, 2022.

[3] Omri Avrahami, Thomas Hayes, Oran Gafni, Sonal Gupta, Yaniv Taigman, Devi Parikh, Dani Lischinski, Ohad Fried, and Xi Yin. Spatext: Spatio-textual representation for controllable image generation. In *CVPR 2023*, pages 18370–18380, June 2023.

[4] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

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

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

[7] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023.

[8] Adrien Bernhardt, Adeline Pihuit, Marie-Paule Cani, and Loïc Barthe. Matisse : Painting 2d regions for modeling free-form shapes. *SBIM 08*, 06 2008.

[9] Alexandre Binninger, Amir Hertz, Olga Sorkine-Hornung, Daniel Cohen-Or, and Raja Giryes. SENS: Part-Aware Sketch-based Implicit Neural Shape Modeling. *Computer Graphics Forum*, 2024.

[10] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

[11] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021.

[12] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, 2022.

[13] Shin-I Cheng, Yu-Jie Chen, Wei-Chen Chiu, Hung-Yu Tseng, and Hsin-Ying Lee. Adaptively-realistic image generation from stroke and sketch with diffusion model, 2022.

[14] Chong Bao and Bangbang Yang, Zeng Junyi, Bao Hujun, Zhang Yinda, Cui Zhaopeng, and Zhang Guofeng. Neumesh: Learning disentangled neural mesh-based implicit field for geometry and texture editing. In *ECCV*, 2022.

[15] Johanna Delanoy, Mathieu Aubry, Phillip Isola, Alexei A Efros, and Adrien Bousseau. 3d sketching using multi-view deep volumetric prediction. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(1):1–22, 2018.

[16] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

[17] Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion, 2023.

[18] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images, 2022.

[19] Yotam Gingold, Takeo Igarashi, and Denis Zorin. Structured annotations for 2D-to-3D modeling. *ACM Transactions on Graphics (TOG)*, 28(5):148, 2009.

[20] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *ICCV*, 2023.

[21] Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta denoising score. 2023.

[22] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. 2022.

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[24] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, page 409–416, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

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

[26] Thomas F Johnston and O. Disney. *Disney Animation: The Illusion of Life*. Abbeville Press, New York, 1st edition, 1981.

[27] Olga Karpenko, John Hughes, and Ramesh Raskar. Free-form sketching with variational implicit surfaces. *Computer Graphics Forum*, 21, 05 2002.

[28] Olga A. Karpenko and John F. Hughes. Smoothsketch: 3d free-form shapes from complex sketches. *ACM Trans. Graph.*, 25(3):589–598, jul 2006.

[29] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. NeurIPS*, 2022.

[30] Oren Katzir, Or Patashnik, Daniel Cohen-Or, and Dani Lischinski. Noise-free score distillation, 2023.

[31] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Conference on Computer Vision and Pattern Recognition 2023*, 2023.

[32] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

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

[34] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[35] Subhadeep Koley, Ayan Kumar Bhunia, Deeptanshu Sekhri, Aneeshan Sain, Pinaki Nath Chowdhury, Tao Xiang, and Yi-Zhe Song. It's All About Your Sketch: Democratising Sketch Control in Diffusion Models. In *CVPR*, 2024.

[36] Zhengfei Kuang, Fujun Luan, Sai Bi, Zhixin Shu, Gordon Wetzstein, and Kalyan Sunkavalli. Palettenerf: Palette-based appearance editing of neural radiance fields, 2023.

[37] John Lasseter. Principles of traditional animation applied to 3d computer animation. *SIGGRAPH Comput. Graph.*, 21(4):35–44, aug 1987.

[38] H. Lee, M. Savva, and A. X. Chang. Text-to-3d shape generation. *Computer Graphics Forum*, n/a(n/a):e15061.

[39] Changjian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. Robust flow-guided neural prediction for sketch-based freeform surface modeling. *ACM Trans. Graph.*, 37(6), dec 2018.

[40] Gang Li, Heliang Zheng, Chaoyue Wang, Chang Li, Changwen Zheng, and Dacheng Tao. 3ddesigner: Towards photorealistic 3d object generation and editing with text-guided diffusion models, 2023.

[41] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *CVPR*, 2023.

[42] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022.

[43] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *ICCV*, 2021.

[44] Zhaoliang Lun, Matheus Gadelha, Evangelos Kalogerakis, Subhransu Maji, and Rui Wang. 3d shape reconstruction from sketches via multi-view convolutional networks. pages 67–77, 10 2017.

[45] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR 2021*, June 2021.

[46] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022.

[47] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. *arXiv preprint arXiv:2211.07600*, 2022.

[48] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. *arXiv preprint arXiv:2112.03221*, 2021.

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

[50] Ashkan Mirzaei, Yash Kant, Jonathan Kelly, and Igor Gilitschenski. Laterf: Label and text driven object radiance fields, 2022.

[51] Ashkan Mirzaei, Riccardo De Lutio, Seung Wook Kim, David Acuna, Jonathan Kelly, Sanja Fidler, Igor Gilitschenski, and Zan Gojcic. Reffusion: Reference adapted diffusion models for 3d scene inpainting, 2024.

[52] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Clip-mesh: Generating textured meshes from text using pretrained image-text models. In *SIGGRAPH Asia 2022 Conference Papers*, SA '22. ACM, November 2022.

[53] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. 2023.

[54] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023.

[55] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[56] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts, 2022.

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

[58] Luke Olsen, Faramarz Samavati, and Joaquim Jorge. Naturasketch: Modeling from images and natural sketches. *Computer Graphics and Applications, IEEE*, 31:24 – 34, 01 2012.

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

[60] Gaurav Parmar, Taesung Park, Srinivasa Narasimhan, and Jun-Yan Zhu. One-step image translation with text-to-image models, 2024.

[61] Or Patashnik, Daniel Garibi, Idan Azuri, Hadar Averbuch-Elor, and Daniel Cohen-Or. Localizing object-level shape variations with text-to-image diffusion models. In *ICCV 2023*, 2023.

[62] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022.

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

[64] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical textconditional image generation with clip latents, 2022.

[65] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. Texture: Text-guided texturing of 3d shapes, 2023.

[66] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2021.

[67] Mehdi Safaee, Aryan Mikaeili, Or Patashnik, Daniel Cohen-Or, and Ali Mahdavi-Amiri. Clic: Concept learning in context. *CVPR*, 2024.

[68] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022.

[69] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

[70] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation, 2023.

[71] R. Schmidt, B. Wyvill, M. C. Sousa, and J. A. Jorge. Shapeshop: Sketch-based solid modeling with blobtrees. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, page 14–es, New York, NY, USA, 2006. Association for Computing Machinery.

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

[73] Etai Sella, Gal Fiebelman, Peter Hedman, and Hadar Averbuch-Elor. Vox-e: Text-guided voxel editing of 3d objects. *arXiv preprint arXiv:2303.12048*, 2023.

[74] J. Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion, 2022.

[75] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

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

[77] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv:2010.02502*, October 2020.

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

[79] StabilityAI. Deepfloyd if: A modular cascaded diffusion model, 2023. https://www.deepfloyd.ai/deepfloyd-if.

[80] Andrea Tagliasacchi and Ben Mildenhall. Volume rendering digest (for nerf), 2022.

[81] Chiew-Lan Tai, Hongxin Zhang, and Jacky Fong. Prototype modeling from sketched silhouettes based on convolution surfaces. *Comput. Graph. Forum*, 23:71–84, 03 2004.

[82] Jiaxiang Tang. Stable-dreamfusion: Text-to-3d with stable-diffusion, 2022. https://github.com/ashawkey/stable-dreamfusion.

[83] Emmanuel Turquin, Jamie Wither, Laurence Boissieux, Marie-paule Cani, and John F. Hughes. A sketch-based interface for clothing virtual characters. *IEEE Computer Graphics and Applications*, 27(1):72–81, 2007.

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

[85] Andrey Voynov, Kfir Aberman, and Daniel Cohen-Or. Sketch-guided text-to-image diffusion models. *ACM Transactions on Graphics*, 42(4), July 2023.

[86] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields, 2022.

[87] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *arXiv preprint arXiv:2212.08070*, 2022.

[88] Chen Wang, Xian Wu, Yuan-Chen Guo, Song-Hai Zhang, Yu-Wing Tai, and Shi-Min Hu. Nerf-sr: High-quality neural radiance fields using super-sampling. *arXiv*, 2021.

[89] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*, 2023.

[90] Ethan Weber, Aleksander Holynski, Varun Jampani, Saurabh Saxena, Noah Snavely, Abhishek Kar, and Angjoo Kanazawa. Nerfiller: Completing scenes via generative 3d inpainting. In *CVPR*, 2024.

[91] L. Williams. Shading in two dimensions. In *Proceedings of Graphics Interface '91*, GI '91, pages 143–151, Toronto, Ontario, Canada, 1991. Canadian Man-Computer Communications Society.

[92] Chufeng Xiao and Hongbo Fu. Customsketching: Sketch concept extraction for sketch-based image synthesis and editing, 2024.

[93] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. ream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. In *CVPR*, 2023.

[94] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023.