SCOPE3D: TRAINING-FREE 3D EDITING WITH STAGE-AWARE CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Generating and editing 3D assets is essential for digital content creation. Yet translating user intent into precise modifications while preserving unrelated geometry and appearance remains challenging. We present a training-free framework that interprets editing instructions and selects appropriate generative operations within a structured 3D model. An instruction-conditioned agent determines the spatial scope and representation requirements of each edit, guiding the choice of generative stages and preservation constraints. When local preservation is needed, the framework estimates an edit mask from the source and edited images and maps it to the source 3D representation through the generator's native cross-attention. The resulting spatial associations guide source-reference-constrained sampling, preserving designated content while allowing edited geometry to extend beyond the original occupied space. We examine the framework across geometry and appearance editing tasks, evaluating region localization, edit fulfillment, and preservation of source content.



Figure 1: **Instruction-guided, training-free 3D editing.** Left: image editing, route selection, and attention-guided 3D editing; local routes protect inferred keep regions. Right: source (top) and edited (bottom) assets illustrate four routes, with camera and scale matched within each pair.

1 INTRODUCTION

3D assets underpin games, digital design, and interactive environments. Their shapes, components, and materials must often be revised as design requirements evolve. Recent progress in image-conditioned 3D generation has made images an effective interface for creating assets (Xiang et al., 2025a;b), while advances in image editing allow users to express visual changes through natural-language instructions and visual guidance (Brooks et al., 2023; Black Forest Labs et al., 2025; Wu et al., 2025; OpenAI, 2025; Google, 2025). Together, these developments offer an accessible route to 3D editing: edit an image of an asset, then transfer the intended modification to its 3D representation.

In 2D, instruction-based editing can be learned from large collections of source images, editing instructions, and target images, with recent models accommodating diverse transformations within a shared architecture (Brooks et al., 2023; Wu et al., 2025; Black Forest Labs et al., 2025). Extending this paradigm to 3D remains more demanding: aligned before-and-after assets with editing

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.

instructions are harder to acquire at scale, and training must capture changes in spatial geometry and surface appearance rather than pixels alone (Ye et al., 2025; Li et al., 2025a). These challenges motivate reusing pretrained 3D priors without additional paired-edit training.

Recent training-free methods follow this direction through different mechanisms for controlling generation. VoxHammer combines edited-image guidance with a specified 3D region and restores source features outside that region (Li et al., 2025a). Nano3D adapts flow-based editing to 3D generation and merges voxel structures and latent features to retain source content (Ye et al., 2025). VecSet-Edit builds on TripoSG (Li et al., 2025b) and automatically identifies editing regions for localized 3D editing (Hsiao et al., 2026). Vinedresser3D uses an instruction-driven agent to coordinate editing tools and selects editable parts from PartField segmentations using a multimodal language model (Chi et al., 2026). We study how editing intent should determine both generative operations and spatial preservation constraints.

We build on TRELLIS, a strong pretrained image-conditioned generator with a structured representation (Xiang et al., 2025a). Sparse occupancy defines spatial support, while structured features encode detail and appearance. This separation supports different interventions for structural and appearance changes; the request's local or global scope determines whether preservation constraints are needed.

Local editing additionally requires a spatial region. Specifying it directly in 3D adds interaction overhead, especially for occluded or interleaved parts; inferring it from one edited image is ambiguous because visible changes reveal only part of its extent. For example, a request to change a car's wheels may concern both sides even when only one side is visible. Restricting an edit to visible surfaces can leave relevant hidden parts unchanged, whereas extending the region indiscriminately risks modifying unrelated geometry.

We present a training-free framework that uses an instruction-conditioned agent to select spatial scope and active generation stages. Local routes lift image-space changes into source edit/keep regions through normalized foreground and background cross-attention. Source-reference constraints protect keep regions while allowing structural changes in the source edit region and originally empty space. Global routes omit these local constraints. Structural edits modify occupancy and its associated features; detail and appearance edits reuse occupancy. Image editing remains a modular frontend, and all generative weights remain frozen.

Our contributions are summarized as follows:

- **An instruction-guided, stage-aware editing framework.** We organize pretrained 3D generative operations according to the spatial scope and representation requirements of an editing request, supporting local revisions and global transformations of structure, geometric detail, and appearance without additional model training.
- **Attention-guided spatial constraints for local editing.** We combine normalized foreground and background cross-attention associations to infer source edit and keep regions from image-space changes. These regions guide source-reference-constrained sampling, protecting designated content while allowing structural modifications beyond the original occupied space.
- **A comparative study of editing and spatial control.** We examine the framework across local and global geometry and appearance edits through comparisons with existing 3D editing methods, visualizations of inferred regions, and measurements of source-content preservation.

2 Related Work

2.1 Generative Image Editing

Image editing controls changes within spatially organized pixels, latent features, or attention maps (Hertz et al., 2023; Lugmayr et al., 2022; Kulikov et al., 2025). TRELLIS similarly attaches features to explicit 3D coordinates, providing a basis for adapting these controls to native asset editing (Xiang et al., 2025a). The transfer must nevertheless account for changing occupancy and geometry hidden from the conditioning image.

Training-free methods regulate departures from a source through sampling interventions. SDEdit uses noise perturbation, Null-text Inversion improves source reconstruction, and Prompt-to-Prompt controls cross-attention to retain spatial organization (Meng et al., 2022; Mokady et al., 2023; Hertz et al., 2023). RePaint replaces known regions with noise-matched references, while FlowEdit constructs an editing flow without explicit inversion (Lugmayr et al., 2022; Kulikov et al., 2025). These mechanisms inform our attention control, source-reference constraints, and global editing routes.

Learned editors instead acquire instruction-following behavior from paired examples. InstructPix2Pix, MagicBrush, and UltraEdit develop this data-driven direction (Brooks et al., 2023; Zhang et al., 2023; Zhao et al., 2024); FLUX.1 Kontext and Qwen-Image accommodate diverse generation and editing tasks within shared architectures (Black Forest Labs et al., 2025; Wu et al., 2025). GPT-4o and Gemini 2.5 Flash Image further support visual modification in multimodal systems (OpenAI, 2025; Google, 2025). Such models provide the edited images used as visual targets in image-guided 3D workflows.

2.2 3D Asset Editing

Image-prior-based approaches connect edited observations to a 3D representation: Vox-E uses diffusion guidance with volumetric regularization, while Instruct-NeRF2NeRF alternates view editing and radiance-field optimization (Sella et al., 2023; Haque et al., 2023). Native generators such as TRELLIS, TRELLIS.2, and TripoSG enable interventions directly in the 3D generation process (Xiang et al., 2025a;b; Li et al., 2025b). VoxHammer combines inversion with source-feature replacement; Nano3D uses flow editing and occupancy/feature merging; VS3D intervenes in velocity space; and 3D-LATTE applies attention control to native Gaussians (Li et al., 2025a; Ye et al., 2025; Liu et al., 2026; Parelli et al., 2026).

These approaches also differ in how editing regions are obtained. VoxHammer accepts a specified 3D region, Nano3D analyzes generated occupancy differences, and 3DMorph estimates bounds from image differences and camera information (Li et al., 2025a; Ye et al., 2025; Preintner et al., 2026). VecSet-Edit uses cross- and self-attention in TripoSG to identify editable latent tokens (Hsiao et al., 2026). Vinedresser3D coordinates tools through an agent and selects parts from PartField segmentations, while EditFlow3D constructs visual and spatial controls and applies trajectory-preserving guidance (Chi et al., 2026; Nie et al., 2026). We connect instruction-guided stage selection with an attention-derived partition of structured source occupancy: local routes apply reference constraints to retained regions, while global routes use the corresponding stages without a local keep constraint.

3 Problem Formulation and Preliminaries

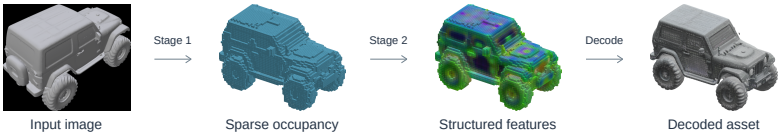
3.1 Image-Guided Asset Editing

Given a source asset $\mathcal{A}_s$, image I_s, and instruction y, we seek an edited asset $\mathcal{A}_e$. An image editor produces $I_e = \mathcal{G}_{\text{img}}(I_s, y)$. For local requests, pixel-wise differences between I_s and I_e are thresholded within source foreground M_s to obtain the image-space edit region $E_2 = \mathcal{D}(I_s, I_e)$, with keep region $K_2 = M_s \setminus E_2$ and background patches B. These frozen, modular components provide visual conditions for the 3D operations.

Local requests preserve part of the source; global requests omit this spatial constraint. This distinction is independent of whether occupancy or only structured features must change. We reuse the source representation (O_s, Z_s) when available from generation.

3.2 Structured 3D Generation

Conditioned on an input image, TRELLIS generates an asset through sparse structure and structured latent features (Xiang et al., 2025a). Let Ω be the voxel domain and $O \subset \Omega$ its occupied coordinates. Stage 1 generates a dense structure latent x^{str}, decoded to O. Stage 2 generates features $Z = \{z_p : p \in O\}$ on these coordinates, and an output decoder produces $\mathcal{A} = D(O, Z)$. Occupancy determines spatial support, while Z encodes geometric detail and appearance. Consequently, fixing occupancy does not strictly fix the decoded surface. The inset summarizes these stages; feature colors denote PCA.



Both stages use conditional rectified flows. We use $t = 1$ for noise and $t = 0$ for a clean latent, with the reference interpolation

$$x_t = (1 - t)x_0 + t\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I). \tag{1}$$

A learned velocity $v_\theta(x_t, t, c)$ transports the state under image condition c. Image information enters through cross-attention,

$$A = \mathrm{softmax}\left(QK^\top/\sqrt{d}\right), \tag{2}$$

where spatial latent queries form Q and image tokens form K and V. Rows of A associate spatial queries with image patches, providing the basis for our spatial constraints.

4 METHOD

Our framework routes image-guided edits through the structure and feature stages of TRELLIS (Figure 2). Local routes share attention-derived spatial constraints; global routes operate without a local keep region. All generative weights remain frozen.

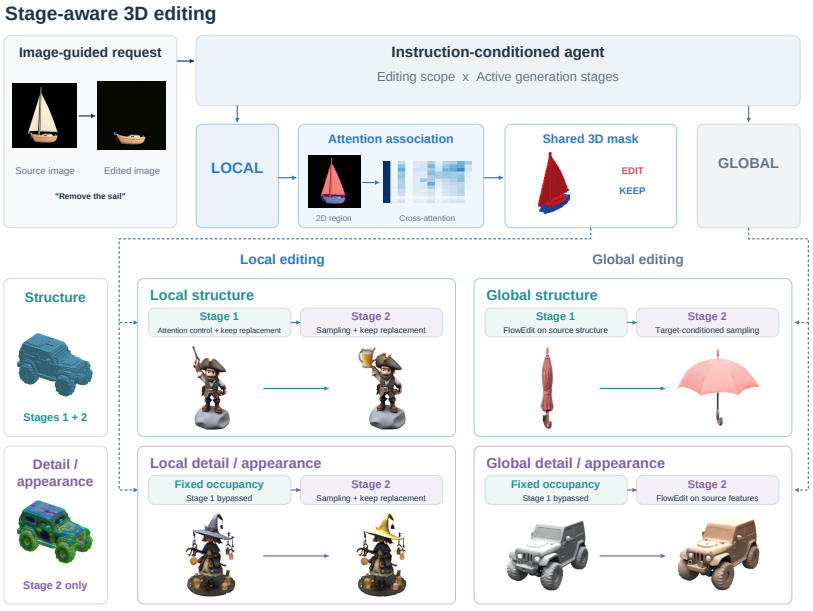


Figure 2: **Four editing routes.** Columns distinguish local and global scope; rows distinguish structure and detail/appearance. Both local routes share attention-based localization and preserve keep regions in their active stages. Global routes use FlowEdit at the corresponding representation level.

4.1 INSTRUCTION-GUIDED EDIT ROUTING

Our routing agent, Gemini-2.5-Flash (Google, 2025), returns $r = (g, \ell)$, where $g \in \{0, 1\}$ indicates whether spatial support must change and $\ell \in \{0, 1\}$ indicates whether local preservation is required. Material edits can retain occupancy, whereas structural changes may require new occupied coordinates.

Using the instruction and source/target images, the agent selects a route in Table 1. Structural routes modify both stages; detail/appearance routes reuse O_s and modify Stage 2. Local routes share one inferred edit/keep partition; global routes omit this constraint.

Table 1: Editing routes determined by scope and representation requirements.

Route	Sparse structure (Stage 1)	Structured features (Stage 2)
Local structural	Attention-guided structure sampling with reference replacement	Coordinate-aligned feature replacement
Global structural	Structure-latent FlowEdit	Target-conditioned feature sampling
Local detail/appearance	Fixed source occupancy	Coordinate-aligned feature replacement
Global detail/appearance	Fixed source occupancy	Feature-latent FlowEdit

4.2 Attention-Guided Spatial Localization

Source-conditioned Stage 1 attention associates the estimated image region with source occupancy. For assets generated from I_s, maps cached or replayed along the source trajectory can be reused across edits in the same image coordinates. Let $A_{ij}^{t,l,h}$ denote attention from spatial query i to image patch j at time t, layer l, and head h. After excluding non-spatial image tokens, we aggregate a selected set $\mathcal{T}$ of these maps:

$$\bar{A}_{ij} = \frac{1}{|\mathcal{T}|} \sum_{(t,l,h)\in\mathcal{T}} A_{ij}^{t,l,h}. \tag{3}$$

The source attention is independent of the requested edit region and can therefore be reused for different edits of the same source. We score only coarse spatial cells containing source occupancy, denoted by $\mathcal{C}_s$. The image masks are mapped to the spatial image-token grid before pooling.

Occupied queries attend to both foreground and background patches. We partition the latter by proximity to the image-space edit and keep regions:

$$B_E = \{j \in B : d(j, E_2) < d(j, K_2)\}, \qquad B_K = B \setminus B_E, \tag{4}$$

where d is image-plane distance on the patch grid. The foreground and background ratios are

$$r_i^{\mathrm{fg}} = \frac{\sum_{j\in E_2} \bar{A}_{ij}}{\sum_{j\in E_2\cup K_2} \bar{A}_{ij} + \varepsilon}, \qquad r_i^{\mathrm{bg}} = \frac{\sum_{j\in B_E} \bar{A}_{ij}}{\sum_{j\in B} \bar{A}_{ij} + \varepsilon}. \tag{5}$$

These ratios have different distributions. We standardize each over $\mathcal{C}_s$ before combining them:

$$s_i = \mathcal{N}(r^{\mathrm{fg}})_i + \mathcal{N}(r^{\mathrm{bg}})_i, \qquad \mathcal{N}(r)_i = \frac{r_i - \mu_{\mathcal{C}_s}(r)}{\sigma_{\mathcal{C}_s}(r) + \varepsilon}. \tag{6}$$

The scores rank edit associations rather than calibrated segmentation probabilities.

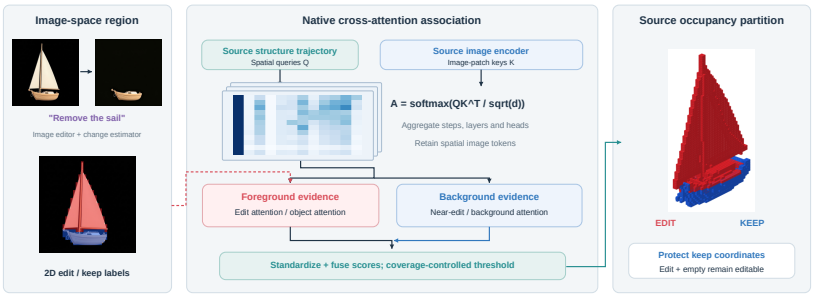


Figure 3: **Image-to-3D region association.** Cross-attention connects image patches to spatial queries. Foreground/background pooling and thresholding yield red edit and blue keep regions. The boat uses recorded data; the layer stack is schematic.

A coverage-controlled quantile converts scores into a coarse mask $m_i = \mathbb{1}[s_i \geq Q_{1-q}(s)]$. Expanding coarse cells to occupancy coordinates with U gives

$$E_3 = O_s \cap U(m), \qquad K_3 = O_s \setminus E_3. \tag{7}$$

Coverage is initialized from the visible edit fraction using the rule in Appendix A. Image area need not reflect the extent of hidden target parts.

4.3 Spatially Constrained Generation

Editable support. Explicit editing volumes, as in VoxHammer (Li et al., 2025a), must cover both the source part and its anticipated extension. We instead preserve K_3 and allow changes in its complement (Figure 4), including originally empty space:

$$F = \Omega \setminus K_3 = E_3 \cup (\Omega \setminus O_s). \tag{8}$$

The complement is taken within the fixed generation domain Ω; it does not permit growth beyond that domain. An optional coarse keep buffer additionally protects neighboring empty cells (Appendix A).



Figure 4: **Anticipating an edit volume versus specifying a source region.** VoxHammer requires a box that covers both the source part and the anticipated new geometry. Our complement-based formulation only requires an edit region on the source asset: preserving the remaining surface leaves both the selected region and empty space editable. The axe illustrates the difference in required spatial specification.

Structural editing. Stage 1 initially reuses source attention weights with target-image values, $A_s V_e$, following the correspondence-control principle of Prompt-to-Prompt (Hertz et al., 2023). Source and target trajectories use the same initial noise. At later steps, source-reference replacement constrains the retained structure. Let $x_K = \mathrm{Enc}_{\mathrm{SS}}(O_s \cap F^c)$ be the encoded protected occupancy and P_K select coarse latent cells containing retained occupied voxels. Before designated velocity evaluations, we replace the current state by

$$\widetilde{x}_t = P_K \odot [(1-t)x_K + t\epsilon] + (1 - P_K) \odot x_t, \tag{9}$$

with fixed noise ϵ, adapting noise-matched replacement (Lugmayr et al., 2022) to the flow trajectory.

Because a latent constraint does not imply an exact decoded occupancy constraint, we correct the decoded structure $\widehat{O}_e$:

$$O_e = (\widehat{O}_e \cap F) \cup (O_s \cap F^c). \tag{10}$$

This restores protected binary occupancy, including protected empty cells when buffering is used. It does not guarantee an identical final continuous surface after feature generation and decoding.

Detail and appearance editing. Stage 2 generates features on O_e. At retained source coordinates, noise-matched source features replace the corresponding current state:

$$\widetilde{z}_t(p) = \begin{cases} (1-t)z_s(p) + t\epsilon_p, & p \in O_e \cap K_3, \\ z_t(p), & \text{otherwise.} \end{cases} \tag{11}$$

Features use the generator's normalized coordinates; new coordinates remain unconstrained. Local detail/appearance edits apply the same operation with $O_e = O_s$. Replacement is released near the end of sampling to permit interface adaptation.

Global editing. Global structural editing applies FlowEdit (Kulikov et al., 2025) to the source structure latent and generates Stage 2 features under the target image. Global detail/appearance editing applies FlowEdit to source features at fixed source coordinates. These routes retain a source-conditioned editing process but omit local edit/keep constraints. FlowEdit is a reused component; our formulation determines when and at which representation level it is applied.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Implementation details. We use frozen TRELLIS models and 25-step sampling schedules. Local editing uses CFG 5; structural edits reuse source attention for 15 steps, and Stage 2 releases keep replacement for its final two steps. Global FlowEdit uses source/target CFG 1.5/5.5, averaging five noise samples per update.

We evaluate sixteen editing requests, with four cases in each category: local and global edits to structure and appearance. Source assets and edited target images are shared where supported by each method. The comparisons evaluate the 3D editing backend with fixed route assignments and image-space regions, rather than frontend prediction accuracy. We compare released Nano3D (Ye et al., 2025), VoxHammer (Li et al., 2025a), and VecSet-Edit (Hsiao et al., 2026) pipelines. For VoxHammer, we corrected 3D coordinate matching and handled the reference-replacement error when the keep region is empty, and restored the recorded input normalization on export.

We evaluate whole-asset alignment and edit consistency using twelve matched views (Li et al., 2025a; Hsiao et al., 2026; Parelli et al., 2026), averaging views then cases. Preservation uses the eight local requests; protocols are in the appendix.

5.2 IMAGE-GUIDED 3D EDITING COMPARISONS

Figure 5 presents all sixteen requests in four groups. Source and edited assets use matched viewing directions within each case. Nano3D and VoxHammer repeatedly leave requested changes unrealized: Nano3D retains the source in several appearance edits, while VoxHammer fails to complete edits such as Empty Cone and Cake Topping. In our global-editing comparisons, VecSet-Edit outputs can also depart from the source orientation or scale, reducing consistency with the original asset.

Table 2 summarizes text/image alignment (Radford et al., 2021; Caron et al., 2021) and cross-view edit consistency over all sixteen requests. Our method achieves the highest mean CLIP-T, CLIP-I, and DINO-I. VoxHammer leads Dir-Con, which does not alone establish edit fulfillment.

Table 2: Whole-asset comparison: means over sixteen cases and twelve views per asset. Feature scores are higher-is-better cosine similarities.

Method	CLIP-T ↑	CLIP-I ↑	DINO-I ↑	Dir-Con ↑
Ours	**0.3577**	**0.8845**	**0.7273**	0.5776
Nano3D	0.3290	0.8217	0.5971	0.5456
VoxHammer	0.3558	0.8678	0.6820	**0.5987**
VecSet-Edit	0.3558	0.8745	0.7249	0.5370

Across eight local requests, ours achieves the best mean CD, PSNR, SSIM, and LPIPS (Table 3); Nano3D achieves the lowest HD95. These diagnostics use shared predicted keep regions, not independent ground truth.

5.3 SPATIAL LOCALIZATION AND ITS EFFECT ON EDITING

We compare our normalized foreground–background attention with 3DMorph, Nano3D, and VecSet-Edit on eight local structural and appearance requests. Figure 6 shows edit/keep partitions under matched cameras. Ours and VecSet-Edit share 2D regions, 3DMorph receives derived boxes,

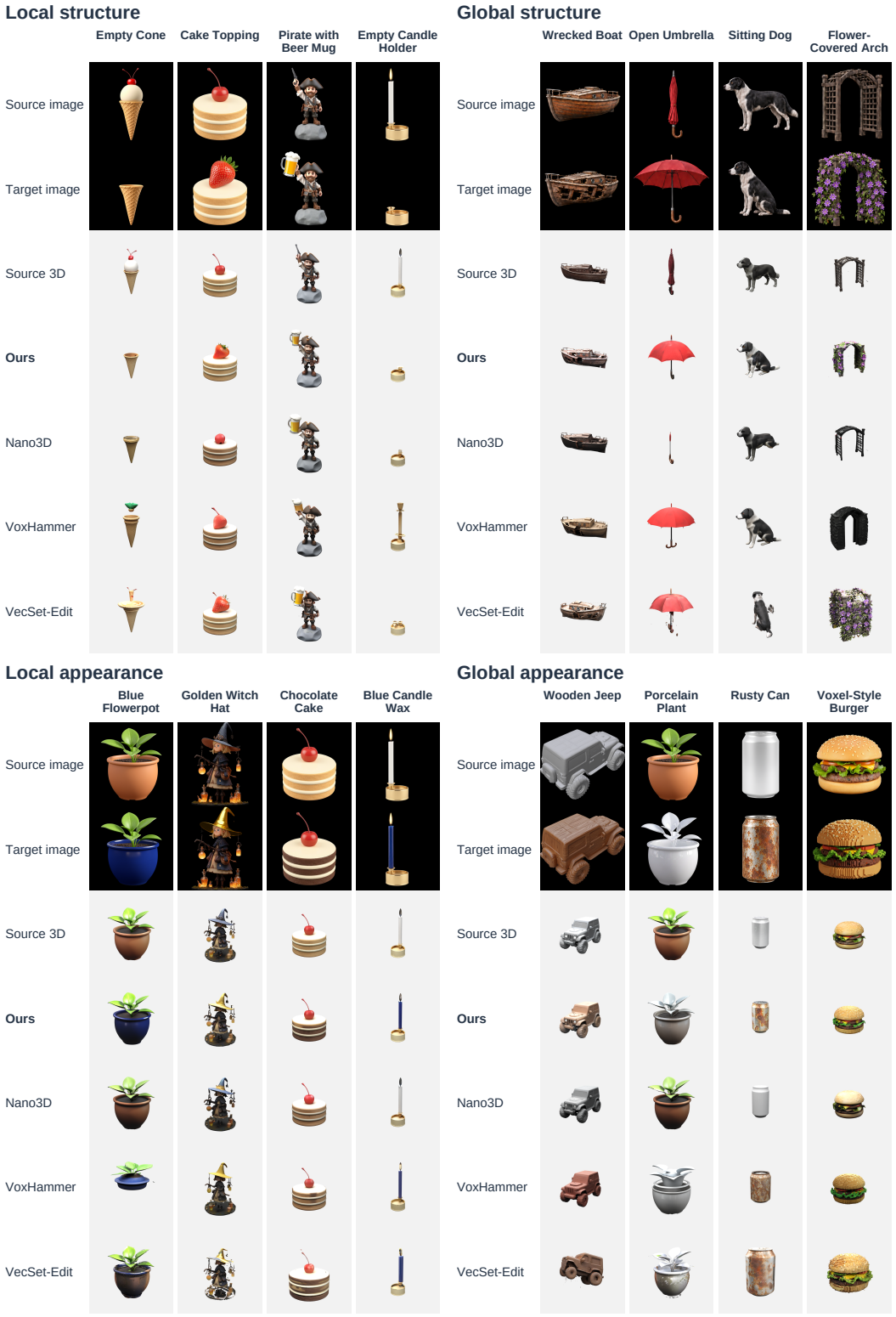


Figure 5: **Image-guided 3D editing on sixteen requests.** Four cases per route compare source and target images, the source asset, and edited assets. Views share camera orientations. VecSet-Edit outputs use its released texture-baking pipeline.

Table 3: Preservation averaged over all eight local requests. CD/HD95 are percentages of source diagonal; appearance uses matched views and shared predicted keep regions, not independent labels.

Method	CD ↓	HD95 ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Ours	**0.0976**	0.7404	**27.764**	**0.9119**	**0.0257**
Nano3D	0.1064	**0.3859**	17.315	0.7284	0.1438
VoxHammer	0.6681	2.6230	14.188	0.5659	0.2166
VecSet-Edit	0.4570	4.2683	19.316	0.7702	0.1046

and Nano3D uses occupancy differences. VecSet-Edit selections are mapped to reconstructed source surfaces through decoder associations; the others use source occupancy. We inspect target coverage and leakage.

Ours gives the most consistent separation in these examples. VecSet-Edit is the closest alternative, but repeatedly labels small target portions as keep, including cone-topping extensions and flowerpot boundaries, constraining content that should remain editable. 3DMorph includes adjacent cake content, while Nano3D misses several appearance targets. These qualitative findings concern the illustrated requests, rather than independently annotated 3D accuracy.

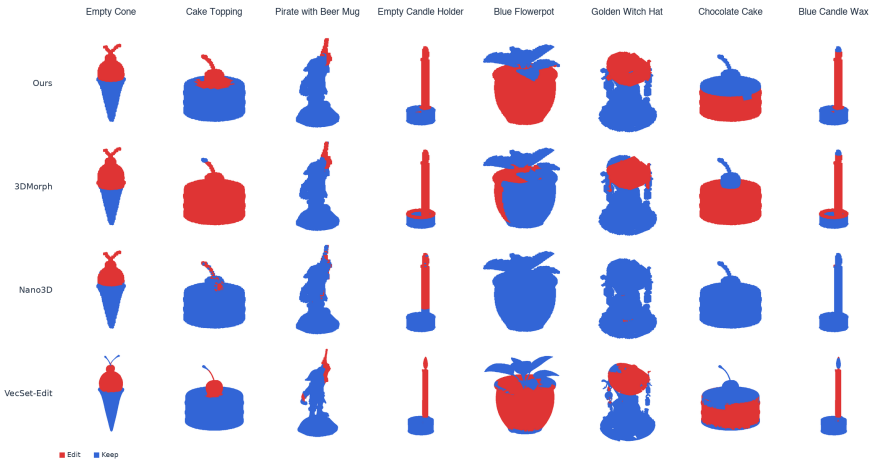


Figure 6: **Region localization on eight local requests.** Columns match Figure 5. Red denotes edit and blue keep; matched views provide a qualitative comparison.

6 Conclusion

Scope3D combines instruction-based planning, stage-aware generation, and attention-derived constraints for training-free 3D editing. Matching requests to a pretrained generator's stages supports local and global changes to structure and appearance. Local routes preserve attention-identified source regions while leaving other content and unoccupied space editable. Experiments illustrate diverse modifications with preservation of unrelated content.

Localization depends on image-space regions and generator attention, while latent preservation does not guarantee identical decoded surfaces. Future directions include multi-view attention aggregation, uncertainty-aware localization, and decoder-aware preservation constraints. Rendered feedback could refine routing and spatial constraints; broader evaluation with independent annotations is needed to assess generalization.

REFERENCES

Jianfeng Xiang et al. Structured 3D Latents for Scalable and Versatile 3D Generation. CVPR, pages 21469–21480, 2025. https://arxiv.org/abs/2412.01506.

Amir Hertz et al. Prompt-to-Prompt Image Editing with Cross-Attention Control. ICLR, 2023. https://arxiv.org/abs/2208.01626.

Andreas Lugmayr et al. RePaint: Inpainting Using Denoising Diffusion Probabilistic Models. CVPR, pages 11461–11471, 2022. https://arxiv.org/abs/2201.09865.

Vladimir Kulikov et al. FlowEdit: Inversion-Free Text-Based Editing Using Pre-Trained Flow Models. ICCV, pages 19721–19730, 2025. https://arxiv.org/abs/2412.08629.

Junliang Ye et al. NANO3D: A Training-Free Approach for Efficient 3D Editing Without Masks. arXiv:2510.15019, 2025. https://arxiv.org/abs/2510.15019.

Yankuan Chi, Xiang Li, Zixuan Huang, and James Matthew Rehg. Vinedresser3D: Towards Agentic Text-guided 3D Editing. CVPR, pages 12673–12683, 2026. https://arxiv.org/abs/2602.19542.

Teng-Fang Hsiao, Bo-Kai Ruan, Yu-Lun Liu, and Hong-Han Shuai. VecSet-Edit: Unleashing Pre-trained LRM for Mesh Editing from Single Image. SIGGRAPH, 2026. https://arxiv.org/abs/2602.04349.

Lin Li et al. VoxHammer: Training-Free Precise and Coherent 3D Editing in Native 3D Space. arXiv:2508.19247, 2025. https://arxiv.org/abs/2508.19247.

Tobias Preintner et al. 3DMorph: Single-Image-Guided Local 3D Shape Editing and Morphing. IJCNN, 2026. https://arxiv.org/abs/2606.07115.

Yangguang Li et al. TripoSG: High-Fidelity 3D Shape Synthesis using Large-Scale Rectified Flow Models. arXiv:2502.06608, 2025. https://arxiv.org/abs/2502.06608.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to Follow Image Editing Instructions. In CVPR, 2023. https://arxiv.org/abs/2211.09800.

Black Forest Labs et al. FLUX.1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space. arXiv:2506.15742, 2025. https://arxiv.org/abs/2506.15742.

Chenfei Wu et al. Qwen-Image Technical Report. arXiv:2508.02324, 2025. https://arxiv.org/abs/2508.02324.

OpenAI. Addendum to GPT-4o System Card: Native Image Generation. System card addendum, 2025. https://openai.com/index/gpt-4o-image-generation-system-card-addendum/.

Google. Gemini 2.5 Flash and Native Capabilities: Audio and Image. Model card, December 2025. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-2-5-Flash-Model-Card.pdf.

Jianfeng Xiang et al. Native and Compact Structured Latents for 3D Generation. Technical report, 2025. https://microsoft.github.io/TRELLIS.2/.

Chenlin Meng et al. SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations. ICLR, 2022. https://arxiv.org/abs/2108.01073.

Ron Mokady et al. Null-text Inversion for Editing Real Images using Guided Diffusion Models. CVPR, 2023. https://arxiv.org/abs/2211.09794.

Kai Zhang et al. MagicBrush: A Manually Annotated Dataset for Instruction-Guided Image Editing. NeurIPS, Datasets and Benchmarks Track, 2023. https://arxiv.org/abs/2306.10012.

Haozhe Zhao et al. UltraEdit: Instruction-based Fine-Grained Image Editing at Scale. NeurIPS, Datasets and Benchmarks Track, 2024. https://arxiv.org/abs/2407.05282.

Hao Liu et al. Velocity-Space 3D Asset Editing. 2026. https://arxiv.org/abs/2605.07385.

Rui Nie et al. EditFlow3D: Automated Local Editing of 3D Assets with Trajectory Preservation. 2026. https://arxiv.org/abs/2608.03179.

Maria Parelli, Michael Oechsle, Michael Niemeyer, Federico Tombari, and Andreas Geiger. 3D-LATTE: Latent Space 3D Editing from Textual Instructions. CVPR, pages 14377–14386, 2026. https://arxiv.org/abs/2509.00269.

Etai Sella et al. Vox-E: Text-Guided Voxel Editing of 3D Objects. ICCV, pages 430–440, 2023. https://arxiv.org/abs/2303.12048.

Ayaan Haque et al. Instruct-NeRF2NeRF: Editing 3D Scenes with Instructions. ICCV, pages 19740–19750, 2023. https://arxiv.org/abs/2303.12789.

Jiangshan Wang et al. Taming Rectified Flow for Inversion and Editing. ICML, 2025. https://arxiv.org/abs/2411.04746.

Mathilde Caron et al. Emerging Properties in Self-Supervised Vision Transformers. ICCV, 2021. https://arxiv.org/abs/2104.14294.

Alec Radford et al. Learning Transferable Visual Models From Natural Language Supervision. ICML, 2021. https://arxiv.org/abs/2103.00020.

A Localization and Sampling Details

A coverage parameter q converts scores into a coarse mask $m_i = \mathbb{1}[s_i \geq Q_{1-q}(s)]$, where the quantile is taken over $\mathcal{C}_s$. We initialize its value from image-space foreground coverage,

$$\rho = \frac{|E_2|}{|E_2| + |K_2|}, \qquad q = \mathrm{clip}[\rho + (\alpha - 1)\rho(1 - \rho), 0.03, 1]\,. \tag{12}$$

Here α controls coverage expansion. Empty edit or keep regions are handled separately by selecting none or all of the source, respectively. In the current representation, coarse 16^3 cells expand into corresponding 4^3 blocks of the 64^3 occupancy grid. With this expansion denoted by U, the source partition is

$$E_3 = O_s \cap U(m), \qquad K_3 = O_s \setminus E_3. \tag{13}$$

Image area is only a coverage heuristic: an occluded counterpart can increase the relevant 3D extent without increasing the visible mask. Threshold sensitivity must therefore be distinguished from the quality of the underlying attention ranking.

An optional one-cell coarse keep buffer can protect neighboring empty cells. Let E_c, K_c be the selected and retained occupied coarse cells and δ a 26-neighbor dilation. We define $E'_c = \delta(E_c) \setminus K_c$ and $K'_c = \delta(K_c) \setminus E'_c$, and use $F = \Omega \setminus U(K'_c)$ when buffering is enabled. This spatial margin changes the protected domain, not the attention scores.

A.1 Sampling Configuration

The reference configuration aggregates Stage 1 attention over 25 sampling steps and 24 layers, averages attention heads, and excludes five non-spatial image tokens. The coverage rule uses $\alpha = 1.1$ as its default; coverage expansion controls the selected spatial extent. Local structural sampling reuses cross-attention for steps 0–14 and applies structure-reference replacement from step 15 onward. The protected occupancy is encoded with the posterior mean. Stage 2 uses fixed coordinate-matched noise for source replacement and releases its final two steps. These schedules specify the reference procedure rather than an optimality claim. A full edit follows route selection, image-space change estimation, optional attention localization, structure editing or reuse, feature editing, and decoding.

B Protocol and Reproducibility Notes

For local preservation diagnostics, all methods are evaluated on a shared keep region predicted by our method. This is not independent 3D ground truth. Geometry is evaluated by bidirectional point-to-triangle distances, with 30,000 area-sampled points per direction, without ICP or output renormalization. We report mean symmetric distance (CD) and the 95th-percentile Hausdorff-style distance (HD95), normalized by the source bounding-box diagonal D. Appearance uses twelve matched views and source-visible keep masks, reporting PSNR, SSIM, and spatial LPIPS. Appearance errors can include geometric displacement and occlusion, and are not pure material errors. Each case contributes equally after view aggregation.

The main editing comparison contains sixteen curated requests. The preservation study uses the eight local requests. Geometry preservation uses symmetric, unsquared point-to-triangle distances; HD95 is a robust percentile distance, not the maximum Hausdorff distance. Appearance renders use fixed source coordinates and cameras. SSIM masks are eroded by three pixels and LPIPS masks by eight pixels to reduce boundary leakage; spatial LPIPS remains an approximate regional measure.

For Figure 6, Ours, 3DMorph, and Nano3D use source-occupancy views; VecSet-Edit uses full-source decoder associations on the surface. Ours and VecSet-Edit share image regions, 3DMorph receives derived boxes, and Nano3D uses occupancy differences. These interfaces are not identical segmentation protocols.

C Whole-Asset Multi-View Evaluation

All sixteen cases contribute equally; categories are not separately weighted. We render twelve orthographic views (six evenly spaced azimuths at elevations 0.30 and -0.20 radians) for each source and

edited asset. Camera parameters are shared across methods within a case. Source bounds determine framing, except that the rusty-can case uses shared union bounds to avoid clipping larger outputs. Raw model sizes and orientations are retained. Renders use 256-pixel base color on white, removing lighting variation but not evaluating full material reflectance. No view is selected by metric scores.

We use official CLIP ViT-B/32 (Radford et al., 2021) and DINO ViT-S/16 final CLS features (Caron et al., 2021), normalized to unit length. Whole square renders are resized to the encoders' 224-pixel inputs; no per-result foreground fitting is applied. Reference-image backgrounds are replaced with white using border-connected color thresholding and square padding. Evaluator-authored source/target descriptions are fixed across methods before scoring and retained with the results; these are not claimed to be original generation prompts.

Let e_v^s, e_v^o be CLIP features of source/output views, t_e the target-text feature, and e_I the target-image feature. We report

$$\text{CLIP-T} = \frac{1}{V}\sum_v \langle e_v^o, t_e\rangle, \qquad \text{CLIP-I} = \frac{1}{V}\sum_v \langle e_v^o, e_I\rangle, \tag{14}$$

$$\text{Dir-Con} = \frac{2}{V(V-1)}\sum_{v<w} \cos(e_v^o - e_v^s, e_w^o - e_w^s). \tag{15}$$

DINO-I uses the CLIP-I formula with DINO features. Zero edit vectors contribute zero direction scores. Dir-Con measures semantic directional agreement, not geometric correspondence. Our explicit protocol follows common metric families rather than claiming an exact reproduction of each reference paper's evaluator.

We retain per-view embeddings and per-case scores. Paired case bootstrap intervals use 10,000 resamples with seed 20260925 and describe variability within this curated set, not independent-test generalization. Single target images do not provide multi-view geometry ground truth; CLIP can miss localized, counting, and spatial errors. Whole-source CD/IoU would penalize intended global geometry changes, so we do not treat them as edit correctness. FID is omitted because sixteen heterogeneous assets do not provide a reliable reference distribution. The existing source/output scale conventions and known VecSet heading differences remain potential confounds.

Table 4: Case-bootstrap 95% intervals for Table 2; these intervals do not imply an independent test set.

Method	CLIP-T	CLIP-I	DINO-I	Dir-Con
Ours	[0.347, 0.368]	[0.859, 0.909]	[0.682, 0.769]	[0.493, 0.661]
Nano3D	[0.304, 0.352]	[0.784, 0.857]	[0.517, 0.669]	[0.450, 0.640]
VoxHammer	[0.336, 0.373]	[0.839, 0.894]	[0.631, 0.732]	[0.513, 0.682]
VecSet-Edit	[0.346, 0.365]	[0.854, 0.893]	[0.696, 0.753]	[0.456, 0.616]

Vertex colors exported from decoded Gaussian RGB are converted from sRGB to linear RGB before being stored as glTF color attributes; texture images retain their standard sRGB interpretation. This ensures consistent color handling across the compared representations.

For Sitting Dog, all source and edited meshes are rotated by 180° about the source-centered vertical axis before rendering.

AI USE STATEMENT

We used AI assistance to aid or polish writing, for retrieval and discovery (e.g., finding related work), and for research ideation or execution. The authors are responsible for verifying the final manuscript, sources, and results.