Omni-3DEdit: Generalized Versatile 3D Editing in One-Pass

Liyi Chen*, Pengfei Wang*, Guowen Zhang, Zhiyuan Ma, Lei Zhang†
The Hong Kong Polytechnic University
liyi0308.chen@connect.polyu.hk, cslzhang@comp.polyu.edu.hk
Code: https://github.com/mt-cly/Omni3DEdit

Abstract

Most instruction-driven 3D editing methods rely on 2D models to guide the explicit and iterative optimization of 3D representations. This paradigm, however, suffers from two primary drawbacks. First, it lacks a universal design of different 3D editing tasks because the explicit manipulation of 3D geometry necessitates task-dependent rules, e.g., 3D appearance editing demands inherent source 3D geometry, while 3D removal alters source geometry. Second, the iterative optimization process is highly time-consuming, often requiring thousands of invocations of 2D/3D updating. We present Omni-3DEdit, a unified, learning-based model that generalizes various 3D editing tasks implicitly. One key challenge to achieve our goal is the scarcity of paired source-edited multi-view assets for training. To address this issue, we construct a data pipeline, synthesizing a relatively rich number of high-quality paired multi-view editing samples. Subsequently, we adapt the pre-trained generative model SEVA as our backbone by concatenating source view latents along with conditional tokens in sequence space. A dual-stream LoRA module is proposed to disentangle different view cues, largely enhancing our model's representational learning capability. As a learning-based model, our model is free of the time-consuming online optimization, and it can complete various 3D editing tasks in one forward pass, reducing the inference time from tens of minutes to approximately two minutes. Extensive experiments demonstrate the effectiveness and efficiency of Omni-3DEdit.

1. Introduction

Instruction-based 3D editing aims to edit a given 3D asset according to user's text prompt, encompassing tasks such as altering object appearances [19], adding new objects to specified locations [6], removing or replacing existing objects [38], *etc*. The predominant approach to this challenge involves leveraging 2D vision-language models to guide the iterative refinement of a 3D representation. However, a fundamental limitation of 2D models is their lack of multi-view consistency. Existing methods [7, 8, 34, 54, 55] rely heavily on an iterative 2D-3D-2D optimization loop to alleviate the inconsistencies arising from per-view edits. For instance, to achieve 3D appearance editing, Instruct-N2N [19], DGE [8], and ViCANeRF [16] repeatedly sample camera views to compute 2D gradients, updating a NeRF [37] or Gaussian [29] representation to preserve the original consistent geometry. In parallel, 3D removal methods [12, 13, 38] often warp foreground masks and employ 2D inpainting models to fill the missing regions, which require explicit 3D updates iteratively as well.

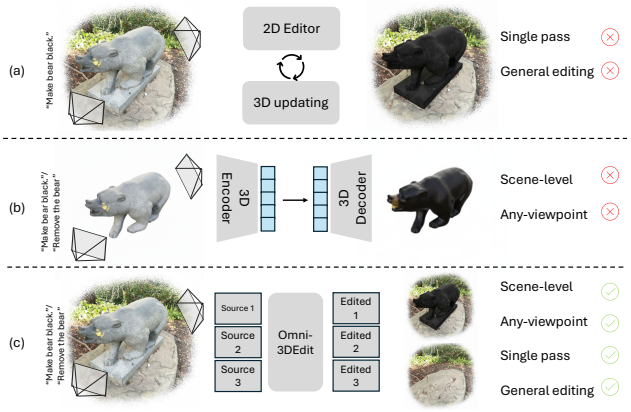

Figure 1. **Motivation of Omni-3DEdit.** (a) 3D editing via iterative 2D-3D-2D optimization with explicit 3D representation lacks generality and is time-consuming. (b) Performing 3D editing in latent space is hard to handle scene-level assets with arbitrary viewpoints. (c) Our Omni-3DEdit aims to solve these issues in multi-view space to perform fast, general, and consistent editing.

While these methods have achieved notable success in their respective domains, they suffer from two critical drawbacks. First, these approaches are task-specific and lack generality. Appearance editing is heavily reliant on the source 3D geometry, whereas object removal requires masks and often involves large-scale geometric deformations. It is difficult to design a universal iterative rule compatible with diverse editing tasks. Second, multi-round iter-

*Equal contribution. †Corresponding author.

ative process leads to excessive computation time and can over-smooth texture details. For example, InstructN2N [19] requires dozens of minutes for a single appearance edit.

We argue that maintaining and updating an explicit 3D representation, while ensuring consistency, is inherently ill-suited and slow for universal and rapid adaptation to various 3D editing commands. Although recent methods such as Tailor3D [40] and CMD [30] have explored editing in a 3D latent space [21, 64] to enable a single-pass unified framework, their models are fitted on object-centric datasets (*e.g.*, ObjectVerse [15]), limiting them to specific camera pose distributions and rendering them incapable of handling general 3D scene inputs.

Instead, in this paper, we introduce **Omni-3DEdit**, a novel framework that addresses the above-mentioned challenges by performing 3D editing directly in the multi-view latent space, as shown in Fig. 1. Our model accepts multi-view images of the original 3D asset from arbitrary viewpoints and an editing instruction, and outputs a set of consistently edited multi-view images. Compared to paradigms that operate in explicit 3D space or on object-level 3D latents, our Omni-3DEdit takes advantage of recent advancements in multi-view generation, 2D editing, and 3D reconstruction. Specifically, we first employ VGGT [51] to acquire camera cues for the input views, which is crucial for ensuring multi-view consistency. We then obtain the reference view by using the recent single-image editor Qwen-Image [56] to perform instruction-guided editing on a randomly selected source view. Subsequently, we introduce OmniNet, a model trained to propagate this edited view consistently across the other viewpoints. OmniNet takes the camera poses, the source multi-view images, and the reference image as input to synthesize remaining edited views. Finally, the resulting view set can be fed into a reconstruction model (*e.g.* AnySplat [25]) to obtain edited 3D asset.

To overcome the scarcity of large-scale paired training data for this task, we adopt a two-pronged strategy. First, we leverage the consistency priors of existing multi-view models to build an offline data synthesis pipeline [25, 56], generating paired training data for various tasks, including 3D removal, addition, and appearance editing. Second, we repurpose the pre-trained multi-view generative model SEVA [69] to reduce data dependency. A dual-stream LoRA [22] module, composing of *Geometry LoRA* and *Guidance LoRA*, is trained to encode the source and target views, preventing the model from disregarding the crucial geometric priors of the source asset.

In summary, our contributions are threefold. First, we propose Omni-3DEdit, a learning-based framework that operates in the multi-view latent space, enabling single-pass, efficient, and unified editing for diverse, scene-level 3D assets. Second, we explore and present effective strategies for training a multi-view consistent editing model *OmniNet* in data-constrained scenarios. Third, extensive experiments demonstrate the superior efficiency and effectiveness of our proposed method in various 3D editing tasks.

2. Related Work

3D Editing in 3D Representation Space. Early works aimed to achieve instruction-driven 3D editing by coupling existing 2D multi-modal generation or editing models with 3D representations such as NeRF [37] or Gaussian Splatting [29]. On one hand, the 2D multi-modal models provide a robust text comprehension interface and editing guidance, while the 3D representation ensures that the editing results adhere to 3D geometry. To ensure multi-view consistency, most methods iteratively evoke 2D models and 3D representations. InstructN2N [19], GaussianEditor [58] and the following works [49, 50, 54, 70] convey view-dependent denoising gradient into 3D representations, iterating thousands of times to achieve appearance editing. While object removing and inpainting methods [6, 11, 38] share a similar workflow based on 2D inpainters, these methods need to maintain a 3D representation during the editing process and update it in an optimization-based manner. In summary, this category of approaches lacks compatibility across different editing tasks and is highly time-consuming.

3D Editing in Object-level 3D Latent Space. Leveraging pre-trained object-level 3D generative models (*e.g.* Shape-E [27], LRM [21], and GS-LRM [64]), methods in [3, 9, 30, 40, 61] explore learning an editing mapping within the latent space of the 3D representation. Compared to maintaining an explicit 3D representation, the latent space is easier to be integrated into editing networks, making it possible to learn a direct mapping from the original 3D latent to the edited latent. However, these methods require view inputs from specific object-centered camera poses and they are constrained by the limitations of base 3D generation model, which can only handle background-free 3D objects [15]. As a result, these methods cannot process scene-level editing from arbitrary viewpoints.

3D Editing in Video/Multi-view Latent Space. Some methods use pre-trained video generation models [4, 20, 48, 60] to predict edited multi-views within the RGB space across consecutive frames. The generative video models enable the handling of scene-level 3D editing. DGE [8], EditCast3D [41], and V2Edit [65] introduce video editing for 3D editing. However, video generation models suffer from (1) a weak prior for 3D consistency, (2) the need for continuous viewpoint transformations, which is computationally expensive, and (3) a lack of understanding camera poses, resulting in suboptimal performance in both computational efficiency and editing quality. DiGA3D [39], Pro3D-Editor [68], and methods [1, 44, 47] share a similar idea of studying editing in multi-view latent space. They fall into the per-scene-optimization paradigm for specific

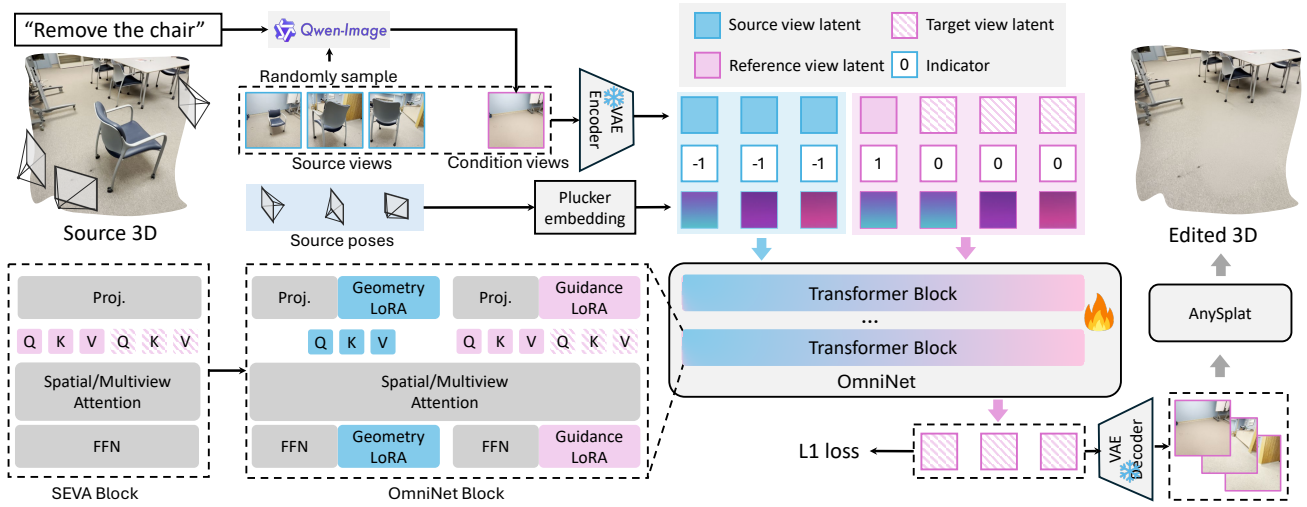

Figure 2. **Overview of Omni-3DEdit.** Given the instruction and multi-view images as inputs, we first employ Qwen-Image to obtain an edited reference image as condition view. Then an OmniNet is trained to map the editing cues from condition view to other views. The outputs of OmniNet are edited multi-view images, which can be used to obtain the edited 3D asset optionally.

editing tasks. Concurrent work Tinker [67] fails to tackle 3D editing that involves significant geometry changes, such as removal or addition. Instead, our Omni-3DEdit achieves unified and generalized 3D editing.

3. Method

Our Omni-3DEdit firstly leverages the 2D multimodal editor Qwen-Image [56] to perform instruction-based editing on an image from a randomly selected view, obtaining a *edited reference image* as the condition. Then, an *OmniNet* is introduced and trained to propagate the edited cues to the other source views. This paradigm not only takes advantage of the recent progress in 2D editing models but also reduces data and resource consumption, so that the OmniNet can focus exclusively on learning within the vision modality. In this section, we first describe the overview of Omni-3DEdit in Sec. 3.1 and then introduce the pipeline to generate paired training data in Sec. 3.2. The proposed dual-stream LoRA for training OmniNet is elaborated in Sec. 3.3.

3.1. Overview of Omni-3DEdit

As shown in Fig. 2, given N source view inputs $I_{src} = \{I^1_{src}, I^2_{src}, ..., I^N_{src}\}$ from the source 3D scene and the editing instruction prompt P, we first employ VGGT [51] to obtain their relative camera poses $p = \{p^1, p^2, ..., p^N\}$. Then, we randomly select a view from the input views and edit it using off-the-shelf Qwen-Image [56], obtaining a conditional image I_{cond} to provide editing cues. These images are fed into a VAE encoder [43] to produce the source view latents $s = \{s^1, s^2, ..., s^N\}$ and the condition view latent c.

During the training phase, noisy target latents $y_\sigma = \{y^1_\sigma, y^2_\sigma, ..., y^N_\sigma\}$ are obtained following EDM [28]:

$$y^n_\sigma = y^n + \sigma\epsilon, \tag{1}$$

where y^n, σ, and ϵ are the n-th clear target view latent, noise level, and random noise, respectively. We concatenate the triplet latents s, c, and y_σ in sequence space to avoid introducing an extra module, taking full advantage of the pre-trained prior for understanding geometry relations among views. To distinguish the different latents, s, c, and y_σ receive -1, 1, and 0 indicators in feature space, respectively. Besides, to supplement the perspective geometry relations among views, p of source views are converted into Plücker embeddings and conveyed to the condition view and noisy target views in the feature space. These input cues are fed into OmniNet $f(\cdot)$ to perform sample prediction. Similar to the training paradigm of SEVA [69], the loss is calculated only for the latents of target views, as shown below:

$$\mathcal{L} = \mathbb{E}\left[\|(f(y_\sigma, s, c, \sigma) - y\|^2_2\right]. \tag{2}$$

During the inference stage, the edited views can be interpolated by feeding the denoised target latent into VAE decoder. AnySplat [25] is optional to obtain edited 3D assets based on edited multi-view in seconds.

Note that Omni-3DEdit makes no assumptions about task priors. Instead, the model needs to implicitly learn to propagate the edit content solely based on the relationship between the reference and source views, thereby ensuring compatibility with versatile tasks.

3.2. Paired Training Data Generation

To drive OmniNet training, we require a dataset of paired 3D multi-view images before and after editing, which

Figure 3. **Data Construction Pipeline.** The original multi-view images are passed through a four-stage pipeline to obtain their paired multi-view counterparts after editing. The pipeline covers tasks of 3D removal, addition, and appearance editing.

should cover diverse scenes and various editing tasks. Given the scarcity of publicly available large-scale scene-level editing datasets, we focus on three common editing categories: 3D addition, 3D removal, and appearance editing. Leveraging existing open 3D multi-view datasets, we establish a data pipeline that integrates existing tools to batch-generate the desired samples for training. Our key insight lies in the observation that both per-view 3D removal and appearance editing usually involve slight multi-view texture inconsistency, which can be alleviated via consistent refinement, while 3D addition data can be obtained by inverting the source and edited views of 3D removal data.

3D Removal. The pipeline is illustrated in Fig. 3, which includes 4 steps. (1) *Instruction gneration.* Given multiple source views, we first employ Gemini-2.5pro [14] to analyze and identify an ideal object for deletion, and concurrently generate a textual editing instruction. Suitable candidates are defined as objects that have clear boundaries, are not truncated, and remain consistently visible across all views. (2) *Per-image editing.* We utilize Qwen-Image [56] to perform per-view foreground removal with generated instructions. (3) *Consistency refinement.* To alleviate inconsistencies introduced by per-view editing (*e.g.*, object-removed background regions exhibit disparate textures or colors), inspired by SDEdit [36], we introduce light-intensity (20%) noise to all edited views and then denoise them using the pre-trained SEVA [69]. (4) *Quality filter.* Due to the inherent stochasticity of 2D editing models and the varying difficulty of editing certain objects, which can lead to undesirable results, we implement an additional quality assessment to filter out undesired or failed samples.

3D Appearance Editing. Similar to the 3D removal task, appearance editing presents challenges in maintaining visual consistency across views when they are edited frame-by-frame. These inconsistencies manifest with variations in texture and color. Fortunately, such issues can be effectively addressed through the consistency refinement step in our 3D removal data pipeline.

Table 1. **Statistics of paired edited multi-views** in our curated training set, categorized by dataset and edit type.

Editing Type	CO3Dv2 [42]	DL3DV [31]	WildRGB-D [59]
3D Addition	15K	7K	6K
3D Removal	15K	6K	6K
3D Appearance Editing	13K	5K	5K

3D Addition. In contrast to 3D removal and appearance editing, 3D addition introduces severe geometric inconsistencies across the edited views, which cannot be effectively alleviated through consistency refinement. Therefore, we adopt a reverse strategy for data generation. Specifically, we utilize the original 3D multi-view images as the target views, and subsequently use the outputs from our 3D removal data pipeline as the corresponding source views. This methodology ensures that the ground-truth target views are inherently multi-view consistent. This strategy is similar to VIVID-10M [23], but we do not need extra masks.

Statistics of Constructed Training Pairs. With the training data generation pipeline, We construct training data based on three off-the-shelf multi-view datasets: CO3Dv2 [42], DL3DV [31], and WildRGB-D [59], to cover a diverse range of 3D scenes from both indoor and outdoor. For each dataset, we begin by uniformly sampling scenes across different categories. From each selected scene, we randomly sample 20 images as training views. The final number of paired images in our training set is shown in Tab. 1. Note that due to the high complexity of the scenes in DL3DV and WildRGB-D, relatively fewer samples passed this quality filtering, resulting in fewer training pairs from

Table 2. **Quantitative comparison (PSNR ↑ / LPIPS ↓) of 360° 3D removal methods on the 360-USID dataset.**

Methods	Carton	Cone	Cookie	Newcone	Plant	Skateboard	Sunflower	Average
SPIn-NeRF [38]	16.659 / 0.539	15.438 / 0.389	11.879 / 0.521	17.131 / 0.519	16.850 / 0.401	15.645 / 0.675	23.538 / 0.206	16.734 / 0.464
2DGS + LaMa [24, 46]	16.433 / 0.499	15.591 / 0.351	11.711 / 0.538	16.598 / 0.670	14.491 / 0.564	15.520 / 0.639	23.024 / 0.194	16.195 / 0.494
2DGS + LeftRefill [5, 24]	15.157 / 0.567	16.143 / 0.372	12.458 / 0.526	16.717 / 0.677	12.856 / 0.666	16.429 / 0.634	24.216 / 0.181	16.282 / 0.518
LeftRefill [5]	14.667 / 0.560	14.933 / 0.380	11.148 / 0.519	16.264 / 0.448	16.183 / 0.463	14.912 / 0.572	18.851 / 0.331	15.280 / 0.468
Gaussian Grouping [62]	16.695 / 0.502	14.549 / 0.366	11.564 / 0.731	16.745 / 0.533	16.175 / 0.440	16.002 / 0.577	20.787 / 0.209	16.074 / 0.480
GScream [53]	14.687 / 0.587	14.655 / 0.476	12.733 / 0.429	13.662 / 0.605	16.238 / 0.437	12.941 / 0.626	18.470 / 0.436	14.758 / 0.514
Infusion [32]	14.191 / 0.555	14.163 / 0.439	12.051 / 0.486	9.562 / 0.624	16.127 / 0.406	13.624 / 0.638	21.195 / 0.238	14.416 / 0.484
Aurafusion360 [57]	17.675 / 0.473	15.626 / 0.332	12.841 / 0.434	17.536 / 0.426	18.001 / 0.322	17.007 / 0.559	24.943 / 0.173	17.661 / **0.388**
Omni-3DEdit (Ours)	18.578 / 0.452	16.296 / 0.360	13.183 / 0.465	17.829 / 0.336	17.598 / 0.411	16.764 / 0.521	23.801 / 0.224	**17.722** / 0.395

these sources. During the training phase, we sample the editing tasks and scenes uniformly.

3.3. Dual-stream LoRA

With the constructed paired data, we repurpose SEVA as our editing model OmniNet, and finetune it to achieve multi-view editing by receiving additional source view latents s alongside the reference (condition) latent c and target view latents y_σ. To incorporate source view cues, there are two typical manners of feature-space concatenation and sequence-space concatenation, which have been adopted in previous novel view generation studies [2, 63]. However, we observe a significant performance degradation for these two architectures. As shown in Fig. 7, repurposed SEVA not only loses generative capability in target regions but also fails to preserve the unedited context from the source view.

We attribute this phenomenon to the use of shared projection layer for processing functionally distinct inputs. First, the source views and condition view serve different purposes. The condition view provides a precise editing signal from a specific perspective, whereas the source view provides comprehensive original context and texture information across camera poses. Forcing OmniNet to process these functionally dissimilar latents with shared layers introduces a learning conflict. Second, the shared weights lack the ability to differentiate bias signals across blocks, so that the model has to distinguish source and condition latents by mapping them to different feature spaces. This mechanism is ineffective in helping the target latent to correctly identify and utilize the source latent features.

Therefore, as shown in Fig. 2, we modify the architecture of SEVA to maintain two distinct sets of parameters within each block to individually encode the source latent and the condition latent. Specifically, OmniNet is based on the pre-trained linear layers of SEVA and introduces a dual-stream LoRA [22] module, including a geometry LoRA to process s to capture geometry priors among source views, and a guidance LoRA to propagate editing guidance from c to y_σ. The features from dual streams will exchange geometry cues and editing guidance in shared multi-view attention layers. This distengled mechanism not only enables OmniNet to learn specialized representations from different views but also introduces a crucial inductive bias, which ensures that the target latent can correctly identify and attend to the features from both view latents.

Discussion. Note that compared to MM-DiT [17], our proposed dual-stream LoRA has two notable distinctions. First, MM-DiT maintains two independent sets of full parameters, but our method utilizes parameter-efficient LoRA modules. This allows OmniNet to leverage the priors of SEVA without full-scale duplication. Second, MM-DiT is designed to handle inputs from different modalities (*e.g.*, text and image). In contrast, we prove that such a dual-stream paradigm is effective for inputs of the same modality (*i.e.*, vision latents) that serve distinct roles.

4. Experiment

4.1. Experiment Setup

Implementation Details. For preprocessing, we follow the pipeline from SEVA [69], normalizing cameras within the same scene to the coordinate range of $[-2, 2]$ and setting $N = 10$. The LoRA rank is set to 8. We train OmniNet for 4,000 iterations with batch size of 32, distributed across 16 NVIDIA H20 GPUs. The number of multi-view denoising steps is set to 50. All images are processed at a resolution of 576×576. We utilize the AdamW [33] optimizer with a constant learning rate of 1×10^{-4} with Eps-weighting MSE loss. We follow SEVA [69] to use SNR shift.

Evaluation Datasets. While Omni-3Dedit can tackle Versatile tasks in an unified manner, we compare it with specific methods respectively. For 3D removal, we utilize the unbounded 360 scene dataset, 360-USID [57], which contains seven 360-degree scenes, comprising three indoor and four outdoor environments. For 3D addition, we follow MVInpainter [6] to study the NVS performance on CO3Dv2 [42] validation set with sampling one scene per object. For 3D appearance editing, we omit numerical results due to the lack of publicly available benchmarks. Instead, we collect a series of complex 3D editing cases involving multi-round editing or significant geometry changes to compare the performance of different methods.

Metrics. We use PSNR, and LPIPS as our evaluation metrics on 3D removal and addition. Following the protocol in prior work [38], we compute these metrics only within the object mask to more accurately evaluate the result of

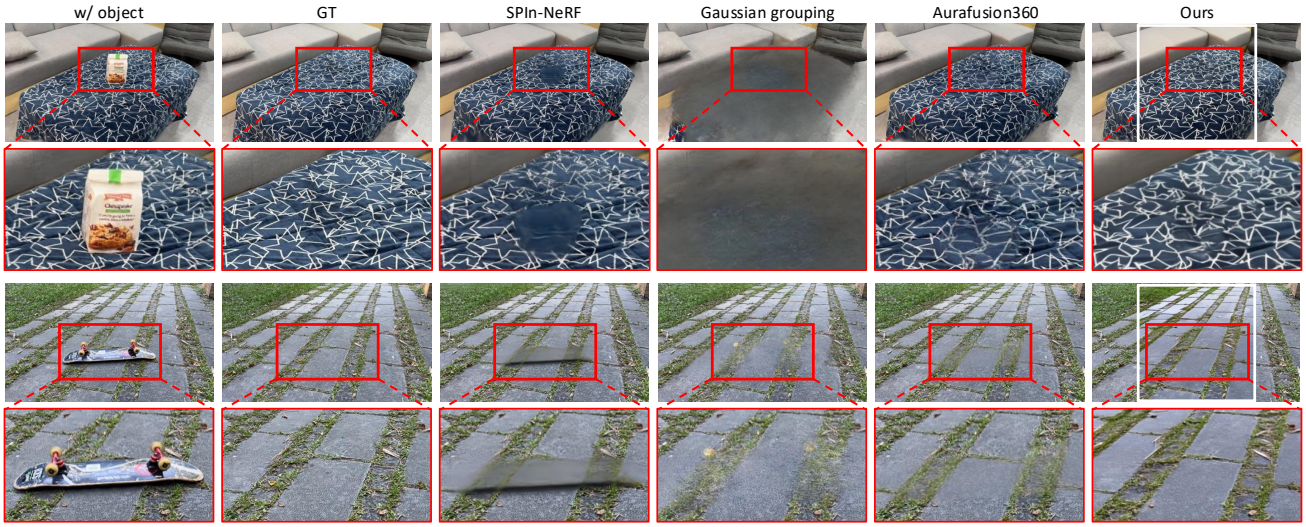

Figure 4. **Qualitative comparisons to 3D removal methods.** Our Omni-3DEdit not only removes the specific object completely but also presents rich details in the removed regions compared to other methods. We center crop views to adapt OmniNet resolution (white box).

3D removal. Besides, we introduce CLIP text-image score, CLIP directional score to study the editing quality on our curated test set. In addition, we leverage mLLM Gemini-2.5pro [14] to conduct a more comprehensive evaluation on the 3D editing quality.

4.2. Experimental Results

3D Removal. Omni-3DEdit is directly applicable to the 3D removal task. By using a 2D editor to perform object erasure on an arbitrary single view, we acquire a reference anchor view. OmniNet can then generate all remaining views. Note that our method operates in a mask-free manner, in contrast to prior works such as MVinpainter [6], SPIN-NeRF [38], and Aurafusion360 [57], which need multi-view object masks to localize the target regions.

We first conduct a quantitative evaluation on the 360-USID dataset [57], comparing our method against specialized 3D removal baselines, including 2DGS [24] + LeftRefill [5], GScream [53], and SPIn-NeRF [38]. As demonstrated in Tab. 2, our method achieves superior 3D removal performance. Compared to Aurafusion360 [57], our approach achieves an advantage in PSNR and costs much less time (2min $v.s.$ 30min) since Omni-3DEdit is free of iterative warping and obtains edited multi-view in a single pass.

We then provide qualitative comparisons in Fig. 4 to illustrate the superiority of our approach. One can observe that compared to Gaussian Grouping [62], Omni-3DEdit correctly identifies the target object for removal without corrupting the content of adjacent objects (*e.g.*, the desk). Furthermore, regarding the visual quality of object-removed regions, Aurafusion360 [57] exhibits significant artifacts and residual contours at the object boundaries, while our method demonstrates a clear advantage in maintaining high-fidelity and consistent details.

Table 3. **Quantitative comparison on CO3Dv2 *val* set.**

Method	PSNR ↑	LPIPS ↓	CLIP-T ↑
ZeroNVS [45]	14.56	0.716	0.196
MVInpainter [6]	19.20	0.344	0.271
Omni-3DEdit (Ours)	**20.67**	**0.278**	**0.277**

3D Addition. To validate the model's capability for 3D object addition, we follow the evaluation methodology established by MVInpainter [6], utilizing the multi-view images from the CO3Dv2 [42] validation set. For each scene, we retain an arbitrary view as the reference image, while the foreground objects in all remaining views are erased and inpainted via Qwen-Image. OmniNet is employed to generate these erased objects conditioned on the reference image, where the target object visible. This experimental setup is to investigate the novel view synthesis (NVS) capability in the context of object addition. As shown in Tab. 3, ZeroNVS [45] fails to fully take the context from source views and generates target views based on the single reference view, achieving the worst performance. MVInpainter [6] heavily relies on complex pre-processing (*e.g.*, point matching, mask propagation), limiting its generalization ability and achieving sub-optimal performance. Our Omni-3DEdit learns the mapping from source views to target edited views and outperforms MVInpainter in synthesized novel view quality. Omni-3DEdit is built upon SEVA, thereby inheriting its original ability for consistent generation under specified camera poses.

3D Appearance Editing. We further demonstrate the applicability of our method to 3D appearance editing. Despite relying on a single reference image that often offers limited-

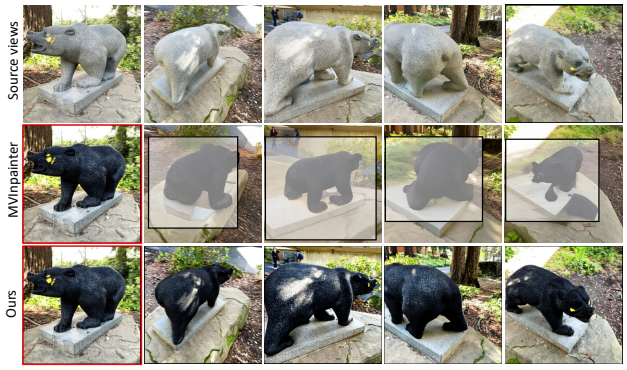

Figure 5. **Comparison of 3D appearance editing.** Red box represents the reference view. White boxes are object masks as additional inputs for MVInpainter.

view guidance, Omni-3DEdit implicitly captures instance-level geometric priors, allowing it to effectively propagate editing signals to other regions unobserved in the reference view. We illustrate this capability with showcase of *"Make bear black."* [19], a scene out of our training data. As shown in Fig. 5, while the reference image solely presents the left perspective, Omni-3DEdit successfully propagates the editing guidance to the entire bear instance, including its rear view. In contrast, prior reference-based 3D editing methods often struggle with 360° appearance editing. This limitation stems from their either heavy reliance on depth warping [52], which fails to handle accumulated errors across large viewpoint changes [18, 35], or their need for explicit masks for instance identification, resulting in the corruption of original geometric information [6, 35]. As demonstrated in the second row, MVinpainter fails to preserve the original geometry, yielding unsatisfactory results.

Complex 3D Editing. Our method supports fundamental operations, including 3D removal, 3D addition, and appearance editing. By combining them, we can achieve more complex editing tasks such as replacement and multi-turn editing. We collect a test benchmark to study the performance of Omni-3DEdit on such tasks.

As shown in Tab. 4, Omni-3DEdit presents significant advantages over previous methods in terms of both Gemini score and time cost. This is mainly due to fact that previous studies can only tackle specific editing tasks such as appearance editing. In addition, they heavily rely on iteratively evoking the 2D editor and explicit 3D representations, resulting in long convergence.

We present a showcase of *"Removing the book."* then *"Adding an apple to desk."* in Fig. 6. DGE [8] fails to achieve clear 3D editing because it relies on the source geometry to find pixel correspondences. Similar failures occur for GaussianEditor [10], although it is equipped with the more powerful editor, Nano-banana [14]. Since each edit might place the apple in a different position on the table, this makes it difficult for the explicit 3D Gaussian to converge, resulting in obvious artifacts. In Nano-banana, we concatenate the source views into a single image, utilizing its in-context capability [66] to edit multi-view at once, thereby improving their 3D consistency. However, the apples in edited views still suffer from inconsistent scale and position. In comparison, Omni-3DEdit maintains high consistency throughout this two-stage editing (removal results are shown in the right-bottom of last column), as evidenced by the coherence of wall tile textures and apple details.

Table 4. **Comparison of methods on complex 3D editing.**

Method	CLIP-T/I	CLIP-Dir.	Gemini score	Time
DGE [8]	0.246	0.132	1.7	5min
GaussianEditor [10]	0.253	0.146	2.0	17min
ViCANeRF [16]	0.257	0.141	2.2	28min
Nano-banana [14]	0.281	0.165	3.8	-
Omni-3DEdit (Ours)	**0.286**	**0.170**	**4.0**	**2min**

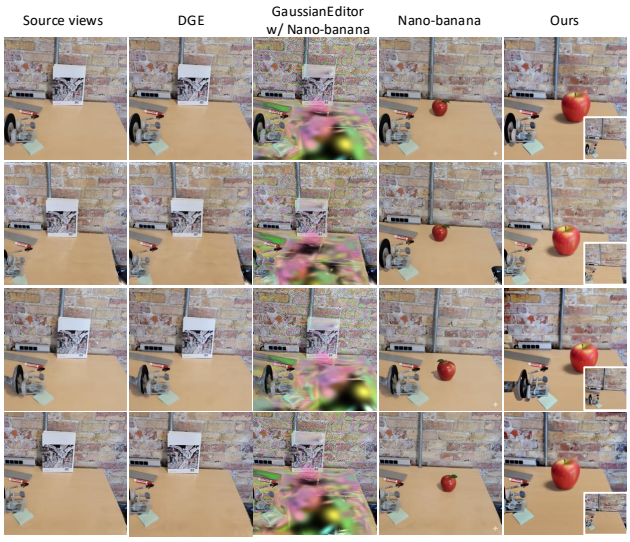

Figure 6. **Results of competing 3D editing methods for a complex task *"Removing the book." then "Adding an apple to desk."***

4.3. Ablation Study

Architecture. We first investigate the impact of architectural choices on model performance by comparing our OmniNet, which is builtg upon sequence concatenation with dual-stream LoRA, with three distinct approaches that incorporate original images. (a) *SEVA zeroshot*: dropping the source view and only feeding the reference view and Gaussian noises to pre-trained SEVA to obtain generated views under given camera poses. (b) *Feature-space concatenation* [63]: concatenating each Gaussian noise with its corresponding source view latent along the channel dimension, followed by feature fusion via a lightweight projection network. (c) *Sequence-space concatenation* [2, 26]: concatenating the source view latent, condition view latent, and Gaussian noises along the sequence dimension, they share

Figure 7. **Ablation study of different architectures on performing multiview editing with edited reference view.** Both plain feature-space concatenation and sequence-space concatenation fail to achieve desired editing results, while dual-stream LoRA improves the editing quality significantly.

the same linear layers in each block.

We provide visualization comparisons over three editing showcases provided in Fig. 7. One can observe that SEVA zero-shot fails to align source view poses since the reliance on single conditional view and normalized camera poses makes SEVA scale-agnostic during the generation process. Feature-space concatenation tends to produce obvious artifacts with blurred details. We suspect it is caused by the fact that the light convolution layers are difficult to fuse cues from source views and target views, and the cues from source view latents are vanishing when passing through the network. A similar phenomenon of bypassing source view information is also observed in sequence-space concatenation. This reveals the difficulty for pre-trained parameters to simultaneously encode both source and target views, which provide distinct information. By introducing dual-stream LoRA, OmniNet brings performance improvement by capturing geometry cues and editing guidance from source and condition views via decoupled LoRA parameters.

Input Signal. Furthermore, we investigate the critical importance of input signals for OmniNet. We conduct ablation studies by dropping the indicator and camera pose to observe their respective impacts on model performance. Experiments on the 360-USID benchmark [57], as presented in Tab. 5, reveal that performance degrades drastically without the indicator due to the lack of explicit signals distinguishing different views. Similarly, excluding camera poses leads to a significant performance drop, since relying solely on appearance to analyze perspective geometric information is too implicit for the model to effectively comprehend.

Table 5. **Ablation study on 360-USID benchmark.**

Settings	SSIM ↑	PSNR ↑	LPIPS ↓
Omni-3DEdit	0.925	17.72	0.395
SEVA zeroshot	0.911	13.99	0.575
Omni-3DEdit w/o indicator	0.917	15.20	0.545
Omni-3DEdit w/o pose	0.903	14.54	0.565

5. Conclusion

In this paper, we proposed Omni-3DEdit, a unified and generalized model capable of handling 3D removal, addition, and appearance editing without relying on additional masks or point matching signals. Specifically, we constructed high-quality paired edited multi-view data across different editing tasks and introduced a dual-stream LoRA module to repurpose the pre-trained multi-view generation model SEVA into a multi-view editing model, OmniNet. Extensive experiments demonstrated that our Omni-3DEdit performs significantly better than existing schemes, showing strong generalization performance with much faster speed.

Limitations. Due to the scarcity of open-source scene-level multi-view data and computational resource constraints, the scale (0.1M) of our constructed dataset is relatively small, which limits Omni-3DEdit's ability to handle very fine-grained editing tasks (*e.g.*, *"adding a bracelet to a human wrist"*). A potential solution is to develop a more sophisticated data construction pipeline to generate a larger corpus of training data. We leave this as our future work.

References

[1] Hadi Alzayer, Yunzhi Zhang, Chen Geng, Jia-Bin Huang, and Jiajun Wu. Coupled diffusion sampling for training-free multi-view image editing. *arXiv preprint arXiv:2510.14981*, 2025. 2

[2] Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, et al. Recammaster: Camera-controlled generative rendering from a single video. *arXiv preprint arXiv:2503.11647*, 2025. 5, 7

[3] Roi Bar-On, Dana Cohen-Bar, and Daniel Cohen-Or. Edit23: 3d editing via propagation of image prompts to multi-view. *arXiv preprint arXiv:2506.20652*, 2025. 2

[4] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 2

[5] Chenjie Cao, Yunuo Cai, Qiaole Dong, Yikai Wang, and Yanwei Fu. Leftrefill: Filling right canvas based on left reference through generalized text-to-image diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 5, 6

[6] Chenjie Cao, Chaohui Yu, Fan Wang, Xiangyang Xue, and Yanwei Fu. Mvinpainter: Learning multi-view consistent inpainting to bridge 2d and 3d editing. *arXiv preprint arXiv:2408.08000*, 2024. 1, 2, 5, 6, 7

[7] Jun-Kun Chen, Samuel Rota Bulò, Norman Müller, Lorenzo Porzi, Peter Kontschieder, and Yu-Xiong Wang. Consistdreamer: 3d-consistent 2d diffusion for high-fidelity scene editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21071–21080, 2024. 1

[8] Minghao Chen, Iro Laina, and Andrea Vedaldi. Dge: Direct gaussian 3d editing by consistent multi-view editing. In *European Conference on Computer Vision*, pages 74–92. Springer, 2024. 1, 2, 7

[9] Minghao Chen, Junyu Xie, Iro Laina, and Andrea Vedaldi. Shap-editor: Instruction-guided latent 3d editing in seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26456–26466, 2024. 2

[10] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21476–21485, 2024. 7

[11] Yuxin Cheng, Binxiao Huang, Taiqiang Wu, Wenyong Zhou, Chenchen Ding, Zhengwu Liu, Graziano Chesi, and Ngai Wong. Perspective-aware 3d gaussian inpainting with multi-view consistency. *arXiv preprint arXiv:2510.10993*, 2025. 2

[12] Yuxin Cheng, Binxiao Huang, Taiqiang Wu, Wenyong Zhou, Chenchen Ding, Zhengwu Liu, Graziano Chesi, and Ngai Wong. Perspective-aware 3d gaussian inpainting with multi-view consistency. *arXiv preprint arXiv:2510.10993*, 2025. 1

[13] Yuxin Cheng, Binxiao Huang, Taiqiang Wu, Wenyong Zhou, Chenchen Ding, Zhengwu Liu, Graziano Chesi, and Ngai Wong. Perspective-aware 3d gaussian inpainting with multi-view consistency. *arXiv preprint arXiv:2510.10993*, 2025. 1

[14] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 4, 6, 7

[15] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13142–13153, 2023. 2

[16] Jiahua Dong and Yu-Xiong Wang. Vica-nerf: View-consistency-aware 3d editing of neural radiance fields. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 7

[17] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 5

[18] Hyojun Go, Byeongjun Park, Jiho Jang, Jin-Young Kim, Soonwoo Kwon, and Changick Kim. Splatflow: Multi-view rectified flow model for 3d gaussian splatting synthesis. *arXiv preprint arXiv:2411.16443*, 2024. 7

[19] Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19740–19750, 2023. 1, 2, 7

[20] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022. 2

[21] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023. 2

[22] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 2, 5

[23] Jiahao Hu, Tianxiong Zhong, Xuebo Wang, Boyuan Jiang, Xingye Tian, Fei Yang, Pengfei Wan, and Di Zhang. Vivid-10m: A dataset and baseline for versatile and interactive video local editing. *arXiv preprint arXiv:2411.15260*, 2024. 4

[24] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accu-

rate radiance fields. In *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. 5, 6

[25] Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *arXiv preprint arXiv:2505.23716*, 2025. 2, 3

[26] Xuan Ju, Weicai Ye, Quande Liu, Qiulin Wang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, and Qiang Xu. Fulldit: Multi-task video generative foundation model with full attention. *arXiv preprint arXiv:2503.19907*, 2025. 7

[27] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 2

[28] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 3

[29] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2

[30] Peng Li, Suizhi Ma, Jialiang Chen, Yuan Liu, Congyi Zhang, Wei Xue, Wenhan Luo, Alla Sheffer, Wenping Wang, and Yike Guo. Cmd: Controllable multiview diffusion for 3d editing and progressive generation. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–10, 2025. 2

[31] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024. 4

[32] 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. 5

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[34] Chaofan Luo, Donglin Di, Xun Yang, Yongjia Ma, Zhou Xue, Chen Wei, and Yebin Liu. Trame: Trajectory-anchored multi-view editing for text-guided 3d gaussian splatting manipulation. *arXiv preprint arXiv:2407.02034*, 2024. 1

[35] Baorui Ma, Huachen Gao, Haoge Deng, Zhengxiong Luo, Tiejun Huang, Lulu Tang, and Xinlong Wang. You see it, you got it: Learning 3d creation on pose-free videos at scale. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2016–2029, 2025. 7

[36] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 4

[37] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[38] 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 *CVPR*, 2023. 1, 2, 5, 6

[39] Jingyi Pan, Dan Xu, and Qiong Luo. Diga3d: Coarse-to-fine diffusional propagation of geometry and appearance for versatile 3d inpainting. *arXiv preprint arXiv:2507.00429*, 2025. 2

[40] Zhangyang Qi, Yunhan Yang, Mengchen Zhang, Long Xing, Xiaoyang Wu, Tong Wu, Dahua Lin, Xihui Liu, Jiaqi Wang, and Hengshuang Zhao. Tailor3d: Customized 3d assets editing and generation with dual-side images, 2024. 2

[41] Huaizhi Qu, Ruichen Zhang, Shuqing Luo, Luchao Qi, Zhihao Zhang, Xiaoming Liu, Roni Sengupta, and Tianlong Chen. Editcast3d: Single-frame-guided 3d editing with video propagation and view selection. *arXiv preprint arXiv:2510.13652*, 2025. 2

[42] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *International Conference on Computer Vision*, 2021. 4, 5, 6

[43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 3

[44] Ahmad Salimi, Tristan Aumentado-Armstrong, Marcus A Brubaker, and Konstantinos G Derpanis. Geometry-aware diffusion models for multiview scene inpainting. *arXiv preprint arXiv:2502.13335*, 2025. 2

[45] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, et al. Zeronvs: Zero-shot 360-degree view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9420–9429, 2024. 6

[46] 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/CVF winter conference on applications of computer vision*, pages 2149–2159, 2022. 5

[47] Zeng Tao, Zheng Ding, Zeyuan Chen, Xiang Zhang, Leizhi Li, and Zhuowen Tu. C3editor: Achieving controllable consistency in 2d model for 3d editing. *arXiv preprint arXiv:2510.04539*, 2025. 2

[48] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang,

Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 2

[49] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3835–3844, 2022. 2

[50] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 2

[51] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025. 2, 3

[52] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5261–5271, 2025. 7

[53] Yuxin Wang, Qianyi Wu, Guofeng Zhang, and Dan Xu. Gscream: Learning 3d geometry and feature consistent gaussian splatting for object removal. In *ECCV*, 2024. 5, 6

[54] 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*, pages 404–420. Springer, 2024. 1, 2

[55] Minghao Wen, Shengjie Wu, Kangkan Wang, and Dong Liang. Intergsedit: Interactive 3d gaussian splatting editing with 3d geometry-consistent attention prior. *arXiv preprint arXiv:2507.04961*, 2025. 1

[56] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. 2, 3, 4

[57] Chung-Ho Wu, Yang-Jung Chen, Ying-Huan Chen, Jie-Ying Lee, Bo-Hsu Ke, Chun-Wei Tuan Mu, Yi-Chuan Huang, Chin-Yang Lin, Min-Hung Chen, Yen-Yu Lin, et al. Aurafusion360: Augmented unseen region alignment for reference-based 360deg unbounded scene inpainting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16366–16376, 2025. 5, 6, 8

[58] Jing Wu, Jia-Wang Bian, Xinghui Li, Guangrun Wang, Ian Reid, Philip Torr, and Victor Adrian Prisacariu. Gaussctrl: multi-view consistent text-driven 3d gaussian splatting editing. *arXiv preprint arXiv:2403.08733*, 2024. 2

[59] Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. Rgbd objects in the wild: Scaling real-world 3d object learning from rgb-d videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22378–22389, 2024. 4

[60] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 2

[61] Junliang Ye, Shenghao Xie, Ruowen Zhao, Zhengyi Wang, Hongyu Yan, Wenqiang Zu, Lei Ma, and Jun Zhu. Nano3d: A training-free approach for efficient 3d editing without masks. *arXiv preprint arXiv:2510.15019*, 2025. 2

[62] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, 2024. 5, 6

[63] 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. 5, 7

[64] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 1–19. Springer, 2024. 2

[65] Yanming Zhang, Jun-Kun Chen, Jipeng Lyu, and Yu-Xiong Wang. V2edit: Versatile video diffusion editor for videos and 3d scenes. *arXiv preprint arXiv:2503.10634*, 2025. 2

[66] Zechuan Zhang, Ji Xie, Yu Lu, Zongxin Yang, and Yi Yang. Enabling instructional image editing with in-context generation in large scale diffusion transformer. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*. 7

[67] Canyu Zhao, Xiaoman Li, Tianjian Feng, Zhiyue Zhao, Hao Chen, and Chunhua Shen. Tinker: Diffusion's gift to 3d–multi-view consistent editing from sparse inputs without per-scene optimization. *arXiv preprint arXiv:2508.14811*, 2025. 3

[68] Yang Zheng, Mengqi Huang, Nan Chen, and Zhendong Mao. Pro3d-editor: A progressive-views perspective for consistent and precise 3d editing. *arXiv preprint arXiv:2506.00512*, 2025. 2

[69] Jensen (Jinghao) Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models. *arXiv preprint arXiv:2503.14489*, 2025. 2, 3, 4, 5

[70] Jingyu Zhuang, Chen Wang, Liang Lin, Lingjie Liu, and Guanbin Li. Dreameditor: Text-driven 3d scene editing with neural fields. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–10, 2023. 2