SLatHarmony: Locality-Guided Harmonization for Local 3D Editing

Anonymous authors
Paper under double-blind review

Abstract

Local 3D editing requires faithfully realizing the intended changes within the target region, preserving non-edited regions, and maintaining coherent shape and texture across the entire asset. Existing methods typically mitigate non-target changes through constrained generation or inversion-based editing. In contrast, we leverage the spatial locality of the structured latent (SLat) representation in TRELLIS.2 to combine source and edited SLat fields through regional composition, thereby preserving source content outside the target region. However, achieving coherent shape and texture integration between target and non-edited regions remains challenging. We therefore propose SLatHarmony, a framework that couples regional SLat composition with cross-field harmonization. Specifically, a shape SLat seamer predicts locally decodable transition features using context from the source and edited sides to bridge geometric discrepancies near the interface, while a texture SLat harmonizer combines channel-wise field alignment with boundary-aware target-region correction to reduce appearance mismatch. Experiments on Edit3D-Bench show that SLatHarmony substantially improves the preservation of non-edited geometry and appearance while maintaining competitive semantic alignment with the edit instruction.

1 Introduction

Local 3D editing aims to modify a target region while preserving the geometry, appearance, and identity of the remaining content. Existing approaches localize changes through spatially constrained optimization, multi-view editing, or structured-latent manipulation Sella et al. (2023); Chen et al. (2024a); Ye et al. (2025); Li et al. (2026); Huang et al. (2026). Although these strategies reduce global drift, they generally reconstruct, inject, or constrain non-edited content during editing rather than preserving its representation directly.

We consider a complementary representation-level perspective. Once an editing process has produced the latent content corresponding to the intended change, regions that should remain unchanged can be retained directly from the source SLat field rather than regenerated. This becomes particularly natural for structured 3D representations whose latent features are explicitly associated with spatial locations. TRELLIS Xiang et al. (2025) represents 3D assets using sparse structured latent (SLat) representations on a 3D grid, and TRELLIS.2 Xiang et al. (2026) further separates shape and texture SLat fields. Such spatial organization makes direct regional reuse a plausible alternative to replacing non-edited content with regenerated latent features.

However, successful regional composition requires two distinct properties: spatial locality and cross-field compatibility. Locality determines whether source features can be reused regionally: modifying latent features in one region should primarily affect the corresponding spatial neighborhood after decoding. Our controlled interventions show that both shape and texture SLat representations exhibit this behavior, making regional composition feasible. Compatibility determines whether independently generated latent regions can form a coherent asset after composition. Intuitively, the source and edited fields may each be valid and decode plausibly in isolation, yet regional composition can create a joint local latent neighborhood near the interface that was not observed during decoder training. Locality only confines any resulting disturbance spatially; compatibility asks whether this composed neighborhood can still serve as a coherent input to the pretrained decoder.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Importantly, such cross-field incompatibility manifests differently in shape and texture SLat spaces. For shape SLat, directly joining independently generated fields can introduce locally incompatible features that decode into cracks, folds, or implausible geometric transitions near the interface. For texture SLat, the discrepancy more often appears as a mismatch in latent statistics or appearance, resulting in abrupt changes in color, brightness, reflectance, or material. Accordingly, shape may require synthesizing transition features unavailable from either field, whereas texture should preserve the intended edited appearance while correcting relative field mismatch. This motivates local completion for shape and alignment plus residual correction for texture, rather than generic blending.

We therefore propose SLatHarmony, a representation-level module that operates after an upstream editor has produced spatially aligned source and edited fields together with a target-region mask. SLatHarmony combines regional SLat composition with cross-field harmonization. Regional composition directly retains source features outside the target region while preserving edited features in its interior, providing explicit control over feature provenance in the non-edited region. Cross-field harmonization then addresses incompatibility at the interface between the source and edited fields. Specifically, a shape SLat seamer predicts locally decodable boundary features using context from both sides, treating geometric integration as a boundary-restricted completion problem. A texture SLat harmonizer instead combines field-level alignment with boundary-aware residual correction over the target region to reduce appearance discontinuities while preserving the edited texture content.

Our contributions are:

- We identify spatial locality and cross-field compatibility as two distinct requirements for composing independently generated structured 3D latents. Controlled interventions verify the locality of shape and texture SLat representations, motivating local 3D editing through direct regional composition.
- We introduce SLatHarmony, which couples explicit source-region preservation with modality-specific cross-field harmonization: contextual local completion for shape SLat fields and conservative field alignment with boundary-aware target-region correction for texture SLat fields.
- Experiments on Edit3D-Bench show substantially improved preservation of non-edited geometry and appearance while maintaining competitive semantic alignment with the edit instruction; controlled component analyses further demonstrate the benefits of shape- and texture-specific interface treatment.

2 Related Work

2.1 Structured 3D generation

Optimization-based methods such as DreamFusion Poole et al. (2023), Magic3D Lin et al. (2023), and ProlificDreamer Wang et al. (2023) lift 2D diffusion priors into 3D through score distillation. Multi-view approaches improve geometric consistency using cross-view constraints or jointly generated observations, as in MVDream Shi et al. (2024), SyncDreamer Liu et al. (2024), and Wonder3D Long et al. (2024). Feed-forward models such as LRM Hong et al. (2023), TripoSR Tochilkin et al. (2024), and LGM Tang et al. (2024) reduce per-instance optimization through direct reconstruction of NeRF-, mesh-, or Gaussian-based representations. Part-aware representations make localized manipulation more natural by organizing shapes into controllable components. PartNeRF Tertikas et al. (2023) learns editable neural parts from 2D supervision, while SALAD Koo et al. (2023) and DiffFacto Nakayama et al. (2023) model shapes through part-level latent or point-based generation. PartGen Chen et al. (2025) and CoPart Dong et al. (2025) support context-aware part generation and composition, while GaussianBlock Jiang et al. (2025) and CompGS Ge et al. (2025) develop compositional and editable Gaussian representations. These approaches emphasize part-level control and composition, whereas our work focuses specifically on compatibility between independently generated SLat fields. TRELLIS Xiang et al. (2025) attaches local features to active cells of a sparse grid and decodes the resulting Structured LATent (SLat) into multiple 3D representations, while TRELLIS.2 Xiang et al. (2026) separates geometry and material into shape and texture SLat fields. Their explicit spatial support is particularly relevant to regional manipulation and composition.

2.2 LOCAL 3D EDITING

Early instruction-driven approaches combine neural fields with 2D diffusion guidance, as in Instruct-NeRF2NeRF Haque et al. (2023) and DreamEditor Zhuang et al. (2023). Other methods use SDS optimization in Vox-E Sella et al. (2023) and FocalDreamer Li et al. (2024), explicit representation editing in GaussianEditor Wang et al. (2024) and TIP-Editor Zhuang et al. (2024), and multi-view constraints in Generic 3D Diffusion Adapter Chen et al. (2024a) and Tailor3D Qi et al. (2024). Preditor3D Erkoc et al. (2025) improves shape-editing efficiency and precision through a learned prior. Recent 3D Gaussian Splatting methods improve localized editing and view consistency through consistent multi-view guidance in DGE Chen et al. (2024b) and EditSplat Lee et al. (2025), direct Gaussian control in GaussianEditor Chen et al. (2024c), and regional or geometry-aware constraints in RoMaP Kim et al. (2025) and InterGSEdit Wen et al. (2025). Moving closer to representation-level editing, 3D-LATTE Parelli et al. (2026) performs instruction-driven editing directly in a native 3D latent space. TRELLIS-based methods exploit structured latent representations to preserve non-target content: Nano3D Ye et al. (2025) merges SLat representations, VoxHammer Li et al. (2026) replaces features in the non-edited region during inversion and denoising, Easy3E Hu et al. (2026) uses region-aware repainting with a soft transition, and Vinedresser3D Huang et al. (2026) performs inversion-guided latent inpainting. Existing methods mainly focus on generating or editing representations while constraining changes outside the target region. SLatHarmony instead focuses on composing independently generated structured latent fields and addressing cross-field compatibility at their interface.

3 PRELIMINARIES

TRELLIS Xiang et al. (2025) represents a 3D object using a sparse structure together with a structured latent (SLat) field. Let $V = \{\mathbf{p}_i\}_{i=1}^{N}$ denote the set of active voxels on a sparse 3D grid, where $\mathbf{p}_i \in \mathbb{Z}^3$. An SLat associates each active voxel with a latent feature, $Z = \{(\mathbf{p}_i, \mathbf{z}_i) \mid \mathbf{p}_i \in V\}$, where $\mathbf{z}_i \in \mathbb{R}^d$. The active voxel set V therefore defines the spatial support of the latent representation, while the attached features encode local 3D content.

TRELLIS.2 Xiang et al. (2026) further factorizes the latent representation into shape and texture SLat fields defined on the same active voxel set V: $Z^{\text{shape}} = \{(\mathbf{p}_i, \mathbf{s}_i) \mid \mathbf{p}_i \in V\}$ and $Z^{\text{tex}} = \{(\mathbf{p}_i, \mathbf{t}_i) \mid \mathbf{p}_i \in V\}$. Shape features encode local geometric information, whereas texture features encode local appearance and material properties. Texture SLat generation is conditioned on the shape representation, but both modalities share the same explicit spatial organization. This shared support makes SLat particularly suitable for regional manipulation and composition, since local regions can be identified and operated on directly through their corresponding active voxels. Spatial indexing enables such regional reuse, but does not by itself guarantee compatibility between features drawn from independently generated fields.

4 METHOD

4.1 SPATIAL LOCALITY OF SHAPE AND TEXTURE SLAT

The spatial locality of decoded responses to local SLat manipulation is first examined. On the sparse active-voxel set V, a contiguous region $\Omega \subseteq V$ is selected, and the features of either the shape or texture SLat within Ω are perturbed using *feature zeroing*, *matched noise*, or *cross-field replacement*, while the other modality and all features outside Ω are kept fixed. For each active voxel $v \in V$, its distance to Ω is defined as the minimum number of 6-neighbor hops from v to any voxel in Ω over the sparse voxel support. The decoded geometric or appearance response is then measured as a function of this sparse-voxel distance and normalized by the response at the perturbation boundary, $R(d)/R(0)$.

Compared with feature zeroing, matched noise and cross-field replacement leave slightly larger long-range shape residuals but follow the same overall decay. Feature zeroing removes local information, whereas matched noise controls for perturbation magnitude without introducing a second coherent field. Cross-field replacement inserts features from an independently generated SLat and therefore most closely approximates regional composition. Its localized inconsistencies suggest that

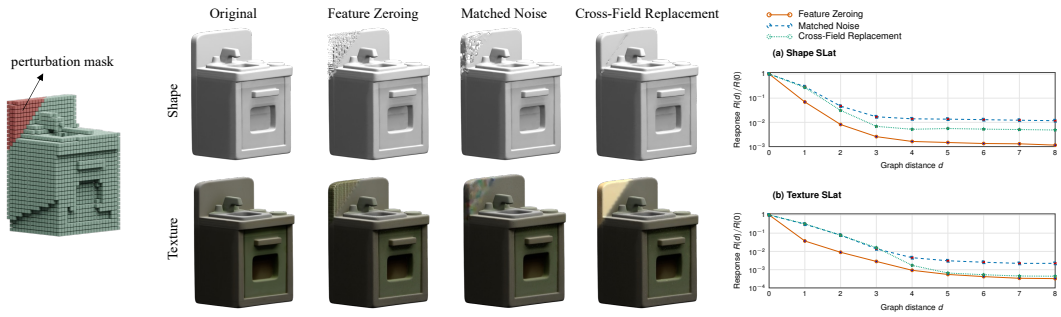

Figure 1: Spatial locality of shape and texture SLat representations. Shape and texture responses to three controlled perturbations are concentrated near Ω and decay rapidly with sparse-voxel distance.

spatial locality makes regional composition feasible while motivating harmonization between independently generated latent fields. Locality does not imply exact output invariance, since sparse decoding can propagate changes beyond the manipulated tokens. Nevertheless, the observed spatial decay provides an empirical basis for regional SLat composition. These observations motivate a two-stage design: deterministic regional composition exploits locality to preserve source content, while interface harmonization addresses the remaining cross-field incompatibility.

4.2 SLatHarmony Pipeline and Problem Formulation

Figure 2 illustrates the complete SLatHarmony editing pipeline. Given spatially aligned source and edited representations, a target image, and a 3D edit mask M, where M denotes the voxel locations allowed to change, SLatHarmony operates entirely at the representation level. The mask and target image may be provided by the user or produced by replaceable upstream components, including part segmentation Liu et al. (2025); Yang et al. (2024), view selection Huang et al. (2026), and image editing or inpainting Lugmayr et al. (2022); Black Forest Labs et al. (2025). Our method then composes the source and edited sparse representations, resolves their interface incompatibility, and decodes the resulting shape and texture SLat fields into the final asset.

Preserving non-edited content first requires preserving its sparse support. Sparse structure determines the active voxel locations at which shape and texture features can exist. If structure generation adds, removes, or displaces active voxels outside M, copying source features afterward cannot recover the original representation because their spatial support has already changed. Let V_s and V_e denote the active voxel sets of the source and edited SLat fields, respectively. The composed support is therefore defined as

$$V_f = (V_s \setminus M) \cup (V_e \cap M), \tag{1}$$

so that the complete source support is retained outside the target region, while structural additions or removals proposed by the edited branch are admitted only inside M.

Feature provenance is deterministic away from the interface. Let $M_b \subseteq M$ denote a narrow boundary band on the edited side, and define

$$V_s^{\text{out}} = V_s \setminus M, \qquad V_e^{\text{bd}} = V_e \cap M_b, \qquad V_e^{\text{int}} = (V_e \cap M) \setminus V_e^{\text{bd}}. \tag{2}$$

Source features are retained on V_s^{out}, while edited features are used on V_e^{int}. Only the edited-side shape features in V_e^{bd} remain to be completed at the interface. Interface processing is restricted to this boundary band, preserving source features outside M while avoiding unnecessary changes to the edited interior.

Since the TRELLIS.2 texture flow is conditioned on the shape representation Xiang et al. (2026), shape composition must be completed before texture generation. A shape edit may add or remove active voxels, so texture should be generated with the composed support and harmonized shape as its conditioning input rather than on the original edited geometry and cropped afterward. The completed shape SLat therefore conditions the generation of the edited texture SLat, which is subsequently composed with the source texture field and harmonized at their interface. The final shape and texture SLat fields are then decoded into the edited asset.

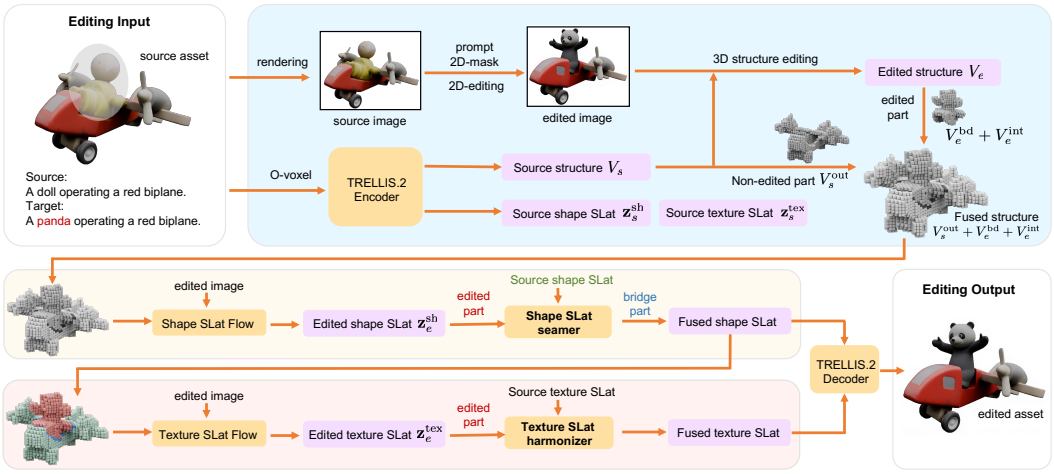

Figure 2: SLatHarmony performs regional SLat composition followed by modality-specific cross-field harmonization.

4.3 Shape SLat Seamer

Once the final support is fixed and the source and edited shape interiors are assigned deterministically, only the edited-side boundary tokens V_e^{bd} remain to be completed. Directly splicing independently generated shape fields at these locations can yield cracks or folded surfaces near the interface. Latent interpolation is also unreliable because smoothness in feature space does not necessarily correspond to a plausible geometric transition after decoding. SLat features are not explicit geometric quantities such as coordinates or signed distances, so even convex or smooth interpolation offers no assurance of a smooth, valid decoded surface. Moreover, the appropriate seam depends on the surrounding source and edited structural context, not only on adjacent feature vectors. The interface is therefore formulated as a boundary-restricted SLat completion problem. On the final support, deterministic composition sets $\mathbf{z}_f^{\text{sh}}(\mathbf{p}) = \mathbf{z}_s^{\text{sh}}(\mathbf{p})$ for $\mathbf{p} \in V_s^{\text{out}}$ and $\mathbf{z}_f^{\text{sh}}(\mathbf{p}) = \mathbf{z}_e^{\text{sh}}(\mathbf{p})$ for $\mathbf{p} \in V_e^{\text{int}}$.

For each boundary query $\mathbf{p} \in V_e^{\text{bd}}$, separate source and edited contexts are constructed on the final sparse support. Let C_s contain tokens $(\mathbf{p}_i, \mathbf{z}_{s,i}^{\text{sh}})$ with $\mathbf{p}_i \in V_s^{\text{out}}$, and let C_e contain tokens $(\mathbf{p}_i, \mathbf{z}_{e,i}^{\text{sh}})$ with $\mathbf{p}_i \in V_e^{\text{int}}$. Keeping the two contexts separate exposes both boundary conditions to the predictor without averaging their features. The shape SLat seamer predicts

$$\hat{\mathbf{z}}^{\text{sh}}(\mathbf{p}) = f_\theta^{\text{sh}}(\mathbf{p}, C_s, C_e), \qquad \mathbf{p} \in V_e^{\text{bd}}. \tag{3}$$

The shape SLat seamer modifies only the boundary features in V_e^{bd}; the sparse support remains fixed by Eq. 1. Structure editing and boundary feature completion are therefore handled by separate stages. Because independently generated fields do not provide a unique ground-truth seam, f_θ^{sh} is learned through within-field masked completion. Given a complete decodable shape SLat, a narrow boundary-like band V^{bd} is hidden, and its features are reconstructed from the remaining context using

$$\mathcal{L}_{\text{shape}} = \frac{1}{|V^{\text{bd}}|} \sum_{\mathbf{p}_i \in V^{\text{bd}}} \text{SmoothL1}(\hat{\mathbf{z}}_i, \mathbf{z}_i^\star). \tag{4}$$

Training uses within-field contexts and hidden targets, whereas inference uses independently generated source and edited contexts; this formulation avoids imposing a unique target seam.

4.4 Texture SLat Harmonizer

The completed shape field then conditions the texture flow, which generates an edited texture field on the composed geometry. Even with shared spatial support, this field may remain incompatible with the independently generated source texture field. The discrepancy is addressed with a channel-wise field-level shift followed by a learned, mask-restricted residual. The analytic shift uses visible

non-edited correspondences to capture the dominant channel-wise field offset while preserving relative variation within the edited features. The learned residual then uses distance to the interface to model spatially varying mismatch unexplained by this offset, limiting freedom to rewrite the intended appearance.

Because the source and edited texture fields share coordinates in the non-edited region, the channel-wise shift is estimated directly from V_s^{out}:

$$\mathbf{s} = \frac{1}{|V_s^{\text{out}}|} \sum_{\mathbf{p} \in V_s^{\text{out}}} \left[\mathbf{z}_s^{\text{tex}}(\mathbf{p}) - \mathbf{z}_e^{\text{tex}}(\mathbf{p}) \right]. \tag{5}$$

For a target-region token $\mathbf{p}$, let $d_{\mathbf{p}}$ be the minimum number of 6-neighbor sparse-voxel hops from $\mathbf{p}$ to the inner boundary. A sparse message-passing network uses the source-visible field, the edited field, their observed exterior offset, and $d_{\mathbf{p}}$ to predict a residual $\mathbf{r}_\theta(\mathbf{p})$. The final texture feature is

$$\mathbf{z}_f^{\text{tex}}(\mathbf{p}) = \begin{cases} \mathbf{z}_s^{\text{tex}}(\mathbf{p}), & \mathbf{p} \notin M, \\ \mathbf{z}_e^{\text{tex}}(\mathbf{p}) + \mathbf{s} + \mathbf{r}_\theta(\mathbf{p}), & \mathbf{p} \in M. \end{cases} \tag{6}$$

The residual predictor r_θ is trained using independently generated texture SLat pairs Z_a^{tex} and Z_b^{tex} from the same conditioning input, which share semantics and active support. For a synthetic target region M, Z_a^{tex} serves as the reference and Z_b^{tex} as the field to align. The shift $\mathbf{s}$ is computed as in Eq. 5, using $V \setminus M$ as the non-edited region and substituting Z_a^{tex} and Z_b^{tex} for the source and edited fields. The target residual is $\boldsymbol{\delta}^\star(\mathbf{p}) = \mathbf{z}_a^{\text{tex}}(\mathbf{p}) - \mathbf{z}_b^{\text{tex}}(\mathbf{p})$, and supervision covers the complete synthetic target region:

$$\mathcal{L}_{\text{tex}} = \frac{1}{|M|} \sum_{\mathbf{p} \in M} \text{SmoothL1}(\mathbf{s} + \mathbf{r}_\theta(\mathbf{p}), \boldsymbol{\delta}^\star(\mathbf{p})). \tag{7}$$

Same-source pairs isolate cross-generation variation without entangling it with a semantic edit. Masking the reference features inside M prevents target leakage, and the same mask-restricted prediction scope is used at inference.

5 Experiments

5.1 Experimental Setup

Benchmark and baselines. SLatHarmony is evaluated on Edit3D-Bench Li et al. (2026), which contains 100 high-quality objects from Google Scanned Objects and PartObjaverse-Tiny, with three editing instances per object. Each instance includes an edit prompt, a target image, and a manually annotated 3D mask. Comparisons are conducted against the TRELLIS-based editing methods VoxHammer Li et al. (2026), Nano3D Ye et al. (2025), and Vinedresser3D Huang et al. (2026). All methods use the same source asset, target image, 3D mask, and evaluation views. SLatHarmony uses the 512-resolution TRELLIS.2 configuration with a 32^3 sparse grid, whereas the compared methods use 1024-resolution configurations with 64^3 sparse grids. Each baseline retains its original editing pipeline under these shared conditions. For SLatHarmony, the sparse-structure generation procedure of Vinedresser3D Huang et al. (2026) is adopted. Conditioned on the edited sparse structure, the pretrained TRELLIS.2 shape SLat flow generates the edited shape field, which is then regionally composed with the source shape field and processed by the shape SLat seamer. After shape harmonization, the pretrained TRELLIS.2 texture SLat flow generates the edited texture field on the composed shape, followed by texture composition and harmonization. For the component analysis, the edited sparse structure and the source and edited SLat fields are fixed, with only the interface treatment being varied.

Shape SLat seamer. The shape SLat seamer is an eight-layer transformer with hidden width 1,024 and 16 attention heads, operating on normalized 32-dimensional TRELLIS.2 shape features. Each context token is represented by its SLat feature, coordinate encoding, and field-origin embedding. Boundary coordinates serve as queries; self-attention couples boundary queries, while cross-attention retrieves contextual patterns from the two field interiors. Each side contributes at most 2,048 context tokens, and training uses at most 512 boundary queries. Inference uses all available context and all queries in the one-voxel-wide boundary band. The masked-completion objective

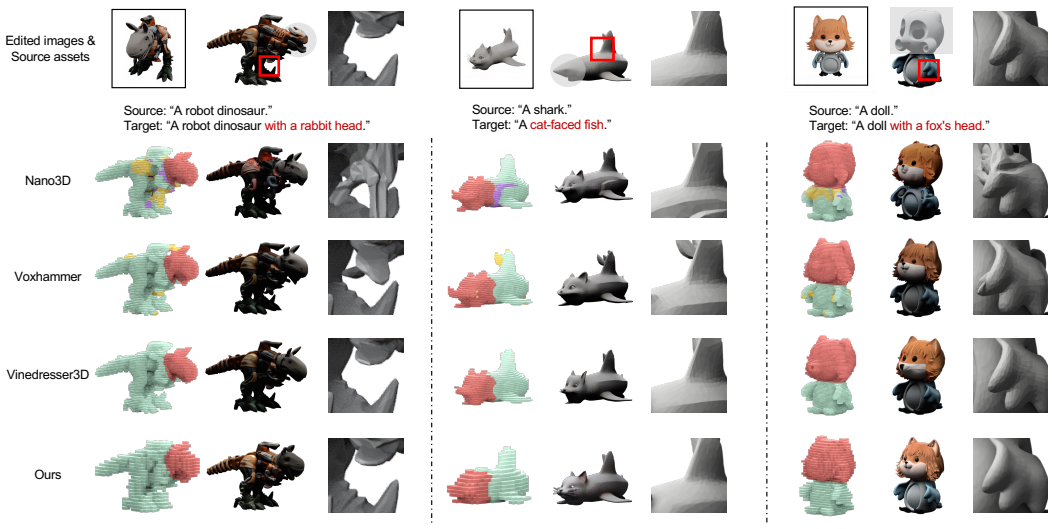

Figure 3: Qualitative comparison with VoxHammer Li et al. (2026), Nano3D Ye et al. (2025), and Vinedresser3D Huang et al. (2026) under the same target image and 3D mask. The voxel visualization shows structural changes relative to the source: red denotes voxels inside the edit mask, green denotes non-edited voxels that remain consistent with the source, yellow denotes spurious added voxels outside the intended edit, and purple denotes source voxels that are incorrectly removed.

in Eq. 4 is optimized using approximately 200k completion examples constructed from about 37k masked samples from Nano3D-Edit-100k Ye et al. (2025) with random-mask augmentation. Training seams have width one or two voxels, with layer-offset augmentation over $\{-2, -1, 0, 1, 2\}$. Training is performed on six NVIDIA A800 GPUs using AdamW with a learning rate of 10^{-4}, a batch size of 64 per GPU, and two-step gradient accumulation, corresponding to an effective batch size of 768.

Texture SLat harmonizer. The texture SLat harmonizer consists of eight sparse message-passing layers with hidden dimension 512. It records boundary distance up to eight sparse-voxel hops as an input feature but predicts the residual over the complete target region. The target-region objective in Eq. 7 is optimized using the same-source texture SLat pairs described in the method. The residual prediction head is zero-initialized, making the initial model equivalent to the analytic channel-wise shift alone. Training is performed on four NVIDIA A800 GPUs for 30k steps using AdamW with a peak learning rate of 2×10^{-4}, a batch size of 1 per GPU, and six-step gradient accumulation, corresponding to an effective batch size of 24. Further architectural and data-construction details are provided in Appendix A.

Metrics. Overall quality and semantic alignment with the edit instruction are measured by FID Heusel et al. (2017), LPIPS Zhang et al. (2018), and CLIP-T Radford et al. (2021), while preservation in the non-edited region is measured by Chamfer Distance (CD), regional LPIPS, and regional SSIM Wang et al. (2004). Each edited asset is rendered from eight shared viewpoints. For metrics requiring correspondence to the target image, DINOv2 image similarity (DINO-I) Oquab et al. (2024) is used to select the closest rendered viewpoint under the same protocol for all methods. Overall LPIPS is computed between the target image and the matched-view render. FID is computed over the sets of target images and matched-view renders, while CLIP-T is averaged over all rendered views. For preservation metrics, the source and edited assets are rendered from the same viewpoints, and the 3D edit mask is projected into each view to identify the non-edited image region. Regional LPIPS and SSIM are computed between the source and edited renders over these projected non-edited regions and averaged across views. Regional CD measures the geometric discrepancy between source and edited surface points outside the 3D edit mask.

Table 1: Quantitative comparison on Edit3D-Bench.

	Overall			Non-edited Region		
Method	FID ↓	LPIPS ↓	CLIP-T ↑	CD ($\times 10^{-3}$) ↓	LPIPS ↓	SSIM ↑
VoxHammer Li et al. (2026)	236.99	0.260	0.226	0.063	0.112	0.694
Nano3D Ye et al. (2025)	244.47	0.257	0.225	0.069	0.129	0.670
Vinedresser3D Huang et al. (2026)	254.03	**0.256**	0.231	0.055	0.113	0.692
Ours	**218.12**	0.257	**0.234**	**0.022**	**0.053**	**0.901**

Table 2: Quantitative component ablation on Edit3D-Bench.

	Overall			Non-edited Region		
Variant	FID ↓	LPIPS ↓	CLIP-T ↑	CD ($\times 10^{-3}$) ↓	LPIPS ↓	SSIM ↑
Feature interpolation + hard splicing	243.90	0.273	0.228	0.023	0.077	0.888
Shape seamer + hard splicing	248.39	0.273	0.230	**0.022**	0.060	0.896
Shape seamer + texture harmonizer	**218.12**	**0.257**	**0.234**	**0.022**	**0.053**	**0.901**

5.2 Main Results

Figure 3 compares SLatHarmony with existing TRELLIS-based editing methods under the same source asset, target image, 3D mask, and evaluation views. SLatHarmony preserves more non-edited structure and exhibits fewer unintended additions or removals outside the target region. The rendered results show the same preservation trend and qualitatively exhibit cleaner transitions around the target region.

This qualitative pattern is also reflected in Table 1, where SLatHarmony improves preservation in the non-edited region while remaining competitive in overall editing quality. In particular, it consistently improves structural and appearance preservation metrics outside the edit mask, while achieving the best FID and CLIP-T and a comparable overall LPIPS. These results indicate that directly retaining source support and latent content outside the target region reduces non-target changes without sacrificing semantic alignment with the edit instruction.

5.3 Ablation Studies

The proposed interface modules are isolated on Edit3D-Bench by fixing the source asset, target image, 3D mask, edited sparse structure, and source and edited SLat fields. Comparing feature interpolation with the shape SLat seamer under the same hard-splicing texture treatment isolates shape completion; replacing hard splicing with the texture SLat harmonizer under the same seamer isolates texture harmonization. Table 2 reports the resulting preservation and overall metrics, while Fig. 4 provides qualitative interface-level comparisons, where independently generated fields do not admit a unique ground-truth transition.

Shape. Feature interpolation enforces latent smoothness without guaranteeing a plausible decoded surface, whereas the shape SLat seamer predicts boundary features using context from the source and edited sides. It reduces regional LPIPS from 0.077 to 0.060 and increases regional SSIM from 0.888 to 0.896, although CD remains nearly unchanged and overall FID does not improve. At the narrow attachments in Fig. 4, the seamer adapts to the available context and produces qualitatively more plausible geometric transitions.

Texture. Hard splicing can expose appearance discontinuities between independently generated texture fields. Under the same shape SLat seamer, the texture SLat harmonizer improves FID from 248.39 to 218.12, overall LPIPS from 0.273 to 0.257, and regional LPIPS from 0.060 to 0.053, while leaving geometric CD unchanged. This is consistent with its texture-specific role: the channel-wise shift compensates for field-level offsets, and the learned correction is restricted to the target region while preserving source texture features elsewhere exactly.

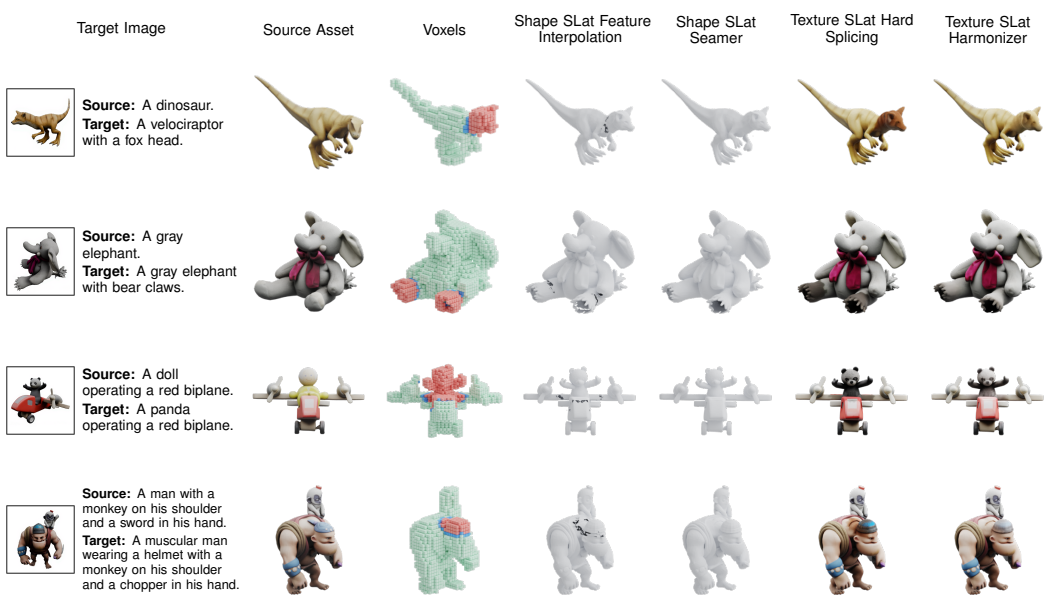

Figure 4: Qualitative component ablation. The shape SLat seamer produces more plausible local geometric transitions than feature interpolation, while the texture SLat harmonizer reduces visible appearance discontinuities compared with hard splicing.

5.4 Discussion and Limitations

SLatHarmony assumes that the upstream edit provides spatially aligned source and edited fields. Severe displacement, topology changes outside the intended region, or inaccurate correspondence remain registration problems that composition cannot resolve. The method repairs a local interface after the edited structure has been established; it does not register two arbitrary assets. The shape SLat seamer also faces a field-identity shift. It learns completion from two contexts within one field but receives contexts from independent generations at inference. The current objective captures local decodability, not the full variation between independently generated shape fields. Future objectives that explicitly model cross-field variation may better address this distribution shift.

The channel-wise texture shift preserves relative feature variation, but it may be insufficient for severe nonlinear appearance mismatch. A more expressive alignment model could handle those cases at the cost of greater freedom to alter the intended edited appearance. Our evaluation covers local edits with a single masked region. Multiple disjoint edits, large structural replacements, and general asset-to-asset composition would require broader correspondence and compatibility modeling.

6 Conclusion

We presented SLatHarmony for local 3D editing through regional SLat composition and cross-field harmonization. Spatial locality allows the method to retain source features outside the target region and use edited features inside it, avoiding reliance on regenerated features for content that should remain unchanged. Independent fields still require a compatible interface. The shape SLat seamer completes boundary features using context from the source and edited sides, and the texture SLat harmonizer aligns the two texture fields and corrects their residual mismatch within the target region. On Edit3D-Bench, SLatHarmony substantially improves preservation of non-edited geometry and appearance while maintaining semantic alignment with the edit instruction. The component analysis supports learned shape completion over feature interpolation and texture harmonization over direct hard splicing. These results suggest that structured 3D latents provide a promising representation for explicit compositional editing, not only for generation.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, manuscript organization, and formatting. They were not used to generate research ideas, experimental results, or scientific conclusions. All AI-assisted edits were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper.

REPRODUCIBILITY STATEMENT

The Method section provides the complete formulation of regional SLat composition, the shape SLat seamer, and the texture SLat harmonizer, including the support and boundary definitions, training objectives, and inference scope. The Experimental Setup section identifies the benchmark, baselines, pretrained models, and model resolutions; specifies the architectures, data construction and augmentation procedures, optimization hyperparameters, effective batch sizes, and training hardware; and describes the rendering protocol and all evaluation metrics. Equations 1–7 define the operations and losses used in the proposed pipeline, and all external datasets, models, and evaluation components are cited to their original sources. The implementation, including the training and inference code for the shape SLat seamer and texture SLat harmonizer, configuration files, and the complete editing pipeline, is provided as supplementary material.

REFERENCES

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space. arXiv preprint arXiv:2506.15742, 2025.

Hansheng Chen, Ruoxi Shi, Yulin Liu, Bokui Shen, Jiayuan Gu, Gordon Wetzstein, Hao Su, and Leonidas Guibas. Generic 3d diffusion adapter using controlled multi-view editing. arXiv preprint arXiv:2403.12032, 2024a.

Minghao Chen, Iro Laina, and Andrea Vedaldi. Dge: Direct gaussian 3d editing by consistent multi-view editing. In *European Conference on Computer Vision (ECCV)*, 2024b.

Minghao Chen, Roman Shapovalov, Iro Laina, Tom Monnier, Jianyuan Wang, David Novotny, and Andrea Vedaldi. Partgen: Part-level 3d generation and reconstruction with multi-view diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5881–5892, 2025.

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 (CVPR)*, pp. 21476–21485, 2024c.

Shaocong Dong, Lihe Ding, Xiao Chen, Yaokun Li, Yuxin Wang, Yucheng Wang, Qi Wang, Jaehyeok Kim, Chenjian Gao, Zhanpeng Huang, Zibin Wang, Tianfan Xue, and Dan Xu. From one to more: Contextual part latents for 3d generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8230–8240, 2025.

Ziya Erkoc, Can Gumeli, Chaoyang Wang, Matthias Niessner, Angela Dai, Peter Wonka, Hsin-Ying Lee, and Peiye Zhuang. Preditor3d: Fast and precise 3d shape editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 640–649, 2025.

Chongjian Ge, Chenfeng Xu, Yuanfeng Ji, Chensheng Peng, Masayoshi Tomizuka, Ping Luo, Mingyu Ding, Varun Jampani, and Wei Zhan. Compgs: Unleashing 2d compositionality for compositional text-to-3d via dynamically optimizing 3d gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18509–18520, 2025.

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 (ICCV)*, pp. 19740–19750, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

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.

Shimin Hu, Yuanyi Wei, Fei Zha, Yudong Guo, and Juyong Zhang. Easy3e: Feed-forward 3d asset editing via rectified voxel flow. arXiv preprint arXiv:2602.21499, 2026.

Zixuan Huang, Xiang Li, Yankuan Chi, and James M. Rehg. Vinedresser3d: Agentic text-guided 3d editing. arXiv preprint arXiv:2602.19542, 2026.

Shuyi Jiang, Qihao Zhao, Hossein Rahmani, De Wen Soh, Jun Liu, and Na Zhao. Gaussianblock: Building part-aware compositional and editable 3d scene by primitives and gaussians. In *International Conference on Learning Representations (ICLR)*, 2025.

Hayeon Kim, Ji Ha Jang, and Se Young Chun. Robust 3d-masked part-level editing in 3d gaussian splatting with regularized score distillation sampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5501–5510, 2025.

Juil Koo, Seungwoo Yoo, Minh Hieu Nguyen, and Minhyuk Sung. Salad: Part-level latent diffusion for 3d shape generation and manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14441–14451, 2023.

Dong In Lee, Hyeongcheol Park, Jiyoung Seo, Eunbyung Park, Hyunje Park, Ha Dam Baek, Sangheon Shin, Sangmin Kim, and Sangpil Kim. Editsplat: Multi-view fusion and attention-guided optimization for view-consistent 3d scene editing with 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11135–11145, 2025.

Lin Li, Zehuan Huang, Haoran Feng, Gengxiong Zhuang, Rui Chen, Chunchao Guo, and Lu Sheng. Voxhammer: Training-free precise and coherent 3d editing in native 3d space. In *IEEE International Conference on 3D Vision*, pp. 1281–1292, 2026.

Yuhan Li, Yishun Dou, Yue Shi, Yu Lei, Xuanhong Chen, Yi Zhang, Peng Zhou, and Bingbing Ni. Focaldreamer: Text-driven 3d editing via focal-fusion assembly. In *AAAI Conference on Artificial Intelligence*, pp. 3279–3287, 2024.

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *IEEE/CVF International Conference on Computer Vision*, pp. 9704–9715, 2025.

Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. In *International Conference on Learning Representations*, 2024.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11461–11471, 2022.

George Kiyohiro Nakayama, Mikaela Angelina Uy, Jiahui Huang, Shi-Min Hu, Ke Li, and Leonidas Guibas. Difffacto: Controllable part-based 3d point cloud generation with cross diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14257–14267, 2023.

Maxime Oquab, Timothee Darcet, Theo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Maria Parelli, Michael Oechsle, Michael Niemeyer, Federico Tombari, and Andreas Geiger. 3d-latte: Latent space 3d editing from textual instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14377–14386, 2026.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *International Conference on Learning Representations*, 2023.

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. arXiv preprint arXiv:2407.06191, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763, 2021.

Etai Sella, Gal Fiebelman, Peter Hedman, and Hadar Averbuch-Elor. Vox-e: Text-guided voxel editing of 3d objects. In *IEEE/CVF International Conference on Computer Vision*, pp. 430–440, 2023.

Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. In *International Conference on Learning Representations*, 2024.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pp. 1–18, 2024.

Konstantinos Tertikas, Despoina Paschalidou, Boxiao Pan, Jeong Joon Park, Mikaela Angelina Uy, Ioannis Emiris, Yannis Avrithis, and Leonidas Guibas. Generating part-aware editable 3d shapes without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4466–4478, 2023.

Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. Triposr: Fast 3d object reconstruction from a single image. *arXiv preprint arXiv:2403.02151*, 2024.

Junjie Wang, Jiemin Fang, Xiaopeng Zhang, Lingxi Xie, and Qi Tian. Gaussianeditor: Editing 3d gaussians delicately with text instructions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20902–20911, 2024.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *Advances in Neural Information Processing Systems*, 2023.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Minghao Wen, Shengjie Wu, Kangkan Wang, and Dong Liang. Intergsedit: Interactive 3d gaussian splatting editing with 3d geometry-consistent attention prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 26136–26145, 2025.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21469–21480, 2025.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3d generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14419–14429, 2026.

Yunhan Yang, Yukun Huang, Yuan-Chen Guo, Liangjun Lu, Xiaoyang Wu, Edmund Y. Lam, Yan-Pei Cao, and Xihui Liu. Sampart3d: Segment any part in 3d objects. arXiv preprint arXiv:2411.07184, 2024.

Junliang Ye, Shenghao Xie, Zhengyi Wang, Ruowen Zhao, 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.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

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*, 2023. doi: 10.1145/3610548.3618190.

Jingyu Zhuang, Di Kang, Yan-Pei Cao, Guanbin Li, Liang Lin, and Ying Shan. Tip-editor: An accurate 3d editor following both text-prompts and image-prompts. *ACM Transactions on Graphics*, 43(4):1–12, 2024.

A Implementation Details

This section describes the exact models and training procedures used in our experiments. Both modules operate on the 32-dimensional SLat features of the 512-resolution TRELLIS.2 configuration, whose sparse feature grid has resolution 32^3. The shape SLat seamer replaces features only in a one-voxel-wide boundary band, whereas the texture SLat harmonizer retains source texture features outside the target region exactly and predicts a correction only for target-region tokens.

A.1 Shape SLat Seamer

Boundary and context construction. We construct the interface on the composed sparse support using 6-neighbor connectivity. Layer zero consists of edited-side voxels adjacent to at least one source-side voxel. Signed negative and positive layers are then obtained by successively expanding into the edited and source sides, respectively. During training, the hidden band has width one or two, and its signed starting layer is sampled from $\{-2, -1, 0, 1, 2\}$. At inference, we use the edited-side layer-zero band with width one. Every band coordinate becomes a query. The remaining edited-side and source-side voxels form two context sets, and their origin is retained explicitly rather than being removed by pooling or interpolation.

For training efficiency, we sample at most 512 boundary queries and 2,048 context tokens from each side. At inference, the complete boundary band and all available context tokens are used. Coordinates are divided by 32 before encoding. Each SLat feature channel is normalized by its training-set mean and standard deviation; the predicted features are transformed back to the original TRELLIS.2 feature scale before composition and decoding.

Token encoding. A context token concatenates its normalized 32-dimensional shape SLat feature with a Fourier encoding of its 3D coordinate. We use six frequency bands $\{2^k\}_{k=0}^{5}$ together with the raw coordinate, producing a 39-dimensional coordinate code. A two-layer MLP maps the concatenated feature and coordinate code to width 1,024. A learned two-entry field-origin embedding is then added to distinguish context from the source and edited sides. Boundary queries contain no hidden ground-truth feature: a separate two-layer MLP maps only their coordinate codes to the same hidden width.

The seamer is additionally conditioned on the target image. We resize the image to 512×512, extract the complete 1,024-dimensional token sequence from a pretrained DINOv3 ViT-L/16 encoder Siméoni et al. (2025), and apply layer normalization followed by a learned linear projection. Image conditioning is dropped independently for 10% of training samples. This conditioning supplies semantic evidence for completing an ambiguous interface without changing the deterministic feature provenance outside the boundary band.

Transformer architecture. The model has eight transformer blocks, hidden width 1,024, 16 attention heads, an MLP expansion ratio of 4, and dropout 0.1. In every block, boundary queries first exchange information through self-attention. They then attend to the concatenated source and edited context tokens through cross-attention, followed by a separate cross-attention operation over the DINOv3 image tokens. All attention and feed-forward sublayers use pre-normalization and residual connections. The output head consists of layer normalization, a linear layer, GELU, and a final linear projection to 32 shape SLat channels. Context features are never overwritten by the network; only the query outputs are inserted into the composed field.

Training data and objective. The training set contains approximately 200k completion examples derived from about 37k masked Nano3D-Edit-100k samples. We supplement the provided masks with two procedurally generated masks per SLat. Their target-region ratio is sampled uniformly from 0.08 to 0.45, and the mask generator is selected from flood fill, ellipsoid, plane cut, and multiple ellipsoids with probabilities 0.50, 0.25, 0.15, and 0.10, respectively. Mask coordinates at resolution 64 are mapped to the 32^3 SLat grid before the boundary layers are constructed. The model reconstructs the held-out, normalized band features with the SmoothL1 objective in Eq. 4.

We optimize with AdamW using learning rate 10^{-4}, weight decay 0.01, $\beta_1 = 0.9$, $\beta_2 = 0.95$, and gradient-norm clipping at 1.0. Training uses automatic mixed precision on six NVIDIA A800

GPUs, a per-GPU batch size of 64, and two-step gradient accumulation, giving an effective batch size of 768.

Inference. We first form the final support using Eq. 1. Source shape features are copied outside the target region and edited shape features are copied in its interior. The one-voxel-wide edited-side boundary band is removed from this deterministic composition and predicted in a single forward pass. The source and edited context sets, the boundary queries, and target-image tokens are passed to the seamer; its denormalized outputs are written back at the query coordinates. Neither the sparse support nor any non-query feature is modified.

A.2 TEXTURE SLAT HARMONIZER

Same-source training pairs. We train with pairs of independently sampled texture SLat fields generated from the same conditioning input. The two fields have identical sparse support and semantics but differ in their latent realization. One field A is treated as the reference and the other field B as the texture field to be aligned. A compact target region containing 5–40% of the active tokens is formed by selecting the nearest tokens to a randomly sampled center in coordinate space. With probability 0.05, we set $A = B$ to regularize identity cases. We use data parts 0–17 for training, part 18 for validation, and part 19 for testing.

The sparse support graph connects 6-neighbor active voxels. For every token inside the synthetic target region, we compute its shortest graph distance to the inner boundary and clamp the result at eight hops. Crucially, the reference feature $A(\mathbf{p})$ and the observed offset $A(\mathbf{p}) - B(\mathbf{p})$ are set to zero inside the target region. Thus, the model cannot copy its supervised target through the input. Outside the target region, both quantities remain visible and define the observed cross-field relationship.

Analytic field-level alignment. For each training or inference sample, the analytic shift is the channel-wise mean of the visible exterior offset,

$$\mathbf{s} = \mathrm{mean}_{\mathbf{p} \notin M} \left[A(\mathbf{p}) - B(\mathbf{p}) \right]. \tag{8}$$

This translation is applied to every target-region token and constitutes a strong parameter-free alignment baseline. The learned network predicts an additional residual. Its final prediction is therefore $\mathbf{s} + \mathbf{r}_\theta(\mathbf{p})$ inside the target region and zero outside it. The zero-initialized output head makes the initial network exactly equivalent to shift-only composition.

Input representation and global context. Each sparse token receives the concatenation of three 32-dimensional vectors: the visible reference feature, the edited feature, and the visible exterior offset. We append a binary target-region indicator and the boundary distance normalized to $[0, 1]$, yielding a 98-dimensional token descriptor. A layer-normalized two-layer MLP projects it to hidden width 512. Coordinates are divided by 32, shifted by -0.5, encoded with four Fourier frequency bands, and projected by a second two-layer MLP; the two embeddings are added.

To describe the field-level offset explicitly, we compute per-channel means and standard deviations over exterior tokens for the concatenation $[A, B, A - B]$. The resulting 192-dimensional statistic is mapped to a 512-dimensional global context vector. This construction supplies both the average convention difference and its channel-wise variation while excluding reference values from the target region.

Sparse message-passing network. The harmonizer contains eight residual message-passing blocks of width 512, with an MLP expansion ratio of 2 and no dropout. Within a block, hidden features are layer-normalized, and each token receives the mean of its 6-neighbor features. Separate linear maps transform the token itself and the neighbor aggregate. The global context produces per-channel affine parameters; its scale is bounded as $1 + 0.1 \tanh(\cdot)$ and its shift is unconstrained. The conditioned message is passed through GELU and a linear projection and added residually, followed by a pre-normalized two-layer residual MLP. After eight blocks, layer normalization and a zero-initialized linear head produce a 32-dimensional correction. We do not apply a distance gate or a magnitude clamp to this correction; boundary distance is instead provided as an input cue, and the output is masked so that non-edited content remains unchanged.

Objective and optimization. The target is the complete 32-channel residual $A(\mathbf{p}) - B(\mathbf{p})$ over the synthetic target region. We use a uniformly weighted SmoothL1 loss with transition parameter 0.1, as written in Eq. 7; no channel reweighting is applied. The model is trained for 30k optimizer steps on four NVIDIA A800 GPUs with bfloat16 mixed precision. AdamW uses a peak learning rate of 2×10^{-4}, weight decay 0.01, $\beta_1 = 0.9$, $\beta_2 = 0.95$, and gradient-norm clipping at 1.0. We linearly warm up for 1,000 steps and then use cosine decay to 10% of the peak learning rate. The per-GPU batch size is 1 with six-step gradient accumulation, giving an effective batch size of 24. The final model has 19.89M parameters.

Inference. The edited texture SLat is generated on the already composed and harmonized shape support. We align the source texture field to this support, retain source features exactly outside the target region, and construct the same visibility-masked inputs used during training. After estimating the exterior channel shift and running the sparse network once, we add the predicted 32-channel correction to edited texture features inside the target region. The source field is then restored outside the mask before decoding, so the texture SLat harmonizer cannot modify non-edited texture tokens.

B Additional Qualitative Results

Figures 5–7 provide additional qualitative results across diverse object categories and target-region edits. For each example, we show the target image and instruction, the source asset, and three views of the edited asset. The multi-view results illustrate that the intended change remains spatially localized while non-edited content is preserved across viewpoints. They also show that the composed shape SLat and texture SLat remain coherent beyond the single presentation view used in the main paper.



Figure 5: Additional multi-view qualitative results (Part I). Each row shows the target image and edit instruction, the source asset, and three views of the SLatHarmony result. The examples cover object replacement, part addition, and local structural editing.

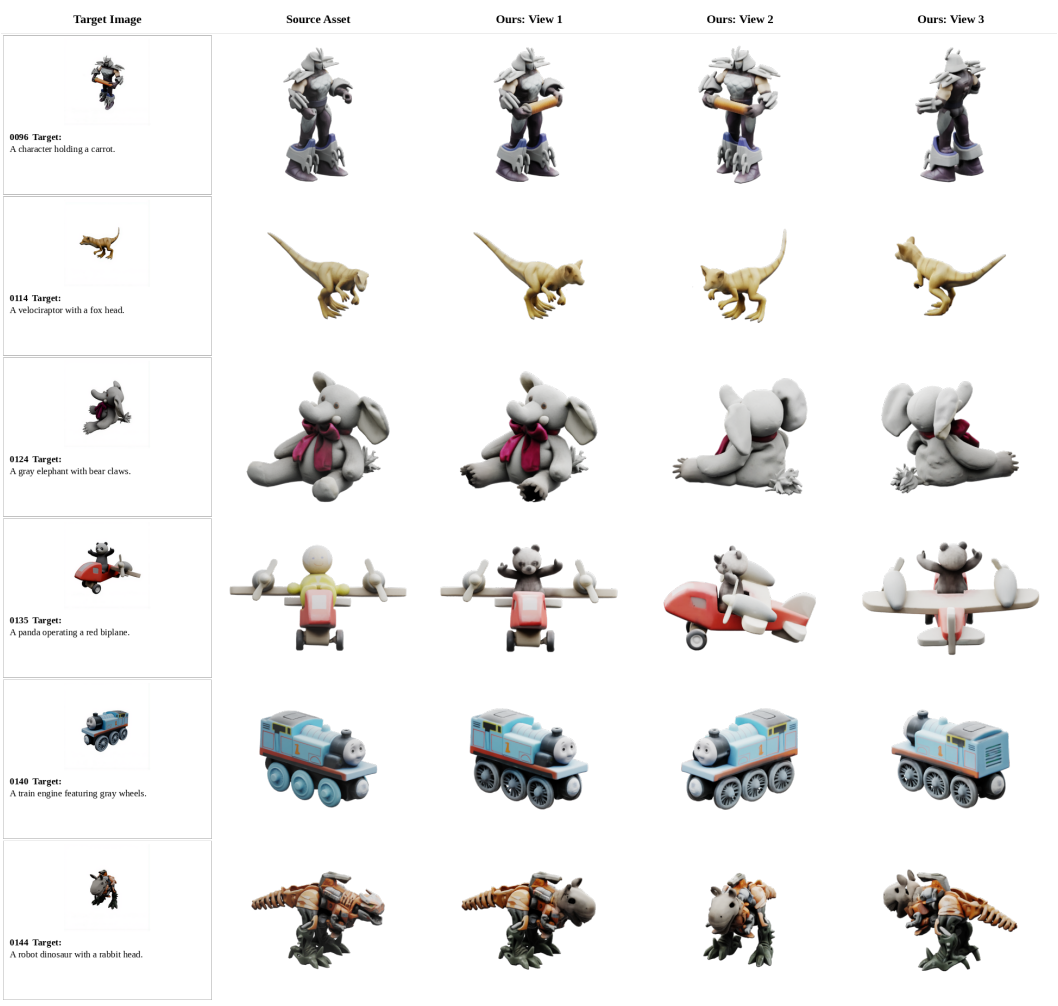

Figure 6: Additional multi-view qualitative results (Part II). SLatHarmony preserves non-edited content while producing target-region changes that remain consistent across viewpoints.

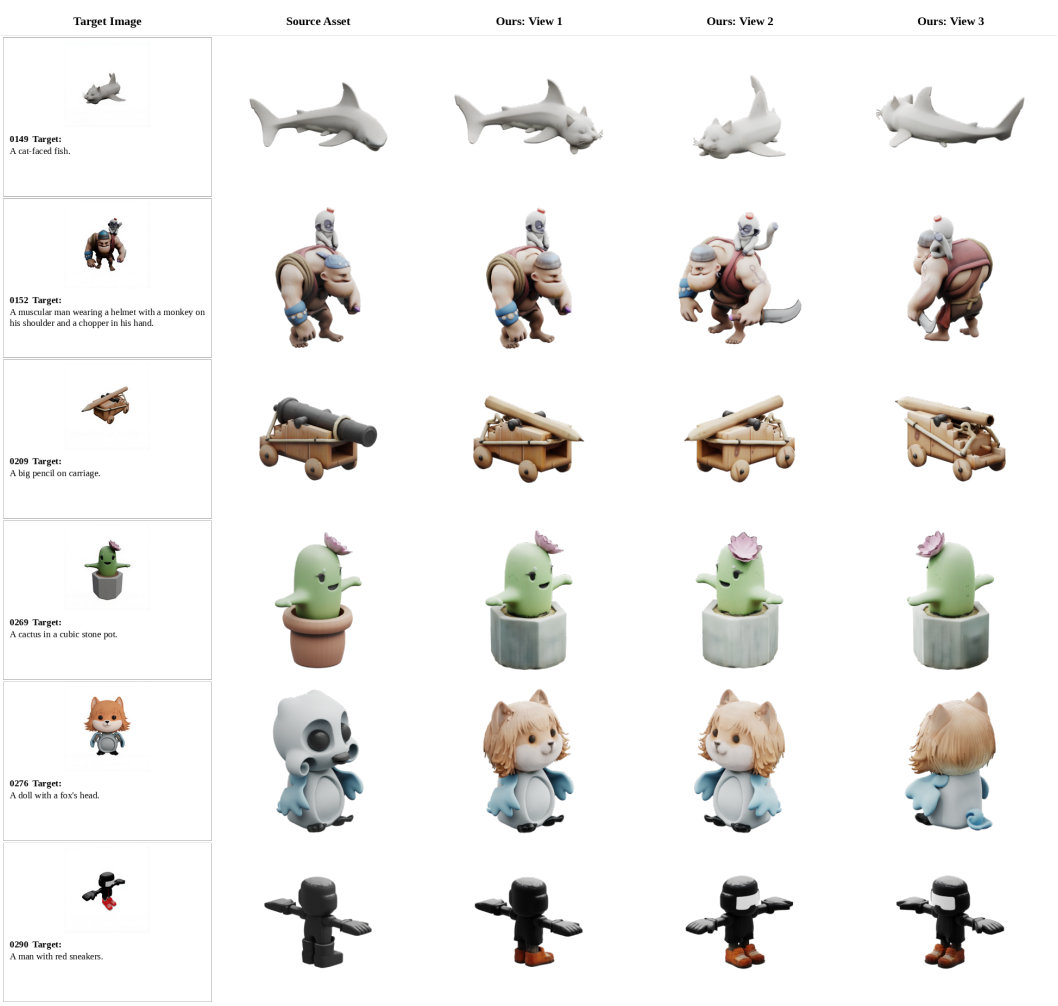

Figure 7: Additional multi-view qualitative results (Part III). These examples include local semantic replacement, articulated objects, and edits involving distinct source and target appearances.