Training-Free 3D Editing via Feature-Divergence Localization and Trajectory Correction

Tri Ton **Siwoo Lim** **Joshua Tian Jin Tee** **Chang D. Yoo***
Korea Advanced Institute of Science and Technology (KAIST)
{tritth, cd_yoo}@kaist.ac.kr

Abstract

Modifying 3D assets requires enacting targeted structural transformations while strictly preserving the geometric identity of unedited regions. While recent probability flow models enable training-free editing, bounding these modifications within a high-dimensional latent space presents severe mathematical challenges. Current inversion-free trajectories rely on global velocity updates, which often cause spatial bleed into preserved structures and induce trajectory under-commitment, causing edits to stall mid-path. To resolve these dual failures, we introduce FocusFlow, a segmentation-free framework that structures the integration trajectory into two disjoint phases. During early coarse structure formation, FocusFlow applies Feature-Divergence Localization (FDL) to spatially gate the velocity field using intrinsic cross-attention. Once coarse structure has formed, FocusFlow applies Reference Generation Guidance (RGG), a trajectory-correction term that drives convergence to the target shape by pulling the trajectory toward an explicit clean-latent target. Evaluated on the joint Google Scanned Objects and PartObjaverse-Tiny benchmarks, FocusFlow obtains the best preservation scores and the largest edit magnitude among compared methods, suggesting that it can reduce the traditional preservation-modification trade-off. By providing stable, localized geometric control without manual spatial annotations, this phase-scheduled approach lowers technical barriers to the modification of synthetic 3D media.

1 Introduction

Modifying existing 3D assets [1, 49, 52] requires enacting targeted structural transformations while strictly preserving the geometric identity of unedited regions. Recent 3D foundation models [15, 46] parameterized as rectified flow transformers provide strong geometric priors for shape generation. However, extending these priors to training-free editing via inversion-free [21, 52, 18, 49] trajectories introduces severe failure modes. Specifically, current unconstrained editing trajectories either stall before realizing the target geometry or incorrectly propagate the transformation into regions that must remain identical to the source shape.

These failures originate from the underlying dynamics of probability flow ordinary differential equations (ODEs). Rectified flows establish deterministic transport maps designed to follow straight paths between empirical distributions. Because global editing updates, such as AnchorFlow [52], apply a uniform semantic velocity across the entire latent representation, structures that should remain unchanged absorb part of the spatial transformation. As shown in Fig. 1 (top), this results in severe spatial bleed where the plane's structure is distorted even though the edit targets only the passenger. Furthermore, flow trajectories driven strictly by velocity differences encode a semantic direction of change but lack an explicit terminal coordinate. Consequently, edits of medium magnitude suffer from trajectory under-commitment in Fig. 1 (bottom), stalling mid-path and failing to physically

*Corresponding Author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

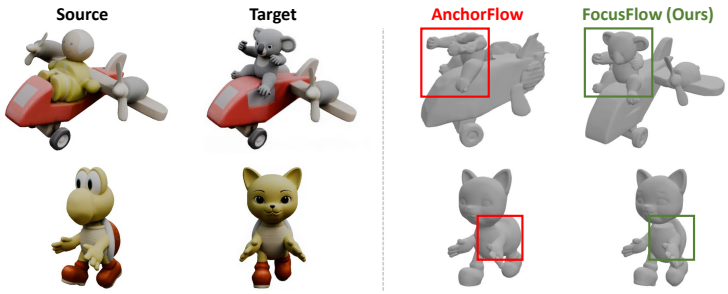

Figure 1: **FocusFlow addresses global editing failures.** The baseline [52] exhibits spatial bleed and trajectory under-commitment (red boxes). FocusFlow confines the velocity field during localization and pulls toward a clean-latent target during guidance, realizing the target with less background distortion and no segmentation masks (green boxes).

realize the target geometry. Naive attempts to resolve this stalling via instantaneous target activation abruptly shift the velocity field, destabilizing the integration path and inducing topological fracturing (the mesh tearing into disconnected fragments) in the decoded mesh.

To resolve these dual failures, we introduce FocusFlow (Fig. 1, right), a phase-scheduled editing framework that strictly decouples spatial gating from terminal convergence. Prior studies in 2D image manipulation demonstrate that edits can be localized without explicit manual annotations, using cross-attention maps [10] or contrasts between noise predictions [3]. Building on this observation, FocusFlow integrates Feature-Divergence Localization (FDL) to enforce intrinsic spatial localization and prevent spatial bleed, alongside Reference Generation Guidance (RGG), a trajectory-correction term that drives explicit endpoint commitment. By scheduling these mechanisms into sequential, disjoint phases of the integration trajectory, FocusFlow avoids the weaker edits we observe when both act at once. Our core technical contributions are threefold. First, we propose a phase-scheduled integration that assigns spatial gating and terminal convergence to disjoint phases of the trajectory, avoiding the weaker edits we observe when both act at once. Second, for intrinsic spatial localization, we propose Feature-Divergence Localization (FDL), a segmentation-free module deriving a soft gate from patch divergence and cross-attention scores to suppress spatial bleed during early integration. Third, to encourage explicit endpoint commitment, we introduce Reference Generation Guidance (RGG), which smoothly pulls the trajectory toward a clean-latent reference target via time-dependent weighting, reducing the risk of topological fracturing. On the Google Scanned Objects (GSO) [4] and PartObjaverse-Tiny [48] benchmarks, FocusFlow obtains the best preservation scores and the largest edit magnitude among compared methods.

2 Related Work

3D generation models. Recent 3D generative models improve the quality and accessibility of digital asset creation. Early optimization-based approaches demonstrated that pretrained 2D diffusion priors [41] can be distilled into 3D representations through rendered-view supervision [39, 24]. However, these methods require costly per-instance optimization and are designed for generating new assets rather than modifying existing ones. More recent native 3D generative models [15, 46, 50, 17, 45, 28, 34] learn generative priors directly over large-scale 3D data. Specific architectures predict 3D representations from a single image [12], introduce structured 3D latents decodable into multiple formats [46], or provide high-fidelity image-to-3D pipelines based on learned shape latents [15]. While these models provide strong geometric priors, they function as generators rather than editors. They synthesize new assets but lack an explicit mechanism to preserve a source asset while modifying only a localized region.

Training-free 3D editing. Existing 3D editing methods balance localized modifications with source geometry preservation. Classical approaches deform or reconstruct meshes under semantic guidance via text-driven differentiable rendering [8], conditional reconstruction [7], region-level fusion [13], or part-aware pipelines [26, 47], but rely on predefined editable regions, segmentation cues, or auxiliary supervision. Recent work instead edits assets in native latent spaces to leverage geometric priors and avoid multi-view inconsistencies [22, 49, 36], with flow-based editors using velocity

differences [21] or latent anchors [52], and feed-forward variants enabling fast edits [14]. Still, these face a strict preservation-modification trade-off, and training-free variants either entangle geometry with texture [22, 49] or require 3D masks [22], drawn by hand or predicted by 3D segmentation models [48, 43]. In 2D, Prompt-to-Prompt [10] shows that cross-attention ties prompt words to image regions, and DiffEdit [3] derives an edit mask by contrasting noise predictions under the two prompts. The 3D latent is an unordered token set with no spatial grid, so FDL instead maps DINO patch differences to latent tokens through the generator's cross-attention. FocusFlow operates strictly on the geometric latent and separates spatial localization from target commitment, enabling precise mask-free editing.

3 Preliminaries

3.1 Inversion-Free 3D Editing

We build on a rectified flow transformer [15, 25, 27, 5] that maps a Gaussian prior to a 3D shape latent distribution via a learned velocity field. Let $z_t \in \mathbb{R}^{N\times d}$ denote the latent state at flow time $t \in [0, 1]$, where N represents the token count and d represents the channel dimension. The generator parameterizes a conditional velocity field $v_\theta(z_t, t, c)$, driven by an image condition c. The probability flow ordinary differential equation (ODE) trajectory is defined as $\mathrm{d}z_t \;=\; v_\theta(z_t, t, c)\mathrm{d}t$. This equation governs the deterministic transport of an initial Gaussian sample $z_0 \sim \mathcal{N}(0, I)$ to the clean 3D shape latent z_1. Integration of this vector field across the unit interval yields a sample from the learned shape distribution.

To edit a 3D shape, the system receives a source condition c^{src} and a target condition c^{tar}. The objective is to compute an edited trajectory $\{z_t^{\mathrm{edit}}\}_{t\in[0,1]}$ that originates at the source clean latent z_1^{src} and evolves into an edited latent z_1^{tar}. We adopt the inversion-free formulation introduced by FlowEdit [21], which defines the edited trajectory dynamically as

$$z_t^{\mathrm{edit}} \;=\; z_1^{\mathrm{src}} \;+\; z_t^{\mathrm{tar}} \;-\; z_t^{\mathrm{src}}. \tag{1}$$

Here $z_t^{\mathrm{src}} = t\, z_1^{\mathrm{src}} + (1-t)\,\epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$ is the source latent noised to level t, and $z_t^{\mathrm{tar}} = z_t^{\mathrm{edit}} + z_t^{\mathrm{src}} - z_1^{\mathrm{src}}$ follows from Equation 1. This formulation constructs a path that begins at the source shape and ends at the edited shape, avoiding the destructive process of transporting the latent back through the Gaussian prior space. By defining $v_t^{\mathrm{src}} = v_\theta(z_t^{\mathrm{src}}, t, c^{\mathrm{src}})$ and $v_t^{\mathrm{tar}} = v_\theta(z_t^{\mathrm{tar}}, t, c^{\mathrm{tar}})$, we obtain the inversion-free edit velocity by differentiating Equation 1 with respect to t: $v_t^{\mathrm{edit}} \;=\; v_t^{\mathrm{tar}} \;-\; v_t^{\mathrm{src}}$.

3.2 Latent Anchor-Aligned Flows

While inversion-free editing prevents Gaussian prior destruction, unconstrained velocity differences frequently push the latent representation off the target data manifold. AnchorFlow [52] constrains this geometric drift by introducing a latent-anchor alignment term. The latent anchor f_t is defined as $f_t \;=\; z_t \;+\; (1-t)v_t$. The anchor f_t computes the linear forecast of the clean latent by extending the current velocity tensor directly to the endpoint of the trajectory. To enforce proximity between the source and target trajectories in this forecasted clean-latent space, AnchorFlow derives an alignment loss $\mathcal{L}_{\mathrm{align}}$. The gradient descent step on this alignment loss evaluates to

$$\nabla\mathcal{L}_{\mathrm{align}} \;=\; (2-t)\big(f_t^{\mathrm{tar}} \;-\; f_t^{\mathrm{src}}\big). \tag{2}$$

This gradient step actively penalizes geometric divergence between the two forecasts, pulling the edit trajectory back toward the structurally stable source manifold. The complete, unconstrained AnchorFlow update is therefore applied at each integration step as

$$\mathrm{d}z_t^{\mathrm{edit}} \;=\; \big(v_t^{\mathrm{edit}} \;-\; \nabla\mathcal{L}_{\mathrm{align}}\big)\mathrm{d}t. \tag{3}$$

While Equation 3 preserves manifold adherence, applying this update globally across all N tokens causes severe *spatial bleed*, altering background structures that must remain preserved. Furthermore, the velocity vector v_t^{edit} directs semantic change but lacks an explicit terminal coordinate. This absence causes *trajectory under-commitment*, where medium-magnitude edits stall mid-path and fail to realize the target shape.

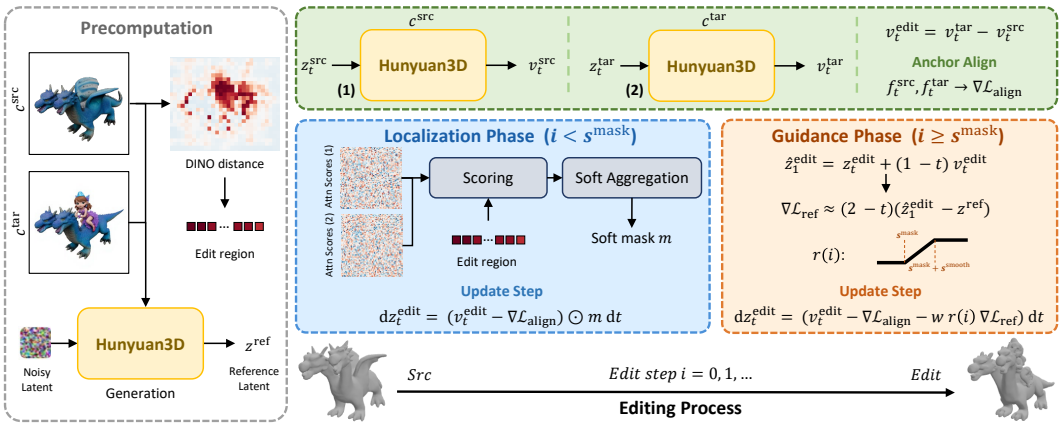

Figure 2: **FocusFlow framework.** *Left:* Precomputation of the DINO distance, the edit region and the reference latent z^{ref}. *Top:* Per-step velocities and the alignment gradient. *Center (localization phase):* FDL gates the velocity field via soft mask m to confine the edit. *Right (guidance phase):* RGG applies a ramped pull toward z^{ref}, driving structural commitment.

4 Method

4.1 Phase-Scheduled Framework

We distinguish continuous flow time $t \in [0, 1]$ from the edit-step index $i = 0, 1, \ldots$, which counts solver steps from the start of the edit. While unconstrained velocity integration theoretically maps the source distribution to the target distribution [21], in practice it often stalls mid-path [52]: the velocity difference encodes a direction of change but no explicit terminal coordinate. Furthermore, because flow models form global layouts under high noise and refine details under low noise, spatial gating should act during early structural formation, whereas pulling toward a terminal target only helps once coarse geometry exists. To resolve spatial bleed and trajectory under-commitment, FocusFlow structures the integration trajectory into two disjoint phases separated by a discrete boundary s^{mask} (5 steps in our experiments), decoupling early spatial localization from subsequent terminal convergence. Figure 2 illustrates the complete phase-scheduled framework:

- **Localization Phase ($i < s^{\text{mask}}$):** During the early integration steps, Feature-Divergence Localization (FDL) constructs a dynamic per-token soft mask m to spatially gate the velocity field, suppressing spatial bleed into preserved regions.
- **Guidance Phase ($i \geq s^{\text{mask}}$):** Once coarse geometry forms, the spatial gate retires. Reference Generation Guidance (RGG) activates via a weight ramped over the step index, pulling the trajectory toward an explicit clean-latent target to drive convergence to the target shape.

We detail the spatial gating and target convergence mechanisms below.

4.2 Feature-Divergence Localization

To prevent spatial bleed during coarse-structure formation, Feature-Divergence Localization (FDL) generates a continuous soft mask m that spatially gates the velocity field. To maintain structural stability, we derive this mask from the divergence of semantic image features rather than the divergence of the high-dimensional velocity field. We construct this intrinsic spatial localization mechanism through three sequential operations: semantic patch selection, multi-block cross-attention scoring, and soft aggregation.

Semantic Patch Selection. Let c^{src} and c^{tar} denote the source and target image conditions. Let $\mathcal{E}_{\text{DINO}}$ denote a frozen DINO encoder [35]. We pass each image condition through $\mathcal{E}_{\text{DINO}}$ to extract per-patch features $\phi^{\text{src}} = \mathcal{E}_{\text{DINO}}(c^{\text{src}})$ and $\phi^{\text{tar}} = \mathcal{E}_{\text{DINO}}(c^{\text{tar}})$, both in $\mathbb{R}^{P \times d_\phi}$, where P is the total number of patches and d_ϕ is the feature dimension. We compute the patch-wise cosine distance to

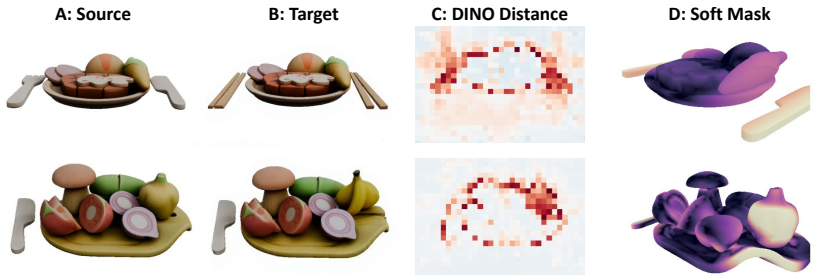

Figure 3: **FDL soft mask construction.** *A:* Source condition. *B:* Target condition. *C:* DINO distance D_p highlighting the semantic modification. *D:* Final soft mask m_n.

measure the semantic deviation:

$$D_p \;=\; 1 \;-\; \frac{\phi_p^{\mathrm{src}} \cdot \phi_p^{\mathrm{tar}}}{\|\phi_p^{\mathrm{src}}\|\,\|\phi_p^{\mathrm{tar}}\|}, \qquad p \in \{1, \ldots, P\}. \tag{4}$$

Because DINO features encode semantic correspondences rather than low-level pixel intensities, patches exhibiting high cosine distance likely reflect semantic changes. Instead of selecting a fixed number of patches, we dynamically define the edit region $K_{\mathrm{changed}} \subseteq \{1, \ldots, P\}$ using an adaptive statistical threshold. We retain all patches where $D_p > \mu(D) + \sigma(D)$, where $\mu(D)$ and $\sigma(D)$ are the mean and standard deviation of D over the P patches. This yields a sparse edit region whose size adapts to the edit.

Multi-Block Cross-Attention Scoring. To translate this image-space divergence into a latent-space relevance score, we intercept the cross-attention queries $q_b \in \mathbb{R}^{N \times d_k}$ from latent tokens and keys $k_b \in \mathbb{R}^{P \times d_k}$ from image patches. We perform this interception across a selected subset of transformer blocks $\mathcal{B}$. For each block $b \in \mathcal{B}$ and latent token n, we compute the unnormalized relevance score by averaging the dot product strictly over the changed patches:

$$a_{b,n} \;=\; \frac{1}{|K_{\mathrm{changed}}|} \sum_{p \in K_{\mathrm{changed}}} q_{b,n} \cdot k_{b,p}. \tag{5}$$

We compute these scores independently for the source and target forward passes and min-max normalize each over the N latent tokens, so that neither pass dominates the fusion through its raw scale. This yields $\bar{a}_{b,n}^{\mathrm{src}}, \bar{a}_{b,n}^{\mathrm{tar}} \in [0, 1]$. To ensure that tokens relevant to either the original structure or the target geometry remain active, we fuse the two streams using an element-wise maximum: $\mathrm{score}_{b,n} \;=\; \max\!\big(\bar{a}_{b,n}^{\mathrm{src}},\, \bar{a}_{b,n}^{\mathrm{tar}}\big)$.

Soft Aggregation. The fused scores define a soft mask for each block b:

$$m_{b,n} \;=\; \alpha \;+\; (1 - \alpha)\,\big(\mathrm{score}_{b,n}\big)^{\gamma}, \tag{6}$$

where the floor $\alpha \in [0, 1]$ is the minimum fraction of the update that every token receives, while the sharpness exponent $\gamma > 0$ controls the response boundary. We aggregate the block-wise masks via a mean operation to yield the final token mask $m_n = \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} m_{b,n}$, visualized in Fig. 3. We apply this intrinsic spatial localization during the localization phase ($i < s^{\mathrm{mask}}$):

$$\mathrm{d}z_t^{\mathrm{edit}} \;=\; \big(v_t^{\mathrm{edit}} \;-\; \nabla \mathcal{L}_{\mathrm{align}}\big) \odot m \,\mathrm{d}t. \tag{7}$$

The operator $\odot$ denotes element-wise multiplication broadcast across the channels. We restrict gating to $i < s^{\mathrm{mask}}$ because flow models establish coarse geometry early and refine fine detail late, so the gate is needed only while coarse geometry forms. Moving the boundary later degrades both preservation and edit strength, whereas retiring the gate at s^{mask} lets the late steps update all tokens.

4.3 Reference Generation Guidance

While intrinsic spatial localization restricts the velocity domain, the differential signal v_t^{edit} strictly encodes a direction of semantic change, causing trajectories to stall before physically realizing the

target shape. Reference Generation Guidance (RGG) resolves this trajectory under-commitment by establishing an explicit terminal coordinate in the continuous latent space. By precomputing a reference latent z^{ref} via the base flow generator [15] on the target condition, we obtain a sample from the generator's prior that drives convergence to the target shape while bypassing the computational overhead and perspective biases of pixel-space differentiable rendering.

Clean Latent Residual. Since z^{ref} is a clean latent, we compare it with a forecast of the clean edited latent rather than with the current state z_t^{edit}. This state is not a noisy sample at level t, so evaluating v_θ on it would query the network off its input distribution. We therefore form the forecast with the edit velocity v_t^{edit}, which the network computes only at the noisy states z_t^{src} and z_t^{tar}:

$$\hat{z}_1^{\mathrm{edit}} \;=\; z_t^{\mathrm{edit}} \;+\; (1-t)v_t^{\mathrm{edit}}. \tag{8}$$

We enforce convergence by defining the clean latent reference loss as the squared Euclidean distance between this forecast and the reference latent:

$$\mathcal{L}_{\mathrm{ref}} \;=\; \frac{1}{2}\big\|\hat{z}_1^{\mathrm{edit}} \;-\; z^{\mathrm{ref}}\big\|^2. \tag{9}$$

Differentiating $\mathcal{L}_{\mathrm{ref}}$ with respect to the current state z_t^{edit} requires applying the chain rule to Equation 8, yielding an expression dependent on the target velocity Jacobian J_θ:

$$\nabla_{z_t^{\mathrm{edit}}}\mathcal{L}_{\mathrm{ref}} \;=\; \big(I \;+\; (1-t)J_\theta^\top\big)\big(\hat{z}_1^{\mathrm{edit}} \;-\; z^{\mathrm{ref}}\big), \tag{10}$$

where $J_\theta = \partial v_\theta(z_t^{\mathrm{tar}}, t, c^{\mathrm{tar}})/\partial z_t^{\mathrm{edit}}$. The exact gradient needs a vector-Jacobian product at every guidance step. We substitute the exact matrix with the scalar identity $(2-t)I$, a standard simplification in the score-distillation literature [39]. This substitution provides a directional pull toward the reference latent and yields a tractable, closed-form gradient:

$$\nabla\mathcal{L}_{\mathrm{ref}} \;\approx\; (2-t)\big(\hat{z}_1^{\mathrm{edit}} \;-\; z^{\mathrm{ref}}\big). \tag{11}$$

To prevent this residual from overriding the differential edit velocity and collapsing the geometry into a direct copy of z^{ref}, we scale the gradient by a fixed balancing coefficient w. The applied guidance term becomes $w(2-t)(\hat{z}_1^{\mathrm{edit}} - z^{\mathrm{ref}})$.

Time-Dependent Weighting. Activating this reference guidance instantaneously at the phase boundary s^{mask} abruptly shifts the velocity field. This sudden transition destabilizes the integration, inducing topological fracturing in the decoded mesh. To ensure trajectory smoothing, we apply a continuous time-dependent weighting schedule $r(i)$ tied to the discrete step index, which scales linearly over a defined duration s^{smooth} (15 steps in our experiments): $r(i) \;=\; \min\Big(1,\; \max\Big(0,\; \frac{i - s^{\mathrm{mask}}}{s^{\mathrm{smooth}}}\Big)\Big)$.

Once the localization phase concludes and the spatial gate retires, this smoothed pull is introduced alongside the manifold alignment constraint. To minimize the distance loss and drive convergence toward the reference latent, we subtract the reference gradient:

$$\mathrm{d}z_t^{\mathrm{edit}} \;=\; \Big(v_t^{\mathrm{edit}} \;-\; \nabla\mathcal{L}_{\mathrm{align}} \;-\; w\, r(i)\nabla\mathcal{L}_{\mathrm{ref}}\Big)\mathrm{d}t. \tag{12}$$

Applying this pull concurrently with the spatial gate during the early integration phase works against the edit: in our *Concurrent* ablation (Table 2), the background stays close to the source, but the edit is the weakest of all variants. Structuring the update into sequential, disjoint phases encourages stronger terminal convergence without violating the localized geometric structure. The complete step-by-step pseudo-code for the phase-scheduled integration is provided in Appendix A.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate on Edit3D-Bench [22], which contains 100 3D models curated from Google Scanned Objects (GSO) [4] and PartObjaverse-Tiny [48], spanning single-part geometries and multi-part assemblies. Each model has 3 editing prompts, giving 300 editing cases. Each prompt provides 2D renderings of the source, a 2D mask of the edit region, an edited image that serves as the target condition c^{tar}, and a human-annotated 3D mask of the edit region. We use this set because the local metrics need explicit 3D region annotations: the 3D mask only partitions sampled surface points into edited and preserved sets for evaluation.

Table 1: Quantitative comparison on GSO and PartObjaverse-Tiny. We mark the **best** and second-best results. *Note:* All reported numbers are scaled by a factor of 10^2.

Method	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$	$\mathrm{CD}_{\mathrm{edit}}\uparrow$	$\mathrm{NC}_{\mathrm{preserve}}\uparrow$	F-score$\uparrow$	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$
Direct Editing [15]	4.66	1.49	54.48	2.98	63.30	23.64
Inversion [16]	3.94	1.38	55.51	2.88	**64.35**	**24.14**
Inversion-free [21]	3.15	1.50	57.96	2.33	61.78	21.72
AnchorFlow [52]	3.81	1.58	56.20	2.60	61.42	22.51
Nano3D [49]	3.22	1.50	57.46	2.50	62.06	22.08
FocusFlow (Ours)	**3.11**	**1.61**	**58.20**	**3.05**	63.85	23.91

Implementation Details. We use an Euler ODE solver configured for $T = 50$ time steps. We initiate the edit trajectory at step $n_{\max} = 41$ (skipping the first 9 integration steps) and stop at step $n_{\min} = 1$, following AnchorFlow [52]. Intrinsic spatial localization extracts cross-attention from transformer blocks $\mathcal{B} = \{4, 5\}$, with mask parameters $\alpha = 0.2$ and $\gamma = 1.0$. The guidance phase activates at the discrete step boundary $s^{\mathrm{mask}} = 5$ and ramps over $s^{\mathrm{smooth}} = 15$ steps with a reference weight $w = 0.3$. The reference latent z^{ref} is generated with 30 denoising steps. The target and source guidance scales for the edit trajectory are 7.5 and 3.5, respectively. Sweeps over α, γ, w, s^{mask}, and the number of reference steps are reported in Appendix B.

Baselines. We compare FocusFlow against five representative 3D editing frameworks. *Direct Editing* [15] generates a new mesh from scratch, conditioned on the target image. *Inversion* [16] performs deterministic latent inversion prior to executing forward generation. For native flow-based editing, we evaluate *Inversion-free* [21], which integrates unconstrained velocity differences, and *AnchorFlow* [52], which stabilizes the trajectory via latent-anchor alignment. We also include *Nano3D* [49] as a contemporary training-free baseline. Notably, FocusFlow needs no manual spatial annotations or segmentation masks during editing.

Metrics. Four geometric metrics are computed on output meshes [6]. No metric uses a ground-truth edited mesh. To quantify structural integrity on the preserved points, we measure the Chamfer Distance ($\mathrm{CD}_{\mathrm{preserve}}\downarrow$) and Normal Consistency ($\mathrm{NC}_{\mathrm{preserve}}\uparrow$) [9]. To measure the size of the change, we compute the same distance on the edited points ($\mathrm{CD}_{\mathrm{edit}}\uparrow$); a higher value indicates a larger change. On the preserved points, we also compute the F-score$\uparrow$ [19] of the edited mesh against the source mesh: the harmonic mean of precision and recall at a distance threshold of $\tau = 0.01$. We measure semantic alignment using the cosine similarity between target embeddings and six uniformly sampled gray renders of the edited mesh ($\mathrm{CLIP}_{\mathrm{img}}\uparrow$ and $\mathrm{CLIP}_{\mathrm{txt}}\uparrow$) [40].

5.2 Research Questions

We organize our empirical study around four research questions:

- **RQ1 (State-of-the-Art Comparison):** Does the phase-scheduled integration of Feature-Divergence Localization (FDL) and Reference Generation Guidance (RGG) yield stronger semantic modification and better source-identity preservation than global baselines?
- **RQ2 (Phase Scheduling Necessity):** Does the discrete temporal boundary s^{mask} between the two phases affect edit strength and semantic alignment? The "Concurrent" ablation removes this boundary to test this.
- **RQ3 (Transition Dynamics):** How do mask softness and temporal weighting shape the resulting geometry? The "Binary Mask" and "Instantaneous Activation" ablations isolate each of these two factors.
- **RQ4 (Component Independence):** Are FDL and RGG jointly necessary to balance semantic editing against geometric preservation, or does either suffice on its own? The "No Mask" and "No Latent Reference" ablations test each component in turn.

5.3 Main Results

To address **RQ1**, we evaluate FocusFlow against external baselines on the GSO and PartObjaverse-Tiny sets in Table 1. FocusFlow has the best value on all four geometric metrics: $\mathrm{CD}_{\mathrm{preserve}}$ of 3.11,

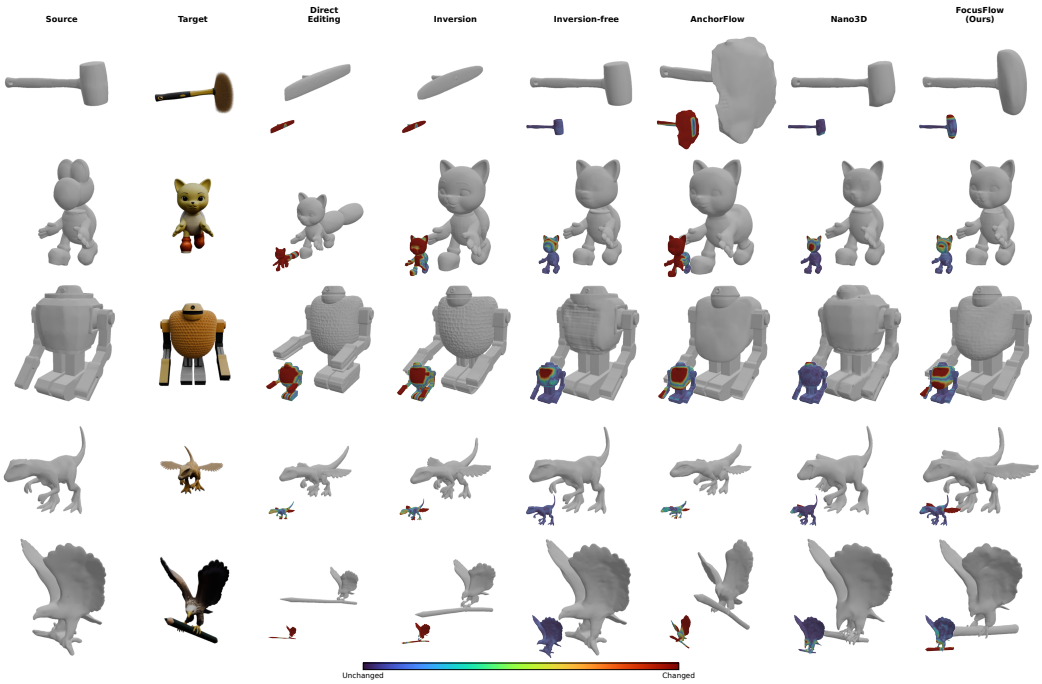

Figure 4: **Qualitative comparison of 3D editing baselines.** The inset under each result shows the per-point deviation of the output from the source mesh (blue: unchanged, red: changed), so a well-localized edit appears blue outside the edit region. While prior methods either alter the source geometry or fail to fully realize the target modification, FocusFlow realizes the edit while leaving unedited regions largely intact.

$\mathrm{CD}_{\mathrm{edit}}$ of 1.61, $\mathrm{NC}_{\mathrm{preserve}}$ of 58.20, and F-score of 3.05. It ranks second to Inversion on both CLIP metrics. These suggest that FocusFlow can reduce the trade-off between structural preservation and semantic modification.

Table 1 also reflects different failure modes among the baselines. Methods that regenerate the latent embeddings from the target condition [15, 16] reach high semantic alignment but distort the unedited regions, yielding the worst $\mathrm{CD}_{\mathrm{preserve}}$ scores (4.66 and 3.94). Conversely, the unconstrained flow method [21] stays close to the source, but has the lowest $\mathrm{CLIP}_{\mathrm{txt}}$ (21.72), consistent with trajectory under-commitment. Nano3D [49] ranks third on $\mathrm{CD}_{\mathrm{preserve}}$ and $\mathrm{NC}_{\mathrm{preserve}}$, with lower semantic alignment ($\mathrm{CLIP}_{\mathrm{txt}}$ of 22.08). By combining intrinsic spatial localization and reference generation guidance on geometry, FocusFlow reaches semantic alignment between the two regeneration methods with less spatial bleed, so Table 1 supports **RQ1** under this protocol.

Table 2: Component ablations on the combined GSO and PartObjaverse-Tiny set.

Variant	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$	$\mathrm{CD}_{\mathrm{edit}}\uparrow$	$\mathrm{NC}_{\mathrm{preserve}}\uparrow$	F-score$\uparrow$	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$
No Mask	3.23	1.55	56.87	2.93	63.40	23.60
Binary Mask	3.38	1.57	57.18	2.86	63.35	23.54
Concurrent	3.13	1.51	57.32	2.72	62.79	23.11
No Latent Ref	3.32	1.60	57.19	2.69	63.26	23.49
Instant. Act.	3.24	**1.77**	57.02	2.79	63.56	23.54
FocusFlow (Ours)	**3.11**	1.61	**58.20**	**3.05**	**63.85**	**23.91**

5.4 Qualitative Evaluation

Figure 4 illustrates, on part addition and replacement, the behaviors quantified in Section 5.3. The baseline methods manifest the failure modes described in Section 3. Regenerative frameworks lack intrinsic spatial localization, distorting unedited regions and degrading source identity (rows 1 and 5, cols 3–4). Conversely, unconstrained flow methods often suffer from trajectory under-commitment, stalling before realizing the semantic target (rows 1, 3, 4, and 5, col 5). Even when stabilized by

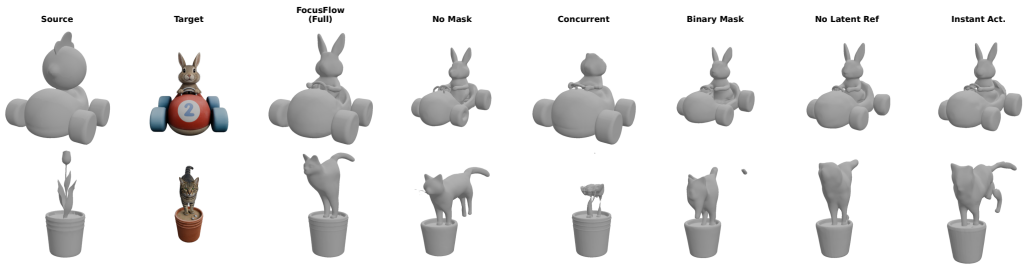

Figure 5: **Qualitative ablation results.** Visual comparison of ablated variants against the full framework on two part-replacement cases.

latent anchoring, global velocity updates often cause spatial bleed, modifying unedited background geometries (rows 1, 2, 4, and 5, col 6). Furthermore, methods operating on coupled representations can under-edit (row 1, col 7) or introduce structural artifacts (row 3, col 7).

FocusFlow reduces these failures. By enforcing intrinsic spatial localization during coarse structure formation, it gates the velocity field to preserve the source geometry. Subsequently applying time-dependent weighting toward the reference latent mitigates the abrupt change in the velocity field, driving convergence to the target shape. In Figure 4, FocusFlow confines most of the change to the edit: in the deviation insets, the handle stays blue while the head changes (row 1, col 8), the body stays blue while wings are added (row 4, col 8), and the eagle stays mostly blue while the branch becomes a pencil (row 5, col 8).

5.5 Ablation Studies

To isolate the marginal contributions of the proposed mechanisms and address **RQ2**, **RQ3**, and **RQ4**, we evaluate five ablated variants on the joint benchmark (Table 2); Fig. 5 shows two example cases.

Phase Scheduling Necessity (RQ2). We evaluate the requirement for the temporal boundary s^{mask} by activating intrinsic spatial localization and reference generation guidance simultaneously across all integration steps (*Concurrent*). This simultaneous activation keeps the background close to the source but suppresses the intended edit: the chick head stays in row 1, and no cat forms in row 2. It yields the weakest edit magnitude and the poorest semantic alignment in Table 2 (CLIP_{txt} of 23.11), which supports separating the mechanisms into disjoint phases. The late pull also shows no clear harm to preservation: the full model has the lowest $\text{CD}_{\text{preserve}}$ in Table 2 (3.11, against 3.32 for *No Latent Ref*, which applies no pull).

Transition Dynamics (RQ3). We assess the impact of transition functions via the *Binary Mask* and *Instant. Act.* ablations. Replacing the soft mask with a strict binary threshold (*Binary Mask*) gives the highest $\text{CD}_{\text{preserve}}$ in Table 2 and, in Fig. 5, leaves a floating fragment (row 2). Furthermore, substituting the time-dependent weighting with an instantaneous activation creates an abrupt shift in the velocity field. While this sudden update gives the highest CD_{edit}, it can induce topological fracturing, the mesh tearing into disconnected fragments, as with the detached leg in row 2, and it lowers $\text{NC}_{\text{preserve}}$ and the F-score against the full model. These results suggest that the smooth ramp helps avoid such artifacts.

Component Independence (RQ4). In Table 2, removing either mechanism worsens every metric against the full model, which suggests that both contribute to the balance. Disabling FDL (*No Mask*) causes spatial bleed into background geometry, shrinking the pot (row 2), and raises $\text{CD}_{\text{preserve}}$ to 3.23. Omitting RGG (*No Latent Ref*) leans toward trajectory under-commitment: integrating velocity differences alone leaves the target shape less fully realized, as with the faceless cat in row 2, and lowers semantic alignment (CLIP_{img}). It also gives the lowest preservation F-score in the table.

On a single A100, FocusFlow takes 29.82 s per edit, about 12% more than Inversion (26.56 s) and about 40% more than AnchorFlow (21.38 s). Appendices D and B detail computational cost and hyperparameter sensitivity. Appendix E reports the external benchmarks, robustness to a degraded reference, the Jacobian check, and the user study.

6 Conclusion

We introduced FocusFlow, a segmentation-free framework that reduces the spatial bleed and trajectory under-commitment of inversion-free probability flow editing. By temporally separating intrinsic spatial localization (Feature-Divergence Localization) and terminal target convergence (Reference Generation Guidance), the integration trajectory protects unedited geometry during coarse structure formation and drives endpoint convergence once coarse geometry has formed. On the joint GSO and PartObjaverse-Tiny benchmark, the phase-scheduled integration obtains the best preservation scores and the largest edit magnitude among compared methods. This suggests that it can reduce the conventional trade-off between geometric preservation and semantic edit magnitude without requiring manual spatial annotations during editing. More broadly, decomposing flow trajectories into temporally specialized phases offers a useful design pattern wherever preservation and modification compete. We view this as a step toward phase-aware scheduling for controllable generative editing more generally.

7 Limitations

Despite improving trajectory stability, the method exhibits specific limitations. First, the final mesh depends strictly on the frozen ShapeVAE [15]. Its compression of high-frequency geometry inevitably degrades fine source details, capping the maximum achievable geometric preservation regardless of latent precision. As a training-free method, however, FocusFlow can adopt stronger base generators without retraining. Second, FDL relies on accurate semantic mapping by the frozen DINO encoder [35]. If the patch-wise cosine distance fails to cleanly isolate the modified region, the spatial gate incorrectly bounds the update, causing localized spatial bleed. Third, RGG adds a one-time 30-step reference-generation pass, which raises runtime by about 40% over AnchorFlow. Finally, relying on a rigidly fixed temporal boundary s^{mask} may retire the spatial gate too early or too late for certain shapes. We discuss future work and broader impacts in Appendices G and H.

Acknowledgments and Disclosure of Funding

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2026-25584577, KAIST Deep Tech Bridge Center).

References

[1] Roi Bar-On, Dana Cohen-Bar, and Daniel Cohen-Or. EditP23: 3D Editing via Propagation of Image Prompts to Multi-View. *arXiv preprint arXiv:2506.20652*, 2025.

[2] Minh-Quan Viet Bui, Jaeho Moon, and Munchurl Kim. AirSplat: Alignment and Rating for Robust Feed-Forward 3D Gaussian Splatting. In *European Conference on Computer Vision*, 2026.

[3] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. DiffEdit: Diffusion-based semantic image editing with mask guidance. In *International Conference on Learning Representations*, 2023.

[4] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B. McHugh, and Vincent Vanhoucke. Google Scanned Objects: A High-Quality Dataset of 3D Scanned Household Items. In *International Conference on Robotics and Automation*, 2022.

[5] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *International Conference on Machine Learning*, 2024.

[6] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[7] Will Gao, Dilin Wang, Yuchen Fan, Aljaz Bozic, Tuur Stuyck, Zhengqin Li, Zhao Dong, Rakesh Ranjan, and Nikolaos Sarafianos. 3D Mesh Editing using Masked LRMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[8] William Gao, Noam Aigerman, Thibault Groueix, Vladimir G. Kim, and Rana Hanocka. TextDeformer: Geometry Manipulation using Text Guidance. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023.

[9] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[10] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-Prompt Image Editing with Cross-Attention Control. In *International Conference on Learning Representations*, 2023.

[11] Ji Woo Hong, Tri Ton, Trung X. Pham, Gwanhyeong Koo, Sunjae Yoon, and Chang D. Yoo. ITA-MDT: Image-Timestep-Adaptive Masked Diffusion Transformer Framework for Image-Based Virtual Try-On. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[12] 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. In *International Conference on Learning Representations*, 2024.

[13] James Jincheng Hu, Yuxiao Wu, Youcheng Cai, and Ligang Liu. CraftMesh: High-Fidelity Generative Mesh Manipulation via Poisson Seamless Fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[14] Shimin Hu, Yuanyi Wei, Fei Zha, Yudong Guo, and Juyong Zhang. Easy3E: Feed-Forward 3D Asset Editing via Rectified Voxel Flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[15] Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, et al. Hunyuan3D 2.1: From Images to High-Fidelity 3D Assets with Production-Ready PBR Material. *arXiv preprint arXiv:2506.15442*, 2025.

[16] Guanlong Jiao, Biqing Huang, Kuan-Chieh Wang, and Renjie Liao. UniEdit-Flow: Unleashing Inversion and Editing in the Era of Flow Models. In *International Conference on Learning Representations*, 2026.

[17] Heewoo Jun and Alex Nichol. Shap-E: Generating Conditional 3D Implicit Functions. *arXiv preprint arXiv:2305.02463*, 2023.

[18] Jeongsol Kim, Yeobin Hong, Jonghyun Park, and Jong Chul Ye. FlowAlign: Trajectory-Regularized, Inversion-Free Flow-based Image Editing. In *International Conference on Learning Representations*, 2026.

[19] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM Transactions on Graphics*, 2017.

[20] Gwanhyeong Koo, Hyunsu Kim, Youngji Kim, Taejae Lee, Siwoo Lim, Sunjae Yoon, Suyong Yeon, and Chang D. Yoo. InSpace: Structure-Aware 3D Indoor Scene Generation from a Single 360° Image. In *European Conference on Computer Vision*, 2026.

[21] Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. FlowEdit: Inversion-Free Text-Based Editing Using Pre-Trained Flow Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[22] 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 *International Conference on 3D Vision*, 2026.

[23] Siwoo Lim, Sunjae Yoon, Gwanhyeong Koo, Hyeonseo Yun, and Chang D. Yoo. TanGO: Training-Free 3D Editing via Tangent-Space Guidance and Optimization. In *European Conference on Computer Vision*, 2026.

[24] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[25] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*, 2023.

[26] Anran Liu, Cheng Lin, Yuan Liu, Xiaoxiao Long, Zhiyang Dou, Hao-Xiang Guo, Ping Luo, and Wenping Wang. Part123: Part-aware 3D Reconstruction from a Single-view Image. In *ACM SIGGRAPH 2024 Conference Papers*, 2024.

[27] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *International Conference on Learning Representations*, 2023.

[28] Shitong Luo and Wei Hu. Diffusion Probabilistic Models for 3D Point Cloud Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[29] Tung M. Luu, Hwanhee Kim, Younghwan Lee, and Chang D. Yoo. Video-Based Optimal Transport for Feedback-Efficient Offline Preference-Based Reinforcement Learning. In *International Conference on Machine Learning*, 2026.

[30] Tung M. Luu, Tri Ton, Duc M. Nguyen, Xuan Thanh Nguyen, Nguyen Gia Binh, Quang Tan Nguyen, Vu Duong, Daniel Sonntag, Duy M. H. Nguyen, Vien Anh Ngo, et al. Progress-Guided Preference Generation for Aligning Generative Control Policies. In *Conference on Robot Learning*, 2026.

[31] Neil A. Macmillan and C. Douglas Creelman. *Detection Theory: A User's Guide*. Psychology Press, 2004.

[32] Tuan Duc Ngo, Chuang Gan, and Evangelos Kalogerakis. VolFill: Single-View Amodal 3D Scene Reconstruction with Volumetric Flow Matching. In *Advances in Neural Information Processing Systems*, 2026.

[33] Thanh Nguyen, Tri Ton, Hongbin Choe, Tung M. Luu, and Chang D. Yoo. Fast and Highly Expressive Policy Learning for Offline Reinforcement Learning via Bootstrapped Flow Q-Learning. In *International Conference on Machine Learning*, 2026.

[34] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-E: A System for Generating 3D Point Clouds from Complex Prompts. *arXiv preprint arXiv:2212.08751*, 2022.

[35] Maxime Oquab, Timothée Darcet, Théo 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.

[36] 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*, 2026.

[37] Jongmin Park, Minh-Quan Viet Bui, Juan Luis Gonzalez Bello, Jaeho Moon, Jihyong Oh, and Munchurl Kim. EcoSplat: Efficiency-controllable Feed-forward 3D Gaussian Splatting from Multi-view Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[38] Trung X. Pham, Tri Ton, and Chang D. Yoo. MDSGen: Fast and Efficient Masked Diffusion Temporal-Aware Transformers for Open-Domain Sound Generation. In *International Conference on Learning Representations*, 2025.

[39] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D Diffusion. In *International Conference on Learning Representations*, 2023.

[40] 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*, 2021.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[42] Joshua Tian Jin Tee, Hee Suk Yoon, Sunjae Yoon, Tri Ton, and Chang D. Yoo. Diffusion Negative Preference Optimization Made Simple. In *International Conference on Learning Representations*, 2026.

[43] Tri Ton, Ji Woo Hong, SooHwan Eom, Jun Yeop Shim, Junyeong Kim, and Chang D. Yoo. Zero-Shot Dual-Path Integration Framework for Open-Vocabulary 3D Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024.

[44] Tri Ton, Ji Woo Hong, and Chang D. Yoo. TARO: Timestep-Adaptive Representation Alignment with Onset-Aware Conditioning for Synchronized Video-to-Audio Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[45] Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Jingxi Xu, Philip Torr, Xun Cao, and Yao Yao. Direct3D: Scalable Image-to-3D Generation via 3D Latent Diffusion Transformer. In *Advances in Neural Information Processing Systems*, 2024.

[46] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[47] Xinhao Yan, Jiachen Xu, Yang Li, Changfeng Ma, Yunhan Yang, Chunshi Wang, Zibo Zhao, Zeqiang Lai, Yunfei Zhao, Zhuo Chen, et al. X-Part: High Fidelity And Structure Coherent Shape Decomposition And Completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[48] 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.

[49] Junliang Ye, Shenghao Xie, Ruowen Zhao, Zhengyi Wang, Hongyu Yan, Wenqiang Zu, Lei Ma, and Jun Zhu. Nano3D: A Training-Free Approach for Efficient 3D Editing Without Masks. In *International Conference on Learning Representations*, 2026.

[50] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. CLAY: A Controllable Large-scale Generative Model for Creating High-quality 3D Assets. *ACM Transactions on Graphics*, 2024.

[51] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3D: Exploring Unified 3D Representation at Scale. In *International Conference on Learning Representations*, 2024.

[52] Zhenglin Zhou, Fan Ma, Chengzhuo Gui, Xiaobo Xia, Hehe Fan, Yi Yang, and Tat-Seng Chua. AnchorFlow: Training-Free 3D Editing via Latent Anchor-Aligned Flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

A Algorithm

Algorithm 1 details the precise execution. By sequencing the integration into the localization phase ($i < s^{\text{mask}}$) and the guidance phase ($i \geq s^{\text{mask}}$), the spatial gate and the endpoint pull never act on the same step. In the guidance phase, the update subtracts the reference gradient to drive convergence to the target geometry.

Algorithm 1: FocusFlow phase-scheduled integration.

Input: Source mesh, source condition c^{src}, target condition c^{tar}, discrete integration steps T, edit range $n_{\max}, n_{\min}$, discrete phase boundary s^{mask}, discrete smoothing duration s^{smooth}, preservation floor α, sharpness exponent γ, reference weight w.

Output: Edited clean latent z_1^{edit}.

`// Initialization`

$\phi^{\text{src}}, \phi^{\text{tar}} \leftarrow \mathcal{E}_{\text{DINO}}(c^{\text{src}}), \mathcal{E}_{\text{DINO}}(c^{\text{tar}})$;

Define active edit region K_{changed} where cosine distance $D_p > \mu(D) + \sigma(D)$; `// Eq. 4`

$z^{\text{ref}} \leftarrow \text{Generator}(c^{\text{tar}})$; `// Reference latent`

$z_1^{\text{src}} \leftarrow$ ShapeVAE encoding of source mesh;

$n_{\text{edit}} \leftarrow n_{\max} - n_{\min}$; `// Number of edit steps`

$z_{t_0}^{\text{edit}} \leftarrow z_1^{\text{src}}$;

for $i = 0, \ldots, n_{\text{edit}} - 1$ **do**

  $t_i \leftarrow \text{TimeSchedule}(T - n_{\max} + i)$; `// Skips the first` $T - n_{\max}$ `steps`

  Sample $\epsilon \sim \mathcal{N}(0, I)$; $z_{t_i}^{\text{src}} \leftarrow t_i z_1^{\text{src}} + (1 - t_i)\epsilon$;

  $z_{t_i}^{\text{tar}} \leftarrow z_{t_i}^{\text{edit}} + z_{t_i}^{\text{src}} - z_1^{\text{src}}$;

  $v_{t_i}^{\text{edit}} \leftarrow v_\theta(z_{t_i}^{\text{tar}}, t_i, c^{\text{tar}}) - v_\theta(z_{t_i}^{\text{src}}, t_i, c^{\text{src}})$;

  Compute AnchorFlow alignment gradient $\nabla\mathcal{L}_{\text{align}}$;

  if $i < s^{\text{mask}}$ **then**

    `// Localization Phase`

    Extract multi-block cross-attention scores $A_{t_i}^{\text{src}}, A_{t_i}^{\text{tar}}$;

    Compute score_n over K_{changed} ; `// Eq. 5`

    Form soft mask m parameterized by (α, γ) ; `// Eq. 6`

    $\Delta z_{t_i} \leftarrow \left(v_{t_i}^{\text{edit}} - \nabla\mathcal{L}_{\text{align}}\right) \odot m \, \mathrm{d}t_i$; `// Eq. 7`

  else

    `// Guidance Phase`

    $\hat{z}_1^{\text{edit}} \leftarrow z_{t_i}^{\text{edit}} + (1 - t_i) v_{t_i}^{\text{edit}}$; `// Clean-latent forecast`

    $r(i) \leftarrow \min\left(1, \max\left(0, \frac{i - s^{\text{mask}}}{s^{\text{smooth}}}\right)\right)$; `// Trajectory smoothing via step index`

    $\nabla\mathcal{L}_{\text{ref}} \leftarrow (2 - t_i)(\hat{z}_1^{\text{edit}} - z^{\text{ref}})$; `// Eq. 11`

    $\Delta z_{t_i} \leftarrow \left(v_{t_i}^{\text{edit}} - \nabla\mathcal{L}_{\text{align}} - w \, r(i) \nabla\mathcal{L}_{\text{ref}}\right)\mathrm{d}t_i$; `// Eq. 12`

  $z_{t_{i+1}}^{\text{edit}} \leftarrow z_{t_i}^{\text{edit}} + \Delta z_{t_i}$;

return z_1^{edit};

B Hyperparameter Sensitivity

We evaluate sensitivity along five axes, varying one hyperparameter at a time while holding the others at the defaults defined in Sec. 5.1 ($\alpha = 0.2$, $\gamma = 1.0$, $w = 0.3$, $s^{\text{mask}} = 5$, $s^{\text{smooth}} = 15$, and 30 denoising steps for the reference latent z^{ref}). All evaluations are performed on the joint GSO and PartObjaverse-Tiny benchmark.

Preservation floor α. The floor α is the minimum fraction of the update that every token receives during the localization phase. Table 3 sweeps $\alpha \in \{0.0, 0.2, 0.4, 0.6\}$. The default $\alpha = 0.2$ achieves the optimal performance across all metrics. Setting $\alpha = 0$ gives low-score tokens almost no update, which degrades $\text{CD}_{\text{preserve}}$ to 3.44. Larger floors ($\alpha \geq 0.4$) allow the edit signal to flood into preservation tokens, monotonically worsening $\text{CD}_{\text{preserve}}$ from 3.51 at $\alpha = 0.4$ to 3.65 at $\alpha = 0.6$.

Table 3: Sensitivity to the preservation floor α.

α	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$	$\mathrm{CD}_{\mathrm{edit}}\uparrow$	$\mathrm{NC}_{\mathrm{preserve}}\uparrow$	F-score$\uparrow$	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$
0.0	3.44	1.50	56.86	2.43	63.06	23.21
0.2	**3.11**	**1.61**	**58.20**	**3.05**	**63.85**	**23.91**
0.4	3.51	1.40	56.73	2.58	62.44	22.97
0.6	3.65	1.45	56.74	2.57	62.62	23.00

Sharpness exponent γ. The exponent γ dictates the transition speed between preservation (near α) and edit (near 1) states. Table 4 reports $\gamma \in \{0.5, 1.0, 2.0\}$. The linear regime ($\gamma = 1.0$) provides the best balance. A flatter mask ($\gamma = 0.5$) inflates $\mathrm{CD}_{\mathrm{edit}}$ to 1.65 by spreading the edit signal but significantly degrades preservation metrics ($\mathrm{CD}_{\mathrm{preserve}}$ 3.52, $\mathrm{NC}_{\mathrm{preserve}}$ 56.65). Sharper masks ($\gamma = 2.0$) approach a binary gate, recovering the degradation patterns discussed in Sec. 5.5.

Table 4: Sensitivity to the sharpness exponent γ.

γ	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$	$\mathrm{CD}_{\mathrm{edit}}\uparrow$	$\mathrm{NC}_{\mathrm{preserve}}\uparrow$	F-score$\uparrow$	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$
0.5	3.52	**1.65**	56.65	2.56	62.32	23.01
1.0	**3.11**	1.61	**58.20**	**3.05**	**63.85**	**23.91**
2.0	3.52	1.53	56.74	2.38	62.83	23.24

Reference weight w. This weight balances the differential edit velocity v_t^{edit} against the clean-latent residual. As shown in Table 5, the default $w = 0.3$ is optimal. While metrics exhibit some non-monotonicity at intermediate weights, $\mathrm{CLIP}_{\mathrm{img}}$ falls from 63.85 at $w = 0.3$ to 60.88 at $w = 1.5$. This confirms the trajectory collapse predicted in Sec. 4.3, where the residual dominates the update and reverts the edit toward a direct copy of z^{ref}, erasing the source identity.

Table 5: Sensitivity to the reference weight w.

w	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$	$\mathrm{CD}_{\mathrm{edit}}\uparrow$	$\mathrm{NC}_{\mathrm{preserve}}\uparrow$	F-score$\uparrow$	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$
0.3	**3.11**	**1.61**	**58.20**	**3.05**	**63.85**	**23.91**
0.5	3.44	1.47	56.69	2.51	62.78	23.05
1.0	3.40	1.45	56.65	2.56	62.24	22.89
1.5	3.52	1.54	56.65	2.51	60.88	22.29

Mask boundary s^{mask}. Table 6 sweeps the discrete phase boundary. The shortest setting ($s^{\mathrm{mask}} = 5$) yields the best results. Extending the localization phase beyond this point ties the spatial gate to fine detail rather than coarse layout, causing $\mathrm{CD}_{\mathrm{preserve}}$ to rise from 3.11 to 3.41 as s^{mask} increases to 10. Similarly, the F-score drops from 3.05 to 2.39 as the guidance pull is delayed until step 15.

Denoising steps for z^{ref}. We vary the number of steps used to precompute the reference latent z^{ref} while holding the editing budget fixed. Table 7 reports results for 20, 30, and 50 steps. Thirty steps provide the optimal balance; reducing to 20 steps leaves z^{ref} under-converged, weakening the guidance pull and degrading $\mathrm{CD}_{\mathrm{preserve}}$ to 3.41. Increasing to 50 steps yields marginal gains in semantic alignment but degrades structural preservation, suggesting diminishing returns relative to the overhead.

C Implementation Details of Baselines

To ensure a rigorous and reproducible comparison, we provide the specific configurations for all external baseline methods. All experiments are conducted using a fixed random seed of 42.

Direct Editing. This baseline generates a new mesh from scratch with the Hunyuan3D-2.1 flow-based generator [15], conditioned on the target image. We utilize a target guidance scale of 5.0 and an integration schedule of 50 denoising steps.

Table 6: Sensitivity to the discrete mask boundary s^{mask}.

s^{mask}	$\text{CD}_{\text{preserve}}\downarrow$	$\text{CD}_{\text{edit}}\uparrow$	$\text{NC}_{\text{preserve}}\uparrow$	F-score$\uparrow$	$\text{CLIP}_{\text{img}}\uparrow$	$\text{CLIP}_{\text{txt}}\uparrow$
5	**3.11**	**1.61**	**58.20**	**3.05**	**63.85**	**23.91**
10	3.41	1.55	56.95	2.52	62.55	23.00
15	3.34	1.51	56.80	2.39	62.49	22.74

Table 7: Sensitivity to the number of denoising steps for z^{ref}.

Steps	$\text{CD}_{\text{preserve}}\downarrow$	$\text{CD}_{\text{edit}}\uparrow$	$\text{NC}_{\text{preserve}}\uparrow$	F-score$\uparrow$	$\text{CLIP}_{\text{img}}\uparrow$	$\text{CLIP}_{\text{txt}}\uparrow$
20	3.41	1.51	56.71	2.54	62.64	23.14
30	**3.11**	1.61	**58.20**	3.05	63.85	23.91
50	3.14	**1.66**	58.16	**3.07**	**64.19**	**23.95**

Editing-by-Inversion. Following the UniEdit-Flow framework [16], we perform conditional inversion using the source image (guidance scale 1.0) for 50 steps. The subsequent forward generation uses the target image condition with a guidance scale of 5.0, resulting in a total of 100 function evaluations (NFEs).

Inversion-free. We implement the velocity difference integration based on the FlowEdit formulation [21] on top of the Hunyuan3D backbone. To ensure a fair comparison with our framework, we use the same schedule as FocusFlow ($T = 50$, $n_{\max} = 41$, $n_{\min} = 1$). We set the target and source guidance scales to 7.5 and 3.5, respectively, with $n_{\text{avg}} = 1$ noise sample per step.

AnchorFlow. We adopt the identical temporal integration schedule ($n_{\max} = 41$) and guidance scales (7.5/3.5) as the Inversion-free implementation. We incorporate the official alignment gradient parameters to stabilize the trajectory through latent anchoring.

Nano3D. Unlike the other baselines, Nano3D [49] is built on TRELLIS [46]. We follow the official implementation for the sparse structure encoder. Assets are processed using the authors' provided voxelization and rendering utilities to ensure structural fidelity.

D Computational Efficiency

Table 8: Computational cost comparison.

Method	Runtime (s)	VRAM (GB)
Inversion [16]	26.56 ± 0.10	11.12
AnchorFlow [52]	21.38 ± 0.10	12.50
Nano3D [49]	20.68 ± 0.68	10.53
FocusFlow	29.82 ± 1.73	13.76

We evaluate the computational efficiency of FocusFlow against the baseline editing frameworks. Table 8 reports the average of 10 inference runtimes and VRAM usage on a single NVIDIA A100 GPU. FocusFlow requires a one-time forward pass (30 steps) to generate the reference latent z^{ref}, followed by the 40-step editing integration. This structure avoids the per-instance optimization overhead typical of score distillation sampling (SDS)-based methods [39] while maintaining competitive runtime against deterministic inversion pipelines.

E Additional Evaluation

This appendix reports additional experiments that support Section 5.

E.1 External Benchmarks

To test FocusFlow beyond Edit3D-Bench, we also run it on two external benchmarks, Eval3DEdit [52] and Nano3D-Edit [49], under their official protocols, with 100 cases each. We report alignment scores based on CLIP [40], Uni3D [51], and DINO [35]. These sets have no 3D region annotations, so we annotated 3D regions manually to compute $\mathrm{CD}_{\mathrm{preserve}}$.

FocusFlow has the best value on all six metrics on both sets (Tables 9 and 10). The CLIP, Uni3D, and DINO scores need no region annotation or reference geometry, so that part of the ranking does not depend on our annotations. The $\mathrm{CD}_{\mathrm{preserve}}$ values here are not comparable with Table 1, because the sets and region annotations differ.

On a resolution-normalized 64^3 voxelization, chosen to stay well-defined on raw scan meshes, all three benchmarks have a median of about one connected component (26-connectivity) and a median surface occupancy of about 0.04, so their shapes are similar in connectivity and spatial extent. We attribute differences in raw face count mainly to scan tessellation.

Table 9: Results on Eval3DEdit (100 cases). We mark the **best** and second-best results.

Method	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$	$\mathrm{Uni3D}_{\mathrm{pc}}\uparrow$	$\mathrm{Uni3D}_{\mathrm{txt}}\uparrow$	$\mathrm{DINO}_{\mathrm{img}}\uparrow$	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$
Direct Editing [15]	56.12	43.08	27.85	17.20	46.88	0.149
Inversion [16]	64.64	45.83	30.03	21.53	44.53	0.082
Inversion-free [21]	64.51	45.60	30.75	21.50	43.14	0.078
AnchorFlow [52]	68.26	48.00	33.75	23.39	52.28	0.124
Nano3D [49]	69.50	47.96	34.45	24.17	53.72	0.075
FocusFlow (Ours)	**71.73**	**49.40**	**35.21**	**25.02**	**57.55**	**0.067**

Table 10: Results on Nano3D-Edit (100 cases). We mark the **best** and second-best results.

Method	$\mathrm{CLIP}_{\mathrm{img}}\uparrow$	$\mathrm{CLIP}_{\mathrm{txt}}\uparrow$	$\mathrm{Uni3D}_{\mathrm{pc}}\uparrow$	$\mathrm{Uni3D}_{\mathrm{txt}}\uparrow$	$\mathrm{DINO}_{\mathrm{img}}\uparrow$	$\mathrm{CD}_{\mathrm{preserve}}\downarrow$
Direct Editing [15]	60.34	40.15	29.80	14.50	45.28	0.2591
Inversion [16]	59.18	42.78	31.72	16.02	42.93	0.1352
Inversion-free [21]	59.44	41.37	31.95	16.00	41.54	0.1287
AnchorFlow [52]	59.43	42.66	31.61	16.05	43.86	0.1265
Nano3D [49]	59.08	42.14	32.38	16.43	41.00	0.1138
FocusFlow (Ours)	**62.02**	**43.43**	**33.56**	**17.06**	**48.01**	**0.0803**

E.2 Robustness to a Misaligned or Failed Reference

The experiments in this subsection and in Appendix E.3 use a subset of 100 of the 300 editing cases. RGG relies on an independently generated z^{ref}, which can be misaligned with the source or fail outright. To test how FocusFlow behaves in these cases at $w = 0.3$, we degrade z^{ref} in three ways: a 60° rigid rotation; over several random seeds, the reference whose preserved region deviates most from the source; and a 5-step under-converged reference. Fracturing would show as the mesh splitting into pieces, so we count connected components, and we also track $\mathrm{CD}_{\mathrm{edit}}$ and CLIP similarity to the source and target. In aggregate across all three degradations, the component count stays flat at about 14 and $\mathrm{NC}_{\mathrm{preserve}}$ barely moves, while the edit weakens: $\mathrm{CD}_{\mathrm{edit}}$ drops from 1.45 to 1.25, and the output stays closer to the source than to the target.

We also label each of 100 test cases under a degraded reference as a warp (preservation breaks and new components appear) or a safe under-commit (weaker edit, source-like output). Table 11 reports about 90% under-commit and about 10% warp. A degraded reference therefore usually costs edit strength rather than source integrity, but it warps the source in 8 to 11 of the 100 cases.

Because z^{ref} is a one-time precompute, a failed reference can also be screened. Keeping the best of K candidates by CLIP similarity to the target, which needs no ground truth, recovers most of the loss by $K = 4$ ($\mathrm{CD}_{\mathrm{preserve}}$ from 3.80 to 3.50, F-score from 2.05 to 2.30).

Table 11: Outcomes over 100 test cases under a degraded z^{ref} at $w = 0.3$ (case counts).

Degraded reference	Under-commit (safe)	Warp (unsafe)	N
5-step z^{ref}	90	10	100
60°-rotated z^{ref}	89	11	100
Worst-seed z^{ref}	92	8	100

E.3 Validity of the Scalar-Identity Jacobian Approximation

We measure the error directly. The exact gradient needs one vector-Jacobian product per guidance step, which we compute over 100 sampled edits and compare with the scalar form. Table 12 reports the cosine similarity (rotation) and norm ratio (stretch), binned by t. In these bins, the cosine similarity averages 0.934 and rises from 0.883 to 0.971 as $t \to 1$, and the norm ratio ranges from 1.05 to 1.48. The bins cover only $t \geq 0.80$, the late part of the guidance phase, which starts near $t = 0.29$.

Using the exact gradient changes the final metrics modestly: $\mathrm{CD}_{\mathrm{preserve}}$ from 3.32 to 3.24, $\mathrm{CD}_{\mathrm{edit}}$ from 1.45 to 1.35, F-score from 2.38 to 2.50, and CLIP within 0.2. It also raises the runtime by about 6.4 times in this experiment (28 to 180 s).

Table 12: Exact gradient g_{exact} against the scalar form g_{approx} over 100 sampled edits, binned by flow time t.

t range	Cosine similarity	$\|g_{\mathrm{exact}}\|/\|g_{\mathrm{approx}}\|$
0.80–0.85	0.883	1.48
0.85–0.90	0.929	1.17
0.90–0.95	0.956	1.28
0.95–1.00	0.971	1.05
Overall	**0.934**	**1.23**

E.4 Sensitivity of the FDL Threshold on Subtle Edits

FDL selects the edit region $\mathcal{K}_{\mathrm{changed}}$ with the adaptive threshold $D_p > \mu(D) + \sigma(D)$. To test this threshold on subtle edits, we stratified the 300 cases by DINO-feature divergence and took the bottom tercile, the 100 smallest-shift cases. The threshold adapts its selection to the edit: it selects 9.5% of the patches on subtle edits versus 16.9% on large edits, and against the 2D masks derived from the ground-truth 3D edit regions it reaches 83.5% recall on subtle edits and 80.0% on large edits. Subtle edits have small footprints: the annotated region covers around 4% of the image, and the selected patches cover about 9%, so the threshold over-covers the region by roughly 5% of the image.

E.5 User Study

Twenty participants evaluated 30 editing pairs sampled from GSO and PartObjaverse-Tiny. Each case was a two-alternative forced choice [31] between FocusFlow and one baseline, shown with the source render and target prompt from fixed viewpoints, with left/right order randomized. Participants answered three questions per case: prompt alignment (which model better applies the intended semantic change), visual quality (which shows fewer artifacts such as surface collapse or mesh tearing), and shape preservation (which better preserves the unedited regions and source identity).

Table 13 reports how often participants preferred FocusFlow: 75.7% to 85.8% across all comparisons and criteria, highest on shape preservation against AnchorFlow (85.0%) and Nano3D (85.8%).

Table 13: User study.

Baseline	Prompt alignment↑	Visual quality↑	Shape preservation↑
vs. Inversion [16]	77.8%	77.2%	77.0%
vs. AnchorFlow [52]	75.7%	78.7%	85.0%
vs. Nano3D [49]	75.8%	84.5%	85.8%

F Extended Qualitative Results

Due to space constraints in the main text, we provide an extended visual comparison of FocusFlow against the established baseline methods in Figure 6.

Consistent with the analysis in Section 5.3, regenerative methods [16] successfully capture the semantic target but frequently distort unedited regions. Conversely, unconstrained flow methods [21] often suffer from trajectory under-commitment, failing to fully realize the target modifications. Global updates with latent anchoring [52] improve target realization but suffer from severe spatial bleed into the background geometry. Across these cases, which span part addition, part replacement, and whole-object changes, FocusFlow realizes the target while leaving most of the unedited source geometry unchanged.

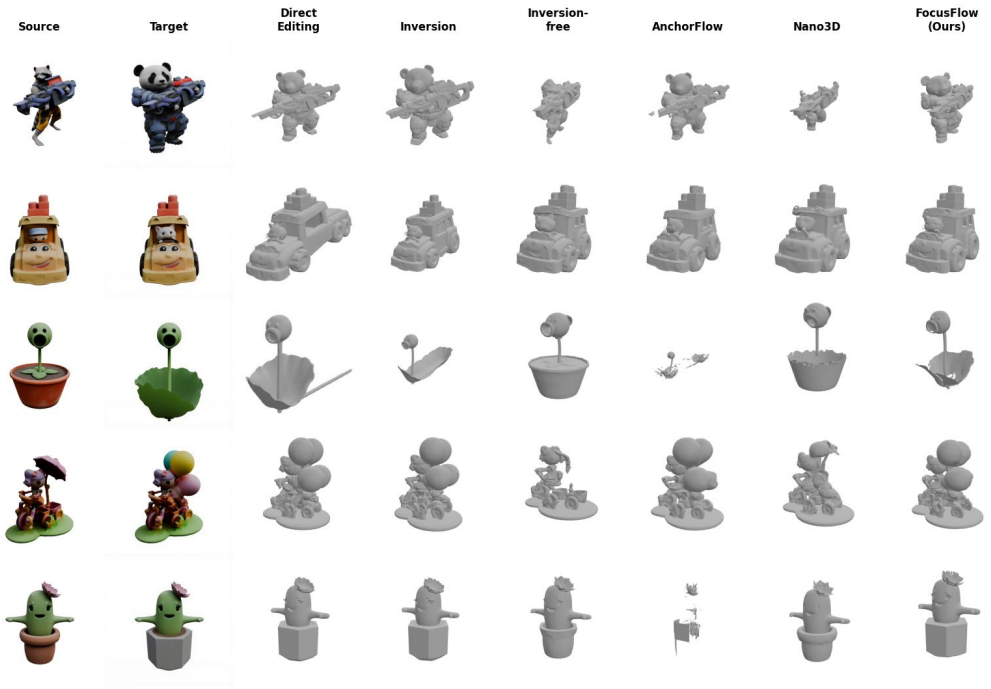

Figure 6: **Extended qualitative comparison of 3D editing baselines.** FocusFlow realizes the target while leaving most of the unedited geometry unchanged; the baselines show the spatial bleed or trajectory under-commitment.

G Future Work

In future work, we plan to extend FocusFlow in three directions. First, FDL and RGG modify only the sampling trajectory, so they could be combined with other training-free methods, such as concurrent tangent-space steering [23], and carried over to scene-level generation [20, 32], provided the generator cross-attends to image features. Editing a captured scene would also require reconstructing it, which feed-forward Gaussian splatting does from multi-view images in a single pass [37, 2]. Second, the phase boundary and the balancing coefficient are fixed across edits. A reward learned from a few pairwise human judgments [29] could select them per edit, and preference optimization [42], with pseudo-preferences generated without human annotation [30], could align the base generator itself. Third, phase-scheduled integration is not specific to 3D: audio synthesis [44, 38] and virtual try-on [11] could adopt it without retraining, changing one sound event or one garment while keeping the rest of the output. Conversely, bootstrapping flow-matching velocities into a one-step map [33] could shorten the reference pass of RGG.

H Broader Impacts

The proposed framework lowers the technical barrier to fine-grained 3D asset editing, which can benefit creators in gaming, spatial computing, and digital design. However, advancing the seamless, training-free modification of synthetic 3D media inherently introduces provenance risks, as the ability to convincingly alter existing structures without obvious artifacts could be misused for unauthorized asset manipulation or deceptive content generation. To mitigate these risks and safeguard digital authenticity, we strongly recommend that future public deployments of latent 3D editing pipelines integrate robust invisible 3D watermarking techniques or cryptographic metadata protocols to ensure verifiable asset lineage.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: They can support the main claims of the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in Section 7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide exhaustive implementation details, including solver configurations and exact hyperparameter settings in Section 5.1, as well as a complete step-by-step pseudocode in Appendix A to facilitate full reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Code is not included with this version of the paper. The method is specified by the pseudo-code in Appendix A, the solver settings and hyperparameters in Section 5.1 and Appendix B, and the baseline configurations in Appendix C.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 5.1 gives the solver configuration and all hyperparameters, Appendix B their sweeps, Appendix C the exact baseline configurations, and Appendix A step-by-step pseudo-code.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Tables 1, 2 report means over 300 editing cases without error bars; only runtime has mean ± std (Table 8).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the compute resources (NVIDIA A100) in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We strictly follow the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to Appendix H.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We strictly follow the corresponding licenses for existing assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: Appendix E.5 describes the task and the three questions asked.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [No]

    Justification: The study collected only pairwise preference choices on rendered 3D objects, with no personal data and no foreseeable risk to participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.