

Edit3r: Instant 3D Scene Editing from Sparse Unposed Images

Jiageng Liu^{1*}, Weijie Lyu^{2*}, Xueting Li³, Yejie Guo⁴, and Ming-Hsuan Yang²

¹ University of California, Los Angeles, Los Angeles, CA, USA

² University of California, Merced, Merced, CA, USA

³ NVIDIA Research, Santa Clara, CA, USA

⁴ Shanghai Jiao Tong University, Shanghai, China

*Equal contribution.

Project Page: <https://edit3r.github.io/edit3r/>



Fig. 1: (a) System overview. *Edit3r* consumes multi-view inconsistent 2D edits and reconstructs a coherent 3D representation in ~ 0.5 seconds. Our framework supports numerous editing tasks, including object insertion, removal, and global stylization (e.g., color and atmospheric transitions), while maintaining cross-view consistency for novel view synthesis. *Prompt: Decorate room for party.* (b) Compared with other optimization-based methods with the same prompt, *Edit3r* provides more diverse and vivid edited scenes. (c) More edit examples. *Prompt: Add flower bed near bench.*

Abstract. We present *Edit3r*, a feed-forward framework that reconstructs and edits a 3D scene in a single pass. Unlike prior approaches that require per-scene optimization, *Edit3r* directly predicts instruction-aligned 3D Gaussians from unposed sparse-view images and a text prompt. A key challenge in training 3D editing models is the lack of view-consistent edited images for supervision, due to the stochastic nature of 2D image editing. We address this by reformulating editing as a recoloring pretext task, and show that a model trained on simple recoloring transfers effectively to 3D-consistent editing at inference time. Specifically, we develop (i) a SAM2-based recoloring pipeline that generates one recolored reference view along with reliable, cross-view-consistent supervision views,

and (ii) an asymmetric input strategy that pairs the recolored reference view with raw auxiliary views, encouraging the network to fuse and align disparate observations. At inference, it supports diverse 2D editors by resolving cross-view conflicts and yielding coherent 3D edits, without requiring multi-view edited supervision during training. For large-scale quantitative evaluation, we introduce ***DL3DV-Edit-Bench***. Comprehensive quantitative and qualitative results show that *Edit3r* achieves superior semantic alignment and enhanced 3D consistency compared to recent baselines, while operating at significantly higher inference speed, making it promising for real-time 3D editing applications.

Keywords: 3D Scene Editing · Feed-forward Reconstruction · Multi-view Consistency

1 Introduction

3D scene editing is a fundamental long-standing problem in computer vision. By offering intuitive control over content creation, it plays a crucial role in diverse applications such as visual effects [48], augmented and virtual reality (AR/VR) [25], game design [40], and digital twins [26]. Recent advancements in 3D scene representation and reconstruction [18, 28] have sparked a rapid surge of interest in editing 3D scenes using text prompts. Existing methods [13, 19, 38] typically combine neural reconstruction with text-driven image editing in a *reconstruct-edit-refit* pipeline. Given a collection of posed images and a textual instruction, these approaches first reconstruct the scene using a neural 3D representation, then apply 2D generative editing to the rendered images, and finally re-optimize the 3D representation to incorporate the edits. While this pipeline achieves impressive visual quality and alignment with instruction prompts, it is limited by slow per-scene optimization and retraining, which limits practical usability. Moreover, these methods often introduce multi-view inconsistencies, revealing a substantial gap between current 3D editing frameworks and the need for fast, interactive scene manipulation in real-world applications.

Conversely, the rapid development of Large Reconstruction Models (LRMs) [4, 5, 16, 37, 42, 46] offers a more efficient paradigm for 3D scene reconstruction by amortizing computation across large datasets and removing per-scene optimization. Notably, Gaussian-based LRMs enable rapid, photorealistic rendering suited for real-time applications, thanks to the Gaussian splatting technique.

Motivated by these advancements, we propose *Edit3r*, a feed-forward approach that reconstructs and edits a 3D scene in a single pass. Beginning with a pair of unposed views, we edit them using a 2D image editor. These potentially inconsistent edited views serve as inputs for *Edit3r*, which directly and efficiently predicts 3D Gaussian Splats aligned with the edited views while eliminating the inconsistency between edited input views, seamlessly unifying reconstruction and editing without the need for per-scene optimization. The semantic content is generated by flexible 2D editors, while the 3D backbone learns to fuse view-level edit proposals into a coherent Gaussian scene without per-scene optimization.

Training such a feed-forward reconstruction-editing model introduces two practical challenges. First, view-consistent ground-truth edited images for supervision are unavailable. Using view-consistent 2D edits for training could lead to unstable training and blurry artifacts. To address this, we propose a SAM2-based [33] recoloring for view-consistent supervision. We find that our model, though trained only on a recoloring task, effectively handles images edited by 2D editing models at inference. Second, edited inputs can remain semantically and visually inconsistent across views. To tackle this, we design an asymmetric input scheme that pairs a recolored reference view with other views in their original appearance, encouraging the network to fuse and align edited and unedited observations, providing clean 3D Gaussians that align with the reference view.

The lack of established benchmarks limits fair and scalable evaluation in 3D scene editing. Hence, we introduce **DL3DV-Edit-Bench**, a standardized benchmark built on the DL3DV [20] test split to systematically assess multi-view consistency and inference efficiency, setting a foundation for reproducible and rigorous comparisons. Our contributions are summarized as follows:

- We introduce *Edit3r*, a feed-forward framework that simultaneously reconstructs 3D scenes and generates edited Gaussians aligned to a text prompt.
- We develop a SAM2-based recoloring technique for view-consistent supervision during training and introduce an asymmetric input scheme to bolster robustness against cross-view inconsistencies.
- We present *DL3DV-Edit-Bench*, a scene-level, multi-view 3D editing benchmark built on the DL3DV test split.
- Extensive quantitative and qualitative experiments show *Edit3r* is significantly faster than previous approaches, achieving higher visual quality and semantic consistency while demonstrating practical real-world potential.

2 Related Work

2D Image and Video Editing. Recent diffusion-based editors enable high-quality, instruction-conditioned edits through techniques such as noise-to-image guidance [27], cross-attention manipulation [14], instruction tuning [3], and other structure-conditional control [47]. While effective for single images, naively applying them per frame leads to flicker and identity drift.

To address temporal coherence, video extensions employ attention/token propagation [11], edit-specific attention alignment [21, 31], optical-flow guidance [41], and implicit canonicalization [30]. Despite strong short-range coherence, these approaches remain fundamentally 2D and lack explicit 3D scene reasoning, leading to view-inconsistent artifacts when lifted to 3D supervision.

3D Scene Editing. Neural reconstruction methods such as NeRF [28] and 3D Gaussian Splatting [18] enable photorealistic rendering, motivating integration with semantic editing. Existing approaches [6, 13, 23, 38, 43] typically follow a reconstruct-edit-refit pipeline. Instruct-NeRF2NeRF [13] reconstructs scenes with NeRF, edits multi-view renderings using InstructPix2Pix [3], and

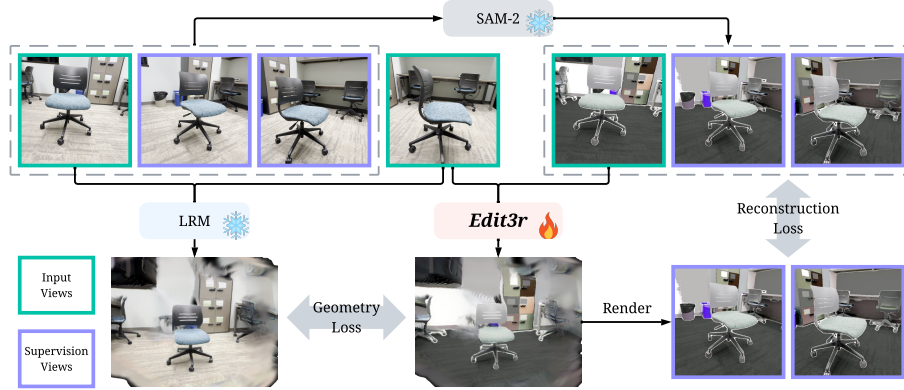


Fig. 2: Training pipeline of *Edit3r*. Green-framed images are input views and purple-framed images are supervision views (ground-truth and rendered results). We first apply SAM2-based recoloring to both inputs and supervision views, then feed an asymmetric pair (one recolored view and one original view) into *Edit3r* to predict a 3D Gaussian scene, which is rendered and supervised with reconstruction losses against the recolored supervision views (right). In parallel, a frozen LRM reconstructs the scene from the original inputs and provides 3D supervision via geometry losses that regularize *Edit3r*’s Gaussian predictions in 3D space.

re-optimizes the scene with edited images. GaussianCtrl [38] employs ControlNet [47] with depth conditioning and attention-based alignment for view consistency, while GaussianEditor [6] introduces Gaussian semantic tracing for precise editing control. Though achieving impressive quality, these methods require slow per-scene optimization and are prone to multi-view inconsistencies.

Generalizable Feed-forward Reconstructors. Large reconstruction models (LRMs) [4, 5, 24, 34, 35, 44, 46] have revolutionized 3D reconstruction by learning generalizable priors from large-scale datasets like Objaverse [9]. Leveraging transformer architectures [10, 36], these models enable rapid inference of NeRF or Gaussian representations from sparse views without per-scene optimization. Methods such as PixelSplat [4] utilize epipolar geometry, while many other methods [5, 7, 8, 39] construct cost volumes for multi-view aggregation. Despite their efficiency in reconstruction, the application of feed-forward reconstructors to 3D scene editing remains largely unexplored, which our work addresses.

3 Method

We propose a novel and efficient feed-forward framework for 3D scene editing, shown in Fig. 2. Given multi-view images and a text prompt, our method first applies 2D edits [2, 3, 29] to the input images, and then reconstructs a 3D scene that is aligned with the text prompt, in a single forward pass, even when the edited images introduce multi-view inconsistencies. In contrast to optimization-based approaches [6, 13, 38], our pipeline enables real-time, instruction-guided editing by leveraging the efficient reconstruction capabilities of LRMs. Furthermore, *Edit3r* is highly modular and can seamlessly integrate other 2D image

editing techniques, such as Instruct-Pix2Pix [3] and FLUX [2], thereby substantially enhancing the diversity and flexibility of the generated results.

3.1 Preliminaries

Problem Setup. Given unposed images with corresponding camera intrinsics $\{(I_v, k_v)\}_{v=0}^{V-1}$, an editing text prompt T , where V is the number of input views, our goal is to reconstruct a geometrically consistent 3D scene S_T that is semantically aligned with T .

During training, our pipeline takes one recolored image (I'_0, k_0) and another original image (I_1, k_1) from the unposed sequence as inputs to a feed-forward reconstruction model f_θ with learnable parameters θ . Instead of explicitly estimating camera poses or meshes, we approach reconstruction by lifting the unposed images into a canonical 3D Gaussian scene. Given views with unknown extrinsics, the network predicts a set of anisotropic Gaussian primitives in a fixed world frame. Each primitive encodes both geometry (location and shape) and appearance attributes (radiance and opacity). Formally, we learn a function that maps image evidence to this representation:

$$\mathcal{G} = f_\theta(\{(I'_0, k_0), (I_1, k_1)\}),$$

where $\mathcal{G} = \{(\mu_j, \Sigma_j, c_j, \alpha_j)\}_{j=0}^{V-1}$ denotes the predicted set of 3D Gaussians. $\mu \in \mathbb{R}^3$ is the center, $\Sigma \in \mathbb{R}^{3 \times 3}$ is the covariance, c is a vector of spherical-harmonic color coefficients, and α is opacity.

Pose-Free 3D Scene Reconstruction. Our approach adopts the pose-free 3D reconstruction framework of NoPoSplat [42]. Given sparse unposed multi-view images, we embed the camera intrinsics for each view using a small MLP $\phi : \mathbb{R}^d \rightarrow \mathbb{R}^D$, and concatenate these embeddings with the image token sequences to obtain the input tokens z_v :

$$z_v = [\text{img_tokens}(I_v) \oplus \phi(k_v)].$$

A vision transformer (ViT) [10] encoder with shared weights processes each view independently, taking z_v as input and producing per-view features f_v in a unified feature space. The set of features $\{f_v\}_{v=0}^{V-1}$ is then fused using a ViT decoder that leverages both self-attention and cross-view attention, enabling the model to resolve occlusions and appearance variations between views. This decoder outputs fused features $\{f_{fused}\}_{v=0}^{V-1}$, which are subsequently passed to two lightweight Gaussian heads. Each Gaussian head contains two DPT-based [37] predictors: one predicts the 3D centers of the Gaussians using only transformer features to ensure geometric stability; the other incorporates both the transformer features and RGB image shortcuts to estimate additional attributes, such as opacity, covariance, and low-order spherical harmonics for view-dependent color. For every input, the model produces a dense set of Gaussian primitives, which are concatenated in a canonical frame and rendered by standard 3D Gaussian splatting. The model is trained using photometric losses, eliminating the need for explicit pose input or pose-based warping and resulting in a pose-free, feed-forward pipeline.



Fig. 3: Example of SAM2-based recoloring. Starting from the original scene (left), SAM2 produces object masks (middle), which are then used to apply recoloring to selected regions, yielding the recolored result (right).

3.2 SAM2-Based Recoloring

A key challenge in training our feed-forward 3D scene editing framework is the lack of multi-view consistent images paired with realistic edited inputs for supervision. To overcome this, we propose to approximate the editing task with recoloring. Specifically, we build upon SAM2 [33] and design an object-aware segmentation and recoloring pipeline: the recoloring branch generates multi-view consistent supervision targets, while SAM2’s masks enable object-level appearance augmentations. These object-level augmentations play a crucial role in bridging the gap between our recolored training data and the edited images encountered at inference, ultimately enhancing the model’s generalization and performance. The overall data processing pipeline is illustrated in Fig. 3. In this setting, recoloring provides a controlled pretext task for learning view-consistent fusion. At inference time, structural edit proposals such as newly added objects are produced by the 2D editor in individual views, and *Edit3r* reconciles these view-level proposals in the canonical 3D Gaussian space.

Object-Level Multi-Frame Segmentation via SAM2. Given multi-view video frames $\{I_v\}_{v=0}^{V-1}$, we apply the automatic mask generator (AMG) from SAM2 on the first frame I_0 to automatically identify all candidate objects, without the need for manual prompts. Each AMG proposal m provides a soft mask, area, stability score, predicted IoU, and a bounding box $b = (x, y, w, h)$, enabling efficient object selection and subsequent recoloring across views.

We keep a proposal if it passes a deterministic multi-criterion filter: (1) area $\geq A_{\min}$; (2) stability $\geq s_{\min}$; (3) predicted IoU $\geq q_{\min}$; (4) aspect ratio w/h within $[r_{\min}, r_{\max}]$; (5) the box has at least m_{edge} pixels of margin to the image boundary. These parameter details are provided in the appendix. All retained instances are assigned persistent IDs and used to prompt the SAM2 video segmentation predictor (VSP) at $t=0$.

Next, we apply the VSP to the video sequence. For each frame I_v , the model identifies the set of active object IDs O_v and per-object mask logits $Z_v^{(r)}$ which indicate foreground probabilities. Binary masks are obtained by thresholding the logits at zero, with pixels having logits greater than zero designated as foreground. To mitigate identity drift, we only process a frame v if its object-ID set O_v sufficiently overlaps with that of the most recently processed frame v^- . Formally, we define the overlap ratio as $\frac{|O_v \cap O_{v^-}|}{\max(|O_v|, |O_{v^-}|)}$. If this ratio falls below 0.5, frame I_v is skipped and excluded from all subsequent stages. For all remain-

ing (non-skipped) frames v in the video, this stage produces per-object binary masks $\{M_v^{(r)}\}_{r \in O_v}$, which guide the subsequent region-wise recoloring process.

Region-Wise Recoloring with Cross-View Consistency. We perform region-aware color augmentation by sequentially applying a composite color transform $\mathcal{C}_{\Theta_r}$ to each region in a deterministic, temporally consistent manner. This transform consists of ColorJitter, gamma correction, PCA-based lighting offset, a fixed RGB-channel permutation, and optional grayscale conversion. The augmentation parameters Θ_r are sampled once per region and then reused across all frames and views to ensure consistent appearance. When regions overlap, we re-normalize the per-pixel soft masks so that the mask weights sum to one over all overlapping regions (with a small ε added to the denominator for numerical stability). The recolored image is then synthesized by soft blending:

$$I'_t = \sum_{r \in \mathcal{R}} \hat{\alpha}_t^{(r)} \odot \mathcal{C}_{\Theta_r}(I_t) + \left(1 - \sum_{r \in \mathcal{R}} \hat{\alpha}_t^{(r)}\right) \odot I_t.$$

Here, $\odot$ denotes element-wise multiplication, and $\hat{\alpha}_t^{(r)}$ is the per-pixel *renormalized* mask weight for region r ; the remaining weight $1 - \sum_{r \in \mathcal{R}} \hat{\alpha}_t^{(r)}$ at each pixel is assigned to the original image. With a single region, this reduces to standard alpha blending between the transformed and original image. To reduce distribution shift, we also recolor the background using the same color transform.

3.3 Loss Functions

We train the model using both 2D supervision signals and 3D geometric constraints. For each view v , given the rendered prediction $\hat{I}_v$ produced by our model and the ground-truth image I_v , we optimize the model parameters by minimizing a weighted sum of three complementary loss terms across all views:

$$\min_{\theta} \sum_{v=0}^{V-1} \left[\mathcal{L}_{\text{CLIP}}(\hat{I}_v, I_v) + \mathcal{L}_{\text{LPIPS}}(\hat{I}_v, I_v) + \mathcal{L}_{\text{MSE}}(\hat{I}_v, I_v) \right] + \mathcal{L}_{\text{center}} + \mathcal{L}_{\text{geom}},$$

where each term is weighted by its corresponding coefficient. The loss functions are defined as follows: a) a CLIP-based image-image loss $\mathcal{L}_{\text{CLIP}}(\hat{I}_v, I_v)$ that aligns the predictions with the targets in a semantic embedding space, providing a robust global supervision signal; b) a VGG-based perceptual loss $\mathcal{L}_{\text{LPIPS}}(\hat{I}_v, I_v)$ that enhances mid- and high-frequency details such as edges and textures, improving perceptual fidelity beyond pixel-level matching; and c) a low-frequency MSE loss $\mathcal{L}_{\text{MSE}}(\hat{I}_v, I_v)$ that enforces consistency in color, exposure, and illumination while remaining tolerant to small reprojection errors.

Beyond the 2D rendering losses, we introduce 3D regularization on the predicted Gaussian centers to stabilize the scene structure during editing. Because the editing applied to input images may distort depth cues and cause geometric inconsistencies across views, we leverage the pretrained LRM (NoPoSplat [42]) to extract reference Gaussian centers $\mathcal{G}_{\text{ref}}$ from the unedited images. During training, the edited Gaussians $\mathcal{G}_{\text{edit}}$ are constrained to align with these reference

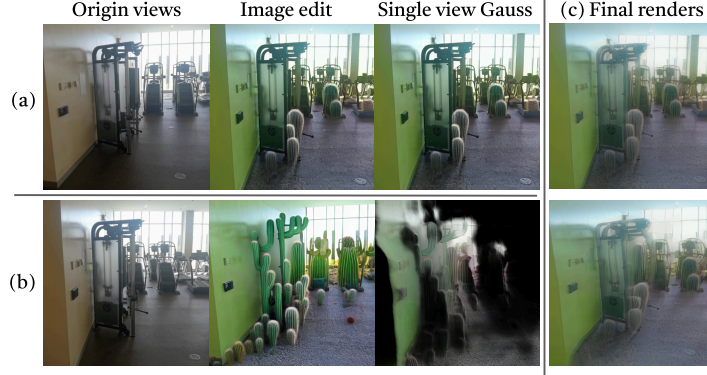


Fig. 4: Example of inference-time editing. (a) First input view, its edited result, and the corresponding single-view Gaussian rendering. (b) Second input view, its edited result, and the corresponding single-view Gaussian rendering. (c) Final rendering obtained by combining Gaussians from both views. *Prompt: Add a cactus garden.*

centers using two complementary regularization terms. First, a center-matching loss encourages each predicted Gaussian to remain close to its corresponding reference location, formulated as a Huber loss [17] between the predicted and reference 3D centers:

$$\mathcal{L}_{\text{center}} = \text{SmoothL1}(\hat{\boldsymbol{\mu}}, \boldsymbol{\mu}_{\text{ref}}).$$

This term preserves geometric fidelity to the original scene while permitting local deformations consistent with the applied edits. Second, a multi-view consistency loss enforces structural coherence across different edited views by minimizing the pairwise Chamfer- L_1 distance between randomly sampled Gaussian centers from each view:

$$\mathcal{L}_{\text{geom}} = \frac{1}{V(V-1)} \sum_{i < j} \text{Chamfer}_{L_1}(\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\mu}}_j).$$

This encourages a consistent 3D configuration across views, reducing undesired depth-layer separation and suppressing “floating” artifacts caused by view-dependent misalignment. Together, $\mathcal{L}_{\text{center}}$ anchors the edited scene to the base geometry learned by the pretrained reconstruction model, while $\mathcal{L}_{\text{geom}}$ encourages smooth geometric alignment across views. These 3D regularization terms complement the 2D perceptual and semantic losses by enforcing structural stability in the canonical Gaussian space, ensuring that scene edits remain spatially coherent, without introducing drift in Gaussian center locations during training.

3.4 Inference

In our training procedure, we construct asymmetric input pairs to deliberately introduce additional cross-view inconsistencies, encouraging the model to learn how to robustly fuse such inconsistencies. Although our model is trained solely on recolored images, it generalizes effectively to images edited by various 2D editing

methods [2, 3]. Unlike optimization-based pipelines that require iterative test-time fitting, inference in our system is fully feed-forward. Since the 2D editing stage operates independently from both the feed-forward reconstruction and the training procedure, our approach is highly flexible and can accommodate diverse image editing techniques [2, 12, 29]. For experiments and evaluations, we primarily employ commonly used IP2P [3] as the 2D editor, while additional editors are included in ablation studies.

As illustrated in Fig. 1, given a multi-view sequence $\{I_v\}_{v=0}^{V-1}$ and a text instruction $\mathcal{E}$, we first generate per-frame edited images $\{I_v^*\}_{v=0}^{V-1}$ with a 2D image editor. This differs from the training asymmetric input pairs setup, where only the first frame is edited; at inference, all frames are edited to maximize cross-view evidence. We then perform a single feed-forward pass of *Edit3r* to produce the edited 3D Gaussians and render novel views.

Fig. 4 provides a more concrete example: we visualize the Gaussians generated from the two input views separately in (a) and (b). Due to the inconsistency introduced by the 2D image editor, pixels in the second view (b) can conflict with those in the first view (a); in these regions, the Gaussians from the second view automatically reduce their opacity to mitigate the conflict, shown in the third image in (b). Panel (c) shows two newly synthesized views obtained by combining Gaussians from both inputs, demonstrating view-consistent results.

3.5 DL3DV-Edit-Bench

Motivation and Scope. Most existing 3D editing evaluations [13] either focus on single-object or toy scenes rather than full scene editing (*e.g.*, limited context and occlusions), or rely on ad-hoc, privately curated data without a standardized protocol. This hampers fair comparison: methods differ in scene diversity, edit categories, and evaluation criteria. Moreover, most prior methods edit an *existing* 3DGS and are therefore constrained by its original geometry, which largely restricts them to appearance-level changes and makes them ill-suited for edits that substantially alter scene structure. We therefore build **DL3DV-Edit-Bench** on top of the DL3DV [20] test split, covering diverse indoor/outdoor real scenes and four text-driven edit types: *Add*, *Remove*, *Modify* (attribute/material/appearance), and *Global* (style/exposure). Unlike methods bound to the initial 3D geometry, our pipeline leverages a 2D editor to introduce richer changes directly in image space, so our benchmark is deliberately designed to include more geometric modifications (*e.g.*, *Add* and *Remove*) that cannot be obtained by merely adjusting an existing 3DGS. The benchmark targets *scene-level* editing under multi-view inputs and measures both edit effectiveness and cross-view consistency in realistic settings.

Generation Pipeline. We start from the DL3DV test set and sample 20 real-world scenes (indoor / outdoor). We first run Grounding-DINO [22] on all views to obtain object-level labels and region proposals. The per-scene image set and extracted labels are then fed to a large language model to synthesize candidate prompts for the four edit categories (*Add* / *Remove* / *Modify* / *Global*). We manually vet and keep 5 valid prompts per scene, resulting in 100 total edit

instances. To ensure multi-view consistency during editing, we fix the 2D image editor, *share* the same random seed and the same textual prompt across all views of a scene, and confine local edits with masks derived from the proposals. All editor hyperparameters (CFG, steps, strength schedule) are documented and kept constant unless otherwise stated.

4 Experiments

4.1 Implementation Details

Our main experiments are conducted on the *DL3DV-Edit-Bench* that we curate to serve as our primary evaluation benchmark. To align with prior work, we report results on the public datasets used in those papers in the supplementary material. For the 2D image-editing module, we use InstructPix2Pix [3] as the default editor to ensure comparability, and we evaluate more editors in ablations. All experiments are run on a single NVIDIA RTX 6000 GPU.

4.2 Baselines

We compare against three methods: (1) EditSplat [19] (optimization-based) with its original 2D editor, InstructPix2Pix, kept unchanged to avoid re-tuning and ensure fairness; (2) GaussCtrl [38] (optimization-based) likewise retaining the authors’ ControlNet module. Both optimization-based baselines are compute-intensive and, in addition to the text prompt, require a pre-reconstructed DL3DV scene, which we supply from standardized baseline reconstructions for parity. (3) NoPoSplat [42] (feed-forward), which runs the released checkpoint and consumes only the multi-view inputs for reconstruction, serving as a lower bound on cross-view edit consistency and isolating the contribution of our cross-view fusion in the feed-forward design. For NoPoSplat and *Edit3r*, the inputs are two sparse views and the edit prompt. For EditSplat and GaussCtrl, we follow their standard setting with the full posed image set, a pre-reconstructed 3DGS scene, and the same prompt. All methods are rendered and evaluated at the same held-out novel views sampled approximately ± 30 video frames from the input views, so the reported metrics reflect novel-view renderings rather than directly edited input images. For GaussCtrl, we also include a prompt-format check using LLM-translated descriptive prompts on several scenes; this produces similar trends to the instruction-prompt setting.

4.3 Main Experiments

Evaluation Metrics. Following prior works [13, 19, 38, 44], we evaluate each method from two angles: (1) *edit effectiveness*, i.e., whether the method performs the intended scene edit; and (2) *reconstruction quality and multi-view consistency*, i.e., whether the edited 3DGS yields high-quality renders that remain consistent across viewpoints. For (1), acknowledging the subjectivity of 3D scene editing, we adopt CLIP [32] image-text similarity, which scores the absolute alignment between the edited render and the target description. For (2), no single score captures all failure modes, so we assess two complementary aspects,

Table 1: Quantitative comparison between optimization-based and feed-forward 3D scene editing methods on DL3DV-Edit-Bench. CLIP_{t2i} measures instruction alignment (higher is better). C-FID and C-KID are scene-conditioned realism/consistency metrics computed between edited renders and reference views (lower is better).

Category	Method	Time (s) (↓)	CLIP _{t2i} (↑)	C-FID (↓)	C-KID (↓)
Optimization	GaussCtrl [38]	325.53	0.227	135.0	0.091
	EditSplat [19]	584.46	0.241	174.1	0.122
Feed-forward	NoPoSplat [42]	0.61	0.253	180.6	0.125
	Edit3r (Ours)	0.51	0.266	171.3	0.116

Table 2: Epipolar consistency with respect to the edited reference view. Symmetric epipolar distance (SED) is computed from SIFT matches and known camera geometry.

Metric	NoPoSplat	Edit3r (Ours)
Mean Med. SED (px) ↓	30.260	23.752
Median Med. SED (px) ↓	3.783	3.595
Med. SED < 3 px ↑	55.56%	59.29%
Inlier ≤ 3 px ↑	42.54%	47.06%

rendering quality and *geometric consistency*. We first quantify quality at the *distribution* level using C-FID [15] and C-KID [1]: both measure how close the edited renders are, *as a set*, to real reference views of the same scene, jointly reflecting visual fidelity and diversity. C-FID computes the Fréchet distance between the two feature distributions, whereas C-KID computes the kernel MMD and is more reliable at moderate sample sizes; the prefix “C-” indicates that both are computed in a scene-conditioned manner. However, distribution-level scores neither penalize per-frame artifacts nor directly constrain cross-view geometry, so we add two further measurements. (i) For *per-frame quality*, we report the no-reference metrics NIQE and BRISQUE together with variance-of-Laplacian sharpness, which detect single-image degradations such as blur. (ii) For *geometric consistency*, we evaluate epipolar consistency between the edited reference view and each rendered novel view, following the symmetric epipolar-distance protocol used in prior novel-view consistency evaluation [45]: we extract SIFT correspondences between the two views, compute the fundamental matrix from the known camera intrinsics and poses, and report symmetric epipolar distance (SED) statistics together with 3-pixel inlier/view ratios.

Quantitative Evaluation. As summarized in Tab. 1, *Edit3r* achieves an optimal quality-speed trade-off. In terms of inference latency, our feed-forward approach is orders of magnitude faster than optimization-based baselines, processing a scene in 0.51s compared to over 325s for iterative methods. *Edit3r* attains the highest CLIP_{t2i} score, reflecting superior instruction following and semantic alignment. While GaussianCtrl maintains low C-FID and C-KID by performing only marginal updates, it suffers from poor text alignment due to restricted edit magnitude. EditSplat exhibits moderate edit strength but remains overly conservative. In contrast, NoPoSplat, primarily designed for reconstruction, tends to average inconsistent multi-view evidence, resulting in blurred artifacts and diminished scores across all metrics. Overall, *Edit3r* successfully balances gener-

Table 3: No-reference image quality and sharpness evaluation. Lower is better for NIQE and BRISQUE; higher is better for sharpness.

Method	NIQE ↓	BRISQUE ↓	Sharpness ↑
Original Scene	2.982	15.402	1506.719
EditSplat	3.439	29.588	327.252
NoPoSplat	3.885	27.659	251.997
GaussCtrl	7.081	66.384	13.718
<i>Edit3r</i> (Ours)	3.402	26.538	314.902

Table 4: Ablation study on different training strategies. “w/o Recolor” utilizes image editing for training instead of recoloring; “w/o 3D Loss” removes the 3D geometric constraint loss; “w/o SAM” disables SAM-based object masking when recoloring; “w/o R-Drop” removes R-Drop regularization.

Method	CLIP _{t2i} (↑)	C-FID (↓)	C-KID (↓)
<i>Edit3r</i>	0.266	171.3	0.116
+w/o Recolor	0.243	215.0	0.141
+w/o 3D Loss	0.237	278.4	0.182
+w/o SAM	0.248	179.6	0.127
+w/o R-Drop	0.252	183.1	0.130

active fidelity with structural integrity, outperforming feed-forward alternatives in C-FID and C-KID while matching the semantic stability of the most conservative optimizers. Tab. 2 shows that *Edit3r* improves geometric consistency over the feed-forward NoPoSplat baseline across all SED and 3px-inlier metrics, indicating more coherent fusion of edited multi-view evidence. Tab. 3 shows that *Edit3r* also obtains the best NIQE and BRISQUE among edited outputs while maintaining competitive sharpness.

Qualitative Comparison. Fig. 5 presents four examples (a–d) of scene editing by EditSplat [19], GaussCtrl [38], NoPoSplat [42], and *Edit3r*. The small inset at the bottom-right of each panel shows the result from another viewpoint of the same scene. EditSplat achieves partially successful edits in (a, c, d), but it also alters regions that should remain unchanged, such as the sky and ground in (c). GaussCtrl is almost entirely unsuccessful across (a–d), producing chaotic or ineffective edits when applied to scenes that are unseen during its training. NoPoSplat generally generates plausible results, but its quality is highly sensitive to the inconsistency between input views: when cross-view inconsistency is large, it suffers from blur in (a,d), whereas for more consistent inputs in (b,c), the reconstruction quality is much better. In contrast, our method delivers stable, high-quality edits across all four scenes, maintaining consistency across views while preserving the original scene structure.

4.4 Ablation Study

We validate our design choices by comparing *Edit3r* against different training and inference variants. Unless otherwise stated, the default configuration uses SAM2-based recoloring for supervision, the asymmetric input scheme, our full training losses (2D appearance + 3D regularization), randomly dropping the



Fig. 5: Qualitative comparison of four methods. The first row shows the original scene and the editing prompt; the remaining four rows show the results, each paired with an additional view.

first input view’s Gaussian with $p=0.5$ during training; and InstructPix2Pix as the editing front-end at inference.

Recolor/Image Editing For Training. Directly training on multi-view images produced by a 2D editor inevitably introduces substantial cross-view inconsistencies that act as label noise and hinder convergence. We therefore supervise training with *SAM2-based recoloring*: object masks are extracted and recolored consistently across views, yielding stable, view-aligned targets. As Tab. 4 shows, in our ablations, replacing recoloring data with direct multi-view editing degrades multi-view consistency and increases artifacts, confirming the need for view-consistent supervision.

Training Loss. Because recoloring/editing can shift predicted Gaussian centers, relying solely on 2D losses (reconstruction/style) permits geometry drift: the rendering may look plausible while Gaussian primitives misalign in 3D. We ablate 3D regularizers, and results are shown in Tab. 4. Removals lead to larger center deviation and diminished cross-view coherence. Using the full loss stabilizes geometry and improves semantic fidelity without sacrificing appearance.



Fig. 6: Scene editing results of *Edit3r*. The left column shows the input scene views, and the remaining columns show the rendered results after applying the edit with 2D image editors (IP2P, FLUX, GPT, and Gemini). *Prompt: Paint the door red.*

SAM2 Segmentation for Augmentation. To narrow the distribution gap between recolored training images and edited images at inference, we augment training data with SAM2-derived mask jittering (dilation/erosion), palette perturbation, and limited background leakage. Ablating these SAM2-based augmentations by training with plain recoloring only reduces robustness and increases failure cases under stronger edits, indicating that modest variability during training is beneficial. Results are shown in Tab. 4.

Random Drop the First Input View’s Gaussian. Given the asymmetric inputs, the edited reference view can dominate the learned style. We introduce random drop during training: with probability 0.5, we discard the Gaussians associated with the first input view when forming supervision, forcing the network to propagate the edit semantics from the reference to the auxiliaries. Ablations on the drop probability, with results shown in Tab. 4, show that removing random drop causes overfitting to the reference viewpoint, while excessively high drop rates weaken edit strength.

Editor Agnosticity during Inference. Our framework decouples 2D editing from feed-forward reconstruction, allowing for modular integration of various generative priors at test time. We benchmark IP2P [3] against GPT-Image-1 [29], Gemini-2.5-Flash-Image [12], and FLUX [2]. As summarized in Tab. 5, while FLUX achieves superior visual fidelity, it frequently exhibits spatial mislocalization. GPT and Gemini offer precise localization but introduce extraneous global artifacts, whereas IP2P demonstrates the lowest perceptual quality. Crucially, *Edit3r* maintains consistent speed and cross-view coherence regardless of the 2D backbone. This modularity indicates that our reconstruction pipeline is editor-agnostic and scales with future improvements in 2D synthesis.

Ablation on All-Edited Input Pairs During Inference. Asymmetric training inputs (recolored view + original view) intentionally expose cross-view inconsistency to enforce robust fusion. At inference, we edit all input views to provide richer edit evidence and reduce inconsistency. Since the backbone is trained to

Table 5: Ablation study on different image editors in our method.

Method	CLIP _{t2i} (↑)	C-FID (↓)	C-KID (↓)
IP2P [3]	0.266	171.3	0.116
GPT [29]	0.261	166.2	0.102
Gemini [12]	0.246	150.3	0.098
FLUX [2]	0.276	169.9	0.112

Table 6: Ablation study on all-edited vs. asymmetric input pairs.

Input	CLIP _{t2i} (↑)	C-FID (↓)	C-KID (↓)
Asymmetric	0.259	189.7	0.149
All-edited	0.266	171.3	0.116

handle content collision, such input pairs make the inference setting easier than the training setting because of less inconsistency. We include an ablation on all-edited vs. asymmetric input during inference, results are shown in Tab. 6.

5 Conclusion

We introduced *Edit3r*, a pose-free, feed-forward framework that unifies reconstruction and instruction-driven editing of 3D scenes from unposed, instruction-edited images. By directly predicting edited 3D Gaussian splats in a single pass—without test-time optimization or pose estimation—our approach departs from the conventional reconstruct-edit-refit pipeline and enables fast, photorealistic, and instruction-aligned rendering. Two design choices are key to making this practical at scale: (i) a SAM2-based recoloring strategy that provides reliable, cross-view-consistent supervision despite the lack of true multi-view edited ground truth; and (ii) an asymmetric input scheme that pairs an edited reference view with unedited auxiliary views, encouraging the network to fuse separate observations while preserving scene structure. Coupled with lightweight 3D regularization on Gaussian centers, these components yield robust multi-view coherence even when the input edits are imperfect or inconsistent.

To enable fair and reproducible comparisons, we proposed *DL3DV-Edit-Bench*, a scene-level benchmark spanning 20 diverse scenes and four edit types (Add / Remove / Modify / Global). On this benchmark, *Edit3r* achieves stronger text-image alignment and improved 3D consistency than recent optimization-based and feed-forward baselines, while being orders of magnitude faster at inference. Ablations confirm the necessity of view-consistent recoloring, asymmetric inputs, and 3D geometric losses, and demonstrate that our method is editor-agnostic and generalizes to a range of 2D editing front-ends.

Limitations and Future Work. Although recoloring provides stable supervision, it does not fully capture large geometric changes (*e.g.*, substantial add / remove edits) or extreme material and illumination shifts. Looking ahead, promising directions include expanding supervision beyond recoloring with view-consistent generative augmentation, learning per-object disentangled controls for precise 3D edits, incorporating uncertainty-aware rendering to handle ambiguous or conflicting edits, extending to dynamic scenes and longer view sequences, and further enriching *DL3DV-Edit-Bench* with more diverse and complex scenes.

References

1. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying MMD GANs. In: ICLR (2018) [11](#)
2. Black Forest Labs, Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., Dockhorn, T., English, J., English, Z., Esser, P., Kulal, S., Lacey, K., Levi, Y., Li, C., Lorenz, D., Müller, J., Podell, D., Rombach, R., Saini, H., Sauer, A., Smith, L.: FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space (2025), <https://arxiv.org/abs/2506.15742> [4](#), [5](#), [9](#), [14](#), [15](#)
3. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: CVPR (2023) [3](#), [4](#), [5](#), [9](#), [10](#), [14](#), [15](#)
4. Charatan, D., Li, S.L., Tagliasacchi, A., Sitzmann, V.: pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In: CVPR (2024) [2](#), [4](#)
5. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: ICCV (2021) [2](#), [4](#)
6. Chen, Y., Chen, Z., Zhang, C., Wang, F., Yang, X., Wang, Y., Cai, Z., Yang, L., Liu, H., Lin, G.: Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In: CVPR (2024) [3](#), [4](#)
7. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: MVSplat: Efficient 3D Gaussian Splatting from sparse multi-view images. In: ECCV (2024) [4](#)
8. Chen, Y., Zheng, C., Xu, H., Zhuang, B., Vedaldi, A., Cham, T.J., Cai, J.: Mvsplat360: Feed-forward 360 scene synthesis from sparse views. In: NeurIPS (2024) [4](#)
9. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., Vanderbilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects. arXiv preprint arXiv:2212.08051 (2022) [4](#)
10. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. ICLR (2021) [4](#), [5](#)
11. Geyer, M., Bar-Tal, O., Bagon, S., Dekel, T.: Tokenflow: Consistent diffusion features for consistent video editing (2023), <https://arxiv.org/abs/2307.10373> [3](#)
12. Google: Gemini 2.5 Flash Image. Gemini API documentation (2025), <https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash-image>, accessed: 2026-06-26 [9](#), [14](#), [15](#)
13. Haque, A., Tancik, M., Efros, A., Holynski, A., Kanazawa, A.: Instruct-nerf2nerf: Editing 3d scenes with instructions. In: ICCV (2023) [2](#), [3](#), [4](#), [9](#), [10](#)
14. Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., Cohen-Or, D.: Prompt-to-prompt image editing with cross attention control (2022), <https://arxiv.org/abs/2208.01626> [3](#)
15. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS (2017) [11](#)
16. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: LRM: large reconstruction model for single image to 3d. In: ICLR (2024) [2](#)

17. Huber, P.J.: Robust estimation of a location parameter. *Annals of Mathematical Statistics* (1964) 8
18. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. In: *ACM TOG* (2023) 2, 3
19. Lee, D.I., Park, H., Seo, J., Park, E., Park, H., Baek, H.D., Shin, S., Kim, S., Kim, S.: Editsplat: Multi-view fusion and attention-guided optimization for view-consistent 3d scene editing with 3d gaussian splatting (2025), <https://arxiv.org/abs/2412.11520> 2, 10, 11, 12
20. Ling, L., Sheng, Y., Tu, Z., Zhao, W., Xin, C., Wan, K., Yu, L., Guo, Q., Yu, Z., Lu, Y., Li, X., Sun, X., Ashok, R., Mukherjee, A., Kang, H., Kong, X., Hua, G., Zhang, T., Benes, B., Bera, A.: D3dv-10k: A large-scale scene dataset for deep learning-based 3d vision (2023), <https://arxiv.org/abs/2312.16256> 3, 9
21. Liu, S., Zhang, Y., Li, W., Lin, Z., Jia, J.: Video-p2p: Video editing with cross-attention control (2023), <https://arxiv.org/abs/2303.04761> 3
22. Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Jiang, Q., Li, C., Yang, J., Su, H., Zhu, J., Zhang, L.: Grounding dino: Marrying dino with grounded pre-training for open-set object detection (2024), <https://arxiv.org/abs/2303.05499> 9
23. Lyu, W., Li, X., Kundu, A., Tsai, Y.H., Yang, M.H.: Gaga: Group any gaussians via 3d-aware memory bank (2024) 3
24. Lyu, W., Zhou, Y., Yang, M.H., Shu, Z.: Facelift: Learning generalizable single image 3d face reconstruction from synthetic heads. In: *ICCV* (2025) 4
25. Madhavaram, V., Rawat, S., Devaguptapu, C., Sharma, C., Kaul, M.: Towards a training free approach for 3d scene editing (2024), <https://arxiv.org/abs/2412.12766> 2
26. Melnik, A., Alt, B., Nguyen, G., Wilkowski, A., Stefańczyk, M., Wu, Q., Harms, S., Rhodin, H., Savva, M., Beetz, M.: Digital twin generation from visual data: A survey (2025), <https://arxiv.org/abs/2504.13159> 2
27. Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.Y., Ermon, S.: Sdedit: Guided image synthesis and editing with stochastic differential equations (2022), <https://arxiv.org/abs/2108.01073> 3
28. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: *ECCV* (2020) 2, 3
29. OpenAI: GPT Image 1. OpenAI API documentation (2025), <https://platform.openai.com/docs/models/gpt-image-1>, accessed: 2026-06-26 4, 9, 14, 15
30. Ouyang, H., Wang, Q., Xiao, Y., Bai, Q., Zhang, J., Zheng, K., Zhou, X., Chen, Q., Shen, Y.: Codef: Content deformation fields for temporally consistent video processing (2024), <https://arxiv.org/abs/2308.07926> 3
31. Qi, C., Cun, X., Zhang, Y., Lei, C., Wang, X., Shan, Y., Chen, Q.: Fatezero: Fusing attentions for zero-shot text-based video editing (2023), <https://arxiv.org/abs/2303.09535> 3
32. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021), <https://arxiv.org/abs/2103.00020> 10
33. Ravi, N., Gabeur, V., Hu, Y.T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., et al.: Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714* (2024) 3, 6
34. Szymanowicz, S., Rupprecht, C., Vedaldi, A.: Splatler image: Ultra-fast single-view 3d reconstruction. In: *CVPR* (2024) 4

35. Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., Liu, Z.: LGM: large multi-view gaussian model for high-resolution 3d content creation. In: ECCV (2024) 4
36. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017) 4
37. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy. In: CVPR (2024) 2, 5
38. Wu, J., Bian, J.W., Li, X., Wang, G., Reid, I., Torr, P., Prisacariu, V.: GaussCtrl: Multi-View Consistent Text-Driven 3D Gaussian Splatting Editing. ECCV (2024) 2, 3, 4, 10, 11, 12
39. Xu, H., Chen, A., Chen, Y., Sakaridis, C., Zhang, Y., Pollefeys, M., Geiger, A., Yu, F.: Murf: Multi-baseline radiance fields. In: CVPR (2024) 4
40. Xu, Y., Ng, Y., Wang, Y., Sa, I., Duan, Y., Li, Y., Ji, P., Li, H.: Sketch2scene: Automatic generation of interactive 3d game scenes from user’s casual sketches (2024), <https://arxiv.org/abs/2408.04567> 2
41. Yang, S., Zhou, Y., Liu, Z., Loy, C.C.: Rerender a video: Zero-shot text-guided video-to-video translation (2023), <https://arxiv.org/abs/2306.07954> 3
42. Ye, B., Liu, S., Xu, H., Li, X., Pollefeys, M., Yang, M.H., Peng, S.: No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. arXiv preprint arXiv:2410.24207 (2024) 2, 5, 7, 10, 11, 12
43. Ye, M., Danelljan, M., Yu, F., Ke, L.: Gaussian grouping: Segment and edit anything in 3d scenes. In: ECCV (2024) 3
44. You, J., Lin, C., Lyu, W., Zhang, Z., Yang, M.H.: InstaInpaint: Instant 3D-scene inpainting with masked large reconstruction model. In: NeurIPS (2025) 4, 10
45. Yu, J.J., Forghani, F., Derpanis, K.G., Brubaker, M.A.: Long-term photometric consistent novel view synthesis with diffusion models. In: ICCV (2023) 11
46. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: Gs-lrm: Large reconstruction model for 3d gaussian splatting. ECCV (2024) 2, 4
47. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models (2023) 3, 4
48. Zhang, X., Lee, J., Joslin, C., Lee, W.: Advancing 3d gaussian splatting editing with complementary and consensus information (2025), <https://arxiv.org/abs/2503.11601> 2