Geometry-Aware Diffusion Models for Multiview Scene Inpainting

AHMAD SALIMI

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

GRADUATE PROGRAM IN COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

OCTOBER, 2025

© Ahmad Salimi, 2025

Abstract

In this thesis, we focus on 3D scene inpainting, where parts of an input image set, captured from different viewpoints, are masked out. The main challenge lies in generating plausible image completions that are geometrically consistent across views. Most recent work addresses this challenge by combining generative models with a 3D radiance field to fuse information across a relatively dense set of viewpoints. However, a major drawback of these methods is that they often produce blurry images due to the fusion of inconsistent cross-view images. To avoid blurry inpaintings, we eschew the use of an explicit or implicit radiance field altogether and instead fuse cross-view information in a learned space. In particular, we introduce a geometry-aware conditional generative model, capable of multiview consistent inpainting using reference-based geometric and appearance cues. A key advantage of our approach over existing methods is its unique ability to inpaint masked scenes with a limited number of views (i.e., few-view inpainting), whereas previous methods require relatively large image sets for their 3D model fitting step. Empirically, we evaluate and compare our scene-centric inpainting method on two datasets, SPIn-NeRF and NeRFiller, which contain images captured at narrow and wide baselines, respectively, and achieve state-of-the-art 3D inpainting performance on both. Additionally, we demonstrate the efficacy of our approach in the few-view setting compared to prior methods.

In memory of the 176 souls lost on

Ukraine International Airlines Flight PS752

January 8, 2020

Their pursuit of knowledge and dreams

continues to inspire us

Acknowledgements

It has been a privilege to work under the supervision of **Konstantinos G. Derpanis**. He guided me to discover my own research direction and taught me that meaningful research is about posing the right questions, not merely advancing metrics. His commitment to excellence and consistent availability for guidance have shaped both this thesis and my development as a researcher. I am deeply grateful for his mentorship throughout my journey.

I thank my committee member, **Marcus A. Brubaker**, for his insightful and valuable contributions to this thesis. His big-picture insights on methodology design provided crucial strategic direction, while his constructive feedback on submissions significantly strengthened the presentation and framing of this research. Each of his comments proved to be exceptionally helpful in refining both the technical approach and the quality of the final work.

I am deeply indebted to my mentor and collaborator, **Tristan Aumentado-Armstrong**, for his exceptional intellectual guidance throughout this research. He was present at every step of the methodology design, contributing substantially to both the conceptual development and the written presentation of this work. Our frequent discussions and collaborative problem-solving sessions were instrumental in shaping many of the core ideas in this thesis.

I also thank the members of the **CVIL Lab at York University** for a welcoming and collaborative environment. Helpful discussions, feedback on drafts, and practical assistance from the entire lab kept the research moving and improved the final outcome. Special gratitude goes to **Jason Yu** for his invaluable assistance with day-to-day questions and steadfast support during tight deadlines for paper submissions.

My deepest appreciation extends to my family, whose love and encouragement sustained me throughout this journey. Above all, I honor my mother, whose constant support and devotion have been the foundation of my strength. Her resilience and grace have been my guiding light and a role model for the values of perseverance, compassion, and integrity that I strive to embody in both my work and life.

Finally, my deepest gratitude goes to my wife, **Summer Kabiri**. Her love, patience, and unwavering belief carried me through late nights, long experiments, and the inevitable detours of research. She celebrated small victories and offered perspective when results disappointed. This thesis reflects her support as much as my effort.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Image inpainting is a long-standing problem in computer vision and graphics [3]. In contrast to unconditional generation, inpainting is constrained by the conditioning input: any output must both be visually plausible and match the partial content. The problem of inpainting *3D scenes* has recently grown in popularity (e.g., [48, 64]) due to the advent of powerful novel view synthesis (NVS) models. Such models are usually implemented as scene models capable of differentiable rendering, e.g., neural radiance fields (NeRFs) [46] or 3D Gaussian splatting (3DGS) [29]. For both NVS and 3D inpainting, the input scene is implicitly represented through a posed set of images; hence, we may consider the equivalent problem of "multiview inpainting", which provides a natural interface between 2D image inpainting and 3D NVS.

Multiview (3D) inpainting is significantly more constrained than even the 2D case: inpainted scene content must be *geometrically* realistic and exhibit *cross-view consistency*. In other words, novel image content in one view not only must be plausible from a dif-

ferent viewpoint, but also have the same visual content, with only physically sensible view-dependent effects. Cross-view consistency is particularly critical when leveraging 2D image inpainters that, by default, are neither aware of the 3D scene structure nor the current state of inpainted content in other images; unsurprisingly, this results in inconsistent scene content. Yet, we still seek to exploit the powerful generative priors learned by 2D inpainters, as the paucity of 3D scene data prevents training inpainters that operate directly in 3D to the same level of quality. This problem is likely exacerbated with increasingly powerful generative inpainters which tend to hallucinate even more aggressively [47]. Thus, how can we balance per-view image quality with consistency across views?

A common approach, often used in NeRF editing (e.g., [85]), is to fuse information across views via 3D radiance fields. A cyclic process, such as iterative dataset update (IDU) [19] or score distillation sampling (SDS) [53], is then used, whereby images are edited, used to inform the fitting process of the NVS model, and NVS renders (which combine information across views) assist with new edits. While this approach ensures consistency via the integrated 3D radiance field, it has a few shortcomings: (i) a tendency towards blurry outputs and (ii) a reliance on the radiance field, which presents inherent difficulties of its own. The blur in (i) is, in part, due to the fusion happening in pixel space, while the downsides of (ii) include the need for accurate camera parameters with sufficient view coverage.

In this thesis, we resolve these problems by fusing cross-view information in a learned space, during a generative diffusion process, and eschewing the necessity of a 3D radiance field, though we can optionally choose to use one as a separate post-fitting step. We further

reduce the tendency of generative inpainters to over-hallucinate, a common cause of 3D inconsistencies, by conditioning them on the 3D scene structure. In particular, we devise a geometry-aware conditional diffusion model, capable of inpainting multiview-consistent images based on geometric and appearance cues from reference images. One key capability of our model is the handling of *uncertain* or *partial* scene information (e.g., missing due to occlusions and viewpoint changes). Since existing 3D and multiview datasets are limited in comparison to 2D datasets, we construct an approach for simulating multiview data from *single-view* images. We integrate our model into a 3D scene inpainting algorithm, performing 3D-consistent propagation of image content across views. Unlike inpainters based on 3D radiance fields, which fuse inconsistencies in appearance space and thus induce blur, our method fuses cross-view information via the generative model, resulting in sharper outputs even when a NeRF is fit to our final inpainted results.

Our approach makes several key contributions: We develop a *geometry-aware conditional inpainting diffusion model* that leverages 3D scene geometry to achieve multiview consistency without requiring explicit 3D radiance field representations. The model conditions on both appearance and geometric cues projected from reference views, enabling direct cross-view information transfer while maintaining sharp, high-quality outputs. It incorporates a hierarchical confidence-based fusion mechanism that combines information from multiple reference views using learned confidence masks that account for occlusions and geometric reliability. We design an *autoregressive inpainting framework for large baselines* that selects a key-view subset for initial content generation, then propagates information to remaining views, enabling effective handling of large camera baselines. We develop a method for creating *synthetic multiview training data* from single-view images

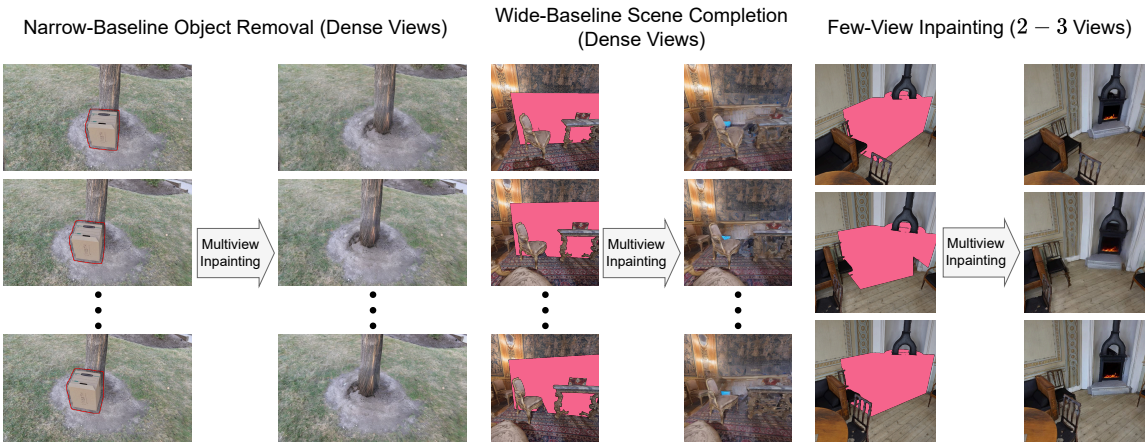

Figure 1.1: Visualization of our target inpainting tasks. We target three tasks: (i) narrow-baseline object removal, (ii) wide-baseline scene completion and (iii) few-view inpainting. Here, we show examples of our inputs and corresponding outputs for each task.

using depth estimation and mesh rendering, addressing the scarcity of multiview datasets for training geometry-aware inpainting models. Additionally, we introduce and evaluate the challenging problem of *few-view inpainting*, demonstrating that our approach can achieve consistent results even with extremely sparse view coverage where traditional radiance field-based methods fail.

In this thesis, we target three main inpainting tasks: narrow-baseline object removal, wide-baseline scene completion, and few-view inpainting. Most previous work focuses on the first task, the second one is a recent addition, and the third task has not been extensively explored with recent innovations. Fig. 1.1 provides a visualization of each of our target tasks.

We evaluate our multiview inpainter on two established benchmarks for 3D inpainting, SPIn-NeRF [48] and NeRFiller [85], which contain narrow and wide baselines, respectively, and achieve state-of-the-art 3D inpainting performance on both. A conference paper

based on the work presented in this thesis has been accepted to the British Machine Vision Conference (BMVC) 2025 for presentation. Please see our project page for visualizations of results: https://geomvi.github.io.

Chapter 2

Background and Related Work

This chapter provides the technical foundation necessary to understand our approach to geometry-aware multiview inpainting. We begin in Sec. 2.1 by introducing diffusion models, the generative framework that underpins our method, covering their fundamental processes, training objectives, and conditional generation capabilities. We then examine in Sec. 2.2 how 2D diffusion priors have been adapted for 3D inpainting tasks, highlighting the challenges of multiview inconsistency and existing approaches to address them. Next, in Sec. 2.3 we survey 3D scene editing techniques, with particular focus on inpainting methods that leverage radiance fields and diffusion models. In Sec. 2.4 we discuss reference-conditioned generative editing approaches that enable consistent generation across multiple views, and in Sec. 2.5 we examine iterative and autoregressive generation paradigms for 3D content creation. Finally, in Sec. 2.6 we provide a detailed technical overview of DUSt3R, the scene geometry estimation method that serves as a key component of our approach. Together, these topics establish the conceptual and technical

groundwork for our geometry-aware multiview inpainting algorithm that achieves consistent multiview inpainting without requiring dense view coverage or explicit 3D model fitting.

2.1 Diffusion Models

Diffusion models [13, 22, 23, 67, 69] are a class of generative models that learn to invert a fixed noising process, progressively transforming random noise into structured data. This section covers the fundamental concepts of diffusion models, starting with the basic forward and reverse processes, followed by training objectives and inference procedures. We then discuss conditional generation through classifier-free guidance and conclude with latent diffusion models, which enable efficient high-resolution synthesis.

2.1.1 Denoising Diffusion Probabilistic Models

The diffusion framework consists of two complementary processes: a forward process that gradually adds noise to a clean data sample until it becomes pure random noise, and a reverse process that learns to remove this noise step by step to recover the original data. Both processes are illustrated in Fig. 2.1. Originally introduced in discrete time as denoising diffusion probabilistic models (DDPMs) [23], the framework has since been generalized to continuous-time formulations using stochastic differential equations (SDEs) [69].

Given a data sample, $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, the forward process progressively adds Gaussian noise to a data point over time, resulting in a noise distribution, typically standard Gaus-

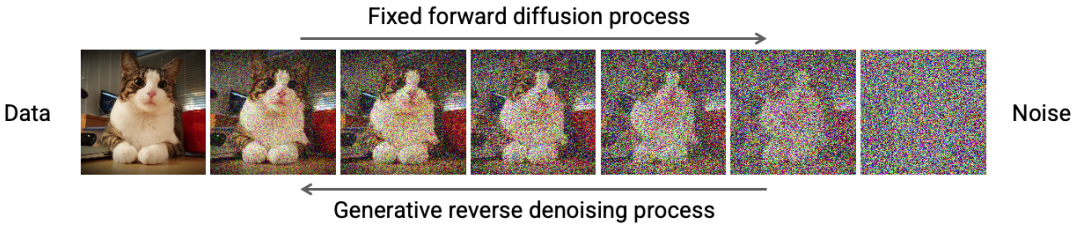

Figure 2.1: Illustration of the diffusion process in DDPMs. The forward process gradually adds Gaussian noise to a data sample over multiple timesteps until the signal becomes indistinguishable from pure noise. The generative model learns to reverse this process, starting from noise and iteratively denoising to recover a coherent data sample. © NVIDIA Corporation. Reprinted from [77].

sian. In the DDPM formulation, the forward process is defined as a Markov chain:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbb{I}), \tag{2.1}$$

where $t \in \{1, \ldots, T\}$ is the diffusion timestep and β_t is a predefined variance schedule. The final step closely approximates an isotropic Gaussian distribution, i.e., $\mathbf{x}_T \sim p_T \approx \mathcal{N}(0, \mathbb{I})$. The forward process admits a closed-form marginal distribution at any timestep:

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbb{I}), \tag{2.2}$$

with $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$. In other words, using the reparameterization trick [30], we can sample $\mathbf{x}_t$ at any timestep t directly from $\mathbf{x}_0$ as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathbb{I}). \tag{2.3}$$

The generative process aims to learn the reverse of this diffusion, effectively denois-

ing from $\mathbf{x}_T$ back to $\mathbf{x}_0$. The reverse process is also a Markovian chain consist of small Gaussian denoising steps, parameterized by θ:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \sigma_t^2 \mathbb{I}), \tag{2.4}$$

where σ_t^2 denotes the variance of the noise added at timestep t, and $\boldsymbol{\mu}_\theta$ is a neural network. Usually, σ_t^2 is also predefined and equals to $(1 - \bar{\alpha}_t)$, similar to the forward process.

Instead of directly regressing the mean using $\boldsymbol{\mu}_\theta$, the model is often trained to predict the noise component added in the forward process using the noise estimation model, $\boldsymbol{\epsilon}_\theta$. The predicted mean is then reparameterized as:

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right). \tag{2.5}$$

To train the model, the objective is to minimize the difference between the true noise added during the forward process and the noise predicted by the model. This leads to the simplified loss function, commonly referred to as the "simple loss":

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \varepsilon \sim \mathcal{N}(0,\mathbb{I}), t \sim \mathcal{U}(1,T)} \left[\left\| \varepsilon - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\, \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon, t) \right\|^2 \right], \tag{2.6}$$

where ε is the Gaussian noise added to the clean sample, $\mathbf{x}_0$, to obtain $\mathbf{x}_t$. The expectation is taken over data samples, noise, and diffusion timesteps.

During inference, the model starts with pure Gaussian noise, $\mathbf{x}_T \sim \mathcal{N}(0, \mathbb{I})$, and iteratively applies the learned reverse diffusion process to generate a data sample. At each timestep, t, the model predicts the noise component, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$, and computes the denoised

estimate, $\mathbf{x}_{t-1}$, using the parameterized reverse process. This iterative denoising continues until reaching $\mathbf{x}_0$, which represents the generated data sample.

2.1.2 Classifier-Free Guidance

For conditional generation tasks, classifier-free guidance [22] enables conditioning the generation process without relying on an external classifier [13]. This is achieved by jointly training the model on conditional and unconditional objectives, enabling extrapolation from the unconditional prediction toward the conditional prediction during sampling. Mathematically, the guided noise estimate is computed as a weighted combination of the conditional and unconditional noise estimates:

$$\boldsymbol{\epsilon}_\theta^{\text{CFG}}(\mathbf{x}_t, t \mid \mathbf{y}) := \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \varnothing) + \gamma \left(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \mathbf{y}) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t \mid \varnothing) \right), \tag{2.7}$$

where γ is the guidance scale that controls the strength of the conditioning, $\mathbf{y}$ is the conditioning signal, and $\varnothing$ denotes the unconditional generation. By adjusting the guidance scale, γ, one can balance between sample fidelity and diversity, effectively controlling the influence of the conditioning information during inference. More specifically, the guidance scale, γ, provides intuitive control over the generation behavior: $\gamma = 0$ results in pure unconditional generation, $\gamma = 1$ corresponds to pure conditional generation, and $\gamma > 1$ penalizes the unconditional generation to further emphasize fidelity to the conditioning signal.

2.1.3 Latent Diffusion Models

Operating diffusion models directly in high-dimensional pixel space becomes computationally prohibitive for high-resolution images due to the quadratic scaling of memory and computation with image resolution. To address these computational challenges, Latent Diffusion Models (LDMs) [60] apply the diffusion process in a learned lower-dimensional latent space. A pretrained variational autoencoder (VAE) [30] compresses the input image, $\mathbf{x}$, into a latent representation, $\mathbf{z} = \mathcal{E}(\mathbf{x})$, where the diffusion and denoising processes are carried out. After sampling, the latent output, $\tilde{\mathbf{z}}$, is decoded back into pixel space via a decoder, $\mathcal{D}$, to obtain the final image, $\tilde{\mathbf{x}} = \mathcal{D}(\tilde{\mathbf{z}})$. This separation of compression and generative modeling enables significantly more efficient training and inference, especially for high-resolution synthesis.

The denoising model, $\boldsymbol{\epsilon}_\theta$, used in LDMs is typically a U-Net [61] architecture, consisting of downsampling and upsampling blocks with skip connections that preserve spatial information across resolutions. As shown in Fig. 2.2, attention mechanisms, particularly cross-attention layers, are integrated throughout the network to enable conditioning on various modalities such as text, semantic maps, or other images. During training, both unconditional and conditional inputs are used to support classifier-free guidance at inference time. This architecture makes LDMs highly flexible and scalable for complex generative tasks.

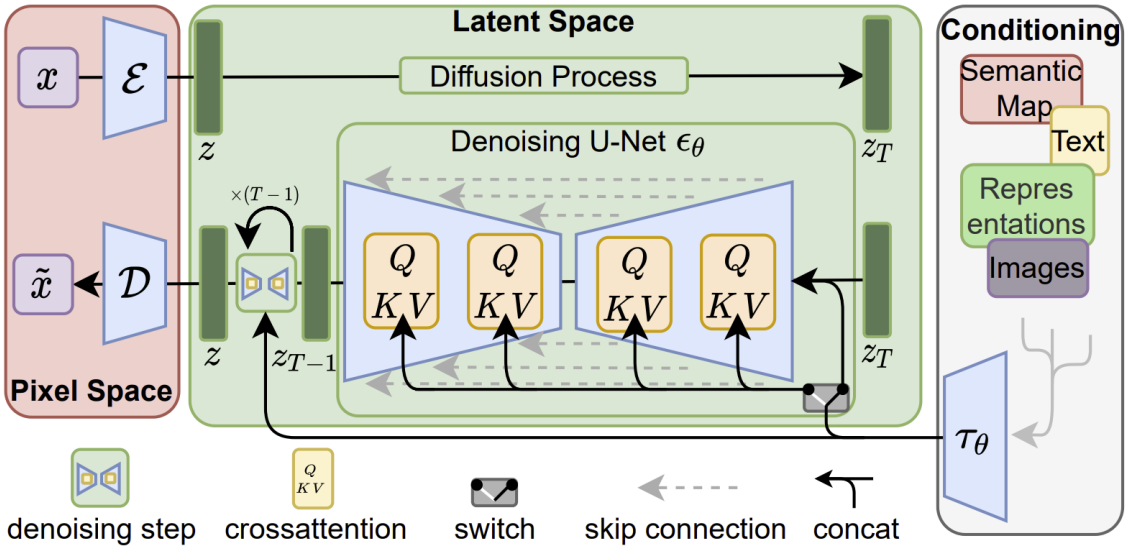

Figure 2.2: Architecture of latent diffusion models (LDMs) as proposed in [60]. The input image, $\mathbf{x}$, is first encoded into a latent representation, $\mathbf{z}$, using the encoder part of a pretrained variational autoencoder, $\mathcal{E}$. The diffusion process operates in latent space, where a denoising U-Net, ϵ_θ, learns to reverse the noising trajectory over T steps. The conditioning on various modalities such as text, semantic maps, or images is performed by either concatenation or a cross-attention mechanism. The denoised latent is decoded by $\mathcal{D}$ to produce the final image, $\tilde{\mathbf{x}}$. © 2022 IEEE. Reprinted, with permission, from [60].

2.2 2D Diffusion Priors for 3D Inpainting

2D image inpainting is a well-studied problem, and recent advances in *diffusion models* have greatly improved the quality of inpainted images [55]. Modern diffusion-based inpainting frameworks can produce photorealistic and semantically coherent completions of masked regions in a single image. For example, *Stable Diffusion* [60], a latent diffusion model originally developed for text-to-image generation, can be adapted for image inpainting tasks and has demonstrated high-quality results [70]. Extensions of Stable Diffusion like *ControlNet* provide additional conditioning (such as edges, depth maps, or segmen-

tation) to guide the generation process [95], enabling more controllable and structurally faithful inpainting outcomes. Another recent model, *BrushNet*, introduces a dual-branch diffusion architecture that explicitly incorporates the masked image content into the generation process, resulting in more coherent and high-fidelity fill-ins of missing regions [27]. These diffusion-based approaches represent the state of the art in 2D inpainting, achieving impressively realistic results on single-view images.

However, applying these 2D diffusion models independently to each view of a 3D scene leads to *multiview inconsistency*, as illustrated in Fig. 2.3. In a multiview inpainting scenario, the same missing content observed from different camera angles should be completed in a consistent manner across all images [47, 48]. Off-the-shelf 2D inpainters have no built-in understanding of the underlying 3D geometry or cross-view relations; therefore, filling holes in each view separately often yields incompatible results (e.g., the object inpainted in one view might appear with a different shape or texture, or not appear at all in another view). This lack of geometric coherence across viewpoints is a fundamental challenge when using 2D priors for 3D inpainting and can manifest as visual discrepancies when the views are assembled into a single scene.

Despite the challenges associated with multiview inconsistency, recent approaches have sought to harness the powerful generative priors of diffusion-based 2D inpainters by integrating them with explicit or implicit 3D radiance fields. For instance, NeRFiller [85] utilizes off-the-shelf 2D generative models to complete missing regions in 3D scenes, effectively distilling 2D inpainting results into a consistent NeRF representation using IDU [19]. Similarly, Inpaint3D [54] employs a 2D diffusion model to guide the inpainting of 3D scenes, optimizing a NeRF to align with the generative priors from 2D inpaint-

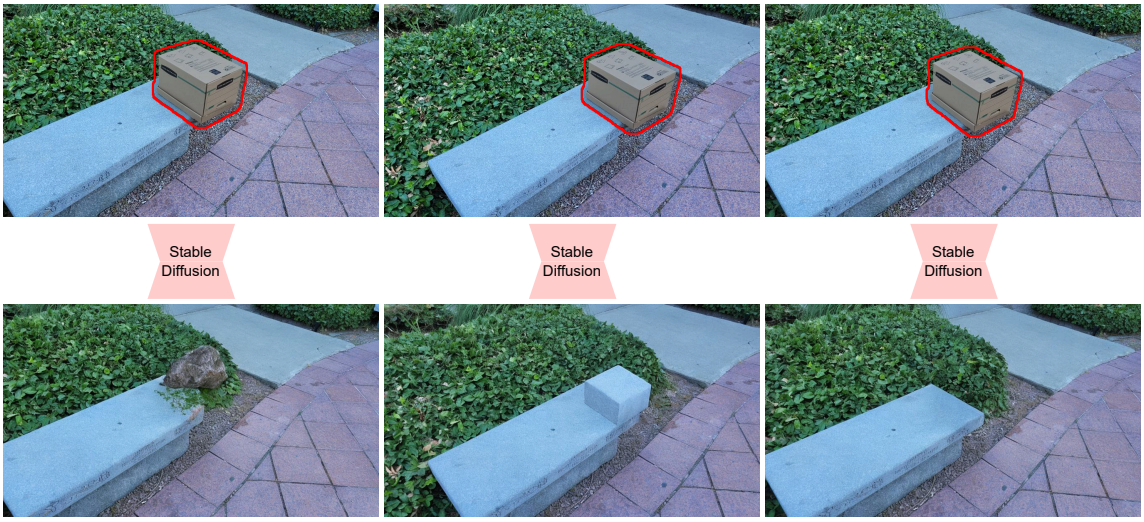

Figure 2.3: Illustration of multiview inconsistency when applying 2D diffusion models independently to each view of a 3D scene. The same missing region produces different completions across viewpoints, leading to geometric and appearance inconsistencies that violate the underlying 3D structure.

ing via SDS [53]. Additionally, RefFusion [50] adapts an inpainting diffusion model to a reference image and uses SDS [73] to inpaint a 3DGS [29] with the reference-adapted diffusion model. Building upon these advancements, MVIP-NeRF [8] introduces a novel approach that leverages diffusion priors for NeRF inpainting, addressing both appearance and geometry aspects by inpainting the rendered images and normal maps. MVIP-NeRF performs joint inpainting across multiple views to achieve a consistent solution through an iterative optimization process based on SDS [53]. Furthermore, MALD-NeRF [35] proposes tempering the stochasticity of diffusion models with per-scene customization and mitigating textural shifts through masked adversarial training, resulting in state-of-the-art NeRF inpainting results.

Although these methods effectively leverage the high-quality priors from 2D diffusion

models within a 3D framework, leading to multiview-consistent inpainting results, they often encounter challenges related to the quality of the final renders. Specifically, the *geometry-unaware* 2D inpainting results in multiview inconsistency, due to the stochastic nature of diffusion models. Fusing these inconsistencies in the pixel space of the 3D radiance field may lead to blurry renders. In this thesis, we address this issue by constraining a diffusion-based inpainter using scene geometry. By incorporating geometric information into the inpainting process, we aim to preserve the strong generative priors of diffusion models while ensuring coherent and realistic completions across multiple views.

2.3 3D Scene Editing

Editing 3D scenes is a fundamental aspect of 3D content creation, with applications spanning video games, virtual reality, and film production. The advent of 3D radiance fields [7, 56], such as NeRFs [46], has catalyzed a surge in innovative editing techniques. These methods encompass a variety of operations, including scene translation [10, 11, 15, 19, 31, 49, 84, 87], super-resolution [25, 34], shape deformation [26, 94, 99], appearance modifications [32, 33, 43, 82], and inpainting [47, 48, 64, 86].

Inpainting, the process of filling in missing or occluded regions within a scene, is particularly the focus of our work. Traditional 3D inpainting methods have focused primarily on object removal [42, 48, 78, 83, 86]. In contrast, our method can insert additional content and perform scene completion. Several specialized approaches have emerged for specific inpainting scenarios. RenderDiffusion [1] employs 2D diffusion models for 3D-aware inpainting but is constrained by weak supervision and struggles with complex scenes. SIGN-

eRF [14] focuses on localized translation and object insertion, offering limited support for broader inpainting tasks. Chen et al. [9] investigate the influence of inpainting masks for deterministic object removal but are restricted to forward-facing scenes. Gaussian Grouping [90] integrates segmentation features directly into the radiance field, facilitating various editing tasks.

For general inpainting, most methods build on 3D radiance fields [8, 35, 39, 50, 54]. Several approaches, particularly relevant to our work, leverage SDS [53, 79] to incorporate 2D diffusion priors. These include RefFusion [50], Inpaint3D [54], and MVIP-NeRF [8]. However, these approaches typically require relatively dense view coverage for effective 3D model fitting. This limits their applicability to real-world scenarios where capturing many viewpoints may be impractical or costly.

In contrast, our method directly inpaints the image set, allowing it to operate effectively even with sparse view coverage. This capability is particularly valuable for practical applications such as editing existing photo collections, virtual reality content creation, or architectural visualization, where obtaining dense multiview data is often challenging or prohibitively expensive.

Similar to our work, recent inpainters have adapted diffusion models for multiview inpainting. For instance, MVInpainter [5] formulates 3D editing as a multiview 2D inpainting task, utilizing reference guidance to partially inpaint multiview images without the need for explicit camera poses, thereby simplifying the complexity of 3D inpainting in real-world scenarios. However, MVInpainter is limited to processing sequences of up to 24 frames, which may restrict its applicability to larger scenes.

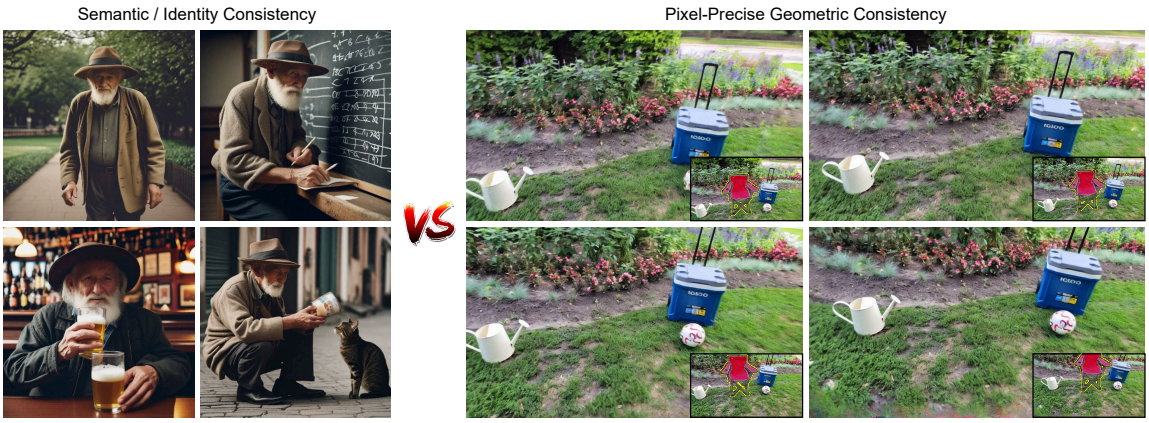

Figure 2.4: Comparison between semantic/identity consistency (left, adapted from [75], with permission. © 2024 ACM.) and pixel-precise geometric consistency (right, our method). Semantic consistency methods maintain conceptual coherence (e.g., person identity) but allow geometric variations across views. Conversely, 3D inpainting requires maintaining precise spatial relationships and geometric properties across all viewpoints.

2.4 Reference-Conditioned Generative Editing

Reference-conditioned generative editing leverages additional reference images to guide the generation or modification of target images. While various methods encourage consistency across multiple generations by sharing features across diffusion processes [2, 62, 75, 101], they primarily enforce *semantic* consistency, which is insufficient for achieving the precise, pixel-level coherence required for 3D-consistent content. As illustrated in Fig. 2.4, semantic consistency ensures that the same conceptual elements (e.g., a person's identity or general appearance) are maintained across views, but allows for variations in precise positioning, orientation, and geometric details. In contrast, pixel-precise geometric consistency, as required for multiview 3D inpainting, demands that the spatial relationships and geometric properties of inpainted content remain coherent across all viewpoints.

Conditional image generators have been employed for novel view synthesis (NVS) by

mapping observed reference images and target camera parameters to generate new views [6, 17, 38, 65, 76, 88, 91–93]. These models are trained on multiview datasets to learn the complex 3D transformations necessary for NVS. However, their primary focus is on generating new views rather than inpainting existing 3D scenes coherently.

While our conditional diffusion model is reminiscent of this, addressing the entirety of the NVS problem is not necessary for 3D inpainting; instead, we assemble reference information from multiple source views in the image coordinate frame of the target view, without relying on explicit camera information. This strategy allows us to disentangle the geometry transformations from the generative inpainting. Hence, our inpainter avoids learning complex 3D transforms (e.g., triangulating and projecting between frames). Instead, it learns to utilize projected reference information entirely in 2D, while accounting for simple 3D constraints, such as relative depth ordering of generated content with respect to existing content.

Finally, reference-based inpainting methods, such as [97, 98, 102], utilize one reference image to inpaint another, sometimes handling transformations beyond viewpoint changes, like lighting variations. Although our method also leverages reference information, it aligns more closely with NeRF-based approaches, operating on full multiview image sets where all views are initially masked, rather than focusing on a single reference-target pair. This enables our approach to achieve coherent and realistic completions across multiple views, addressing the challenges associated with multiview-consistent inpainting.

2.5 Iterative 3D Generation

3D content generation processes are typically performed iteratively or autoregressively. Iterative approaches repeatedly refine the generated content through multiple cycles, progressively enhancing quality and consistency. These methods have prominently been employed in text-to-3D generation [44, 53] and instruction-driven 3D editing [19]. Recently, similar iterative techniques have also been adapted for 3D inpainting tasks. For example, SDS [53] has been effectively utilized in methods like Inpaint3D [54], RefFusion [50], and MVIP-NeRF [8], providing iterative refinement based on learned diffusion priors. Another iterative paradigm, known as IDU [19], has been employed by methods such as NeRFiller [85], where iterative updates to the training dataset progressively improve the 3D completion.

In contrast, autoregressive approaches sequentially generate new data conditioned on previously generated content. This approach has mainly been explored in the context of full-scene 3D generation [12, 24] and generative novel view synthesis tasks [37, 58, 59, 76, 91]. Autoregressive models inherently maintain consistency through conditioning on previous outputs, which makes them well-suited for tasks involving sequential or spatially coherent generation.

In this thesis, we introduce an autoregressive inpainting framework specifically designed for large-baseline multiview scenarios. Unlike previous iterative approaches that rely on extensive refinement cycles or dataset updates, our method generates missing regions sequentially in an autoregressive manner, leveraging strong diffusion-based priors and explicit geometric constraints. This approach enables effective handling of sparse or large-baseline view settings, maintaining high coherence and visual fidelity across all

generated views.

2.6 Scene Geometry Estimation with DUSt3R

Multiview stereo reconstruction aims to recover 3D scene geometry from multiple images, providing essential depth information and camera parameters needed for 3D scene understanding. Traditional structure-from-motion (SfM) pipelines like COLMAP [63] achieve this through a sequential process: first extracting and matching sparse feature points across images, then estimating camera poses through bundle adjustment, and finally computing dense depth maps via stereo matching. While highly successful for well-textured scenes with sufficient viewpoint overlap, these classical approaches can struggle with challenging scenarios such as textureless regions, wide camera baselines, or scenes with repetitive patterns.

DUSt3R [81] represents a modern learning-based approach to multiview stereo reconstruction that addresses many of these limitations. Unlike traditional methods that rely on hand-crafted features and explicit geometric constraints, DUSt3R leverages deep learning to directly predict dense 3D correspondences between image pairs. The method's end-to-end learning approach enables more robust performance in challenging conditions where traditional feature-based methods might struggle.

For our multiview inpainting application, DUSt3R offers several advantages over classical approaches like COLMAP. First, DUSt3R's ability to handle complex settings such as sparse view sets aligns well with practical scenarios where dense image capture may be impractical. Second, DUSt3R provides significantly faster runtime compared to COLMAP's

sequential feature extraction, matching, and bundle adjustment pipeline, which is crucial for our iterative inpainting framework that requires repeated geometry updates.

Given a set of N images, $\{I_i \in \mathbb{R}^{3\times H\times W}\}_{i=1}^{N}$, of a static scene and a scene graph, $G = (V, E)$, where $V = \{i\}_{i=1}^{N}$ is the set of view indices and $E \subseteq V \times V$ is the set of edges connecting views, DUSt3R uses a network, $\mathcal{P}$, to predict pairwise 3D pointmaps for an edge, $e = (i, j) \in E$, as

$$(X^{i,e}, C^{i,e}), (X^{j,e}, C^{j,e}) = \mathcal{P}(I_i, I_j), \tag{2.8}$$

where $X^{v,e} \in \mathbb{R}^{H\times W\times 3}$ is the 3D pointmap of I_v in the coordinate system of the first element of e, i.e., I_i, and $C^{v,e} \in \mathbb{R}_{+}^{H\times W}$ is the confidence map indicating the reliability of the predicted points.

The method then performs a global optimization to obtain the global camera parameters and dense depth maps. The optimization objective seeks to minimize the error between the predicted 3D points and their corresponding locations when transformed into a consistent global coordinate system, weighted by the confidence of each prediction. Mathematically, this is formulated as

$$\mathcal{G} = \underset{\mathbf{R},\mathbf{t},\mathbf{K},D,\sigma,\mathbf{P}}{\arg\min} \sum_{e\in E}\sum_{v\in e}\sum_{\mathbf{x}\in\Omega} C^{v,e}[\mathbf{x}] \left\| \mathbf{R}_v^{\top}\left(D_v[\mathbf{x}]\mathbf{K}_v^{-1}h(\mathbf{x}) - \mathbf{t}_v\right) - \sigma_e \mathbf{P}_e X^{v,e}[\mathbf{x}] \right\|_2^2, \tag{2.9}$$

where $\Omega = \{1, \ldots, H\} \times \{1, \ldots, W\}$ is the set of pixel coordinates, $h : \mathbb{R}^n \to \mathbb{R}^{n+1}$ is the homogeneous mapping, and $\mathcal{G}$ is the set of all optimized geometry parameters, including view extrinsics, $(\mathbf{R}_v, \mathbf{t}_v) \in \mathrm{SE}(3)$, intrinsics, $\mathbf{K}_v \in \mathbb{R}^{3\times 3}$, and dense depth maps, $D_v \in \mathbb{R}^{H\times W}$, for all views, $v \in V$, as well as pairwise poses, $\mathbf{P}_e \in \mathrm{SE}(3)$, and scale factors,

$\sigma_e \in \mathbb{R}_+$, for all edges, $e \in E$. In short, we denote this entire process as

$$\mathcal{G} = \mathrm{DUSt3R}(\{I_i\}_{i=1}^{N}, G), \tag{2.10}$$

where $\mathcal{G} = \{(\mathbf{R}_v, \mathbf{t}_v, \mathbf{K}_v, D_v)\}_{v \in V} \cup \{(\mathbf{P}_e, \sigma_e)\}_{e \in E}$ and G is the scene graph.

This two-stage approach, first predicting pairwise correspondences, then globally optimizing for consistent geometry, allows DUSt3R to handle challenging scenarios with wide baselines and sparse view coverage while maintaining computational efficiency. The method's ability to provide both dense depth maps and camera parameters makes it particularly suitable for applications requiring comprehensive scene understanding.

2.7 Summary

This chapter has established the technical foundation for our geometry-aware multiview inpainting approach by surveying key concepts and related work across six complementary areas. We began with diffusion models, the generative framework that underpins our method, covering their fundamental forward and reverse processes, training through noise prediction, conditional generation via classifier-free guidance, and efficient high-resolution synthesis through latent diffusion models. These concepts are essential for understanding how our method leverages powerful 2D generative priors for 3D-consistent inpainting.

We then examined the challenges of applying 2D diffusion priors to 3D inpainting, particularly the fundamental problem of multiview inconsistency that arises when independently applying 2D models to each view of a 3D scene. Existing approaches address

this by integrating 2D diffusion models with 3D radiance fields through techniques like Score Distillation Sampling (SDS) and Iterative Dataset Update (IDU), but often suffer from blurry results due to the fusion of inconsistent cross-view predictions in the pixel space of the 3D representation.

Our survey of 3D scene editing techniques revealed a diverse landscape of methods for manipulating radiance fields, with inpainting being particularly challenging due to the need for both high-quality generation and geometric consistency. Most existing 3D inpainting methods require dense view coverage for effective 3D model fitting, limiting their applicability to real-world scenarios where capturing many viewpoints may be impractical.

Reference-conditioned generative editing approaches offer insights into achieving consistency across multiple generations, though existing methods primarily enforce semantic rather than pixel-level coherence. Our approach builds on these concepts by assembling reference information from multiple source views in the image coordinate frame of the target view, allowing us to disentangle geometry transformations from generative inpainting while maintaining precise spatial consistency.

We explored iterative and autoregressive paradigms for 3D generation, with the latter being particularly relevant to our sequential inpainting approach. Unlike iterative methods that require extensive refinement cycles, autoregressive generation maintains consistency through conditioning on previous outputs, making it well-suited for our large-baseline multiview scenarios.

Finally, we provided a comprehensive technical overview of DUSt3R, the scene geometry estimation method that forms a crucial component of our approach. DUSt3R's two-stage process, pairwise correspondence prediction followed by global optimization,

enables robust geometry estimation from sparse or wide-baseline views, making it particularly suitable for our multiview inpainting framework where geometry serves as a key constraint for ensuring cross-view consistency.

Together, these background topics motivate our key contributions: a geometry-aware conditional diffusion model that achieves multiview-consistent inpainting without requiring dense view coverage or explicit 3D model fitting, operating directly on image sets while leveraging geometric constraints to ensure coherent completions across sparse/dense view sets or small/large-baseline viewpoints.

Chapter 3

Technical Approach

This chapter presents our geometry-aware multiview-consistent 3D scene inpainting algorithm. Our approach addresses the fundamental challenge of maintaining geometric consistency across multiple viewpoints while inpainting 3D scenes, without relying on explicit 3D radiance field representations.

We begin in Sec. 3.1 by formally defining the problem and establishing our assumptions and objectives. We then provide in Sec. 3.2 an overview of our approach, contrasting it with existing methods and highlighting our key innovations: avoiding explicit 3D radiance fields, fusing information in learned generative spaces, and enabling direct appearance transfer across views using estimated scene geometry.

Our method consists of two main components working in tandem: a scene geometry estimator based on DUSt3R [81] (Sec. 3.3) and a geometry-aware inpainting diffusion model (Sec. 3.4), combined within an autoregressive framework (Sec. 3.5). The geometry estimator provides dense depth maps and camera parameters from incomplete views, while

the diffusion model leverages this geometric information to perform multiview-consistent inpainting. At each iteration, we inpaint a subset of not-yet-inpainted views, conditioned on the other views (whether inpainted or not), followed by updating the estimated geometry. This cyclic process gradually fills in the scene's geometry and appearance while maintaining consistency across all viewpoints.

3.1 Problem Definition

As with prior work [48], we assume as given (i) a set of N views, $\left\{ I_i \in \mathbb{R}^{3\times H\times W} \right\}_{i=1}^{N}$, representing a static scene, and (ii) their corresponding inpainting masks, $\left\{ M_i \in \{0,1\}^{H\times W} \right\}_{i=1}^{N}$, demarcating the 3D region to be inpainted. While other methods may require additional inputs, such as camera parameters or depths per image, these are optional for our approach (though we can nonetheless use them). Our objective is to jointly inpaint the given views, thus inpainting the 3D scene, ideally in a consistent manner across views.

3.2 Overview

Prior works (e.g., [8, 47, 48, 50, 54, 85]) often have two disparate components: (i) an implicit or explicit 3D radiance field (e.g., NeRF [46] and 3DGS [29]), which fuses information across views to ensure consistency, and (ii) a 2D image inpainter, which alters the source views, potentially conditioned on the state of (i) (e.g., [19, 49, 85]). However, this separation has shortcomings: first, (i) fuses information in pixel space (due to the supervision mechanism), leading to blurry results, and second, (ii) is only aware of the other views through (i), meaning any mechanism for enforcing consistency is *indirect*. In contrast, our

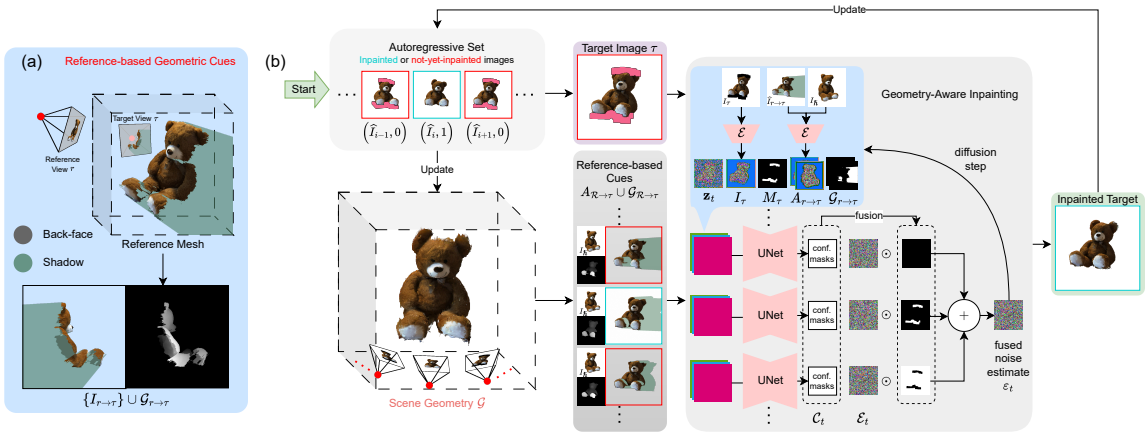

Figure 3.1: Overview of our geometry-aware 3D scene inpainter. (a) A visualization of the reference-based geometric cues. Back-faces are always obscured by shadow volumes. $I_{r\to\tau}$ denotes the rendered photometric content, and $\mathcal{G}_{r\to\tau} = \left\{F_{r\to\tau}, B_{r\to\tau}, S_{r\to\tau}, \widehat{D}_{r\to\tau}\right\}$ denotes the geometric cues (front-face mask, back-face mask, shadow mask, and normalized inverse depth, respectively) for the reference view, r, projected to target view, τ. (b) A step-by-step visualization of our autoregressive inpainting process. Note that the scene geometry consists of separate meshes for each image, not a single harmonized mesh, as shown here for simplicity. Here, we are only showing one diffusion step for the geometry-aware inpainting model. $\left\{\left(\widehat{I}_i, b_i\right)\right\}_{i=1}^{N}$ denotes the autoregressive set where $\widehat{I}_i$ is the image (whether inpainted or not) and b_i is the inpainted-status indicator; $\mathcal{G}$ denotes the scene geometry estimate; $\mathcal{E}$ denotes the VAE encoder (which maps an image to the latent space of Stable Diffusion); $\mathbf{z}_t$ denotes the diffusion latent at timestep t; I_τ, M_τ denote the target image and mask; $I_\hbar$ denotes the hint image; $A_{r\to\tau}, \mathcal{G}_{r\to\tau}$ denote the appearance and geometric cues from reference view, r, to target view, τ; $A_{\mathcal{R}\to\tau}, \mathcal{G}_{\mathcal{R}\to\tau}$ denote the sets of appearance and geometric cues from all reference views to target view, τ; $\mathcal{C}_t = \{\mathbf{C}_t^f, \mathbf{C}_t^b, \mathbf{C}_t^s\}, \mathcal{E}_t$ denote the sets of predicted confidence masks and noise estimates, respectively; and ε_t denotes the fused noise estimate at diffusion timestep t; see Fig. 3.5 for an example of autoregressive steps.

algorithm relaxes the need for a radiance field, fuses information in the *learned* space of a generative model (avoiding the blur of pixel space), and enables *direct* appearance transfer across views, using estimated scene geometry.

To achieve this, our approach combines two key models: a scene geometry estimator

and a geometry-aware inpainting model. For the geometry estimator, we use the performant DUSt3R [81], which efficiently provides dense depth, with or without the presence of camera poses, utilizing views (inpainted or not) directly. For the latter, we fine-tune a latent 2D diffusion-based inpainter, to condition on the other views. However, naively conditioning the inpainter forces it to learn both inpainting and generative NVS (itself a non-trivial task [17, 74, 92]), by learning to map one view to another using an internal scene model. Instead, we use our scene geometry estimator to feed appearance information from other views to our inpainter, by directly projecting information from source views, as well as passing explicit geometric information pertinent to the inpainting (e.g., occlusion).

Given these two models, we devise a simple autoregressive scene inpainting algorithm. At each iteration, we inpaint a subset of not-yet-inpainted views, conditioned on the other views (whether inpainted or not), followed by updating the estimated geometry. As this cyclic process continues, the scene geometry and appearance in each view are gradually filled in; see Fig. 3.1 for a schematic of our approach.

3.3 Scene Geometry Estimator

We utilize DUSt3R [81] for scene geometry reconstruction, as detailed in Sec. 2.6. As we inpaint the scene, we iteratively reapply it to update the scene geometry throughout the process.

DUSt3R is not trained on incomplete views (i.e., not-yet-inpainted images); however, we adapt it to use incomplete views by passing the images to the model with the masked

area set to zero, i.e., for an incomplete view, v, we pass $I_v \odot \neg M_v$ to $\mathcal{P}$, where $\odot$ and $\neg$ are the element-wise multiplication and logical negation operators, respectively. Also, for each edge, e, containing the incomplete view, v, we suppress DUSt3R's predicted confidence maps in the masked area by setting the confidence to zero, i.e., we use

$$\tilde{C}^{v,e} = C^{v,e} \odot \neg \left((1 - b_v) \cdot M_v \right), \quad \forall e \in E; \forall v \in e, \tag{3.1}$$

as the confidence maps for global optimization, where b_i is the indicator of whether I_i has been inpainted. This will prevent incomplete parts from contributing during the optimization phase.

Optionally, we also pre-set ground-truth information for the optimization phase. For pre-setting camera parameters, we initialize and freeze the corresponding parameters in the optimization. For pre-setting incomplete dense depth, we only initialize the depth (wherever provided), while allowing it to change during the optimization. We denote our modified version of DUSt3R as

$$\mathcal{G} = \mathrm{DUSt3R}(\{(I_i, M_i, b_i)\}_{i=1}^{N}, G, \mathcal{G}_{\text{init}}), \tag{3.2}$$

where $\mathcal{G}_{\text{init}}$ is a subset of geometry parameters that we wish to pre-set.

3.4 Geometry-Aware Inpainting Diffusion Model

At each step of our autoregressive inpainting process, after updating the scene geometry from the previous step (Sec. 3.3), we apply our geometry-aware inpainting diffusion model

to inpaint a subset of views. This model is designed to leverage both the estimated scene geometry and appearance information from reference views to ensure multiview consistency.

Our goal is to devise an image inpainting model that conditions on a multiview image set and the current scene geometry estimate. At any given autoregressive step, the reference view set may include both masked (not-yet-inpainted) and complete (previously inpainted) images, all of which are used to inform the inpainting of the target image. This enables a form of reference-based inpainting with explicit 3D geometric constraints derived from our scene geometry estimator.

Our conditioning approach leverages geometric knowledge of the scene. In particular, *using scene geometry, we project appearance and geometric information from references into the target camera's image plane*; this alleviates the need for the network to learn geometry estimation and view transforms (as in NVS), saving its capacity for inpainting. Given depth maps and camera parameters from Sec. 3.3, this projection is performed via mesh rendering. However, three issues remain: (i) selecting geometry-driven cues to condition the model on, (ii) fusing cues across multiple reference views, and (iii) training the inpainter to handle errors in estimated scene geometry.

To obtain our geometry-driven cues, we first construct a surface mesh derived from the depth maps. This mesh also implies a "shadow volume" [18, 45], representing the 3D space *hidden from the reference camera* (see Fig. 3.1(a) for a visualization). Thus, we also construct a shadow volume mesh from each reference's surface mesh. This dual-mesh representation enables us to compute geometric cues that encode the reliability of projected appearance information. We first establish our notation (Sec. 3.4.1), then detail our mesh

construction approach (Sec. 3.4.2) and shadow volume construction (Sec. 3.4.3) which together enable the projective mapping of the photometric and geometric information across views. We then define the conditioning signals (Sec. 3.4.4). The remaining issues of multi-reference fusion and robust training are explained in Sections 3.4.5 and 3.4.6, respectively.

3.4.1 Notation

Formally, let I_τ be the target image to be inpainted (i.e., inpainting a single image at a time), with mask, M_τ, and camera parameters, $(\mathbf{R}_\tau, \mathbf{t}_\tau) \in \mathrm{SE}(3), \mathbf{K}_\tau \in \mathbb{R}^{3\times3}$ (i.e., extrinsics and intrinsics). Each element of the reference view-set,

$$\mathcal{R} = \{(I_r, M_r, b_r, \mathbf{R}_r, \mathbf{t}_r, \mathbf{K}_r, D_r)\}_r, \tag{3.3}$$

consists of an image (I_r), mask (M_r), indicator of whether I_r has been inpainted (b_r), extrinsic camera parameters ($\mathbf{R}_r, \mathbf{t}_r$), intrinsic camera parameters ($\mathbf{K}_r$), and depth map (D_r). Camera and depth information come from our geometry estimator (Sec. 3.3). Thus, our inpainter inpaints I_τ, within the masked region, M_τ, conditioning on the reference view-set, $\mathcal{R}$ and the current state of the estimated scene geometry, $\mathcal{G}$.

3.4.2 Surface Mesh Construction

To enable coordinate-aligned conditioning, we establish the geometric foundation for our projective mapping approach. For each reference image, I_r, we construct a triangle mesh, $\mathcal{M}_r$, using its corresponding geometry and camera information, and render the mesh to the target frame to map appearance and geometric information from the reference view.

Given the camera parameters, $(\mathbf{R}_r, \mathbf{t}_r) \in \mathrm{SE}(3), \mathbf{K}_r \in \mathbb{R}^{3\times3}$, and dense depth map, $D_r \in \mathbb{R}^{H\times W}$, from our geometry estimator (Sec. 3.3), we lift pixel coordinates to the world coordinate system via the lifting function

$$\chi(\mathbf{x}) := \mathbf{R}_r^\top \left(D_r[\mathbf{x}] \mathbf{K}_r^{-1} h(\mathbf{x}) - \mathbf{t}_r \right), \tag{3.4}$$

where $h : \mathbb{R}^n \to \mathbb{R}^{n+1}$ is the homogeneous mapping. Following GeoFill [98], we build a triangle mesh with a regular grid, where the mesh vertices are provided by the lifting function, $\chi(\mathbf{x})$. To create discontinuities on object boundaries, we drop mesh edges wherever there is a sudden change in depth. We use the same criterion as GeoFill [98], where we drop the edge between two adjacent vertices, v_j and v_k, if

$$\frac{2\left|D_r[\mathbf{x}_j] - D_r[\mathbf{x}_k]\right|}{D_r[\mathbf{x}_j] + D_r[\mathbf{x}_k]} > \epsilon_{\text{edge}}, \tag{3.5}$$

where ϵ_{edge} is a user-defined hyperparameter (see Tab. A.1). This process results in the 3D triangle mesh, $\mathcal{M}_r$, that represents the scene geometry as observed from reference view, r.

Several pieces of information can be extracted when rendering the mesh, $\mathcal{M}_r$, to a target camera frame with extrinsics, $(\mathbf{R}_\tau, \mathbf{t}_\tau)$, and intrinsics, $\mathbf{K}_\tau$: (i) appearance information, $I_{r\to\tau} \in \mathbb{R}^{3\times H\times W}$, (ii) the map of front-facing triangles, $F_{r\to\tau} \in \{0,1\}^{H\times W}$, (iii) the map of back-facing triangles, $B_{r\to\tau} \in \{0,1\}^{H\times W}$, and (iv) the depth map, $D_{r\to\tau} \in \mathbb{R}^{H\times W}$. Mathematically,

$$I_{r\to\tau}, F_{r\to\tau}, B_{r\to\tau}, D_{r\to\tau} = \mathrm{Render}(\mathcal{M}_r, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}_\tau), \tag{3.6}$$

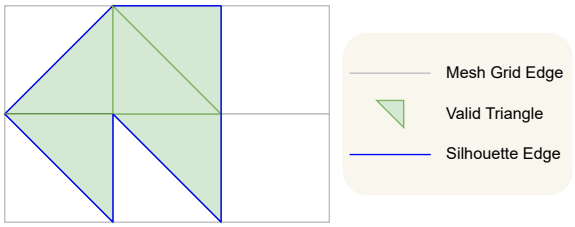

Figure 3.2: Illustration of silhouette edge detection in a triangle mesh. Silhouette edges (highlighted in blue) are identified as edges that belong to only one triangle, marking the boundaries where the surface geometry transitions from visible to occluded regions when viewed from a specific camera position.

where $\mathrm{Render}(\cdot)$ is a mesh rendering function that projects the mesh into the target camera frame.

3.4.3 Shadow Volume Construction

In addition to the surface mesh, $\mathcal{M}_r$, we also need to construct shadow volume meshes that represent the 3D space hidden from each reference camera. These shadow volumes are essential for computing the shadow masks that indicate regions where photometric information may be occluded and thus unreliable.

To create the shadow volume mesh, $\mathcal{M}_r^S$, for a reference view, we first identify the silhouette edges of the surface mesh, $\mathcal{M}_r$. When building the surface mesh as described in Sec. 3.4.2, we also identify the triangle edges at surface boundaries (i.e., silhouette edges) by looking for edges that belong to only one triangle. Fig. 3.2 illustrates the selection of silhouette edges. Let $E_{\mathcal{S}} = \left\{ \left(v_1^{(i)}, v_2^{(i)} \right) \right\}_i$ denote the set of silhouette edges in the world coordinate system. Given the camera extrinsics, the camera center is at $-\mathbf{R}_r^\top \mathbf{t}_r$ in the world coordinate system. Our goal is to draw a ray from the camera center to each point on the edge of the shape (i.e., its occlusion boundaries), and then extend the ray past it

(forming part of the boundary of the shadow volume induced by that geometric element; see Fig. 3.1 for a visualization).

For each vertex, v, in a silhouette edge, we extrude it along its corresponding ray direction according to the reference camera, as

$$v' = v + \varepsilon_d \frac{v + \mathbf{R}_r^\top \mathbf{t}_r}{\|v + \mathbf{R}_r^\top \mathbf{t}_r\|_2}, \tag{3.7}$$

where ε_d is a sufficiently large value to ensure the rendered shadow volume covers the relevant part of other views. For each silhouette edge, $(v_1, v_2) \in E_{\mathcal{S}}$, we form a quad, (v_1, v_2, v_2', v_1'), forming the side walls of the shadow volume. We then split each shadow quad into two triangles and use the resulting triangle mesh, $\mathcal{M}_r^S$, to render the shadow volumes, as

$$S_{r\to\tau} = \mathrm{Render}(\mathcal{M}_r^S, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}_\tau), \tag{3.8}$$

where $S_{r\to\tau} \in \{0,1\}^{H\times W}$ is the shadow mask projected to the target camera frame. The shadow mask indicates pixels in the projected photometric content, $I_{r\to\tau}$, that may be occluded from the reference camera, r, and thus not necessarily reliable for inpainting.

3.4.4 Appearance and Geometric Cues

Building on our mesh-based projection framework, we now define the conditioning signals that inform our diffusion model of the 3D scene. We extract both appearance cues that provide photometric information and geometric cues that encode the reliability and validity of this photometric content.

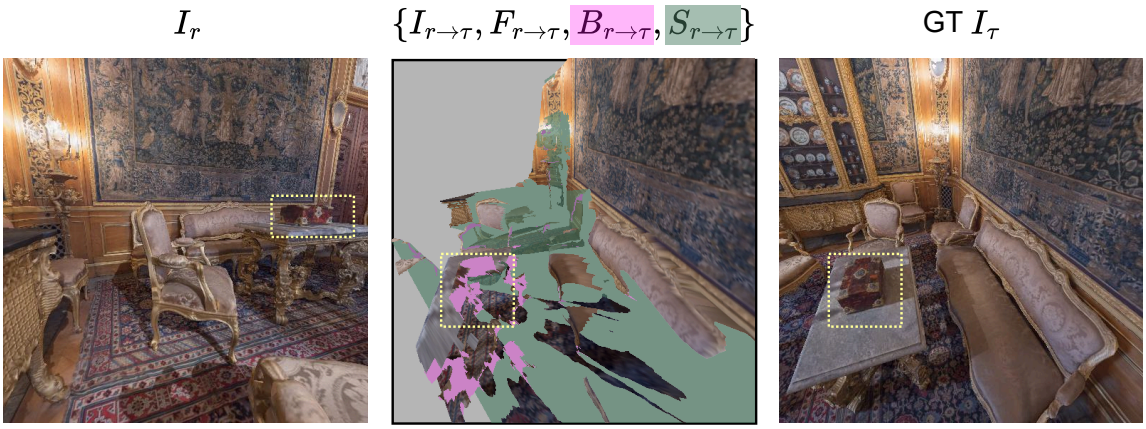

Figure 3.3: Example of insufficient projected information from distant viewpoints. The reference image, I_r (left), contains a highlighted decorative box that appears clearly visible. However, when projecting reference information to the target view, τ, through the rendered cues, $\{I_{r\to\tau}, F_{r\to\tau}, B_{r\to\tau}, S_{r\to\tau}\}$ (middle), the highlighted object predominantly shows back-face triangles, providing insufficient appearance information for the inpainter to maintain stylistic consistency with the original object in the reference view. The ground-truth target view, I_τ (right), shows how the decorative box should appear from the target viewpoint. This limitation motivates our stylistic hint mechanism.

Appearance Cues

We provide the inpainter with two appearance cues: (a) direct reference pixels and (b) a stylistic hint. For (a), we pass $I_{r\to\tau}$ instead of I_r, giving the inpainter direct access to view-aligned pixel colours for localized convolutional processing. However, with large camera baselines, projected information from distant viewpoints may be insufficient (e.g., only the back-face triangles of an object are visible, as illustrated in Fig. 3.3). To ensure a *stylistic* harmony, we introduce (b), an optional "hint image", $I_\hbar$, which is not projected through the scene geometry but rather provides global appearance characteristics.

In the first autoregressive step, as we do not have any inpainted images, we use an empty image (zero-valued) as the hint. For all other steps, we select the furthest inpainted

image to the target image, τ. We use the rotation (orientation) difference between the cameras as our camera distance measure, since the cameras nearly always point to a central point in the scene. Specifically, for a pair of views, i and j, with corresponding extrinsic rotation matrices $\mathbf{R}_i, \mathbf{R}_j \in \mathrm{SO}(3)$, we define the *view distance function* as

$$d(i, j) = \left\| A\left(\mathbf{R}_i^\top \mathbf{R}_j\right) \right\|_2, \tag{3.9}$$

where $A : \mathrm{SO}(3) \to [0, 2\pi) \times [0, 2\pi) \times [0, \pi)$ is a function mapping a rotation matrix to Euler angles. In other situations, this heuristic could be altered (e.g., to use an estimate of overlapping image content, or camera positional information). Therefore, we formalize the selection of the hint image as

$$\hbar = \arg\max_h b_h \cdot d(\tau, h), \tag{3.10}$$

where $b_h \in \{0, 1\}$ indicates whether the image, I_h, has been inpainted. We denote our appearance cues as

$$A_{\mathcal{R}\to\tau} = \{I_{r\to\tau}\}_{r=1}^{R} \cup \{I_\hbar\}, \tag{3.11}$$

where $R = |\mathcal{R}|$ is the number of reference views.

Geometric Cues

While appearance cues provide photometric information, they alone are insufficient for reliable multiview conditioning due to ambiguities in projection validity and occlusion. To address this, we leverage geometric cues that encode the reliability and validity of the

photometric content, illustrated in Fig. 3.1 (a). Specifically, per reference, we compute a (i) front-face mask, $F_{r\to\tau}$, (ii) back-face mask, $B_{r\to\tau}$, (iii) normalized inverse depth (ordering), $\widehat{D}_{r\to\tau}$, computed as

$$\tilde{D}_{r\to\tau} = \frac{1}{D_{r\to\tau}}, \tag{3.12}$$

$$\widehat{D}_{r\to\tau} = \frac{\tilde{D}_{r\to\tau} - \min_{\mathbf{x}}\{\tilde{D}_{r\to\tau}[\mathbf{x}]\}}{\max_{\mathbf{x}}\{\tilde{D}_{r\to\tau}[\mathbf{x}]\} - \min_{\mathbf{x}}\{\tilde{D}_{r\to\tau}[\mathbf{x}]\}} \odot \left(F_{r\to\tau} \vee B_{r\to\tau}\right), \tag{3.13}$$

where $\vee$ is the logical OR operator, and (iv) shadow mask, $S_{r\to\tau}$, all in the target coordinate frame. Front- and back-face masks indicate the validity of the projected photometric content (i.e., the former has valid appearance from I_r, while the latter merely implicates the presence of geometry), the normalized inverse depth provides a measure of relative depth ordering, and the shadow mask indicates potential occlusion. These maps form our geometric cue set,

$$\mathcal{G}_{\mathcal{R}\to\tau} = \left\{ \left(F_{r\to\tau}, B_{r\to\tau}, \widehat{D}_{r\to\tau}, S_{r\to\tau} \right) \right\}_{r=1}^{R}, \tag{3.14}$$

another input to our diffusion model. Importantly, these cues *also* enable a hierarchical, confidence-based fusion of reference information, based on the uncertainty induced by the geometric structure (see Sec. 3.4.5). With both appearance cues, $A_{\mathcal{R}\to\tau}$, and geometric cues, $\mathcal{G}_{\mathcal{R}\to\tau}$, defined, we next address how to effectively fuse information across multiple reference views.

3.4.5 Uncertainty-Aware Multi-Reference Conditioning

Using our cue-based notation, our method inpaints an image, I_τ, within the masked region, M_τ, conditioned on the appearance cues, $A_{\mathcal{R}\to\tau}$, and geometric cues, $\mathcal{G}_{\mathcal{R}\to\tau}$. However, fusing information across multiple references is challenging, especially with conflicting content. We address this by running parallel per-reference diffusion processes and fusing noise estimates at each diffusion step. Ideally, this fusion is geometry-aware (e.g., front-faces of $\mathcal{M}_r$ provide high-certainty photometric content, while back-faces only upper bound target view depth). Since geometric maps may be slightly misaligned due to errors in estimated geometry, we alter the diffusion model to predict confidence maps for each reference, emulating the aligned geometric masks. In the following, we describe how our inpainting is implemented, including confidence estimation and fusion.

Hierarchical Confidence Estimation

The geometric cues, $\mathcal{G}_{\mathcal{R}\to\tau}$, indicate the reliability of the projected photometric content. We utilize three geometric signals: first, the *front-facing confidence mask*, $\mathbf{C}^f$, indicates a pixel is either outside the inpainting mask or guided by a front-facing rendered pixel. In other words, the model has copied the photometric content from the target image itself or the rendered photometric content ($I_{r\to\tau}$). Second, the *back-facing confidence mask*, $\mathbf{C}^b$, indicates a pixel is geometrically restricted by a rendered *back*-face, suggesting new geometry to be generated *in front* of it. Finally, the *shadow confidence mask*, $\mathbf{C}^s$, signals the model's certainty in trusting photometric information despite *potential* occlusion (shadow volume). Notice the confidence masks are closely related to the geometric cues, $F_{r\to\tau}$, $B_{r\to\tau}$, and $S_{r\to\tau}$. Importantly, though, these cues are often slightly misaligned with the

actual target image due to geometry estimation errors. Thus, we instead modify our diffusion model to *estimate these confidence masks* at every diffusion step, in addition to the noise estimate. Please see Sec. 3.4.6 for details on supervising the confidence masks, and Fig. 3.4 for a visualization.

Parallel Diffusion Processing

We now formalize the inpainting process. Given our diffusion model, $\boldsymbol{\epsilon}_\theta$, we split the inpainter into R independent streams to inpaint a target image, I_τ. Let

$$\left(\varepsilon_{r,t}, \mathbf{C}^f_{r,t}, \mathbf{C}^b_{r,t}, \mathbf{C}^s_{r,t}\right) = \boldsymbol{\epsilon}_\theta\left(\mathbf{z}_t, t \mid \mathbf{y}, I_\tau \odot \neg M_\tau, M_\tau, A_{r\to\tau}, \mathcal{G}_{r\to\tau}\right), \tag{3.15}$$

be the output of the rth process at diffusion timestep t, where $\varepsilon_{r,t}$ is the estimated noise for reference, r, $\mathbf{z}_t$ is the latent representation at diffusion timestep t, $A_{r\to\tau} = \{I_{r\to\tau}, I_\hbar\}$, $\mathcal{G}_{r\to\tau} = \{F_{r\to\tau}, B_{r\to\tau}, \widehat{D}_{r\to\tau}, S_{r\to\tau}\}$, and $\mathbf{y}$ is the text prompt. Denote

$$\mathcal{E}_t = \left[\varepsilon_{r,t}\right]_{r=1}^{R}, \tag{3.16}$$

$$\mathbf{C}^\rho_t = \left[\mathbf{C}^\rho_{r,t}\right]_{r=1}^{R}, \quad \rho \in \{f, b, s\}, \tag{3.17}$$

as the combined noise estimates and confidence maps. We then fuse the noise estimates as

$$\varepsilon_t = \Gamma(\mathcal{E}_t, \mathbf{C}^f_t, \mathbf{C}^b_t, \mathbf{C}^s_t), \tag{3.18}$$

thus fusing multiview information in the *learned, generative* space of the diffusion model, rather than in pixel space. The fusion, Γ, follows a four-level confidence hierarchy: (i)

front-face confidence, (ii) back-face confidence, (iii) shadow confidence, and (iv) no confidence. This hierarchy reflects decreasing geometric certainty: front-faces provide direct photometric evidence, back-faces constrain depth bounds, shadows indicate potential occlusion, and no confidence represents completely unseen regions. For each patch at each level, we select the noise estimate from the closest camera among the views at the same level. To that end, we also utilize the view distances of the reference views to the target view, computed as

$$d_\tau = \Big[d(\tau, r) \Big]_{r=1}^{R}, \tag{3.19}$$

where $d(i, j)$ is the view distance function defined in Eq. (3.9). This ensures fusion of the most reliable reference information during denoising. Algorithm 1 summarizes the steps taken by the fusion operator, Γ. Fig. 3.4 visualizes the fusion operator.

We then compute the next noisy latent, $\mathbf{z}_{t-1}$, by applying the denoising step of a diffusion scheduling algorithm, e.g., DDIM [68], as

$$\mathbf{z}_{t-1} = \mathrm{BackwardStep}(\mathbf{z}_t, \varepsilon_t, t). \tag{3.20}$$

3.4.6 Multiview-Aware Training

We initialize our geometry-aware inpainter with Stable Diffusion v2, fine-tuned for inpainting [60, 70]. Following prior work [4], we condition on cues by adding zero-initialized channels to the first convolutional layer of the U-Net [61]. We also modify the U-Net to output confidence masks alongside noise estimates at each timestep. A key challenge in training our model is the limited availability of multiview or 3D datasets. To address this, we develop a comprehensive training strategy that combines real 3D data with synthetic

Algorithm 1 Pseudo-code for fusing the noise estimates, each conditioned on a specific reference view. We denote $\mathcal{E}$ as the noise estimates, each conditioned on a specific reference view, $\mathbf{C}^f, \mathbf{C}^b, \mathbf{C}^s$ as the front-facing, back-facing, and shadow confidence masks, d_τ as the view distances of the reference views to the target view, τ, R as the number of reference images used to inpaint the image, $\vee, \wedge, \neg$ as logical OR, AND, and negation, $\odot, \oslash$ as Hadamard product and division, and $\mathrm{OneHot}(i, N) : \mathbb{N}^{\cdots} \to \{0,1\}^{N \times \cdots}$ as a function that encodes an index, i, into an N-length one-hot vector, respectively. For the shadow confidence mask, "one" means that although the ray intersects the shadow volume, and the content is *uncertain*, the model has decided that there is no occluded content, and the shadow background is valid; see Sec. 3.4.6.

procedure $\Gamma(\mathcal{E} \in \mathbb{R}^{R \times C \times H \times W}, \mathbf{C}^f, \mathbf{C}^b, \mathbf{C}^s \in \{0,1\}^{R \times H \times W}, d_\tau \in \mathbb{R}^R)$
  $\widehat{\mathbf{C}}^f = \bigvee_r \mathbf{C}^f[r]$ ▷ At least one front-face exists. $\in \{0,1\}^{H \times W}$
  $\tilde{\mathbf{C}}^b = \mathbf{C}^b \wedge \neg \widehat{\mathbf{C}}^f$ ▷ Back-faces but not front-faces. $\in \{0,1\}^{R \times H \times W}$
  $\widehat{\mathbf{C}}^b = \bigvee_r \tilde{\mathbf{C}}^b[r]$ ▷ At least one back-face exists. $\in \{0,1\}^{H \times W}$
  $\tilde{\mathbf{C}}^s = \mathbf{C}^s \wedge \neg(\widehat{\mathbf{C}}^f \vee \widehat{\mathbf{C}}^b)$ ▷ Shadow but no front- or back-faces. $\in \{0,1\}^{R \times H \times W}$
  $\widehat{\mathbf{C}}^s = \bigvee_r \tilde{\mathbf{C}}^s[r]$ ▷ At least one shadow exists. $\in \{0,1\}^{H \times W}$
  $\mathbf{C}^\varnothing = \neg(\widehat{\mathbf{C}}^f \vee \widehat{\mathbf{C}}^b \vee \widehat{\mathbf{C}}^s)$ ▷ No confidence. $\in \{0,1\}^{H \times W}$
  $\mathbf{C} = \mathrm{Concat}(\mathbf{C}^f, \tilde{\mathbf{C}}^b, \tilde{\mathbf{C}}^s, \mathbf{C}^\varnothing)$ ▷ Confidence hierarchy. $\in \{0,1\}^{4 \times R \times H \times W}$
  $\mathbf{W} = \mathbf{C} \oslash d_\tau$ ▷ Weight map (prefer closer views). $\in \mathbb{R}^{4 \times R \times H \times W}$
  $\mathbf{F} = \sum_{i=1}^{4} (\arg\max_r \mathbf{W}[i, r]) \odot (\bigvee_r \mathbf{C}[i, r])$ ▷ Selected reference indices. $\in \mathbb{N}^{H \times W}$
  $\widehat{\mathbf{F}} = \mathrm{OneHot}(\mathbf{F}, R)$ ▷ Fused mask. $\in \{0,1\}^{R \times H \times W}$
  $\varepsilon = \sum_r \mathcal{E}[r] \odot \widehat{\mathbf{F}}[r]$ ▷ Fused noise estimate. $\in \mathbb{R}^{C \times H \times W}$
  return ε
end procedure

multiview data from monocular real images, enabling us to leverage abundant single-view data while maintaining our multiview conditioning framework. To that end, we use a mixture of two datasets: (i) MS COCO [36], which is a monocular image dataset, and (ii) Google Scanned Objects [16], which is a 3D mesh dataset.

Our training approach consists of three main components: first, we describe how we synthesize multiview training pairs from monocular images to leverage abundant single-

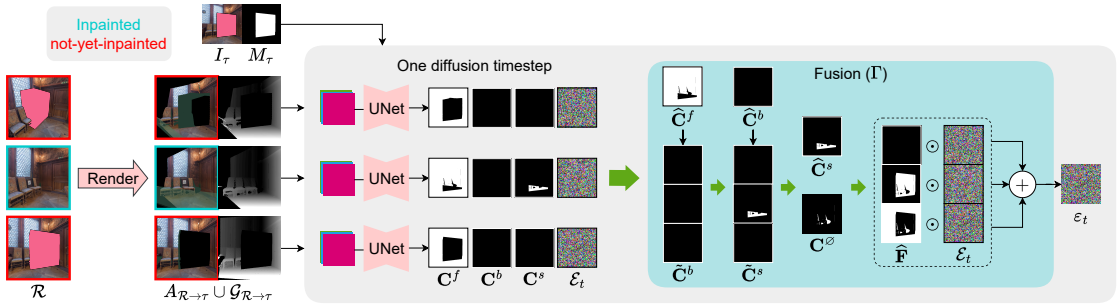

Figure 3.4: Illustration of the fusion of multiple reference views using the confidence masks. We denote I_τ as the *incomplete* target image, M_τ as the inpainting mask, $\mathcal{R}$ as the set of reference images, $A_{\mathcal{R}\to\tau}, \mathcal{G}_{\mathcal{R}\to\tau}$ as reference-based appearance and geometric cues, Γ as the fusion operator, $\mathbf{C}^f, \mathbf{C}^b, \mathbf{C}^s$ as the front-face, back-face, and shadow confidence masks, $\mathcal{E}_t$ as the noise estimates at diffusion timestep t, $\widehat{\mathbf{F}}$ as the fused mask, and ε_t as the fused noise estimate at diffusion timestep t, respectively. $\widehat{\mathbf{C}}^f, \widehat{\mathbf{C}}^b, \widehat{\mathbf{C}}^s, \tilde{\mathbf{C}}^b, \tilde{\mathbf{C}}^s, \mathbf{C}^\varnothing$ represent intermediate variables of the fusion process; see Algorithm 1 for details. As shown, given a target image and a set of reference images, we first render the reference-based appearance and geometric cues, and then at each diffusion step, we fuse the noise estimates conditioned on different reference views using the predicted confidence masks.

view datasets; second, we detail our sampling strategy for direct utilization of 3D mesh datasets; and third, we present our mask generation strategies for creating realistic inpainting scenarios. To ensure robustness to geometry estimation errors encountered during inference, we then explain how we simulate geometric perturbations during training and supervise the predicted confidence masks using *un*perturbed ground-truth geometry. Finally, we present our complete loss function that jointly optimizes noise estimation and confidence prediction.

Single-View Data Synthesis

To ensure greater training data diversity beyond 3D datasets, we synthesize geometric and appearance cues from single-view images via monocular depth estimation. This approach

allows us to leverage large-scale single-view image datasets for training our geometry-aware inpainter. Specifically, given an image, $I_\tau \in \mathbb{R}^{3\times H\times W}$, we first compute a monocular metric depth estimate, $D_\tau \in \mathbb{R}^{H\times W}$. We assume the focal length to be $f = \frac{W+H}{2}$, and the principal point to be in the center of the image, $\mathbf{p} = \frac{1}{2}(W, H)$. We then create a triangle mesh, $\mathcal{M}_\tau$, in the image's coordinate frame (i.e., identity extrinsics), following the mesh construction procedure detailed in Sec. 3.4.2. We assume there is a second camera (i.e., a synthetic reference view) from which the geometric and photometric cues are actually coming. To generate a random reference pose, we sample angles, $\mathbf{a}_r \sim \mathcal{N}\left(0, \sigma_{a_r}^2 \mathbb{I}_3\right)$, and translation, $\mathbf{t}_r \sim \mathcal{N}\left(0, \left(\sigma_{t_r} \cdot \min D_\tau\right)^2 \mathbb{I}_3\right)$, where σ_{a_r} and σ_{t_r} are user-defined hyperparameters (see Tab. A.1). We then form the rotation matrix, $\mathbf{R}_r$, from the sampled Euler angles, $\mathbf{a}_r$, and render the mesh as

$$I_r, D_r = \mathrm{Render}(\mathcal{M}_\tau, \mathbf{R}_r, \mathbf{t}_r, \mathbf{K}), \tag{3.21}$$

to obtain the synthetic reference image, I_r, and its corresponding depth map, D_r. This process will automatically occlude parts of the target view. We do not use hint images when the training sample is from a single-image dataset.

3D Data Sampling

For a mesh-based 3D dataset, given a mesh, $\mathcal{M}$, we uniformly sample the reference and target cameras (in a sphere centered at the object's centroid), directly rendering the reference and target views, I_r, I_τ, respectively, along with the reference depth map, D_r. With a probability of $p_h = 0.95$, we also uniformly sample another camera to render a hint image for training.

Mask Generation

We consider two mask generation strategies: (i) the 2D image-based approach from LaMa [72], which generates large and diverse masks on the target image, and (ii) a 3D-based approach, designed to obtain a 3D consistent mask across a multiview image set. We focus on (ii) for the remainder of this section. This strategy is straightforward, given known 3D geometry: we sample a 3D convex polyhedron, place it in the scene, and obtain an inpainting mask by rendering this occluder volume to the target view. Specifically, let $B \in \mathbb{R}^{2\times 3}, C \in \mathbb{R}^3$ be the bounding box around the scene point cloud and the scene point cloud's centroid, respectively. We first uniformly sample a bounding box, B_o, for the occluder inside the scene's bounding box. We restrict the size of the bounding box to be in the range

$$\left[o_{\text{min}}(B_2 - B_1), o_{\text{max}}(B_2 - B_1)\right], \tag{3.22}$$

where o_{min} and o_{max} are user-defined hyperparameters (see Tab. A.1). The approach differs between datasets due to their distinct characteristics. For Google Scanned Object [16], which contains well-bounded 3D objects, we can directly use the scene's bounding box for occluder placement. In contrast, MS COCO [36] includes outdoor scenes where the scene's bounding box does not properly represent meaningful boundaries. For such cases, we first uniformly sample a point from the scene's point cloud as the occluder's centroid, C_o, and restrict the sampled occluder bounding box to the camera frustum instead of the scene's bounding box. This results in a different size range:

$$\left[o_{\text{min}}\frac{\left(C_o^\top \hat{\mathbf{k}}\right)(B_2 - B_1)}{B_2 - B_1}, o_{\text{max}}\frac{\left(C_o^\top \hat{\mathbf{k}}\right)(B_2 - B_1)}{B_2 - B_1}\right], \tag{3.23}$$

where $\hat{\mathbf{k}}$ is the unit vector in the direction of the z-axis. The specific parameter values are adjusted for each dataset type (see Tab. A.1). We then uniformly sample N_o points within B_o, and fit a convex hull around the sampled points, resulting in the occluder volume. We render the sampled convex hull to the target view, yielding the inpainting mask. With a probability of 0.2, we sample a 3D occluder volume; otherwise, we use LaMa's mask generator.

Simulating Geometric Errors for Domain Adaptation

Our data generation techniques, whether on 3D scenes or single images, are slightly out-of-distribution compared to real-world multiview datasets (on which our method is evaluated). In particular, the geometric errors (whether in scene or camera parameters) from DUSt3R are not naturally present. We therefore consider how to include such errors synthetically.

Given a reference image, I_r, and its corresponding depth map, D_r, we create a triangle mesh, $\mathcal{M}_r$, along with its shadow mesh, $\mathcal{M}_r^S$, (as described in Sec. 3.4.2 and Sec. 3.4.3, respectively). To simulate geometry estimation errors, we also create a perturbed version of the reference mesh, $\widetilde{\mathcal{M}}_r$, and shadow mesh, $\widetilde{\mathcal{M}}_r^S$, by sampling perturbation angles, $\mathbf{a}_p \sim \mathcal{N}\left(0, \sigma_{a_p}^2 \mathbb{I}_3\right)$, and translation, $\mathbf{t}_p \sim \mathcal{N}\left(0, \left(\sigma_{t_p} \cdot \min D_r\right)^2 \mathbb{I}_3\right)$, forming the rotation matrix, $\mathbf{R}_p$, and perturbing the mesh vertices, $\mathbf{V}_r \in \mathbb{R}^{V \times 3}$, as

$$\widetilde{\mathbf{V}}_r = \mathbf{V}_r \mathbf{R}_p^\top + \mathbf{t}_p, \tag{3.24}$$

where σ_{a_p} and σ_{t_p} are user-defined hyperparameters (see Tab. A.1). We then render the

perturbed mesh and shadow as

$$\widetilde{I}_{r\to\tau}, \widetilde{F}_{r\to\tau}, \widetilde{B}_{r\to\tau}, \widetilde{D}_{r\to\tau} = \text{Render}(\widetilde{\mathcal{M}}_r, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}), \tag{3.25}$$

$$\widetilde{S}_{r\to\tau} = \text{Render}(\widetilde{\mathcal{M}}_r^S, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}), \tag{3.26}$$

to obtain the rendered cues, $\widetilde{I}_{r\to\tau}, \widetilde{\mathcal{G}}_{r\to\tau} = \left\{\widetilde{F}_{r\to\tau}, \widetilde{B}_{r\to\tau}, \widehat{D}_{r\to\tau}, \widetilde{S}_{r\to\tau}\right\}$.

Supervising Predicted Confidence Masks

We use the unperturbed meshes, $\mathcal{M}_r$, and $\mathcal{M}_r^S$, to compute the ground-truth confidence masks. Specifically, we render $\mathcal{M}_r$ and $\mathcal{M}_r^S$ as

$$F_{r\to\tau}, B_{r\to\tau} = \text{Render}(\mathcal{M}_r, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}), \tag{3.27}$$

$$S_{r\to\tau} = \text{Render}(\mathcal{M}_r^S, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}), \tag{3.28}$$

to obtain the unperturbed front-face, back-face, and shadow masks. Given the sampled inpainting mask, M_τ, the ground-truth front-face confidence mask is computed as

$$\mathbf{C}^f = (F_{r\to\tau} \wedge \neg S_{r\to\tau}) \vee \neg M_\tau, \tag{3.29}$$

where $\wedge, \vee, \neg$ denote the logical AND, OR, and negation, respectively. This mask highlights the regions that are either outside the inpainting mask, or exclusively guided by front-facing surfaces (excluding the parts intersecting the shadow mask). The ground-

truth back-face confidence mask is computed as

$$\mathbf{C}^b = B_{r\to\tau} \wedge M_\tau. \tag{3.30}$$

This mask highlights the regions inside the inpainting mask that are guided by back-facing surfaces. To compute the ground-truth shadow confidence mask, first note that this mask indicates the model's certainty in trusting the photometric information. To obtain such information, let $\mathcal{F}, \mathcal{F}_r \in \mathbb{N}^{H\times W}$ be the map of triangle face indices rendered to the target view, from meshes $\mathcal{M}$ and $\mathcal{M}_r$, respectively. We can trust the photometric information of a pixel, if and only if $\mathcal{F}$ and $\mathcal{F}_r$ are equal in that pixel and the pixel has valid photometric information (i.e., not disoccluded), meaning both reference and target see the same triangle face at that pixel. Therefore, the ground-truth shadow confidence mask is computed as

$$\mathbf{C}^s = S_{r\to\tau} \wedge \mathbb{1}\left\{\mathcal{F}[\mathbf{x}] = \mathcal{F}_r[\mathbf{x}]\right\}_{\mathbf{x}} \wedge F_{r\to\tau} \wedge M_\tau, \tag{3.31}$$

where $\mathbb{1}\{\cdot\}$ denotes the indicator function.

Complete Loss Function

After obtaining all input and corresponding ground-truth outputs, we sample a random noise, $\varepsilon \sim \mathcal{N}(0, \mathbb{I})$, and timestep, $t \sim \mathcal{U}(1, T)$, training the model via

$$\mathbf{z}_t = \mathrm{AddNoise}(\mathcal{E}(I), \varepsilon, t) \tag{3.32}$$

$$\left(\widetilde{\varepsilon}, \widetilde{\mathbf{C}}^f, \widetilde{\mathbf{C}}^b, \widetilde{\mathbf{C}}^s\right) = \boldsymbol{\epsilon}_\theta \left(\mathbf{z}_t, t \mid \mathbf{y}, I \odot \neg M, M, \widetilde{A}_{r\to\tau}, \widetilde{\mathcal{G}}_{r\to\tau}\right), \tag{3.33}$$

$$\mathcal{L}(\theta) = \|(\varepsilon - \widetilde{\varepsilon}) \odot (M + \lambda(1 - M))\|_2^2 + \eta \sum_{\rho\in\{f,b,s\}} \left\|\mathbf{C}^\rho - \widetilde{\mathbf{C}}^\rho\right\|_2^2, \tag{3.34}$$

where $\mathcal{E}$ is the VAE encoder of Stable Diffusion, $\mathrm{AddNoise}(\mathbf{z}, \varepsilon, t)$ is the forward diffusion step, yielding the latent noisy diffusion intermediate, $\mathcal{L}$ is the training loss, λ and η are user-defined hyperparameters (see Tab. A.1), controlling the weights of the pixels outside the inpainting mask and confidence masks, respectively, relative to the pixels inside the inpainting mask, $\widetilde{A}_{r\to\tau}, \widetilde{\mathcal{G}}_{r\to\tau}$ denote the appearance and geometric cues including the perturbed renders, and $\mathbf{y}$ denotes the text prompt, respectively.

3.5 Autoregressive Scene Inpainting

With our geometry estimator (Sec. 3.3) and geometry-aware inpainter (Sec. 3.4), we can now iteratively inpaint the entire scene. To begin, we initialize the scene geometry by running DUSt3R on the *in*complete input views, as

$$\mathcal{G} \leftarrow \mathrm{DUSt3R}(\{(I_i, M_i, 0)\}_{i=1}^N, G, \mathcal{G}_{\text{pre-set}}), \tag{3.35}$$

where $\mathcal{G}_{\text{pre-set}}$ is the set of ground-truth geometry parameters that we wish to pre-set. For small-baseline scenes and the few-view inpainting task, we compute DUSt3R on a complete symmetric scene graph, $G = (V, E)$, with

$$E = \{(i, j) \in V \times V; i \neq j\}, \tag{3.36}$$

connecting each view to all other views. However, for large-baseline scenes, we restrict the connections of each view to their k closest views, as

$$E = \{(i, j) \in V \times V; i \neq j, j \in \underset{j}{\text{TopK}}(V, -d(i, j))\}, \tag{3.37}$$

where $\text{TopK}_i(S, f(i))$ denotes the top k elements of the set S according to function f, and $d(i, j)$ is the view distance function defined as Eq. (3.9).

We also initialize the "autoregressive set", which holds a tuple for each view containing the image and an inpainted-status indicator, as

$$\left\{\left(\widehat{I}_i, b_i\right)\right\}_{i=1}^{N} \leftarrow \{(I_i \odot \neg M_i, 0)\}_{i=1}^{N}, \tag{3.38}$$

which will be autoregressively updated. A random view, i_1, is then selected to start the inpainting. Algorithm 2 provides a high-level overview of our complete autoregressive scene inpainting procedure. Each iteration consists of three steps: (i) inpainting a subset of not-yet-inpainted images, $\mathcal{T}$, conditioned on a set of reference images, $\mathcal{R}$, and updating the autoregressive set, as explained in Sec. 3.5.1, (ii) updating the scene geometry with the inpainted images, as explained in Sec. 3.5.2, and (iii) selecting a subset of images

Algorithm 2 A high-level pseudo-code for our autoregressive scene inpainting procedure. We denote $\neg$ as logical negation, $\odot$ as Hadamard product, $\setminus$ as set subtraction, $|$ as conditioning indicator, and $(\mathbf{R}_k, \mathbf{t}_k) \in \mathrm{SE}(3), \mathbf{K}_k \in \mathbb{R}^{3\times3}$ as the camera parameters of the kth view of the estimated geometry, $\mathcal{G}$.

procedure INPAINTAUTOREGRESSIVELY($\{(I_i, M_i)\}_{i=1}^N, G, \mathcal{G}_{\text{pre-set}}$)
  $\left\{\left(\widehat{I}_i, b_i\right)\right\}_{i=1}^N \leftarrow \{(I_i \odot \neg M_i, 0)\}_{i=1}^N$ ▷ Initializing the autoregressive set and inpainted indicators
  $\mathcal{G} \leftarrow$ DUST3R($\{(I_i, M_i, b_i)\}_{i=1}^N, G, \mathcal{G}_{\text{pre-set}}$) ▷ Initializing the geometry
  $i_1 \leftarrow$ RANDINT$(1, N)$ ▷ Randomly select the first view
  $\mathbf{D} \leftarrow \left[d(\mathbf{R}_i, \mathbf{R}_j)\right]_{i,j=1}^N$ ▷ Form the view distance matrix
  $K \leftarrow$ SELECTKEYVIEWS$(\mathbf{D}, i_1)$ ▷ Key-view subset (Algorithm 3)
  for τ in K **do**
    $\mathcal{R} \leftarrow \{1, \ldots, N\} \setminus \{\tau\}$ ▷ All other views are references
    $A_{\mathcal{R}\to\tau}, \mathcal{G}_{\mathcal{R}\to\tau} \leftarrow$ RENDERCUES$(\left\{\left(\widehat{I}_i, M_i, b_i\right)\right\}_{i=1}^N, \mathcal{G}, \mathcal{R}, \tau)$
    $\widehat{I}_\tau \leftarrow$ GEOAWAREINPAINT$(I_\tau, M_\tau, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}_\tau \mid A_{\mathcal{R}\to\tau}, \mathcal{G}_{\mathcal{R}\to\tau})$
    $b_\tau \leftarrow 1$ ▷ Mark the view as inpainted
    $\mathcal{G} \leftarrow$ UPDATEDUST3R $(\mathcal{G}, \left\{\left(\widehat{I}_i, M_i, b_i\right)\right\}_{i=1}^N, G, \{\tau\})$
  end for
  $\mathcal{T} \leftarrow \{1, \ldots, N\} \setminus K$ ▷ All remaining views are targets
  $\{(A_{K\to\tau}, \mathcal{G}_{K\to\tau})\}_{\tau\in\mathcal{T}} \leftarrow \left\{\text{RENDERCUES}(\left\{\left(\widehat{I}_i, M_i, b_i\right)\right\}_{i=1}^N, \mathcal{G}, K, \tau)\right\}_{\tau\in\mathcal{T}}$
  $\left\{\widehat{I}_\tau\right\}_{\tau\in\mathcal{T}} \leftarrow \{\text{GEOAWAREINPAINT}(I_\tau, M_\tau, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}_\tau \mid A_{K\to\tau}, \mathcal{G}_{K\to\tau})\}_{\tau\in\mathcal{T}}$
  return $\left\{\widehat{I}_i\right\}_{i=1}^N$
end procedure

to be inpainted at the next autoregressive iteration, as explained in Sec. 3.5.3. A high-level illustration of these steps and a step-by-step example are provided in Fig. 3.1 (b) and Fig. 3.5, respectively.

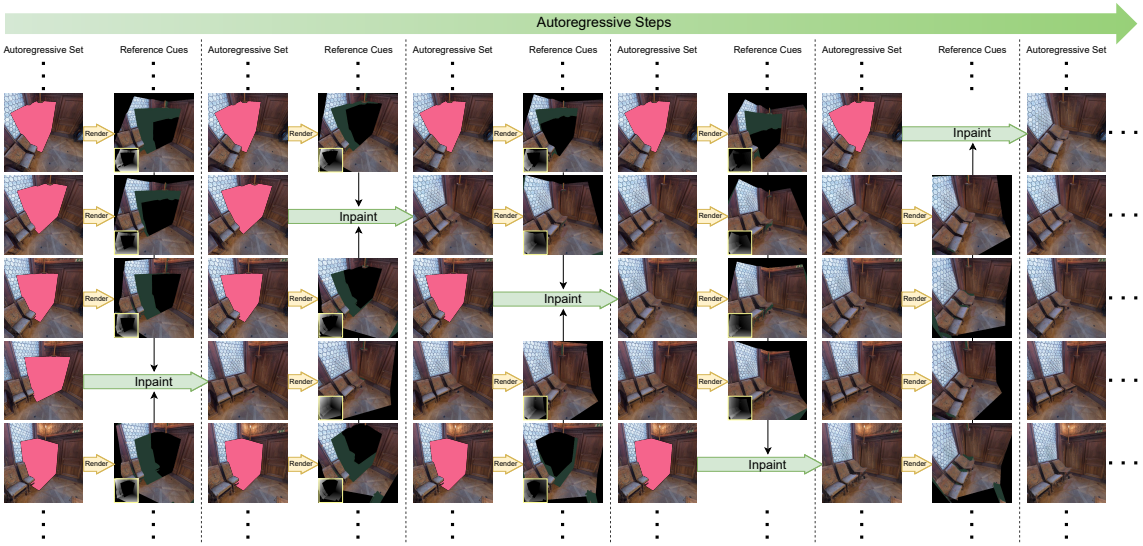

Figure 3.5: A step-by-step illustration of autoregressive inpainting in the first stage, where a key-view subset of the scene is progressively inpainted. Note the consistency preserved throughout the process.

3.5.1 Geometry-Aware Inpainting

For each target view, $\tau \in \mathcal{T}$, we select a subset of reference views, $\mathcal{R}_\tau$, from the autoregressive set, prioritizing those already inpainted (see Sec. 3.5.3). Following Sec. 3.4, we render appearance and geometric cues from all reference views, run parallel diffusion processes, and fuse noise estimates at each step via the predicted confidences. In short, we denote this process as

$$\widehat{I}_\tau \leftarrow \text{GeoAwareInpaint}(I_\tau, M_\tau, \mathbf{R}_\tau, \mathbf{t}_\tau, \mathbf{K}_\tau \mid A_{\mathcal{R}_\tau \to \tau}, \mathcal{G}_{\mathcal{R}_\tau \to \tau}). \tag{3.39}$$

3.5.2 Scene Geometry Update

In this step, we update the geometry of the target views using the inpainted images. In the first autoregressive step, before any image is inpainted, we run DUSt3R on the entire scene graph, yielding $\mathcal{G}$. In each remaining step, where the views $\mathcal{T} \subset V$ are being inpainted, we only update their corresponding vertices, $\mathcal{T}$, and edges, $E_{\mathcal{T}} = \{(i,j) \in E; i \in \mathcal{T}\}$, of the scene graph, initialized by the current state of $\mathcal{G}$, while freezing other parameters. We denote this update as

$$\mathcal{G} \leftarrow \mathrm{UpdateDUSt3R}(\mathcal{G}, \left\{\left(\widehat{I}_i, M_i, b_i\right)\right\}_{i=1}^{N}, G, \mathcal{T}). \tag{3.40}$$

This greatly reduces the run-time of the geometry update, which is significant, since it is performed after every iteration of autoregressive inpainting.

3.5.3 Selecting the Next Images to Inpaint

We employ a two-stage strategy for the autoregressive process. First, we inpaint a key-view subset of the scene, one by one, to *generate* the missing content of the scene. Then, we *propagate* the generated content to the remaining views simultaneously. At the beginning of the first stage, we use a greedy min-max approach to select the key-view subset. Let N be the number of views in the scene. We first form the view distance matrix

$$\mathbf{D} = \left[d(i,j)\right]_{i,j=1}^{N}, \tag{3.41}$$

Algorithm 3 Pseudo-code for selecting a key-view subset of the scene. $\mathbf{D}, i_1, \setminus$ denote the view distance matrix, the initial image to be inpainted, and set subtraction, respectively.

procedure SELECTKEYVIEWS($\mathbf{D} \in \mathbb{R}^{N \times N}, i_1 \in \mathbb{N}$)
  $K = [i_1]$ ▷ Initialize the key-view set
  for n in $[1, \ldots, N-1]$ **do**
    $i_n = \arg\max_{i \notin K} \min_{j \in K} \mathbf{D}[i, j]$ ▷ Next key-view using min-max
    $K = \text{Concat}(K, [i_n])$ ▷ Extend the key-view set
  end for

  $\widehat{K} = [i_1]$ ▷ Initialize the sorted key-view set
  for n in $[1, \ldots, N-1]$ **do**
    $i_n = \arg\min_{i \in K \setminus \widehat{K}} \frac{1}{|\widehat{K}|} \sum_{j \in \widehat{K}} \mathbf{D}[i, j]$ ▷ Minimize the mean distance to the previous sorted views
    $\widehat{K} = \text{Concat}(\widehat{K}, [i_n])$ ▷ Extend the sorted key-view set
  end for
  return $\widehat{K}$
end procedure

where $d(i, j)$ is the view distance function defined in Eq. (3.9). Beginning with the randomly selected view to start the inpainting, i_1, we iteratively add the view that maximizes the minimum distance to the existing subset. Once selected, we order them to minimize the mean distance to the views in the previous steps. This procedure is summarized in Algorithm 3. The key-view stage follows the sorted order, with each target view conditioned on the entire autoregressive set, inpainted or not. In the propagation step, the remaining views are conditioned only on the inpainted key-view images.

3.6 Summary

This chapter presented a geometry-aware approach for multiview-consistent 3D scene inpainting. Our method combines three key components: (1) a scene geometry esti-

mator adapted from DUSt3R that handles incomplete views, (2) a geometry-aware diffusion model that leverages projected appearance and geometric cues with hierarchical confidence-based fusion, and (3) an autoregressive framework that iteratively inpaints the key-view subset before propagating content to remaining views.

Chapter 4

Empirical Evaluation

This chapter presents a comprehensive empirical evaluation of our geometry-aware 3D scene inpainting method. We evaluate our approach across multiple challenging scenarios to demonstrate its effectiveness in achieving multiview-consistent inpainting results while maintaining high visual quality.

We begin in Sec. 4.1 by detailing our implementation specifics, including training configurations, hardware requirements, and hyperparameter settings that enable our method to achieve robust performance across diverse scene types. Sec. 4.2 establishes our evaluation framework, covering the datasets used for assessment, baseline methods for comparison, and metrics employed to measure both visual quality and multiview consistency. The comprehensive evaluation spans three distinct tasks: narrow-baseline object removal, wide-baseline scene completion, and few-view inpainting, each presenting unique challenges that test different aspects of our approach.

In Sec. 4.3, we present our main experimental findings, demonstrating superior perfor-

mance across all evaluation tasks through both quantitative metrics and qualitative comparisons. Finally, Sec. 4.4 provides detailed ablation studies that analyze the key components driving our method's success, validating our architectural decisions and design choices.

Our evaluation demonstrates that our geometry-aware approach successfully addresses the fundamental challenges of 3D scene inpainting while maintaining computational efficiency and practical applicability across diverse real-world scenarios.

4.1 Implementation Details

We use PyTorch3D [57] for mesh rendering. As mentioned in Sec. 3.4.6, we initialize our reference-guided inpainter as a Stable Diffusion v2, fine-tuned for inpainting [60, 70]. For training, we use a mixture of two datasets, MS COCO [36] and Google Scanned Objects [16], where we upsample Google Scanned Objects (GSO) so that the ratio of GSO to MS COCO samples in each epoch is $1 : 10$. Since GSO lacks text captions, we use Kosmos-2 [51] to caption a front-facing view of each object. For MS COCO, we use the existing dataset captions as text prompts during training and synthesize the reference-based geometric cues using DepthAnything V2 [89]. Training and evaluation are performed on NVIDIA L40 GPUs (16 for training, one for inference). To make the model robust towards color and texture discrepancies across multiple views, we randomly augment the texture of the reference mesh using color jitter (brightness, contrast, saturation, and hue). To preserve classifier-free guidance [22] capabilities, we randomly drop the text prompt with a probability of 0.1. If the generated mask for an image is a 3D occluder volume, we randomly drop the *reference* content occluded by the occluder volume with a probability

of 0.2. This will enable conditioning the model on not-yet-inpainted reference views. Notice that our model must be capable of both transferring reference information and also inpainting when no reference information is present. We use AdamW [40] with a constant learning rate schedule. During inference, we adopt the DDIM sampler [68] with 50 denoising steps. For all hyperparameters including data augmentation factors, training configuration, and loss weights, please refer to Tab. A.1.

4.2 Evaluation Protocol

To comprehensively assess our geometry-aware 3D scene inpainting method, we design an evaluation protocol that spans multiple challenging scenarios and datasets. Our evaluation encompasses three distinct tasks that test different aspects of our approach: narrow-baseline object removal on front-facing scenes, wide-baseline scene completion on complex environments, and few-view inpainting with limited viewpoint coverage.

For each task, we follow established evaluation protocols and compare against state-of-the-art methods specifically designed for 3D scene inpainting and related tasks. Our evaluation metrics are carefully chosen to assess both visual quality and multiview consistency, addressing the core challenges of 3D scene inpainting. We employ established metrics for 3D inpainting tasks (both object removal and scene completion), extending the evaluation by introducing additional metrics to assess image quality and multiview consistency, with appropriate adaptations for the inpainting context.

The following subsections detail our datasets (Sec. 4.2.1), baseline comparisons (Sec. 4.2.2), and evaluation metrics (Sec. 4.2.3), establishing the foundation for our empirical analysis.

4.2.1 Datasets

To evaluate our method for narrow-baseline inpainting, we use the SPIn-NeRF [48] dataset, a widely used benchmark for object removal in front-facing real-world scenes. This dataset consists of 10 scenes, each with 60 images featuring the object to be removed and corresponding object masks. Each scene also includes 40 other views as ground truth for object removal, without the presence of the mentioned object.

In addition, we also use the scene-centric portion of the NeRFiller [85] dataset to further investigate the performance of our method on more complex scenes, particularly those with larger baselines. This dataset serves as a benchmark for the scene completion task. Specifically, we use the following scenes as the *scene-centric* portion of the NeRFiller dataset for the wide-baseline scene completion task: "backpack", "billiards", "drawing", "norway", and "office".

For the few-view inpainting task, we use both SPIn-NeRF and the scene-centric portion of NeRFiller, resulting in 15 scenes. For each scene, we uniformly sample eight two-view subsets and eight three-view subsets, yielding a total of 240 few-view sets.

4.2.2 Baselines

For object removal, we compare our method to state-of-the-art approaches on the SPIn-NeRF dataset, specifically: SPIn-NeRF [48], which inpaints images and depth maps to supervise the NeRF fitting process; RefFusion [50], which adapts an inpainting diffusion model to a reference image and uses SDS to inpaint a 3DGS [29] with the reference-adapted diffusion model; InFusion [39], which inpaints a 3DGS by inpainting the depth map of an inpainted reference image; MVIP-NeRF [8], which uses SDS to inpaint a NeRF

by inpainting the rendered images and normal maps via a diffusion prior, and MALD-NeRF [35], which inpaints a NeRF by performing masked adversarial training to customize a diffusion model for each scene. Additional results from other methods are provided in Appendix A.3.

For scene completion, we compare our method to Stable Diffusion [60], which is the naive baseline of independent 2D inpainting, and NeRFiller [85], which uses IDU [19] to alternate between editing input images and updating the NeRF supervised by the edited images.

Finally, for the few-view task, we compare our method to SPIn-NeRF [48] and NeRFiller [85]. For this task, we use NeRFiller with no changes. However, since SPIn-NeRF relies on COLMAP's [63] sparse depth as a training signal, we disable this supervision for the few-view task, as scenes from the NeRFiller dataset lack COLMAP information.

4.2.3 Metrics

For the SPIn-NeRF dataset, we use the same evaluation protocol as reported in the original paper [48]. Specifically, we compute LPIPS [96] (with VGG-16 [66]) and FID [21] between the inpainted images and the ground-truth images, cropped by the inpainting mask's bounding box. The bounding box's size is increased by 10% before cropping, uniformly in each direction. We additionally assess the sharpness of the inpainted images within the inpainting mask using the Laplacian variance [52]. Since our inpainting method does not explicitly enforce 3D consistency via a 3D radiance field, we evaluate the consistency based on epipolar geometry, using the TSED metric [91]. TSED evaluates the consistency of adjacent pairs in a view set [91]. On the SPIn-NeRF dataset, as the scenes have very

small baselines, all possible view pairs are considered adjacent. In addition, our feature correspondences are limited to those inside the inpainting masks. Unlike [91], which considers a minimum of 10 feature matches for consistency, we only consider a minimum of two feature matches, as the inpainting mask is significantly smaller than the whole frame.

For the NeRFiller dataset, we use the same evaluation metrics as NeRFiller [85]. Since these metrics are computed on NeRF renders, we fit a NeRF on our inpainted images directly. Specifically, we compute PSNR, SSIM, and LPIPS, by comparing the rendered training views of the fitted NeRF to the inpainted images (i.e., the dataset used to fit the NeRF), along with MUSIQ [28] and Correspondence Score (Corrs) [85] on a video rendered from the NeRF. In addition to the metrics used by NeRFiller, we also evaluate the sharpness of both inpainted images and the NeRF renders. Finally, we compute TSED to evaluate consistency across images. As scenes in the NeRFiller dataset have a wide baseline, we only consider the two closest views to a view as adjacent views.

Finally, for the few-view inpainting task, as there is no ground truth available, we only compute sharpness and MUSIQ to evaluate image quality. We also compute Corrs and TSED on all possible image pairs in the few-view set, to evaluate geometric and semantic consistency across views. All metrics are computed only inside the bounding box around the inpainting mask for the few-view inpainting task.

For detailed explanations of the specialized metrics used in our evaluation, we refer readers to Appendix A.2, which provides comprehensive descriptions of TSED (Appendix A.2.1), MUSIQ (Appendix A.2.2), and Corrs (Appendix A.2.3).

4.3 Results

In this section, we present comprehensive experimental results demonstrating the effectiveness of our geometry-aware 3D scene inpainting method across three challenging tasks. We begin with narrow-baseline object removal (Sec. 4.3.1), followed by wide-baseline scene completion (Sec. 4.3.2), and conclude with few-view inpainting (Sec. 4.3.3). For each task, we provide both quantitative metrics and qualitative comparisons against state-of-the-art baselines, highlighting our method's superior performance in achieving multiview-consistent, high-quality inpainting results.

4.3.1 Object Removal on Narrow-Baseline Scenes

Due to the narrow baseline, we perform single-reference inpainting for this task. As our method does not rely on fitting a NeRF on the training views, and the evaluations are performed on the test views, we follow SPIn-NeRF's [48] procedure to evaluate image inpainters, e.g., LaMa [72]. Specifically, we first fit a NeRF on the training views and render the test views, which will now contain the unwanted object. The rendered test views are then used as inputs to our inpainting method. To evaluate SPIn-NeRF, InFusion, and MVIP-NeRF, we run their official code to reproduce the results. Since RefFusion does not provide publicly available code, we report the numbers provided by the paper. For MALD-NeRF, we evaluate their publicly available inpainted images for the SPIn-NeRF dataset. As shown in Tab. 4.1 and Fig. 4.1, our method outperforms all baselines on the SPIn-NeRF benchmark.

We obtain comparable sharpness and LPIPS to MALD-NeRF, while significantly out-

Table 4.1: Quantitative evaluation of the object removal task on SPIn-NeRF dataset with directly comparable results using consistent evaluation protocols. We denote sharpness ($\times 10^{-5}$) as σ, the percentage of consistent image pairs (TSED) at $T_{\text{error}} = 2.0$px as $T_{2\text{px}}$, and average run time per scene as τ. Additional results from other methods are provided in Appendix A.3.

Method	LPIPS ↓	FID ↓	σ ↑	$T_{2\text{px}}$ ↑	τ ↓
SPIn-NeRF [48]	0.4864	160.42	13.74	61.04	1h 40m
RefFusion [50]	0.4283	-	-	-	-
InFusion [39]	0.6692	244.19	9.30	35.88	14m
MVIP-NeRF [8]	0.5268	215.60	11.96	58.33	17h 38m
MALD-NeRF [35]	0.3996	130.95	35.27	58.22	-
Ours	0.4028	108.36	34.50	67.35	12m

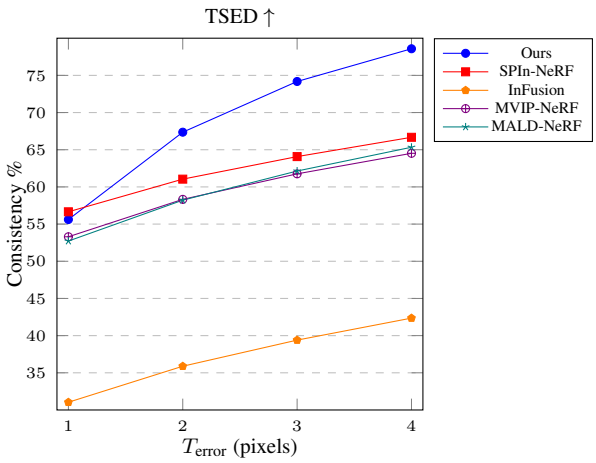

Figure 4.1: Evaluation of 3D consistency of the object removal task on the SPIn-NeRF dataset, using TSED, for various values of T_{error}.

performing it on other metrics (FID and TSED). Further, we demonstrate the ability to handle sparse-view inpainting (see Sec. 4.3.3), which cannot be easily handled by NeRF-based approaches. Please see Appendix A.4 for further analysis.

Furthermore, our method is also efficient, achieving faster scene inpainting compared to other methods. In Fig. 4.1, we observe that when $T_{\text{error}} = 1.0$px, almost all other methods achieve the same consistency as our method. Since all other methods are inherently

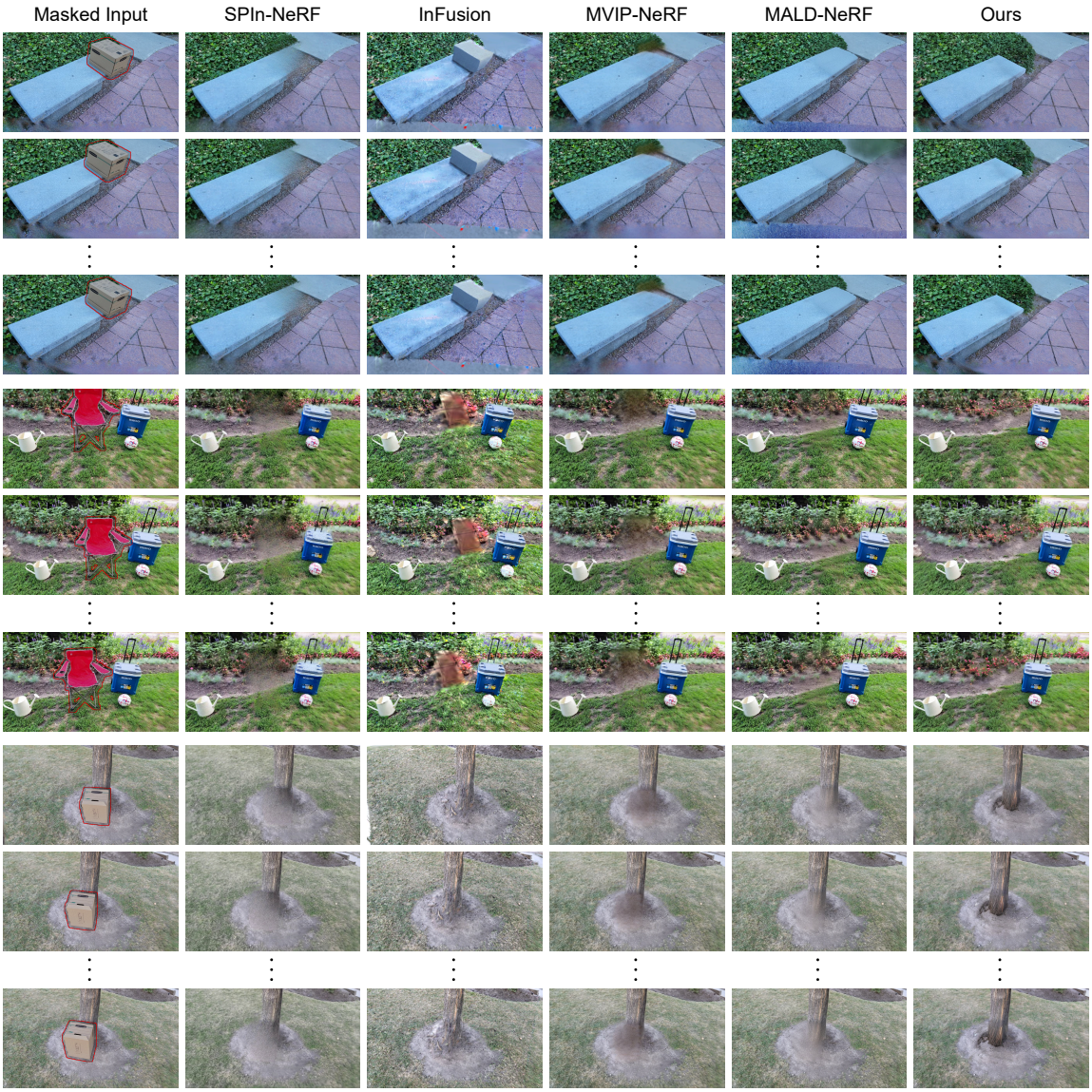

Figure 4.2: Qualitative object removal comparisons on the SPIn-NeRF dataset. Notice some other methods produce blurry regions (e.g., at the bench end) due to multiview inconsistencies, while ours preserves sharpness and visual plausibility. Please zoom in for details.

3D-consistent through 3D radiance fields as the core of their approach, this means our method also achieves a 3D-consistent inpainting. However, in contrast to our method,

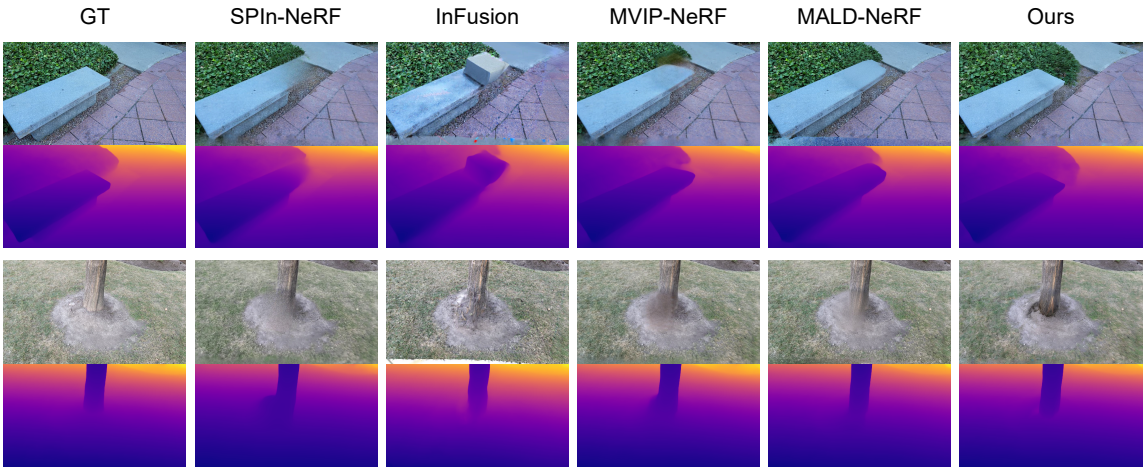

Figure 4.3: Visualized depth maps of various methods' inpaintings on the SPIn-NeRF dataset. The depth maps are obtained by running DUSt3R on the inpainted images. The corresponding inpainted images are also shown.

other methods do not improve (increase) as much as the error threshold increases. This is primarily because other methods enforce 3D consistency by fusing cross-view information through a 3D radiance field, resulting in blurry output renders. Since TSED computes SIFT features, naturally fewer such features will be detected from a blurry image, damaging the TSED score. In contrast, our method produces significantly sharper images (see Tab. 4.1 for details), resulting in more detected features and thus a higher TSED. Finally, we present a set of qualitative results in Fig. 4.2, demonstrating that our method produces sharper images than the baselines, indicating the efficacy of our cross-view fusion, which operates in a *learned* space, rather than pixel space. Furthermore, the depth map visualizations in Fig. 4.3 further highlight that our method produces inpainted images with realistic and coherent geometry, as evidenced by the plausible depth structure.

Table 4.2: Evaluation of scene completion on the NeRFiller dataset. We denote sharpness ($\times 10^{-5}$) as σ, the percentage of consistent image pairs (TSED) at $T_{\text{error}} = 2.0$px as $T_{2\text{px}}$, average run time per scene as τ, independent 2D inpainting as "2D", $\cdot^D$ for direct outputs of the inpainting model, and $\cdot^N$ for NeRF renders.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	MUSIQ ↑	Corrs ↑	σ^D ↑	σ^N ↑	$T^D_{2\text{px}}$ ↑	$T^N_{2\text{px}}$ ↑	τ ↓
Stable Diffusion (2D) [60]	24.69	0.85	0.10	3.77	1120	44.55	25.55	9.81	86.55	3m
NeRFiller w/o depth [85]	27.96	0.88	0.07	3.68	1146	1.20	3.18	14.63	93.51	1h 30m
NeRFiller [85]	27.68	0.87	0.08	3.69	1185	1.25	3.31	16.53	96.04	1h 30m
Ours	28.59	0.89	0.05	3.80	1250	38.96	26.45	67.80	98.25	55m

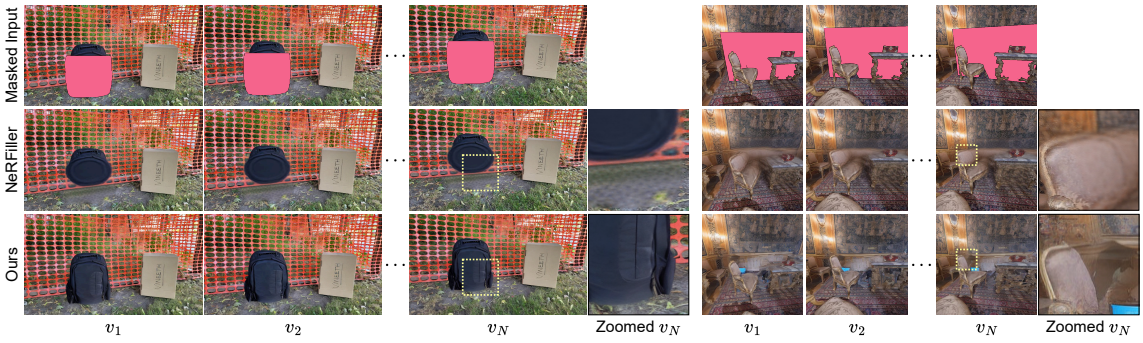

Figure 4.4: Qualitative scene completion comparisons on the NeRFiller dataset. NeRFiller can converge to blurry content, due to mixing divergent views, while ours generates and then propagates sharp content (e.g., see details in the backpack or window glass in the zoomed patches). Each view in a scene is denoted by v_i, where i is the view index.

4.3.2 Scene Completion on Wide-Baseline Scenes

The quantitative results for the scene completion task are presented in Tab. 4.2, demonstrating the superiority of our method on all metrics. We also show that our method is less time-consuming than NeRFiller. Moreover, although fitting a NeRF on our inpaintings reduces sharpness, they still remain significantly sharper than NeRFiller's. We qualitatively illustrate this in Fig. 4.4.

To evaluate 3D consistency, we report TSED on two sets of images, (i) the NeRF datasets (i.e., direct outputs of the inpainting models), and (ii) the NeRF renders. For

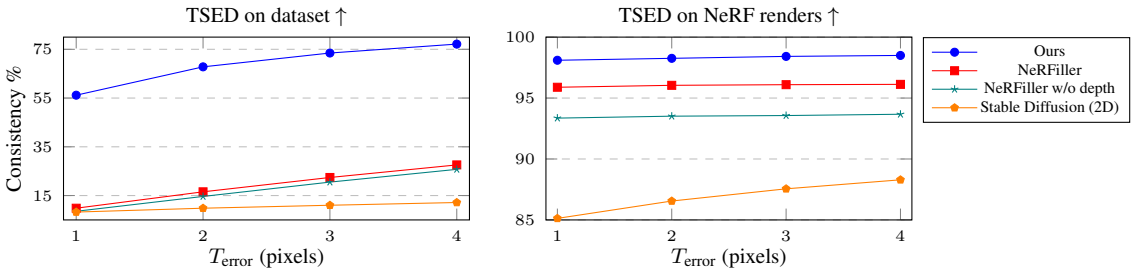

Figure 4.5: Evaluation of 3D consistency of the scene completion task on the NeRFiller dataset using TSED. The left inset shows the consistency of the "source images" (dataset) used to train the NeRF (i.e., the direct outputs of the generative inpainter used in each method). For NeRFiller, we use the dataset from the latest Dataset Update iteration. The right inset shows the consistency of the NeRF renders, from a NeRF fit to those source images.

NeRFiller, we use the dataset produced in the latest Dataset Update iteration. As shown in Tab. 4.2 and further visualized in Fig. 4.5, our dataset used to fit the NeRF is significantly more consistent than that of NeRFiller, resulting in a higher consistency score for the final NeRF renders. The left inset of Fig. 4.5 compares the source images (direct outputs of the generative inpainter) used for NeRF fitting, revealing a substantial gap in multiview consistency between NeRFiller and ours. This improvement stems from our geometry-aware inpainting model, which is specifically trained to propagate information across views in a multiview consistent manner. In contrast, NeRFiller employs a geometry-*un*aware inpainting model, which cannot directly leverage the knowledge encoded in the 3D scene to inform the inpainting. The right inset of Fig. 4.5 shows that fitting a NeRF on these datasets increases overall consistency, but our consistently inpainted images still yield more consistent NeRF renders. We also significantly outperform the naive 2D-only baseline (independent inpaintings; see Sec. 4.4.1 for details).

Table 4.3: Quantitative evaluation of the few-view task. We denote the presence of camera parameters as $(\mathbf{R}, \mathbf{t}, \mathbf{K})$ and depth maps as D, sharpness as σ, the percentage of consistent image pairs (TSED) at $T_{\text{error}} = 2.0$px as $T_{2\text{px}}$, and average run time per scene as τ.

Method	$(\mathbf{R}, \mathbf{t}, \mathbf{K})$	D	$\sigma \uparrow$	MUSIQ $\uparrow$	Corrs $\uparrow$	$T_{2\text{px}} \uparrow$	$\tau \downarrow$
SPIn-NeRF [48]	✓	✗	17.18	3.26	278	25.42	15m
NeRFiller [85]	✓	✗	5.05	3.41	187	18.54	13m
NeRFiller [85]	✓	✓	5.41	3.42	183	18.96	13m
Ours	✗	✗	48.2	3.84	400	52.92	12s

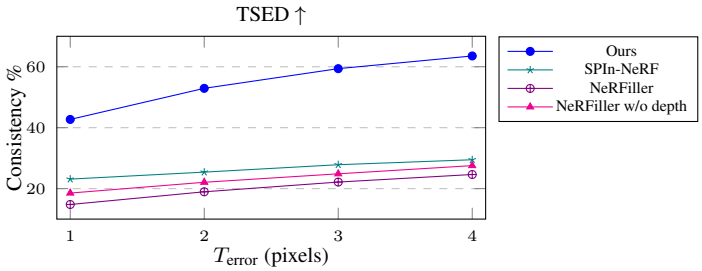

Figure 4.6: Evaluation of 3D consistency on the few-view inpainting task.

4.3.3 Few-View Inpainting

We use single-reference inpainting for the few-view inpainting task. As depicted in Tab. 4.3, we outperform all the baselines on the few-view inpainting task, even without the need to use the ground-truth camera parameters and depth maps, making it more self-contained. The TSED results demonstrate our method achieves a higher consistency score, mainly due to the other methods relying on fitting a NeRF, which is suboptimal for extremely sparse views. This is evident in Fig. 4.6, which illustrates the success of our method on inpainting scenes with very few views. We achieve a noticeable improvement over the baselines in terms of TSED consistency, as the difficulties encountered when fitting NeRFs on very few views result in both inconsistency and blurriness. The qualitative results shown in Fig. 4.7 further confirm the higher quality of our inpainted images, which are sharper and more visually plausible.

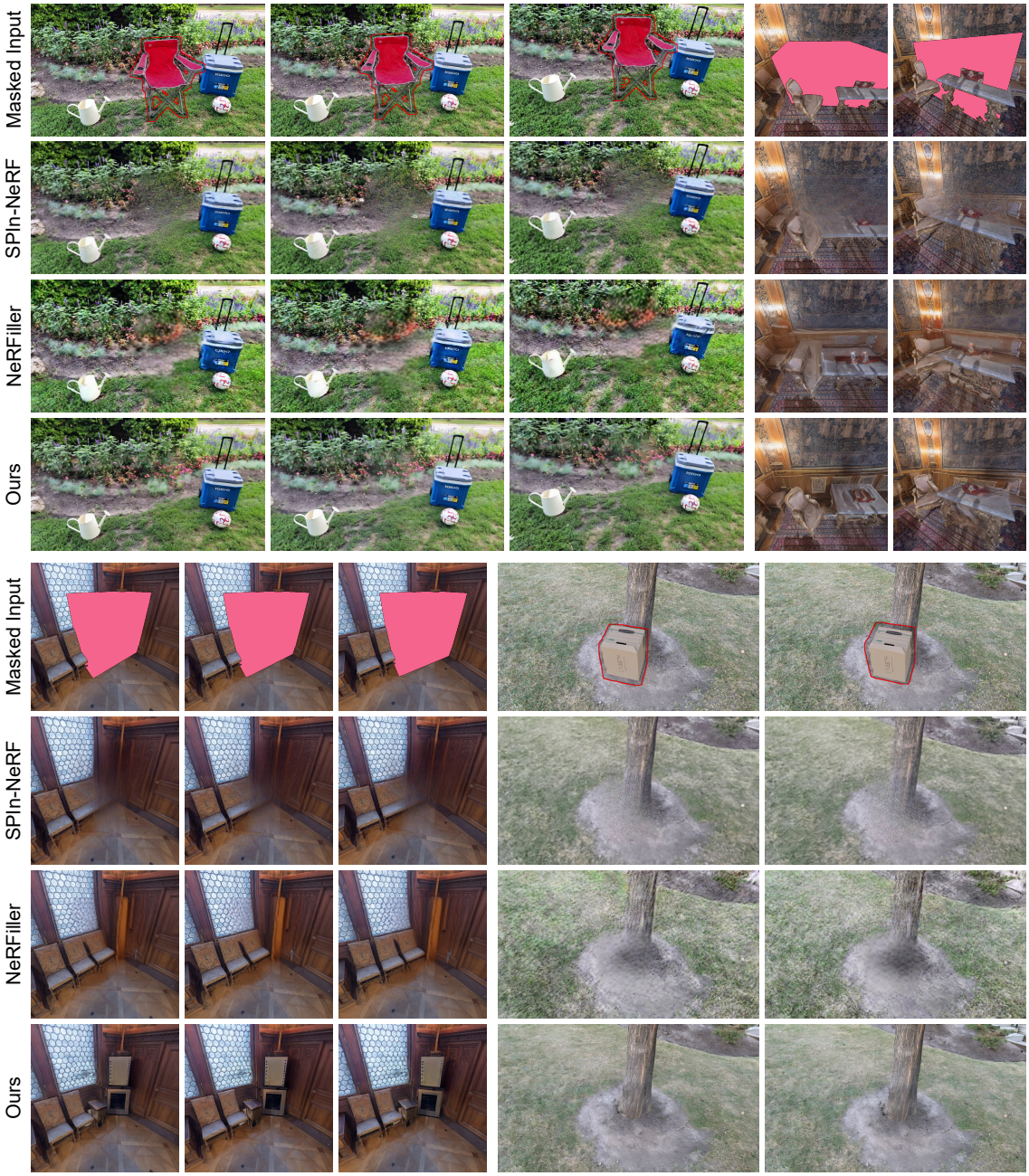

Figure 4.7: Qualitative comparisons for few-view inpainting. Our inpainted images are sharper and more visually plausible.

4.4 Ablation Studies

Having demonstrated the superior performance of our method across diverse tasks, we now analyze the key components that drive its success through comprehensive ablation studies. Our analysis spans three key dimensions: First, we establish the fundamental importance of geometry-aware cross-view fusion by comparing against independent 2D inpainting baselines (Sec. 4.4.1). Second, we examine the individual contributions of our inference-time strategies, including autoregressive inpainting and the role of ground-truth geometric information (Sec. 4.4.2). Finally, we validate our core architectural decisions by ablating various design choices in training datasets, conditioning mechanisms, and fusion strategies (Sec. 4.4.3). These studies collectively reveal the importance of our design choices and validate the effectiveness of our approach.

4.4.1 Comparison with Independent 2D Inpainting

As observed in prior work [85], independent inpainting fails to produce consistent content across views. Since *reference-based geometry-awareness* is one of the core components of our approach, we also present a comparison between our geometry-aware inpainting and the naive baseline of geometry-*un*aware inpainting (i.e., independent 2D inpainting), in Tab. 4.4. We compare our approach to three state-of-the-art diffusion-based 2D inpainting methods: Stable Diffusion [60] (which our model is based on), ControlNet [95], and BrushNet [27]. Note that our model, just as for the 2D inpainter baselines, is also a latent diffusion model with a similar overall architecture, operating on a single image at a time. In other words, we do not use an explicit or implicit 3D radiance field when inpainting

Table 4.4: Evaluating our method against independent inpainting on scene completion. "2D" indicates independent 2D inpainting.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	MUSIQ ↑	Corrs ↑
Stable Diffusion (2D) [60]	24.69	0.85	0.10	3.77	1120
ControlNet (2D) [95]	21.33	0.83	0.14	3.70	1024
BrushNet (2D) [27]	22.84	0.83	0.13	3.77	1081
Ours	28.59	0.89	0.05	3.80	1250

the views. The difference lies only in the conditioning signals: our diffusion model is informed by the 3D world and other views through the various cues passed to the generator at inference time. After inpainting all the views, similar to our method, a NeRF is fit to the inpainted views and the rendered images and videos are assessed to compute the evaluation metrics (refer to Sec. 4.2 for details). As demonstrated, our method significantly outperforms all baselines, confirming its superior quality in the context of 3D inpainting.

4.4.2 Ablation of Inference-Time Strategies

In Tab. 4.5, we ablate various inference-time strategies of our inpainting pipeline on scene completion. We observe that, for wide-baseline datasets like NeRFiller, our autoregressive procedure (Sec. 3.5) is essential, as a single reference lacks sufficient information for a wide baseline (first row). We also find that providing DUSt3R with ground-truth depth maps has little effect on performance, highlighting the robustness of our method. However, ground-truth camera parameters have a more significant impact (second to fourth rows). This is mainly because optimizing camera parameters in DUSt3R involves complex global alignment, whereas, when camera parameters are known, optimizing the depth maps becomes a much simpler task. The qualitative example in Fig. 4.8 also confirms our findings.

Table 4.5: Ablation of available inputs and inpainting strategies on the NeRFiller scenes dataset. We denote the camera parameters as $(\mathbf{R}, \mathbf{t}, \mathbf{K})$, depth maps as D, inpainting strategy as St., single-reference inpainting as SR, and autoregressive inpainting as AR. Note that the last row represents our full strategy.

$(\mathbf{R}, \mathbf{t}, \mathbf{K})$	D	St.	PSNR ↑	SSIM ↑	LPIPS ↓	MUSIQ ↑	Corrs ↑
✓	✓	SR	27.42	0.88	0.07	3.77	1232
✗	✗	AR	28.32	0.88	0.05	3.78	1235
✗	✓	AR	28.29	0.88	0.05	3.79	1231
✓	✗	AR	28.44	0.89	0.05	3.80	1252
✓	✓	AR	28.59	0.89	0.05	3.80	1250

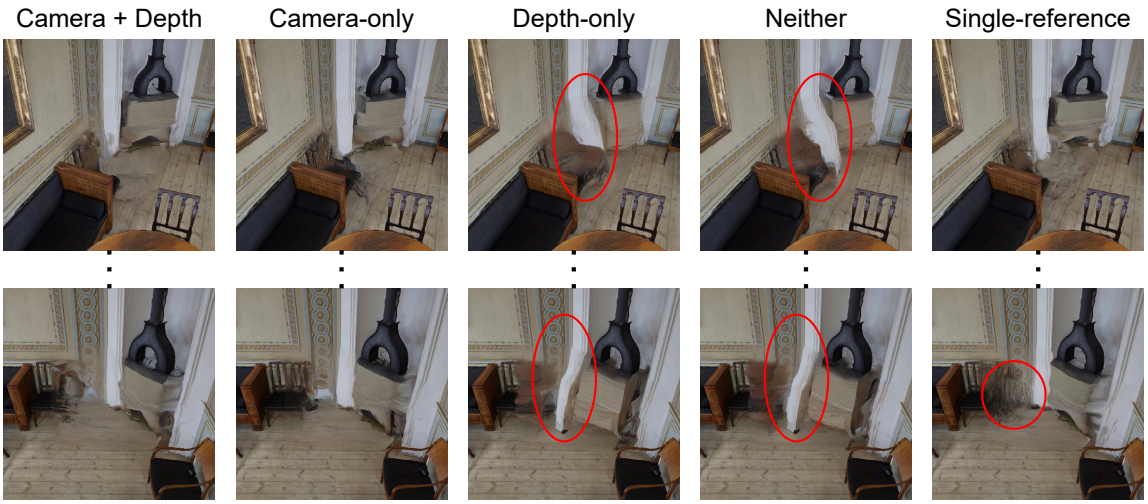

Figure 4.8: A qualitative example from the ablation of inference-time strategies, confirming ground-truth camera parameters and the autoregressive procedure affect the quality of the inpainted scene, whereas ground-truth depth maps have little impact.

4.4.3 Model Design Ablation

Finally, we ablate or vary several design decisions in our training and fusion strategies, including training datasets, availability of geometric cues ($\mathcal{G}_{\mathcal{R}}$) in the conditioning signals, whether to align the reference images to the coordinate of the target image by reprojection, and whether to use mesh perturbation. According to Tab. 4.6, we find that although mesh perturbation (Sec. 3.4.6) does not improve image-based metrics, it has a significant impact

Table 4.6: Ablation of various design choices in training and fusion, including conditioning signals, datasets, and other algorithmic components. We denote the presence of geometric cues in the conditioning signals as $\mathcal{G}_{\mathcal{R}}$. 'Closest camera' and 'Weighted average' ignore the predicted confidence masks; the former solely conditions on the closest view that has been inpainted, and the latter takes a weighted average of the noise estimates, proportional to the inverse view distance between the reference view, r, and the target view, τ, i.e., $\frac{1}{d(r,\tau)}$ (Eq. (3.9)). 'Single confidence' means that, since back-face and shadow masks are disabled, there is only one confidence signal, derived from the front-face mask. The last row represents our full model, which is superior on nearly all metrics, compared to other variants.

Training datasets	$\mathcal{G}_{\mathcal{R}}$	Reference	Perturbation	Fusion	PSNR ↑	SSIM ↑	LPIPS ↓	MUSIQ ↑	Corrs ↑
RealEstate10K + GSO	✗	Naive	N/A	Weighted average	24.34	0.85	0.10	3.79	1113
RealEstate10K + GSO	✓	Reprojected	✓	Hierarchical	27.12	0.87	0.06	3.76	1182
COCO + GSO	✗	Reprojected	✓	Single confidence	28.36	0.88	0.05	3.78	1223
COCO + GSO	✓	Reprojected	✓	Weighted average	29.45	0.89	0.06	3.77	1166
COCO + GSO	✓	Reprojected	✓	Closest camera	27.36	0.88	0.06	3.78	1204
COCO + GSO	✓	Reprojected	✗	Hierarchical	29.25	0.89	0.05	3.72	1222
COCO + GSO	✓	Reprojected	✓	Hierarchical	28.59	0.89	0.05	3.80	1250

on video-based metrics, i.e., a higher image quality and consistency.

Moreover, we find that it is essential to use our hierarchical fusion method (Sec. 3.4.5), as alternative approaches such as "weighted average" and "closest camera" lead to lower performance. On the other hand, "weighted average" achieves the highest PSNR and SSIM among all settings. This is primarily because averaging multiple noise estimates may fuse inconsistent information, producing a blur artifact similar to NeRF renders. Since the inpainted images are already significantly blurred, fitting a NeRF does not introduce additional blurriness. This will result in higher consistency between the inpainted images and their corresponding NeRF renders, leading to higher PSNR and SSIM, though at the cost of lower overall image quality, as reflected in other metrics.

We also observe the importance of conditioning the inpainter on the geometric cues (Sec. 3.4.4). Note that in this case, fusion is performed using a single confidence mask;

with back-face and shadow masks disabled, the only confidence signal comes from the front-face mask.

As mentioned in Sec. 3.4.6, we use a single-view image dataset instead of a multiview one, to ensure a greater data diversity. To explore the effectiveness of a single-view dataset, we compare our base model with the same model trained on RealEstate10K [100] instead of COCO [36]. RealEstate10K, which includes a large set of scenes represented as posed multiview image sets, is commonly used for training large-scale cross-dataset novel view synthesis models (e.g., [80, 92]). To obtain multiview depth maps for RealEstate10K, we run DUSt3R on all the videos as a pre-processing stage. Tab. 4.6 shows that our base model outperforms the one trained on RealEstate10K.

Finally, we evaluate a model naively conditioned on the reference image without any 3D reprojection, instead of our coordinate-aligned conditional inpainting. As reference-based photometric and geometric cues are synthesized for a single-view image dataset like COCO, and no actual reference image exists, we train this model on RealEstate10K. In other words, for naive conditioning, we must use a dataset with multiple posed views per scene, so that one may be used as target and the other as conditioning; our synthetic single-image reprojections, which have only one frame, therefore cannot be used. As presented in Tab. 4.6 (the first two rows), coordinate-aligned conditional inpainting outperforms naive conditioning on RealEstate10K.

Chapter 5

Conclusion

In this thesis, we have presented a novel geometry-aware approach to multiview-consistent 3D scene inpainting that addresses fundamental challenges in the field while offering practical advantages over existing methods. Our work represents a significant departure from traditional approaches that rely on 3D radiance field representations, instead leveraging the power of learned generative spaces and geometry estimation to achieve superior visual quality and geometric consistency, respectively.

The core innovation of our approach lies in the development of a geometry-aware conditional diffusion model that performs cross-view information fusion in the learned latent space of a generative model, rather than in pixel space as done by existing radiance field-based methods. This fundamental design choice enables us to maintain the sharp, high-quality outputs characteristic of modern diffusion models while ensuring multiview consistency through explicit geometric constraints derived from scene geometry estimation.

Our method successfully addresses the primary limitations of existing approaches in several key ways. First, we eliminate the dependency on dense view coverage that constrains radiance field-based methods, enabling effective inpainting even with sparse viewpoint sets through our few-view capability. Second, we avoid the blurry outputs that result from pixel-space fusion of inconsistent cross-view inpainting by operating in the learned space of our geometry-aware diffusion model. Third, we achieve computational efficiency through our autoregressive framework that sequentially processes the key-view subset before propagating content to remaining viewpoints, avoiding the extensive iterative refinement cycles required by methods based on Score Distillation Sampling (SDS) [53] or Iterative Dataset Update (IDU) [19].

Our comprehensive empirical evaluation across three challenging tasks (narrow-baseline object removal, wide-baseline scene completion, and few-view inpainting) demonstrates state-of-the-art performance on established benchmarks. On the SPIn-NeRF dataset for object removal, our method achieves superior visual quality and consistency while maintaining significantly faster runtime compared to existing approaches. For scene completion on the NeRFiller dataset, we outperform all baselines across quantitative metrics while producing sharper, more visually plausible results. Most notably, our method excels in the few-view inpainting scenario, where traditional radiance field-based approaches struggle due to insufficient viewpoint coverage for reliable 3D model fitting.

The technical contributions of our work extend beyond the core algorithmic innovations. Our approach to synthesizing multiview training data from single-view images enables leveraging large-scale 2D datasets for training geometry-aware multiview inpainters, addressing the scarcity of 3D scene data. Our hierarchical confidence estimation mecha-

nism provides robust fusion of information from multiple reference views while adapting to geometry estimation errors.

Our ablation studies validate the importance of each component in our framework. The comparison with independent 2D inpainting methods confirms the critical role of geometry-aware cross-view fusion in achieving consistent 3D inpainting results. The analysis of inference-time strategies demonstrates the necessity of our autoregressive approach for wide-baseline scenarios and highlights the robustness of our method to imperfect geometric information. The model design ablations reveal the effectiveness of our hierarchical fusion strategy and the value of coordinate-aligned conditional inpainting over naive reference conditioning approaches.

While our approach achieves strong performance across all evaluated scenarios, it is not without limitations, as outlined next in Sec. 5.1. The dependency on external tools (Stable Diffusion for generative inpainting and DUSt3R for geometry estimation) means that failures or limitations in these components are propagated through our pipeline. While our method demonstrates graceful degradation in the presence of such failures, extreme errors in geometry estimation or highly out-of-distribution inpainting scenarios can impact the quality of results. Nevertheless, the modular nature of our approach means that future improvements to these foundational tools will directly benefit our method.

The broader impact of our work extends across multiple domains, as discussed later in Sec. 5.2. Our method's unique capabilities (particularly the few-view inpainting capability and computational efficiency) make it well-suited for practical applications in cultural heritage preservation, virtual and augmented reality content creation, film production, and autonomous systems research. The ability to work with sparse view coverage addresses

real-world constraints where dense image capture may be impractical or impossible, significantly expanding the applicability of 3D scene inpainting technology.

Looking forward, our work opens several promising research directions, as explored below in Sec. 5.3. Most notably, the geometry-aware approach demonstrated for static scenes could be extended to dynamic scenarios, enabling 4D inpainting that maintains both spatial and temporal consistency for video editing applications. Additionally, while our method achieves improved computational efficiency over existing approaches, further work toward real-time performance could unlock new interactive applications in augmented reality and immersive content creation.

In conclusion, this thesis has demonstrated that geometry-aware diffusion models represent a powerful paradigm for multiview-consistent 3D scene inpainting. By carefully integrating scene geometry estimation with conditional generative modeling, we have developed an approach that achieves state-of-the-art performance while offering practical advantages in terms of view coverage requirements, computational efficiency, and visual quality. Our work contributes to the broader goal of making high-quality 3D content creation and manipulation more accessible and practical for real-world applications, while maintaining the rigorous evaluation standards necessary for advancing the field of computer vision and graphics.

The success of our geometry-aware approach suggests that the future of 3D scene manipulation lies not necessarily in more complex 3D representations, but in the thoughtful integration of geometric understanding with powerful 2D generative priors. This thesis provides a foundation for continued exploration of this promising direction, with the potential to unlock new capabilities in 3D content creation and editing for years to come.

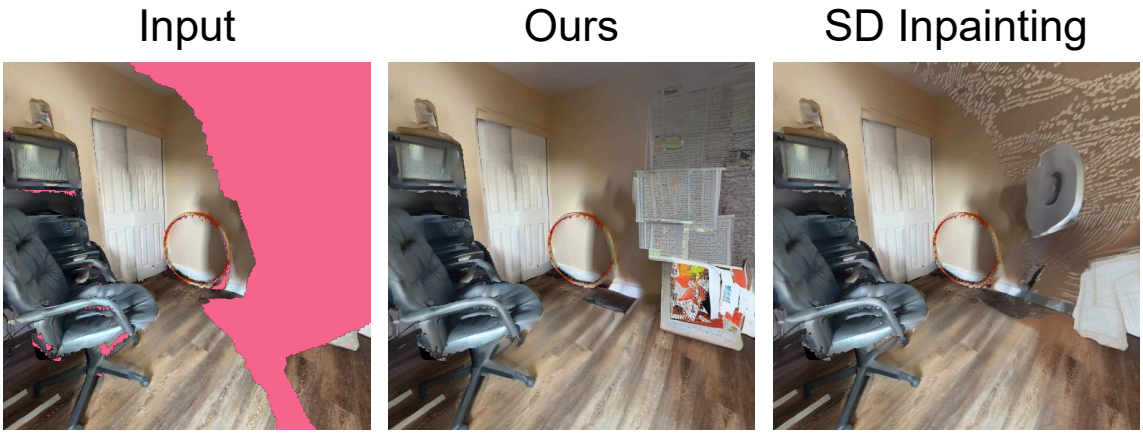

Figure 5.1: Our inpainting model inherits the limitations of Stable Diffusion for inpainting.

5.1 Limitations

While we have demonstrated improved image quality and cross-view consistency over existing baselines, in addition to applicability to the few-view scenario, some shortcomings remain in our approach. Our method relies on two external tools, Stable Diffusion [60] for inpainting and DUSt3R [81] for geometry estimation. Hence, errors caused by these tools (e.g., highly implausible inpaintings or errors in depth estimation) will be propagated throughout the process, resulting in degraded outcomes. In the case of extreme failure in these external tools (e.g. extreme errors in camera and depth estimation), our method is unable to recover.

For instance, consider Fig. 5.1, showing the *initial* view for inpainting the "office" scene from the NeRFiller dataset. Clearly, the failure of our method is inherited from the Stable Diffusion inpainter, likely caused by out-of-distribution conditioning (highly irregular inpainting mask in this case). Nevertheless, our method does not catastrophically fail in such cases, even as extreme as this one, maintaining geometry and appearance

quality elsewhere in the scene. We also expect that future improvements to the mentioned tools will also benefit our approach.

5.2 Broader Impact

Our geometry-aware 3D scene inpainting method has the potential to generate significant positive societal impact across various domains that rely on high-quality 3D content creation and manipulation. The ability to perform multiview-consistent inpainting opens new possibilities for applications that require realistic and coherent 3D scene editing.

In the realm of cultural heritage preservation, our method can be instrumental in digitally restoring damaged historical sites and artifacts. Museums and archaeological institutions can use our approach to remove unwanted elements from 3D scans (such as modern infrastructure or tourists) while maintaining the authentic appearance of historical environments. The few-view capability of our method is particularly valuable in this context, as many heritage sites have limited photographic documentation.

Virtual and augmented reality applications can benefit significantly from our approach. Content creators can efficiently edit 3D scenes by removing or replacing objects while ensuring visual consistency across all viewpoints. This capability is crucial for creating immersive experiences where users can freely navigate the environment without encountering visual artifacts or inconsistencies. Our method's ability to handle uncertain or partial scene information makes it particularly suitable for real-time AR applications where complete scene coverage may not be available.

In the film and entertainment industry, our method enables efficient post-production

workflows for removing unwanted objects from scenes captured with multiple cameras. The geometry-aware nature of our approach ensures that edits remain consistent regardless of viewing angle, reducing the manual labor typically required for such tasks. The sharper outputs achieved by our learned-space fusion approach, compared to traditional pixel-space methods, are particularly valuable for high-quality content production.

Autonomous driving and robotics research can leverage our method for data augmentation and simulation. Researchers can modify existing 3D datasets by removing or altering objects to create diverse training scenarios, potentially improving the robustness of perception systems. The ability to perform consistent edits without requiring perfect camera calibration makes our approach more practical for real-world robotics applications where precise geometric information may be challenging to obtain.

However, our work also presents potential negative societal implications that must be acknowledged. The ability to convincingly edit 3D scenes raises concerns about the creation of misleading or deceptive content. Malicious actors could use such technology to fabricate realistic 3D evidence of events that never occurred, potentially undermining trust in visual documentation. The multiview consistency provided by our method makes such manipulations particularly convincing, as they would appear authentic from any viewing angle.

Furthermore, the technology could be misused for surveillance applications, where scenes are altered to hide or fabricate activities. The potential for creating deepfake-like 3D content poses risks to privacy and could contribute to the spread of misinformation.

As with many advances in generative AI, the responsible development and deployment of 3D scene editing technologies requires careful consideration of ethical guidelines, po-

tential regulatory frameworks, and education about the capabilities and limitations of such systems. We encourage the research community to continue developing detection methods and establishing best practices for the ethical use of 3D content manipulation technologies.

5.3 Future Directions

The geometry-aware approach to 3D scene inpainting presented in this thesis opens several promising avenues for future research that could further advance the field of 3D content manipulation and generation.

Extension to Dynamic Scenes (4D Inpainting). In this thesis, we have demonstrated that equipping generative models with geometry-awareness is key to achieving geometric consistency in static 3D scene inpainting. A natural and compelling next step would be extending these ideas to dynamic scenes, effectively enabling 4D inpainting that maintains both spatial and temporal consistency. This would involve developing methods that can handle the additional complexity of motion and deformation while preserving the geometric constraints that make our current approach successful. Such capabilities would be particularly valuable for video editing applications, where objects need to be removed or modified across both multiple viewpoints and temporal sequences. The challenge lies in developing efficient representations that can capture 4D scene dynamics while maintaining the quality and consistency advantages of our geometry-aware diffusion framework.

Real-Time Efficiency Improvements. Although our method has demonstrated significant improvements in computational efficiency compared to existing 3D inpainting approaches, particularly those based on iterative optimization techniques like Score Dis-

tillation Sampling [53], it still falls short of real-time performance requirements for interactive applications. Working on improving this efficiency could be an interesting future direction, making it closer to real-time applications. Real-time 3D inpainting would unlock new interactive applications in augmented reality, live broadcasting, and immersive content creation, where users expect immediate feedback from their editing operations.

These future directions represent natural extensions of our core contributions and have the potential to significantly impact the broader field of 3D computer vision and graphics, making high-quality 3D content creation and manipulation even more accessible and practical for real-world applications.

Bibliography

[1] Titas Anciukevičius, Zexiang Xu, Matthew Fisher, Paul Henderson, Hakan Bilen, Niloy J Mitra, and Paul Guerrero. RenderDiffusion: Image diffusion for 3D reconstruction, inpainting and generation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 15

[2] Omri Avrahami, Amir Hertz, Yael Vinker, Moab Arar, Shlomi Fruchter, Ohad Fried, Daniel Cohen-Or, and Dani Lischinski. The Chosen One: Consistent characters in text-to-image diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. 17

[3] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. *Proceedings of SIGGRAPH*, 2000. 1

[4] Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 40

[5] Chenjie Cao, Chaohui Yu, Fan Wang, Xiangyang Xue, and Yanwei Fu. MVIn-

painter: Learning multi-view consistent inpainting to bridge 2D and 3D editing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 16

[6] Eric R Chan, Koki Nagano, Matthew A Chan, Alexander W Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. Generative novel view synthesis with 3D-aware diffusion models. In *International Conference on Computer Vision (ICCV)*, 2023. 18

[7] Guikun Chen and Wenguan Wang. A survey on 3D Gaussian Splatting. *arXiv preprint arXiv:2401.03890*, 2024. 15

[8] Honghua Chen, Chen Change Loy, and Xingang Pan. MVIP-NeRF: Multi-view 3D inpainting on NeRF scenes via diffusion prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 14, 16, 19, 26, 58, 62

[9] Jiafu Chen, Tianyi Chu, Jiakai Sun, Wei Xing, and Lei Zhao. Single-mask inpainting for voxel-based neural radiance fields. In *European Conference on Computer Vision (ECCV)*, 2024. 16

[10] Minghao Chen, Iro Laina, and Andrea Vedaldi. DGE: Direct Gaussian 3D editing by consistent multi-view editing. *European Conference on Computer Vision (ECCV)*, 2024. 15

[11] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. GaussianEditor: Swift and controllable 3D editing with Gaussian Splatting. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 15

[12] Jaeyoung Chung, Suyoung Lee, Hyeongjin Nam, Jaerin Lee, and Kyoung Mu Lee. LucidDreamer: Domain-free generation of 3D Gaussian Splatting scenes. *arXiv preprint arXiv:2311.13384*, 2023. 19

[13] Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 7, 10

[14] Jan-Niklas Dihlmann, Andreas Engelhardt, and Hendrik Lensch. SIGNeRF: Scene integrated generation for neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 16

[15] Jiahua Dong and Yu-Xiong Wang. ViCA-NeRF: View-consistency-aware 3D editing of neural radiance fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 15

[16] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google Scanned Objects: A high-quality dataset of 3D scanned household items. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2022. 41, 44, 56

[17] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. CAT3D: Create anything in 3D with multi-view diffusion models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 18, 28

[18] Michael Haller, Stephan Drab, and Werner Hartmann. A real-time shadow approach for an augmented reality application using shadow volumes. In *Proceedings of the ACM symposium on Virtual Reality Software and Technology*, 2003. 30

[19] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-NeRF2NeRF: Editing 3D scenes with instructions. *International Conference on Computer Vision (ICCV)*, 2023. 2, 13, 15, 19, 26, 59, 75

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 103

[21] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 59

[22] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 7, 10, 56

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 7

[24] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2Room: Extracting textured 3D meshes from 2D text-to-image models. In *International Conference on Computer Vision (ICCV)*, 2023. 19

[25] Xudong Huang, Wei Li, Jie Hu, Hanting Chen, and Yunhe Wang. RefSR-NeRF: Towards high fidelity and super resolution view synthesis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 15

[26] Clément Jambon, Bernhard Kerbl, Georgios Kopanas, Stavros Diolatzis, Thomas Leimkühler, and George Drettakis. NeRFshop: Interactive editing of neural radiance fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2023. 15

[27] Xuan Ju, Xian Liu, Xintao Wang, Yuxuan Bian, Ying Shan, and Qiang Xu. BrushNet: A plug-and-play image inpainting model with decomposed dual-branch diffusion. In *European Conference on Computer Vision (ECCV)*, 2024. 13, 69, 70

[28] Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. MUSIQ: Multi-scale image quality transformer. In *International Conference on Computer Vision (ICCV)*, 2021. `www.kaggle.com/models/google/musiq/TensorFlow2/ava/1`. 60, 103

[29] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 2023. 1, 14, 26, 58

[30] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 8, 11

[31] Juil Koo, Chanho Park, and Minhyuk Sung. Posterior distillation sampling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 15

[32] Zhengfei Kuang, Fujun Luan, Sai Bi, Zhixin Shu, Gordon Wetzstein, and Kalyan Sunkavalli. PaletteNeRF: Palette-based appearance editing of neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 15

[33] Jae-Hyeok Lee and Dae-Shik Kim. ICE-NeRF: Interactive color editing of NeRFs via decomposition-aware weight optimization. In *International Conference on Computer Vision (ICCV)*, 2023. 15

[34] Jie Long Lee, Chen Li, and Gim Hee Lee. DiSR-NeRF: Diffusion-guided view-consistent super-resolution NeRF. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 15

[35] Chieh Hubert Lin, Changil Kim, Jia-Bin Huang, Qinbo Li, Chih-Yao Ma, Johannes Kopf, Ming-Hsuan Yang, and Hung-Yu Tseng. Taming latent diffusion model for neural radiance field inpainting. In *European Conference on Computer Vision (ECCV)*, 2024. 14, 16, 59, 62, 108, 109

[36] Tsung-Yi Lin, Michael Maire, Serge J Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 41, 44, 56, 73

[37] Andrew Liu, Richard Tucker, Varun Jampani, Ameesh Makadia, Noah Snavely, and Angjoo Kanazawa. Infinite Nature: Perpetual view generation of natural scenes from a single image. In *International Conference on Computer Vision (ICCV)*, 2021. 19

[38] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. In *International Conference on Computer Vision (ICCV)*, 2023. 18

[39] Zhiheng Liu, Hao Ouyang, Qiuyu Wang, Ka Leong Cheng, Jie Xiao, Kai Zhu, Nan Xue, Yu Liu, Yujun Shen, and Yang Cao. InFusion: Inpainting 3D Gaussians via learning depth completion from diffusion prior. *arXiv preprint arXiv:2404.11613*, 2024. 16, 58, 62, 107

[40] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. 57

[41] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 2004. 101

[42] Yiren Lu, Jing Ma, and Yu Yin. View-consistent object removal in radiance fields. In *Proceedings of the ACM International Conference on Multimedia*, 2024. 15

[43] Alessio Mazzucchelli, Adrian Garcia-Garcia, Elena Garces, Fernando Rivas-Manzaneque, Francesc Moreno-Noguer, and Adrian Penate-Sanchez. IReNe: Instant recoloring of neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 15

[44] David McAllister, Songwei Ge, Jia-Bin Huang, David Jacobs, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Rethinking score distillation as a bridge between image distributions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 19

[45] Michael D McCool. Shadow volume reconstruction from depth maps. *ACM Transactions on Graphics (TOG)*, 2000. 30

[46] 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 *European Conference on Computer Vision (ECCV)*, 2020. 1, 15, 26

[47] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Marcus A Brubaker, Jonathan Kelly, Alex Levinshtein, Konstantinos G Derpanis, and Igor Gilitschenski. Reference-guided controllable inpainting of neural radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023. 2, 13, 15, 26

[48] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Konstantinos G Derpanis, Jonathan Kelly, Marcus A Brubaker, Igor Gilitschenski, and Alex Levinshtein. SPIn-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 4, 13, 15, 26, 58, 59, 61, 62, 67

[49] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Marcus A Brubaker, Jonathan Kelly, Alex Levinshtein, Konstantinos G Derpanis, and Igor Gilitschenski. Watch your steps: Local image and scene editing by text instructions. In *European Conference on Computer Vision (ECCV)*, 2024. 15, 26

[50] Ashkan Mirzaei, Riccardo De Lutio, Seung Wook Kim, David Acuna, Jonathan Kelly, Sanja Fidler, Igor Gilitschenski, and Zan Gojcic. RefFusion: Ref-

erence adapted diffusion models for 3D scene inpainting. *arXiv preprint arXiv:2404.10765*, 2024. 14, 16, 19, 26, 58, 62

[51] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, Qixiang Ye, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. In *International Conference on Learning Representations (ICLR)*, 2024. 56

[52] Said Pertuz, Domenec Puig, and Miguel Angel Garcia. Analysis of focus measure operators for shape-from-focus. *Pattern Recognition*, 2013. 59

[53] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. *International Conference on Learning Representations (ICLR)*, 2023. 2, 14, 16, 19, 75, 82

[54] Kira Prabhu, Jane Wu, Lynn Tsai, Peter Hedman, Dan B Goldman, Ben Poole, and Michael Broxton. Inpaint3D: 3D scene content generation using 2D inpainting diffusion. *arXiv preprint arXiv:2312.03869*, 2023. 13, 16, 19, 26, 107

[55] Weize Quan, Jiaxi Chen, Yanli Liu, Dong-Ming Yan, and Peter Wonka. Deep learning-based image and video inpainting: A survey. *International Journal of Computer Vision (IJCV)*, 2024. 12

[56] AKM Rabby and Chengcui Zhang. BeyondPixels: A comprehensive review of the evolution of neural radiance fields. *arXiv preprint arXiv:2306.03000*, 2023. 15

[57] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo,

Justin Johnson, and Georgia Gkioxari. Accelerating 3D deep learning with PyTorch3D. *arXiv:2007.08501*, 2020. 56

[58] Xuanchi Ren and Xiaolong Wang. Look outside the room: Synthesizing a consistent long-term 3D scene video from a single image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 19

[59] Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-free view synthesis: Transformers and no 3D priors. In *International Conference on Computer Vision (ICCV)*, 2021. 19

[60] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 11, 12, 40, 56, 59, 65, 69, 70, 78

[61] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 11, 40

[62] Rahul Sajnani, Jeroen Vanbaar, Jie Min, Kapil D Katyal, and Srinath Sridhar. GeoDiffuser: Geometry-based image editing with diffusion models. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2025. 17

[63] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 20, 59

[64] I-Chao Shen, Hao-Kang Liu, and Bing-Yu Chen. NeRF-In: Free-form inpainting for pretrained NeRF with RGB-D priors. *IEEE Computer Graphics and Applications*, 2023. 1, 15

[65] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv preprint arXiv:2310.15110*, 2023. 18

[66] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations (ICLR)*, 2015. 59

[67] Jascha Sohl-Dickstein, Eric A Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *International Conference on Machine Learning (ICML)*, 2015. 7

[68] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *International Conference on Learning Representations (ICLR)*, 2021. 40, 57

[69] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. 7

[70] StabilityAI. stabilityai/stable-diffusion-2-inpainting, Released

2023. URL https://huggingface.co/stabilityai/stable-diffusion-2-inpainting. 12, 40, 56

[71] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with Transformers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 105

[72] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with Fourier convolutions. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2022. 44, 61

[73] Jiaxiang Tang. Stable-dreamfusion: Text-to-3D with stable-diffusion, 2022. https://github.com/ashawkey/stable-dreamfusion. 14

[74] Ayush Tewari, Tianwei Yin, George Cazenavette, Semon Rezchikov, Josh Tenenbaum, Frédo Durand, Bill Freeman, and Vincent Sitzmann. Diffusion with forward models: Solving stochastic inverse problems without direct supervision. *Advances in Neural Information Processing Systems*, 2023. 28

[75] Yoad Tewel, Omri Kaduri, Rinon Gal, Yoni Kasten, Lior Wolf, Gal Chechik, and Yuval Atzmon. Training-free consistent text-to-image generation. *ACM Transactions on Graphics (TOG)*, 2024. 17

[76] Hung-Yu Tseng, Qinbo Li, Changil Kim, Suhib Alsisan, Jia-Bin Huang, and Jo-

hannes Kopf. Consistent view synthesis with pose-guided diffusion models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 18, 19

[77] Arash Vahdat and Karsten Kreis. Improving diffusion models as an alternative to GANs, part 1. https://developer.nvidia.com/blog/improving-diffusion-models-as-an-alternative-to-gans-part-1/, 2022. [Accessed 28-03-2025]. 8

[78] Dongqing Wang, Tong Zhang, Alaa Abboud, and Sabine Süsstrunk. InNeRF360: Text-guided 3D-consistent object inpainting on 360-degree neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 15

[79] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score Jacobian chaining: Lifting pretrained 2D diffusion models for 3D generation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 16

[80] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning multi-view image-based rendering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 73

[81] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 20, 25, 28, 78

[82] Xiangyu Wang, Jingsen Zhu, Qi Ye, Yuchi Huo, Yunlong Ran, Zhihua Zhong, and

Jiming Chen. Seal-3D: Interactive pixel-level editing for neural radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023. 15

[83] Yuxin Wang, Qianyi Wu, Guofeng Zhang, and Dan Xu. Learning 3D geometry and feature consistent Gaussian Splatting for object removal. In *European Conference on Computer Vision (ECCV)*, 2024. 15

[84] Yuxuan Wang, Xuanyu Yi, Zike Wu, Na Zhao, Long Chen, and Hanwang Zhang. View-consistent 3D editing with Gaussian Splatting. In *European Conference on Computer Vision (ECCV)*, 2024. 15

[85] Ethan Weber, Aleksander Holynski, Varun Jampani, Saurabh Saxena, Noah Snavely, Abhishek Kar, and Angjoo Kanazawa. NeRFiller: Completing scenes via generative 3D inpainting. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 4, 13, 19, 26, 58, 59, 60, 65, 67, 69, 105

[86] Silvan Weder, Guillermo Garcia-Hernando, Aron Monszpart, Marc Pollefeys, Gabriel J Brostow, Michael Firman, and Sara Vicente. Removing objects from neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 15

[87] Jing Wu, Jia-Wang Bian, Xinghui Li, Guangrun Wang, Ian D Reid, Philip Torr, and Victor Adrian Prisacariu. GaussCtrl: Multi-view consistent text-driven 3D Gaussian splatting editing. In *European Conference on Computer Vision (ECCV)*, 2024. 15

[88] Jiayu Yang, Ziang Cheng, Yunfei Duan, Pan Ji, and Hongdong Li. ConsistNet:

Enforcing 3D consistency for multi-view images diffusion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 18

[89] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 56

[90] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3D scenes. *European Conference on Computer Vision (ECCV)*, 2024. 16, 107

[91] Jason J Yu, Fereshteh Forghani, Konstantinos G Derpanis, and Marcus A Brubaker. Long-term photometric consistent novel view synthesis with diffusion models. In *International Conference on Computer Vision (ICCV)*, 2023. 18, 19, 59, 60, 101

[92] Jason J Yu, Tristan Aumentado-Armstrong, Fereshteh Forghani, Konstantinos G Derpanis, and Marcus A Brubaker. PolyOculus: Simultaneous multi-view image-based novel view synthesis. *European Conference on Computer Vision (ECCV)*, 2024. 28, 73

[93] Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. ViewCrafter: Taming video diffusion models for high-fidelity novel view synthesis. *arXiv preprint arXiv:2409.02048*, 2024. 18

[94] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao.

NeRF-editing: geometry editing of neural radiance fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 15

[95] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *International Conference on Computer Vision (ICCV)*, 2023. 13, 69, 70

[96] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 59

[97] Liang Zhao, Xinyuan Zhao, Hailong Ma, Xinyu Zhang, and Long Zeng. 3DFill: Reference-guided image inpainting by self-supervised 3D image alignment. *arXiv preprint arXiv:2211.04831*, 2022. 18

[98] Yunhan Zhao, Connelly Barnes, Yuqian Zhou, Eli Shechtman, Sohrab Amirghodsi, and Charless Fowlkes. GeoFill: Reference-based image inpainting with better geometric understanding. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2023. 18, 32

[99] Chengwei Zheng, Wenbin Lin, and Feng Xu. EditableNeRF: Editing topologically varying neural radiance fields by key points. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 15

[100] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *Proceedings of SIGGRAPH*, 2018. 73

[101] Yupeng Zhou, Daquan Zhou, Ming-Ming Cheng, Jiashi Feng, and Qibin Hou. StoryDiffusion: Consistent self-attention for long-range image and video generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 17

[102] Yuqian Zhou, Connelly Barnes, Eli Shechtman, and Sohrab Amirghodsi. TransFill: Reference-guided image inpainting by merging multiple color and spatial transformations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 18

Appendix

This appendix provides supplementary material in four sections: complete hyperparameter configuration for reproducibility (Appendix A.1), detailed explanations of specialized evaluation metrics TSED, MUSIQ, and Corrs (Appendix A.2), additional object removal results from prior works (Appendix A.3), and additional qualitative comparisons with MALD-NeRF (Appendix A.4).

A.1 Hyperparameter Configuration

Tab. A.1 provides a comprehensive list of all hyperparameters used in our method, including their values, descriptions, and references to where they are applied in the technical approach.

A.2 Evaluation Metrics

This section provides detailed explanations of the specialized evaluation metrics used throughout our empirical evaluation that may not be familiar to all readers. While standard computer vision metrics such as PSNR, SSIM, LPIPS, FID, and Laplacian variance

Table A.1: Complete hyperparameter configuration used in our multiview inpainting method.

Parameter	Value	Usage	Description
Geometry Processing			
ϵ_{edge}	4×10^{-2}	Mesh construction (Sec. 3.4.2)	Edge threshold for geometry processing
Dataset Preparation			
σ_{a_r}	0.3	Reference poses (Sec. 3.4.6)	Standard deviation for rotation angle sampling
σ_{t_r}	0.2	Reference poses (Sec. 3.4.6)	Standard deviation for translation sampling
o_{min}	0.6	3D occluders (Sec. 3.4.6)	Minimum occluder size parameter (both datasets)
o_{max} (GSO)	1.0	3D occluders (Sec. 3.4.6)	Maximum occluder size parameter for GSO dataset
o_{max} (COCO)	0.8	3D occluders (Sec. 3.4.6)	Maximum occluder size parameter for COCO dataset
Reference dropout probability	0.2	3D occluders (Sec. 3.4.6)	Probability of dropping occluded reference content
σ_{a_p}	0.2	Mesh perturbation (Sec. 3.4.6)	Angular standard deviation for mesh perturbation
σ_{t_p}	0.01	Mesh perturbation (Sec. 3.4.6)	Translational standard deviation for mesh perturbation
Color jitter factor	0.1	Texture augmentation	Factor for brightness, contrast, saturation, hue
Text dropout probability	0.1	Classifier-free guidance	Probability of dropping text prompts
Training Setup			
λ	0.1	Loss function (Sec. 3.4.6)	Loss weight for pixels outside inpainting mask
η	0.2	Loss function (Sec. 3.4.6)	Weight for confidence mask loss component
Learning rate	10^{-4}	AdamW optimizer	Base learning rate
Weight decay	0.01	AdamW optimizer	L2 regularization parameter
β_1	0.9	AdamW optimizer	First moment decay rate
β_2	0.999	AdamW optimizer	Second moment decay rate
Batch size	96	Training setup	Number of samples per batch
Training iterations	8000	Training setup	Total number of training iterations
Gradient accumulation steps	2	Training setup	Steps before parameter update

are well-established and widely understood, we focus here on the more specialized metrics used in our evaluation, which encompass both multiview consistency assessment and advanced image quality evaluation.

A.2.1 TSED

Thresholded Symmetric Epipolar Distance (TSED) [91] is a metric designed to evaluate the consistency of multiview image sets by measuring the geometric consistency of feature correspondences across views. This metric is particularly valuable for assessing whether generated multiview content maintains geometric coherence, as inconsistent geometry will manifest as violations of epipolar constraints.

The computation of TSED begins by extracting SIFT features [41] from each view, $v \in V$, in the image set. For each pair of views, (u, v), the algorithm establishes feature correspondences, $\mathcal{C}$, between the two views using standard feature matching techniques. Given the camera parameters for each view, specifically the extrinsics, $(\mathbf{R}_u, \mathbf{t}_u), (\mathbf{R}_v, \mathbf{t}_v) \in \mathrm{SE}(3)$, and intrinsics, $\mathbf{K}_u, \mathbf{K}_v \in \mathbb{R}^{3\times3}$, the method computes the relative camera motion between the views as:

$$\mathbf{R} = \mathbf{R}_v \mathbf{R}_u^\top, \tag{A.1}$$

$$\mathbf{t} = \mathbf{t}_v - \mathbf{R}\mathbf{t}_u. \tag{A.2}$$

The fundamental matrix $\mathbf{F}$ is then computed as:

$$\mathbf{F} = \mathbf{K}_v^{-\top} [\mathbf{t}]_\times \mathbf{R} \mathbf{K}_u^{-1}, \tag{A.3}$$

where $[\mathbf{t}]_\times$ is the skew-symmetric matrix corresponding to the cross-product operation

with vector $\mathbf{t}$:

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}, \tag{A.4}$$

for $\mathbf{t} = [t_x, t_y, t_z]^\top$.

The fundamental matrix encodes the epipolar constraint that governs the geometric relationship between corresponding points in two views. For any true correspondence between points $\mathbf{p}_u$ and $\mathbf{p}_v$, the epipolar constraint, $\mathbf{p}_v^\top \mathbf{F} \mathbf{p}_u = 0$, must hold, meaning that $\mathbf{p}_v$ lies exactly on the epipolar line, $\mathbf{F}\mathbf{p}_u$. Deviations from this constraint indicate geometric inconsistencies, making the epipolar distance a reliable measure of multiview geometric coherence. The larger the epipolar distance, the less consistent the geometry between the views.

To quantify these deviations, for each feature correspondence, $(\mathbf{p}_u, \mathbf{p}_v) \in \mathcal{C}$, the symmetric epipolar distance (SED) is computed as:

$$\delta(\mathbf{p}_u, \mathbf{p}_v) := \frac{1}{2} \left[d(\mathbf{p}_v, \mathbf{F}\mathbf{p}_u) + d(\mathbf{p}_u, \mathbf{F}^\top \mathbf{p}_v) \right], \tag{A.5}$$

where $d(\mathbf{p}, \ell)$ represents the minimum Euclidean distance between point, $\mathbf{p}$, and line, ℓ. This symmetric formulation ensures that the distance measurement is invariant to the order of the views.

A pair of views, (u, v), is declared geometrically consistent if two conditions are met: first, the number of feature correspondences must exceed a minimum threshold,

$$|\mathcal{C}| \geq T_{\text{matches}}, \tag{A.6}$$

ensuring sufficient feature density for reliable assessment; second, the median of the symmetric epipolar distances across all correspondences must be below an error threshold,

$$\underset{(\mathbf{p}_u,\mathbf{p}_v)\in\mathcal{C}}{\text{median}}\,\{\delta(\mathbf{p}_u,\mathbf{p}_v)\} < T_{\text{error}}. \tag{A.7}$$

The TSED metric then reports the percentage of view pairs that satisfy both consistency criteria, providing a comprehensive measure of multiview geometric coherence.

A.2.2 MUSIQ

MUSIQ [28] is a no-reference image quality assessment metric that leverages multi-scale image representations and transformer architectures to predict perceptual image quality without requiring a reference image. Unlike traditional metrics such as PSNR or SSIM that require a ground truth reference, MUSIQ operates in a blind fashion, making it particularly suitable for evaluating generated content where perfect references may not exist.

The core innovation of MUSIQ lies in its multi-scale approach to image analysis. The method processes input images at multiple resolution levels to capture both fine-grained details and global structural information that contribute to perceived image quality. For an input image, I, the method generates a set of scaled versions, $\{I_s\}_{s=1}^{S}$, where each scale, s, corresponds to a different resolution level.

At each scale, MUSIQ extracts patch-based features using a convolutional backbone, typically a ResNet [20] architecture. These features are then organized into spatial grids that preserve the spatial relationships within the image. The multi-scale feature extraction

can be formalized as:

$$\mathbf{F}_s = \mathrm{CNN}(I_s), \quad s = 1, 2, \ldots, S, \tag{A.8}$$

where $\mathbf{F}_s \in \mathbb{R}^{H_s \times W_s \times D}$ represents the feature map at scale, s, with spatial dimensions, $H_s \times W_s$, and feature dimension, D.

The extracted multi-scale features are then processed by a transformer architecture that can model both local and global dependencies across different scales. The transformer takes flattened spatial feature tokens from all scales and applies self-attention mechanisms to learn relationships between different image regions and scales. This allows the model to understand how local image degradations affect overall perceptual quality and how different scales contribute to the final quality assessment.

The transformer architecture processes the concatenated multi-scale features:

$$\mathcal{F} = \mathrm{Concat}(\mathrm{Flatten}(\mathbf{F}_1), \mathrm{Flatten}(\mathbf{F}_2), \ldots, \mathrm{Flatten}(\mathbf{F}_S)), \tag{A.9}$$

where $\mathrm{Flatten}(\cdot)$ converts the spatial feature maps into sequences of tokens. The transformer then applies multiple layers of self-attention and feed-forward networks to produce a final quality prediction:

$$q = \mathrm{Transformer}(\mathcal{F}), \tag{A.10}$$

where $q \in \mathbb{R}$ represents the predicted quality score.

MUSIQ is trained on large-scale image quality datasets with human perceptual ratings, enabling it to learn the mapping between low-level image features and human quality judgments. The training process involves minimizing the discrepancy between predicted quality scores and ground truth human ratings, typically using regression losses such as

mean squared error.

In our evaluation, MUSIQ provides a complementary perspective to traditional metrics by assessing the perceptual quality of generated content without requiring reference images, which is particularly valuable for novel view synthesis and inpainting tasks where perfect ground truth may not be available.

A.2.3 Corrs

The Correspondence Score (Corrs) metric, introduced by NeRFiller [85], evaluates the consistency of generated content by measuring the quality and quantity of feature correspondences across rendered views. The computation of Corrs relies on the LoFTR (Local Feature Matching with Transformers) network [71], a learned feature matching approach that combines convolutional and transformer architectures to establish dense correspondences between images. The method first converts all images to grayscale representations to focus on structural rather than photometric information. For each view, $v \in V$:

$$G_v = \mathrm{RGB2Gray}(I_v). \tag{A.11}$$

For each pair of views, (u, v), the LoFTR network processes the corresponding grayscale images, G_u and G_v, to extract local features and establish correspondences. The network architecture consists of several stages: feature extraction using convolutional layers, feature enhancement through self- and cross-attention mechanisms, and final correspondence

prediction. The mathematical formulation can be expressed as:

$$\{\mathcal{C}, \mathcal{S}\} = \mathrm{LoFTR}(G_u, G_v), \tag{A.12}$$

where $\mathcal{C}$ represents the set of pixel correspondences between views u and v, and the confidence scores, $\mathcal{S}$, provide a measure of reliability for each correspondence.

To compute the Corrs score for a pair of views, (u, v), a confidence threshold, τ_{conf}, is applied to filter out unreliable correspondences. The final metric counts the number of correspondences that exceed this threshold:

$$\mathrm{Corrs}(u, v) = \sum_{s \in \mathcal{S}} \mathbb{1}\left\{s \geq \tau_{\text{conf}}\right\}, \tag{A.13}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function. In practice, a threshold of $\tau_{\text{conf}} = 0.8$ is commonly used to ensure that only high-quality correspondences contribute to the final score.

The Corrs metric directly measures the matchability of features between views by counting high-confidence correspondences established through the LoFTR network. In our evaluation, it provides a complementary measure to TSED: while TSED focuses on whether correspondences satisfy epipolar constraints, Corrs measures whether correspondences can be established at all.

A.3 Additional Object Removal Results

This section presents additional object removal results that are not directly comparable to our main results in Tab. A.2, primarily due to differences in evaluation code or repro-

Table A.2: Object removal results from prior works using potentially different evaluation code. The first section reports numbers from Inpaint3D as published in their paper. The second section reports numbers from InFusion as published in their paper, including their evaluation of Gaussian Grouping. The third section shows our reproduction of InFusion using their official code. The numbers in the first two sections are not directly comparable to our main results in Tab. 4.1.

Method	LPIPS ↓	FID ↓
Inpaint3D [54]	0.5150	226.04
InFusion (reported) [39]	0.4210	92.62
Gaussian Grp. [90]	0.4540	123.48
InFusion (reproduced) [39]	0.6692	244.19

ducibility issues.

We include results from Inpaint3D [54] and InFusion [39] as reported in their original papers. Additionally, InFusion evaluates Gaussian Grouping [90], a 3DGS-based segmentation method that was not originally evaluated on SPIn-NeRF. However, *these results may not be directly comparable to our evaluation due to potentially different evaluation code implementations*. We include these values only for completeness.

For Inpaint3D, public code is not available, but the paper states that metrics are computed on a bounding box surrounding the masked region, which differs from our protocol (see Sec. 4.2) and should be considered when comparing results.

For InFusion, we provide both the reported results and our own reproduction using their official code[1]. However, the reported results were not reproducible, neither quantitatively nor qualitatively. Notably, others have reported similar reproducibility issues on the project's GitHub repository[2], and the gap in performance we observe may be due to differences in evaluation protocol or per-scene hyperparameter optimization. We note that the paper does not explicitly mention this optimization, nor does it provide a hyperparameter

[1] `https://github.com/ali-vilab/infusion`

[2] see, e.g., #25 and #9

table for reproduction. We include these observations respectfully, as reproducibility can be challenging and implementation details may vary.

A.4 Qualitative Comparison with MALD-NeRF

Fig. A.1 presents a direct qualitative comparison to MALD-NeRF [35], the current state of the art. Here, we highlight the inconsistencies in MALD-NeRF's outputs, primarily caused by common NeRF artifacts, such as floaters or flawed geometry, which compromise geometric plausibility. As a result, the artifacts in MALD-NeRF prevent SIFT from detecting sufficient high-quality feature correspondences, leading to a lower TSED metric.

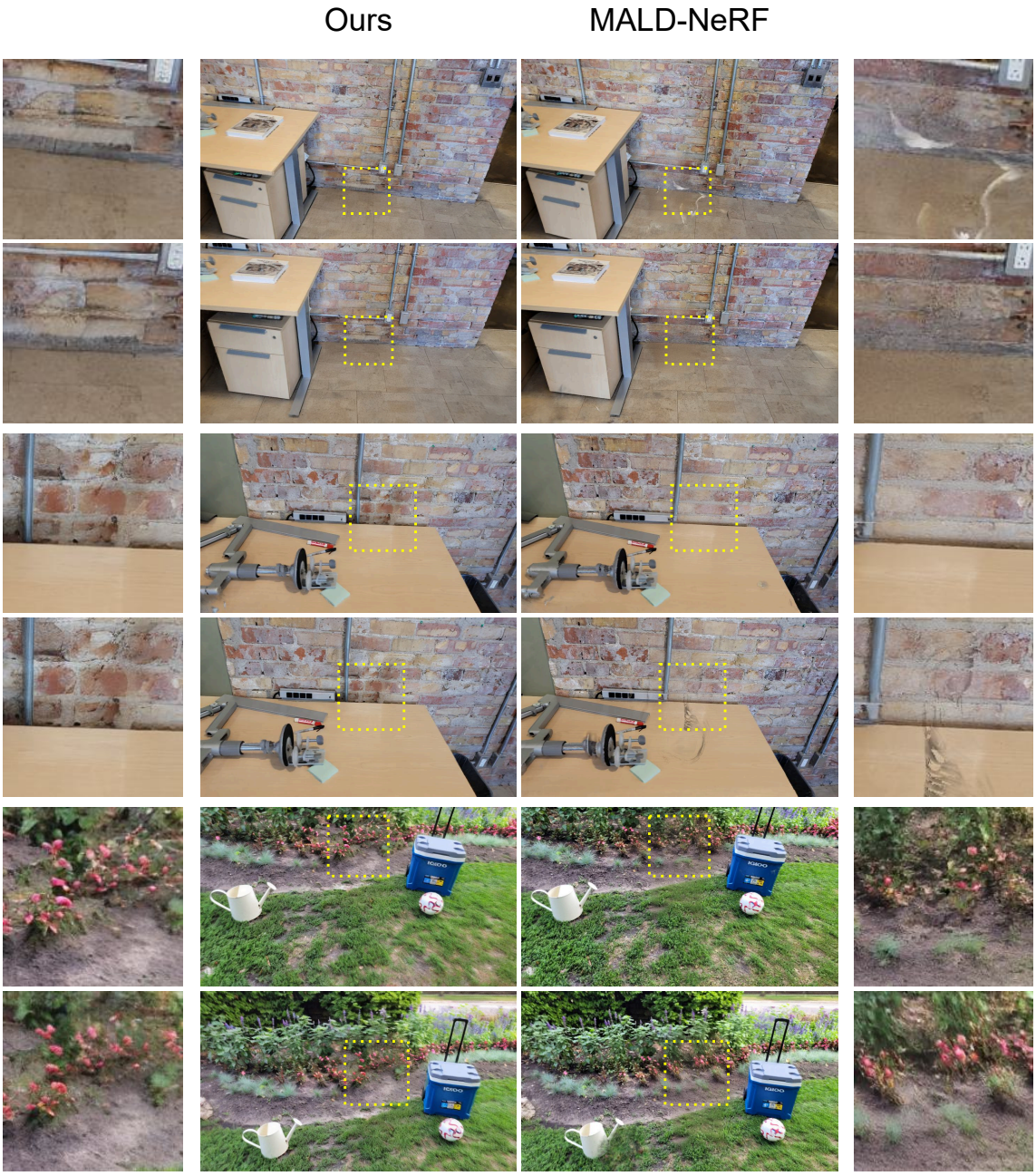

Figure A.1: Qualitative comparison of our method with MALD-NeRF [35]. Each inpainted view is accompanied by a zoomed-in version on the sides, highlighting inconsistencies.