CONCORDFLOW: VIEW-AWARE COMPLEX 3D EDITING VIA FLOW AGGREGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing image-guided 3D editing methods provide effective control when a requested change can be clearly specified from a single view. Complex instructions, however, may involve multiple changes best observed from different viewpoints. Applying these changes sequentially through multi-round editing can lead to error accumulation, while jointly processing multiple view-specific conditions requires resolving spatially varying editing effects and potential conflicts among updates. To address these challenges, we introduce **ConcordFlow**, a mask-free and training-free framework built on TRELLIS for view-aware complex 3D editing that jointly integrates multiple conditions in a single 3D editing process. In the sparse-structure stage, trajectory-guided spatial weighting (TGSW) derives spatial weights from independent editing and restoration trajectories to identify each condition's editing influence. Disagreement-guided reweighting (DGR) then adjusts these weights during joint editing using source–target velocity disagreement. In the subsequent structured-latent stage, disagreement–visibility adaptive fusion (DVAF) likewise compares source- and target-conditioned velocities and combines the resulting disagreement with geometric visibility to guide feature generation. To evaluate our framework, we introduce ComplexEdit3D-Bench for view-aware complex 3D editing. On this benchmark, ConcordFlow achieves stronger condition alignment and better overall source preservation than the compared multi-round methods.

1 INTRODUCTION

Recent advances in 3D generation (Hong et al., 2024; Long et al., 2024; Xu et al., 2024; Li et al., 2025b; Hunyuan3D et al., 2025; Xiang et al., 2025; 2026) have enabled high-quality 3D asset synthesis from a single reference image. These models provide a practical foundation for editing 3D assets according to user instructions. A typical approach adopted by prior works (Chi et al., 2026; Zhou et al., 2026; Nie et al., 2026) is to render the source asset from multiple viewpoints, select an informative view for the requested edit, and apply an image editor to the selected rendering. The resulting source–target image pair forms a view-specific condition that guides a pretrained image-to-3D model (Xiang et al., 2025; 2026; Hunyuan3D et al., 2025) to produce the edited asset. This single-view editing pipeline provides effective guidance when the requested edit can be clearly specified from a single view.

However, users may need to make multiple changes to the same 3D asset. We refer to an instruction that jointly specifies these changes as a complex editing instruction, which can be decomposed into multiple atomic editing instructions, each describing a single desired change. We consider view-aware complex 3D editing, where the requested changes are best observed from different viewpoints and cannot all be clearly specified by a single target image. One natural strategy is multi-round editing, which applies this single-view editing pipeline sequentially under the view-specific conditions, with each round carrying out the atomic instructions assigned to one view. A key limitation of this strategy is that each round operates on the output of the previous one, causing unintended deviations to be inherited by subsequent edits. This process can weaken earlier edits and accumulate unintended changes in unrelated regions. Fig. 1 (left) illustrates how later edits can alter earlier modifications and unrelated source content.

To mitigate error accumulation across editing rounds, we study single-round editing for complex instructions: all target images are constructed beforehand and jointly integrated in a single 3D editing

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

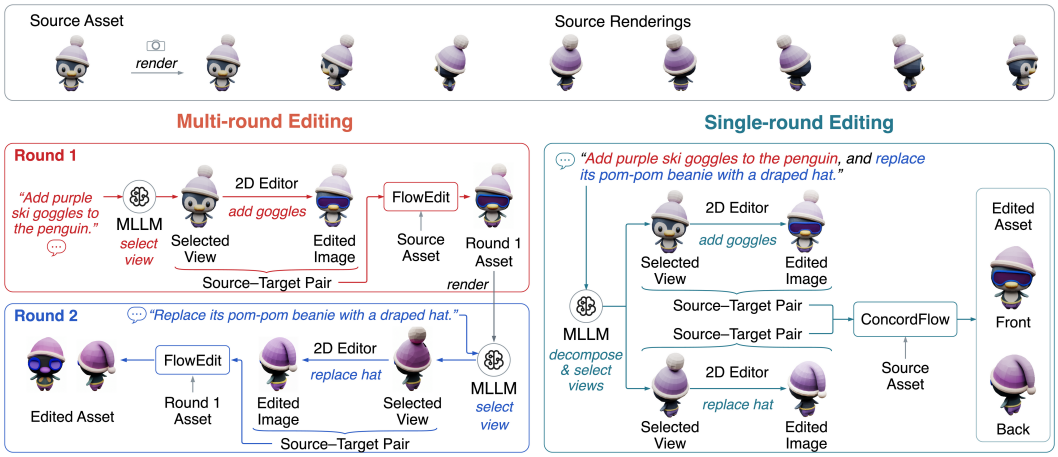


Figure 1: Multi-round versus single-round 3D editing. Left: sequential editing alters previously added goggles and unrelated facial appearance. Right: ConcordFlow jointly integrates view-specific conditions in a single round, fulfilling both editing requests while better preserving source appearance.

process. Each target image specifies only its assigned edits and may retain source content in regions modified under other conditions. Jointly integrating multiple view-specific conditions in a single round poses two challenges. First, their editing effects vary across 3D locations, so the shared update must account for where each condition is relevant. Second, different conditions may induce conflicting updates in the same 3D region, causing mutual interference.

We address these challenges with ConcordFlow, a mask-free and training-free framework built upon TRELLIS (Xiang et al., 2025) for single-round complex 3D editing. Our framework introduces three components across TRELLIS's sparse-structure and structured-latent (SLat) stages. In the sparse-structure stage, trajectory-guided spatial weighting (TGSW) derives spatial weights from the displacements along independent editing and restoration trajectories, assigning higher relative weights where these displacements are substantial and approximately cancel each other. To mitigate conflicts among condition-specific updates, disagreement-guided reweighting (DGR) uses source–target velocity disagreement at the same latent state to refine the spatial weights during joint editing. In the subsequent SLat stage, we generate features on the edited structure under multiple view-specific target conditions. Disagreement–visibility adaptive fusion (DVAF) combines source–target velocity disagreement with geometric visibility to weight the resulting velocity predictions. Together, these components jointly integrate multiple target images to edit a 3D asset. Fig. 1 (right) shows that ConcordFlow fulfills both editing requests while better preserving source appearance.

Our main contributions are summarized as follows:

- We study view-aware complex 3D editing, jointly integrating multiple view-specific conditions in a single round to realize a complex instruction on one 3D asset.
- We propose ConcordFlow, a mask-free and training-free framework that coordinates view-specific contributions through trajectory- and disagreement-guided weighting during sparse-structure editing, followed by disagreement- and visibility-guided fusion during SLat generation.
- We construct a benchmark for view-aware complex 3D editing to comprehensively evaluate our method in terms of both source preservation and condition alignment.

2 Related Work

2.1 Complex Image Editing

Text-guided image editing has been developed through stochastic noising and inversion (Meng et al., 2022; Mokady et al., 2023), attention control (Hertz et al., 2023), instruction tuning (Brooks et al., 2023), and rectified-flow transport (Wang et al., 2025; Kulikov et al., 2025; Kim et al., 2026). Most of these methods formulate editing as a single source–target transformation. For complex instructions,

prior works either apply decomposed edits sequentially (Zhou et al., 2025) or integrate multiple editing effects within a single process (Huang et al., 2024; Jiang et al., 2026; Yoon et al., 2025). In particular, FlowDC (Jiang et al., 2026) and SplitFlow (Yoon et al., 2025) decompose complex target prompts and use the resulting sub-prompts to coordinate editing directions in flow space. Nevertheless, these effects are expressed within a single image coordinate frame. Extending this decomposition to 3D editing with image-to-3D models requires assigning atomic instructions to informative viewpoints and coordinating view-specific conditions over a shared 3D representation.

2.2 3D Generation and Editing

DreamFusion (Poole et al., 2023) and subsequent text-to-3D methods (Chen et al., 2023; Lin et al., 2023; Shi et al., 2024; Tang et al., 2024; Wang et al., 2023) optimize a 3D representation using guidance from pretrained 2D diffusion models. In contrast, 3DShape2VecSet (Zhang et al., 2023) and XCube (Ren et al., 2024) learn generative priors using compact vector-set and hierarchical sparse-voxel representations, respectively. Subsequent native 3D models (Zhang et al., 2024; Li et al., 2025b; Hunyuan3D et al., 2025; Xiang et al., 2025; 2026) further advance generation in learned 3D latent spaces, with our framework built upon TRELLIS (Xiang et al., 2025).

Early 3D editing approaches use generative optimization (Haque et al., 2023; Sella et al., 2023), learned latent-space transformations (Chen et al., 2024b), or edited renderings to guide reconstruction and asset updates (Chen et al., 2024a; Qi et al., 2024; Gao et al., 2025; Barda et al., 2025; Bar-On et al., 2025; Huang et al., 2025; Zheng et al., 2025; Li et al., 2025a). More recent methods operate directly within pretrained 3D generative models. Inversion-based methods such as VoxHammer (Li et al., 2026) and VineDresser3D (Chi et al., 2026) reuse source information along inversion trajectories to preserve unedited regions. Inversion-free methods include Nano3D (Ye et al., 2026), which combines FlowEdit with voxel- and SLat-level merging; AnchorFlow (Zhou et al., 2026), which aligns source and target flows using latent anchors; and VS3D (Liu et al., 2026) and TanGO (Lim et al., 2026), which regulate editing signals in velocity and tangent spaces, respectively. Learned approaches include EditVerse3D (Yin et al., 2026) and PartFlow (Weng et al., 2026), which learn region-aware flow models and semantic-part transformations, respectively. Our focus is instead on jointly coordinating multiple view-specific conditions over one shared 3D representation.

3 Method

3.1 Problem Formulation

Given a source 3D asset and a complex instruction specifying multiple desired changes, a multimodal large language model (MLLM) receives the instruction and calibrated renderings of the asset. It decomposes the instruction into atomic editing instructions and assigns each to an informative view. Multiple instructions may be assigned to the same view. We consider settings with $M \geq 2$ selected views, indexed by $i \in \{1, \ldots, M\}$.

Rendering the source asset from selected view i gives a source image. An image editor jointly applies the instructions assigned to that view, producing a target image. This source–target image pair defines a view-specific condition, with corresponding conditioning signals C_i^{src} and C_i^{tar} used by the flow model. Our goal is to produce an edited 3D asset that satisfies the complete instruction while preserving unrelated geometry and appearance.

3.2 Preliminaries

FlowEdit. We consider a pretrained rectified-flow model (Liu et al., 2023; Lipman et al., 2023) with latent state Z_t evolving according to $\mathrm{d}Z_t = V(Z_t, t, C)\,\mathrm{d}t$, where V is the conditional velocity field, $t \in [0, 1]$, and C is a conditioning signal. Given a clean source latent Z^{src}, source condition C^{src}, and target condition C^{tar}, FlowEdit (Kulikov et al., 2025) evolves a direct editing trajectory Z_t^{FE}. At time t, it samples $\epsilon_t \sim \mathcal{N}(0, I)$ and forms paired states

$$Z_t^{\mathrm{src}} = (1-t)Z^{\mathrm{src}} + t\epsilon_t, \qquad Z_t^{\mathrm{tar}} = Z_t^{\mathrm{FE}} + Z_t^{\mathrm{src}} - Z^{\mathrm{src}}. \tag{1}$$

The velocity difference is defined as

$$\Delta V_t = V(Z_t^{\mathrm{tar}}, t, C^{\mathrm{tar}}) - V(Z_t^{\mathrm{src}}, t, C^{\mathrm{src}}). \tag{2}$$

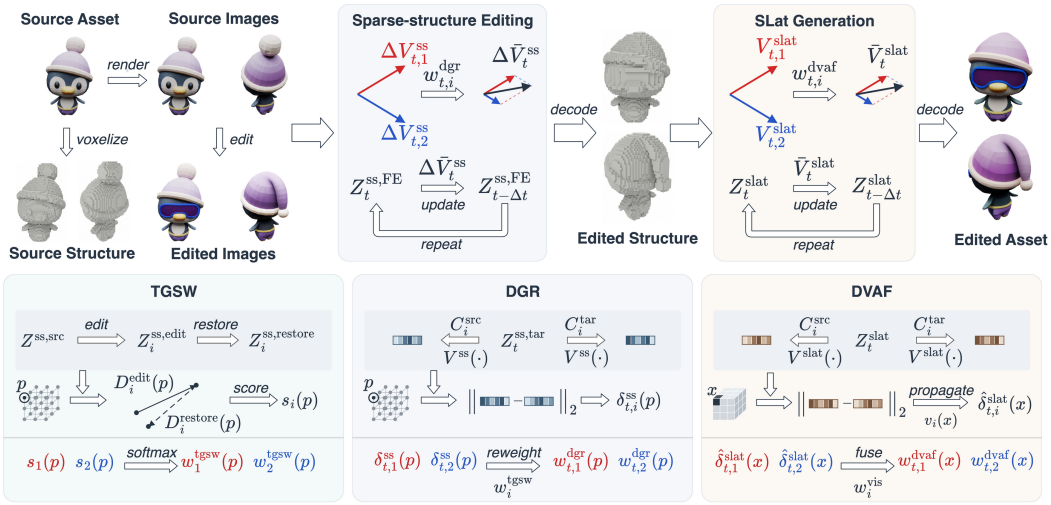


Figure 2: Overview of ConcordFlow. TGSW derives spatial weights from independent editing and restoration trajectories. DGR adjusts these weights using source–target velocity disagreement during joint sparse-structure editing. DVAF combines velocity disagreement with geometric visibility to weight condition-specific velocities during SLat generation.

For a descending Euler schedule with step size $\Delta t > 0$, the editing update is $Z^{\mathrm{FE}}_{t-\Delta t} = Z^{\mathrm{FE}}_t - \Delta t\, \Delta V_t$, initialized with $Z^{\mathrm{FE}}_{t_{\max}} = Z^{\mathrm{src}}$. Here, $t_{\max}$ is the starting timestep.

TRELLIS. TRELLIS (Xiang et al., 2025) represents a 3D asset as a structured latent (SLat): active locations on a sparse 3D grid, each with a local feature. Its pipeline has two rectified-flow stages. The first uses a dense flow transformer to generate a continuous structure latent, which is decoded into active voxel locations. The second uses a sparse flow transformer to generate SLat features at these locations. These features encode geometry and appearance and are decoded into the final 3D asset. V^{ss} and V^{slat} denote the respective velocity fields of the two stages.

3.3 ConcordFlow

Our framework, ConcordFlow, is illustrated in Fig. 2. We first use trajectory-guided spatial weighting (TGSW) to derive spatial weights from independent editing and restoration trajectories. During joint sparse-structure editing, disagreement-guided reweighting (DGR) adjusts these weights using source–target velocity disagreement. Finally, disagreement–visibility adaptive fusion (DVAF) combines predictions from different conditions for SLat generation.

Trajectory-Guided Spatial Weighting. Different view-specific conditions specify different changes, whose influence varies across 3D locations. To determine how to weight their contributions during sparse-structure editing, we first run FlowEdit (Kulikov et al., 2025) separately for each condition and measure the resulting latent displacement. Let $\mathrm{FE}(\cdot)$ denote FlowEdit's terminal latent. For condition i, the edited latent $Z_i^{\mathrm{ss,edit}}$ and forward displacement D_i^{edit} are

$$Z_i^{\mathrm{ss,edit}} = \mathrm{FE}(Z^{\mathrm{ss,src}}; C_i^{\mathrm{src}}, C_i^{\mathrm{tar}}), \qquad D_i^{\mathrm{edit}} = Z_i^{\mathrm{ss,edit}} - Z^{\mathrm{ss,src}}, \tag{3}$$

where $Z^{\mathrm{ss,src}}$ denotes the source sparse-structure latent. Ideally, each condition induces larger displacements in edited regions than elsewhere. In practice, stochastic variation in FlowEdit can also displace unrelated latent elements, making the forward displacement magnitude alone an unreliable measure of local spatial influence.

We therefore introduce TGSW, which complements the forward displacement with a restoration trajectory computed from $Z_i^{\mathrm{ss,edit}}$ by swapping the source and target conditions. This yields a restored latent $Z_i^{\mathrm{ss,restore}}$ and restoration displacement D_i^{restore}:

$$Z_i^{\mathrm{ss,restore}} = \mathrm{FE}(Z_i^{\mathrm{ss,edit}}; C_i^{\mathrm{tar}}, C_i^{\mathrm{src}}), \qquad D_i^{\mathrm{restore}} = Z_i^{\mathrm{ss,restore}} - Z_i^{\mathrm{ss,edit}}. \tag{4}$$

Intended edits follow the source–target image differences, and swapping the conditions provides a corresponding reverse editing signal. In contrast, unintended changes do not match the target image, so swapping the conditions does not explicitly specify their reversal. At a spatial location p in the sparse-structure latent grid, we favor condition i when its editing and restoration displacements are both substantial and approximately opposite. Accordingly, the spatial-influence score is

$$s_i(p) = \min\left(\left\|D_i^{\text{edit}}(p)\right\|_2, \left\|D_i^{\text{restore}}(p)\right\|_2\right) \times \left(1 - \frac{\left\|D_i^{\text{edit}}(p) + D_i^{\text{restore}}(p)\right\|_2}{\left\|D_i^{\text{edit}}(p)\right\|_2 + \left\|D_i^{\text{restore}}(p)\right\|_2 + \epsilon}\right), \tag{5}$$

where $\|\cdot\|_2$ is the channel-wise ℓ_2 norm and $\epsilon > 0$ ensures numerical stability. We then normalize across conditions to obtain the spatial weights:

$$w_i^{\text{tgsw}}(p) = \frac{\exp(s_i(p)/\tau^{\text{tgsw}})}{\sum_{j=1}^{M} \exp(s_j(p)/\tau^{\text{tgsw}})}, \tag{6}$$

where $\tau^{\text{tgsw}} > 0$ controls the sharpness of the spatial-weight distribution.

Disagreement-Guided Reweighting. For removal or replacement edits involving geometry removal, TGSW's restoration step requires reconstructing the removed structure from image conditions, which do not fully constrain its 3D shape and placement. The reconstructed structure may therefore deviate from the original, reducing the relative weight assigned to the view requesting removal. During joint editing, other views that observe the region but retain its original geometry in their target images can counteract the removal. Inspired by VS3D (Liu et al., 2026), we use source–target velocity disagreement to reweight view contributions. DGR compares source- and target-conditioned velocities at the same latent state. This avoids reconstructing the removed structure. At time t, the disagreement at location p is

$$\delta_{t,i}^{\text{ss}}(p) = \left\|V^{\text{ss}}\left(Z_t^{\text{ss,tar}}, t, C_i^{\text{tar}}\right)(p) - V^{\text{ss}}\left(Z_t^{\text{ss,tar}}, t, C_i^{\text{src}}\right)(p)\right\|_2. \tag{7}$$

Here, $Z_t^{\text{ss,tar}}$ is the target-branch state constructed from the shared editing latent. For a visible region, a view requesting removal may produce a larger source–target disagreement than one whose target retains the original geometry. In regions unseen by a view, however, a large disagreement need not reflect the requested edit. To combine their complementary strengths, we reweight the trajectory-based spatial weights using disagreement scores and normalize across conditions:

$$w_{t,i}^{\text{dgr}}(p) = \frac{w_i^{\text{tgsw}}(p)\exp\left(\delta_{t,i}^{\text{ss}}(p)/\tau^{\text{dgr}}\right)}{\sum_{j=1}^{M} w_j^{\text{tgsw}}(p)\exp\left(\delta_{t,j}^{\text{ss}}(p)/\tau^{\text{dgr}}\right)}. \tag{8}$$

Here, $\tau^{\text{dgr}} > 0$ controls the sharpness of disagreement-based reweighting. The per-condition FlowEdit velocity difference is $\Delta V_{t,i}^{\text{ss}}(p) = V^{\text{ss}}(Z_t^{\text{ss,tar}}, t, C_i^{\text{tar}})(p) - V^{\text{ss}}(Z_t^{\text{ss,src}}, t, C_i^{\text{src}})(p)$. We fuse these differences as

$$\Delta \bar{V}_t^{\text{ss}}(p) = \sum_{i=1}^{M} w_{t,i}^{\text{dgr}}(p)\Delta V_{t,i}^{\text{ss}}(p). \tag{9}$$

The resulting velocity difference is used in the FlowEdit update. We observe that disagreement magnitudes decrease during later sampling steps, which may weaken their effect on reweighting at a fixed temperature. We compute these weights during the first K^{dgr} joint editing steps, use each step's weights for its update, and reuse their average over these steps at each latent location thereafter.

Disagreement–Visibility Adaptive Fusion. Since sparse-structure editing can change active voxel locations, SLat features need to be regenerated on the edited structure. A straightforward way to combine the target-conditioned SLat velocity predictions from multiple views is to average them with equal weights at each active voxel. However, equal-weight averaging can weaken an edit when other views cannot observe the region or their target images retain its original content.

Similar to DGR, DVAF uses source–target velocity disagreement to estimate local edit relevance during SLat generation. At time t, we evaluate source- and target-conditioned velocities at the same SLat state Z_t^{slat}, defining the disagreement at active location x as

$$\delta_{t,i}^{\text{slat}}(x) = \left\|V^{\text{slat}}\left(Z_t^{\text{slat}}, t, C_i^{\text{tar}}\right)(x) - V^{\text{slat}}\left(Z_t^{\text{slat}}, t, C_i^{\text{src}}\right)(x)\right\|_2. \tag{10}$$

However, in regions unseen by a given view, this disagreement can be large without necessarily reflecting the requested edit. Direct weighting can thus favor a view that does not observe the region. To reduce this ambiguity, we estimate geometric visibility $v_i(x) \in \{0, 1\}$ by projecting the edited voxels into each camera view and applying a z-buffer visibility test, where $v_i(x) = 1$ indicates that voxel x is visible from view i. At visible voxels, we retain the measured disagreement, while at occluded voxels we estimate it so that each value equals the average of its active neighbors. Within each connected component containing visible voxels, the resulting disagreement estimates $\hat{\delta}^{\mathrm{slat}}_{t,i}(x)$ and disagreement weights $w^{\mathrm{dis}}_{t,i}(x)$ are defined as

$$\hat{\delta}^{\mathrm{slat}}_{t,i}(x) = \begin{cases} \delta^{\mathrm{slat}}_{t,i}(x), & v_i(x) = 1, \\ \dfrac{1}{|\mathcal{N}(x)|} \displaystyle\sum_{y \in \mathcal{N}(x)} \hat{\delta}^{\mathrm{slat}}_{t,i}(y), & v_i(x) = 0, \end{cases} \qquad w^{\mathrm{dis}}_{t,i}(x) = \frac{\exp(\hat{\delta}^{\mathrm{slat}}_{t,i}(x)/\tau^{\mathrm{dvaf}})}{\sum_{j=1}^{M} \exp(\hat{\delta}^{\mathrm{slat}}_{t,j}(x)/\tau^{\mathrm{dvaf}})}, \tag{11}$$

where $\mathcal{N}(x)$ contains the active six-neighbors of x, and $|\mathcal{N}(x)| = \sum_{y \in \mathcal{N}(x)} 1$. The temperature $\tau^{\mathrm{dvaf}} > 0$ controls the sharpness of the disagreement weights. For connected components containing no voxels visible from view i, we set $\hat{\delta}^{\mathrm{slat}}_{t,i}(x) = 0$.

When these disagreement estimates are similar across views, disagreement-based weights approach uniform averaging. We therefore define visibility weights to favor views that observe the voxel:

$$w^{\mathrm{vis}}_i(x) = \begin{cases} \dfrac{v_i(x)}{\sum_{j=1}^{M} v_j(x)}, & \sum_{j=1}^{M} v_j(x) > 0, \\ 1/M, & \text{otherwise.} \end{cases} \tag{12}$$

The normalized entropy $H_t(x) = -\big[\sum_{i=1}^{M} w^{\mathrm{dis}}_{t,i}(x) \log w^{\mathrm{dis}}_{t,i}(x)\big] / \log M$ of the disagreement-based weights balances these distributions:

$$w^{\mathrm{dvaf}}_{t,i}(x) = \big(1 - H_t(x)\big) w^{\mathrm{dis}}_{t,i}(x) + H_t(x) w^{\mathrm{vis}}_i(x). \tag{13}$$

Higher entropy favors visibility, while lower entropy favors disagreement. As in DGR, we compute the disagreement weights $w^{\mathrm{dis}}_{t,i}(x)$ during the first K^{dvaf} SLat sampling steps. For subsequent steps, we use their average at each active voxel to compute the entropy and fused DVAF weights. Writing $V^{\mathrm{slat}}_{t,i}(x) = V^{\mathrm{slat}}\big(Z^{\mathrm{slat}}_t, t, C^{\mathrm{tar}}_i\big)(x)$, we fuse the target-conditioned velocities as $\bar{V}^{\mathrm{slat}}_t(x) = \sum_{i=1}^{M} w^{\mathrm{dvaf}}_{t,i}(x) V^{\mathrm{slat}}_{t,i}(x)$.

4 Experiments

Additional benchmark, evaluation, and implementation details, along with further experiments and analyses, are provided in the appendix.

4.1 Experimental Setups

Datasets. We introduce ComplexEdit3D-Bench, a benchmark for evaluating view-aware complex 3D editing. It contains 100 editing samples spanning three types of editing operations: addition, removal, and replacement. The source 3D assets are selected from Google Scanned Objects (GSO) (Downs et al., 2022) and Objaverse-XL (Deitke et al., 2023). Each source asset is rendered from eight viewpoints, with an elevation of $15°$ and azimuth intervals of $45°$. Given the complex editing instruction and source renderings for each sample, GPT-5.6 Sol (OpenAI, 2026a) decomposes the instruction into atomic editing instructions and assigns each to an informative viewpoint. GPT-Image-2 (OpenAI, 2026b) then edits the source rendering from each selected viewpoint according to its assigned instructions, producing the corresponding target image. Each sample comprises a source 3D asset, a complex editing instruction, the decomposed atomic instructions, source–target image pairs, and manually annotated 3D editing masks.

Metrics. We evaluate source preservation using Chamfer distance (CD) (Fan et al., 2017) for geometry and PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) for rendered images, with the source asset as reference. Condition alignment is assessed using CLIP-T (Radford et al., 2021) for text alignment and Local DINO-I (Oquab et al., 2024) for similarity to the target images. Local DINO-I is computed only within the masked editing regions to focus on the requested changes.

Table 1: Quantitative comparison on ComplexEdit3D-Bench. We compare multi-round and single-round editing on source preservation and condition alignment.

Method	Source Preservation				Condition Alignment
	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	CLIP-T ↑
Multi-round Editing					
FlowEdit (Kulikov et al., 2025)	0.013	22.08	0.921	0.074	0.245
VoxHammer (Li et al., 2026)	0.012	24.66	0.951	0.052	0.246
VineDresser3D (Chi et al., 2026)	**0.010**	23.99	0.936	0.058	0.246
Nano3D (Ye et al., 2026)	**0.010**	24.92	0.951	0.048	0.245
PartFlow (Weng et al., 2026)	0.013	23.83	0.941	0.056	0.244
Single-round Editing					
ConcordFlow	**0.010**	**26.51**	**0.954**	**0.045**	**0.255**

Baselines. We adapt FlowEdit (Kulikov et al., 2025) to TRELLIS and apply it sequentially for multi-round editing. Because unintended changes can accumulate across rounds, our comparison also includes four TRELLIS-based methods with dedicated source-preservation mechanisms: VoxHammer (Li et al., 2026) and VineDresser3D (Chi et al., 2026), which use spatial masks to preserve unedited regions; Nano3D (Ye et al., 2026), which employs Voxel/SLat-Merge to reduce unintended changes; and PartFlow (Weng et al., 2026), which learns source-conditioned control branches for 3D editing. For mask-based methods, we provide manually annotated 3D editing masks. For single-round methods, all target images are generated in advance from the original source asset and shared across methods. Multi-round methods share the same target image in the first round, while subsequent targets are generated by editing renderings of each method's preceding output as instructed. We report Local DINO-I only for comparisons with shared target images, including our ablations.

4.2 Main Results

Quantitative Comparison. Tab. 1 reports quantitative results on ComplexEdit3D-Bench. ConcordFlow achieves better overall source preservation and higher condition alignment than the compared multi-round methods, indicating that it better fulfills the requested edits while retaining unrelated source content. Multi-round FlowEdit exhibits the poorest source preservation across all four metrics, which may result from unintended changes accumulating over successive editing rounds. The four existing multi-round methods improve source preservation over FlowEdit through dedicated preservation mechanisms, but remain less effective than ConcordFlow in this complex editing setting.

Qualitative Comparison. Fig. 3 compares the visual results on ComplexEdit3D-Bench. The compared methods may fulfill only part of a complex instruction or introduce unintended changes to the source asset. In contrast, ConcordFlow better balances the fulfillment of the requested edits with the preservation of unrelated source content.

User Study. We conduct a user study involving 20 participants and 25 cases to complement the quantitative evaluation. For each case, participants are shown the source 3D asset, the complex editing instruction, and anonymized results from VoxHammer (Li et al., 2026), VineDresser3D (Chi et al., 2026), Nano3D (Ye et al., 2026), PartFlow (Weng et al., 2026), and ConcordFlow. Participants independently select the best result for instruction alignment, visual quality, and source preservation. Tab. 2 reports the percentage of votes received by each method for each criterion; ConcordFlow receives the highest share of votes across all three criteria.

Efficiency. Tab. 3 compares runtime for a single case with three view-specific conditions on a single NVIDIA A800 GPU, with and without TRELLIS's additional 150-view rendering cost. Reported runtimes exclude MLLM-based instruction decomposition, view assignment, and target-image editing. Because multi-round editing uses the previous round's output as the next round's source, the 150-view rendering is repeated in each round, whereas ConcordFlow performs it only once. Even without this additional rendering cost, ConcordFlow is faster than the four multi-round methods in this setting; including it further widens the runtime gap.

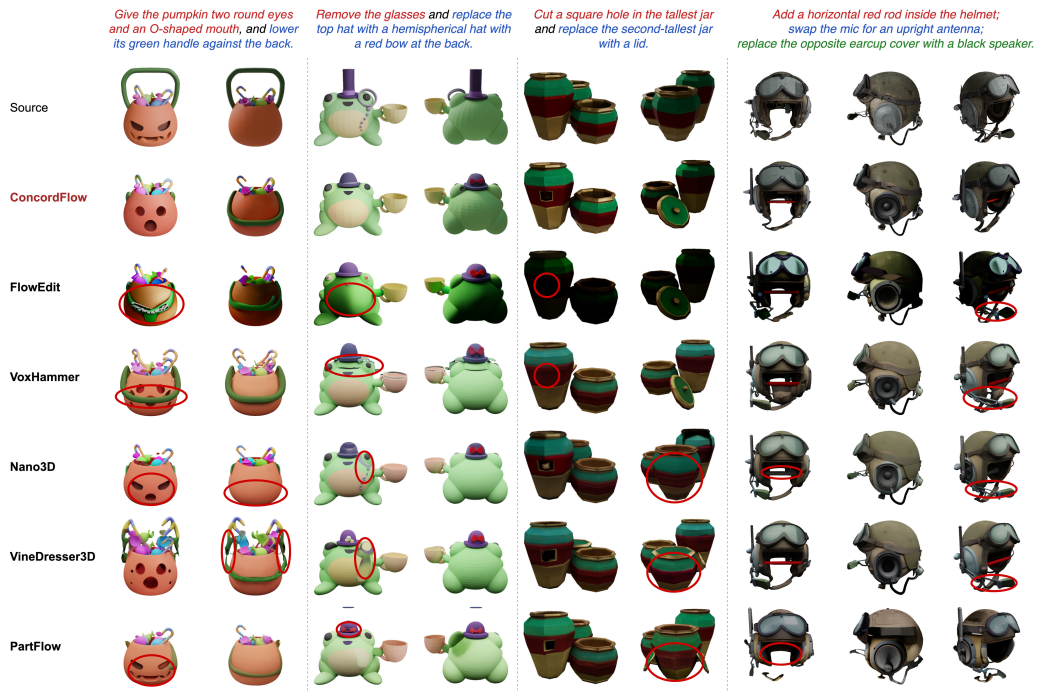


Figure 3: Qualitative comparison on ComplexEdit3D-Bench. ConcordFlow better balances the fulfillment of complex editing instructions with the preservation of unedited geometry and appearance.

Table 2: User study. Percentage of votes received by each method for instruction alignment, visual quality, and source preservation.

Method	Instruction Alignment ↑	Visual Quality ↑	Source Preservation ↑
VoxHammer	8.4%	9.6%	9.0%
VineDresser3D	3.0%	1.2%	1.8%
Nano3D	6.0%	8.4%	7.8%
PartFlow	9.0%	7.8%	8.4%
ConcordFlow	**73.6%**	**73.0%**	**73.0%**

Table 3: Efficiency comparison. Runtime in seconds for a single case with three view-specific conditions.

Method	150-view Rendering	
	w/o ↓	w/ ↓
VoxHammer	387	872
VineDresser3D	379	864
Nano3D	137	622
PartFlow	177	662
ConcordFlow	**128**	**289**

4.3 Ablation Studies

Effects of Different Components. Tab. 4 reports the ablation results of the three components, while Fig. 4 illustrates their effects through progressive addition. Disabling TGSW or DVAF replaces its weights with uniform weights; disabling DGR skips reweighting, leaving the TGSW weights unchanged. The "All" column reports Local DINO-I averaged over cases across the full benchmark.

Removing TGSW, DGR, or DVAF reduces overall Local DINO-I and CLIP-T, supporting the contribution of each component to instruction fulfillment. The full method achieves the highest overall alignment scores, with modest trade-offs in source preservation. Fig. 4 illustrates the contributions of the three components. With TGSW, the hands are moved behind the back and a new flat tail is generated, although remnants of the original arms and tail remain. Adding DGR removes these remnants, enabling more complete geometric edits. Finally, DVAF improves appearance consistency across views, producing a consistently yellow helmet and more complete red boxing gloves.

Complementarity of TGSW and DGR. We categorize view-specific edits according to whether the intended editing region is visible from other conditioning views and whether the requested changes involve voxel removal. Tab. 4 reports Local DINO-I for three categories: VR denotes edits involving voxel removal in regions visible from other conditioning views; VO denotes other edits in

Table 4: Ablation study of ConcordFlow. L. DINO-I denotes Local DINO-I.

Sparse Structure		SLat	Source Preservation				Condition Alignment				
TGSW	DGR	DVAF	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	L. DINO-I ↑				CLIP-T ↑
							All	VR	VO	NV	
✓	–	✓	0.010	26.63	0.954	0.042	0.738	0.699	0.755	0.762	0.253
–	✓	✓	0.010	26.68	0.954	0.044	0.750	0.759	0.751	0.745	0.252
✓	✓	–	0.010	26.55	0.954	0.044	0.749	0.751	0.747	0.751	0.249
✓	✓	✓	0.010	26.51	0.954	0.045	0.767	0.768	0.772	0.765	0.255

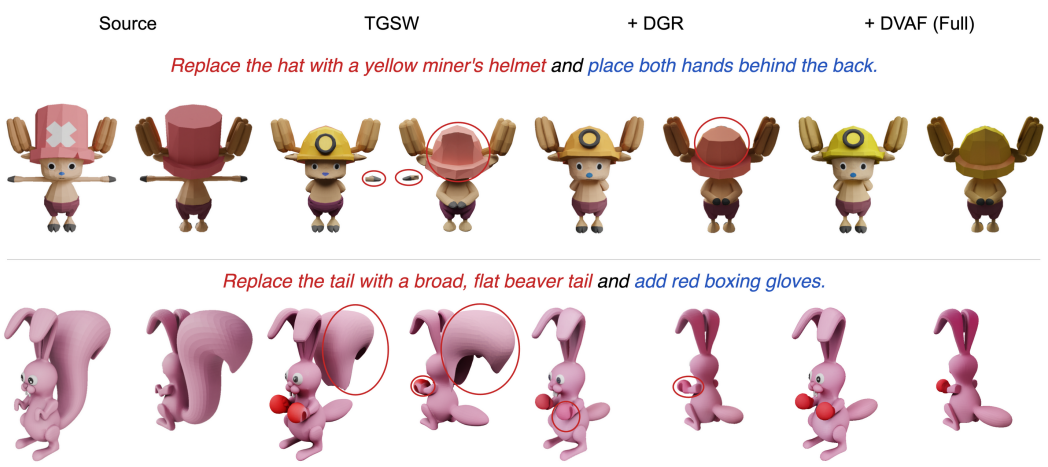


Figure 4: Qualitative ablation of the proposed components. Starting with TGSW, we progressively add DGR and DVAF to obtain the full method. Red circles mark artifacts and incomplete edits.

such regions; and NV denotes edits in regions not visible from other conditioning views. VR, VO, and NV are averaged over editing views within each category.

In VR, DGR outperforms TGSW in Local DINO-I. Errors in reconstructing the removed structure during restoration may weaken TGSW's weight discrimination, leaving opposing updates from other visible views insufficiently suppressed. When other conditioning views also observe the edited region, same-state velocity disagreement may provide greater contrast between conditions requesting removal and those retaining the original geometry. Fig. 4 illustrates this difference: original tail and arm remnants remain with TGSW but disappear after adding DGR.

In NV, DGR performs worse than TGSW, possibly because disagreement from non-observing views can yield excessive weights unrelated to the requested edit. Performance is similar in VO, while combining both components improves all three categories.

5 Conclusion and Limitations

We presented ConcordFlow, a mask-free and training-free framework for view-aware complex 3D editing in a single round. Our framework combines trajectory-guided spatial weighting (TGSW) and disagreement-guided reweighting (DGR) for joint sparse-structure editing with disagreement–visibility adaptive fusion (DVAF) for SLat generation. We also introduced ComplexEdit3D-Bench to evaluate view-aware complex 3D editing. Experiments demonstrate that ConcordFlow achieves better overall source preservation and condition alignment than the compared multi-round baselines, enabling multiple requested changes to be realized within one shared 3D asset.

Our framework constructs all image conditions from the same source asset before joint editing. It therefore does not support editing sequences in which a later image condition can only be constructed after an earlier edit has been completed. Extending our framework to support such dependencies is an important direction for future work.

AI USE STATEMENT

During the preparation of this work, the authors used GPT-5.6 Sol for language editing and clarity improvement. For benchmark construction, GPT-5.6 Sol was used to decompose complex editing instructions and select corresponding viewpoints, while GPT-Image-2 was used to generate target images. The authors reviewed and edited the AI-assisted content as needed and take full responsibility for the content of this paper.

ETHICS STATEMENT

ConcordFlow could be misused to create misleading content or infringe intellectual property rights; responsible use should respect applicable asset and model licenses. Participation in our user study was voluntary, and participants provided informed consent. Responses were analyzed and reported in aggregate.

REPRODUCIBILITY STATEMENT

We describe our method, evaluation protocols, and implementation details in the main paper and supplementary material. We will release the code and the evaluation dataset ComplexEdit3D-Bench.

REFERENCES

Roi Bar-On, Dana Cohen-Bar, and Daniel Cohen-Or. Editp23: 3d editing via propagation of image prompts to multi-view, 2025. URL https://arxiv.org/abs/2506.20652.

Amir Barda, Matheus Gadelha, Vladimir G. Kim, Noam Aigerman, Amit H. Bermano, and Thibault Groueix. Instant3dit: Multiview inpainting for fast editing of 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16273–16282, June 2025.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18392–18402, June 2023.

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

Minghao Chen, Junyu Xie, Iro Laina, and Andrea Vedaldi. Shap-editor: Instruction-guided latent 3d editing in seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26456–26466, June 2024b.

Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22246–22256, October 2023.

Yankuan Chi, Xiang Li, Zixuan Huang, and James Matthew Rehg. Vinedresser3d: Towards agentic text-guided 3d editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12673–12683, June 2026.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 35799–35813. Curran Associates, Inc., 2023. doi: 10.52202/075280-1554. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/70364304877b5e767de4e9a2a511be0c-Paper-Datasets_and_Benchmarks.pdf.

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 *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2553–2560, 2022. doi: 10.1109/ICRA46639.2022.9811809.

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 (CVPR)*, July 2017.

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 (ICCV)*, pp. 7154–7165, October 2025.

Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19740–19750, October 2023.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=_CDixzkzeyb.

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 B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 50678–50702, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/dcad3425f5c8c36b5b3885c091bf1257-Paper-Conference.pdf.

Junchao Huang, Xinting Hu, Shaoshuai Shi, Zhuotao Tian, and Li Jiang. Edit360: 2d image edits to 3d assets from any angle. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16618–16628, October 2025.

Mingzhen Huang, Jialing Cai, Shan Jia, Vishnu Suresh Lokhande, and Siwei Lyu. Paralleledits: Efficient multi-aspect text-driven image editing with attention grouping. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 22569–22595. Curran Associates, Inc., 2024. doi: 10.52202/079017-0710. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2847043899e1171183ceadf86bdbb280-Paper-Conference.pdf.

Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, Qingxiang Lin, Zeqiang Lai, Xianghui Yang, Huiwen Shi, Zibo Zhao, Bowen Zhang, Hongyu Yan, Lifu Wang, Sicong Liu, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Dongyuan Guo, Junlin Yu, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Shida Wei, Chao Zhang, Yonghao Tan, Yifu Sun, Lin Niu, Shirui Huang, Bojian Zheng, Shu Liu, Shilin Chen, Xiang Yuan, Xiaofeng Yang, Kai Liu, Jianchen Zhu, Peng Chen, Tian Liu, Di Wang, Yuhong Liu, Linus, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material, 2025. URL https://arxiv.org/abs/2506.15442.

Yilei Jiang, Zhen Wang, Yanghao Wang, Jun Yu, Yueting Zhuang, Jun Xiao, and Long Chen. Flowdc: Flow-based decoupling-decay for complex image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25757–25766, June 2026.

Jeongsol Kim, Yeobin Hong, Jonghyun Park, and Jong Chul YE. Flowalign: Trajectory-regularized, inversion-free flow-based image editing. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 121814–121846, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/c613667b7f11fbf478293266a156aabe-Paper-Conference.pdf.

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 (ICCV)*, pp. 19721–19730, October 2025.

Lin Li, Zehuan Huang, Haoran Feng, Gengxiong Zhuang, Rui Chen, Chunchao Guo, and Lu Sheng. Voxhammer: Training-free precise and coherent 3d editing in native 3d space. In *2026 International Conference on 3D Vision (3DV)*, pp. 1281–1292, 2026. doi: 10.1109/3DV69130.2026.00126.

Peng Li, Suizhi Ma, Jialiang Chen, Yuan Liu, Congyi Zhang, Wei Xue, Wenhan Luo, Alla Sheffer, Wenping Wang, and Yike Guo. Cmd: Controllable multiview diffusion for 3d editing and progressive generation. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, SIGGRAPH Conference Papers '25, New York, NY, USA, 2025a. Association for Computing Machinery. ISBN 9798400715402. doi: 10.1145/3721238.3730722. URL https://doi.org/10.1145/3721238.3730722.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, and Yan-Pei Cao. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–18, 2025b. doi: 10.1109/TPAMI.2025.3633512.

Siwoo Lim, Sunjae Yoon, Gwanhyeong Koo, Hyeonseo Yun, and Chang D. Yoo. Tango: Training-free 3d editing via tangent-space guidance and optimization. In Paolo Favaro, Zuzana Kukelova, Atsuto Maki, Anna Rohrbach, Konrad Schindler, and Federico Tombari (eds.), *Computer Vision – ECCV 2026*, pp. 369–385, Cham, 2026. Springer Nature Switzerland. ISBN 978-3-032-37235-2.

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 (CVPR)*, pp. 300–309, June 2023.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Hao Liu, Yuxuan Lin, Jingfeng Guo, Ruihang Chu, Junjie Wang, Ruotong Li, and Yujiu Yang. Velocity-space 3d asset editing, 2026. URL https://arxiv.org/abs/2605.07385.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, and Wenping Wang. Wonder3d: Single image to 3d using cross-domain diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9970–9980, June 2024.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=aBsCjcPu_tE.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6038–6047, June 2023.

Rui Nie, Chuang Wang, Haitao Zhou, Jiahe Song, Buyu Li, Sheng Wang, and Qian Yu. Editflow3d: Automated local editing of 3d assets with trajectory preservation, 2026. URL https://arxiv.org/abs/2608.03179.

OpenAI. GPT-5.6 Sol model. OpenAI API documentation, 2026a. URL https://developers.openai.com/api/docs/models/gpt-5.6-sol. Accessed September 6, 2026.

OpenAI. GPT-Image-2 model. OpenAI API documentation, 2026b. URL https://developers.openai.com/api/docs/models/gpt-image-2. Accessed September 6, 2026.

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.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FjNys5c7VyY.

Zhangyang Qi, Yunhan Yang, Mengchen Zhang, Long Xing, Xiaoyang Wu, Tong Wu, Dahua Lin, Xihui Liu, Jiaqi Wang, and Hengshuang Zhao. Tailor3d: Customized 3d assets editing and generation with dual-side images, 2024. URL https://arxiv.org/abs/2407.06191.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4209–4219, June 2024.

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

Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 39838–39859, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/adbe936993aa7cf41e45054d8b72f183-Paper-Conference.pdf.

Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 33879–33896, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/905202e21386913d8eac637c2b50f590-Paper-Conference.pdf.

Jiangshan Wang, Junfu Pu, Zhongang Qi, Jiayi Guo, Yue Ma, Nisha Huang, Yuxin Chen, Xiu Li, and Ying Shan. Taming rectified flow for inversion and editing. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan LI, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8406–8441. Curran Associates, Inc., 2023. doi: 10.52202/075280-0368. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1a87980b9853e84dfb295855b425c262-Paper-Conference.pdf.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

Jiawei Weng, Saining Zhang, Zhenxin Diao, Peishuo Li, Henghaofan Zhang, Junhao Chen, and Hao Zhao. Feedforward 3d editing learns from semantic-part transformation, 2026. URL https://arxiv.org/abs/2605.27351.

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 (CVPR)*, pp. 21469–21480, June 2025.

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

Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models, 2024. URL https://arxiv.org/abs/2404.07191.

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 C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 46900–46922, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/4e3157021c5f833bb2204081f1dda573-Paper-Conference.pdf.

Youtan Yin, Yanning Zhou, Jiacheng Wei, Xiaofeng Yang, Jun Zhang, Jiayang Bai, Jingwen Ye, Weidong Zhang, and Guosheng Lin. Editverse3d: High-quality 3d object editing with region-aware learning. In Paolo Favaro, Zuzana Kukelova, Atsuto Maki, Anna Rohrbach, Konrad Schindler, and Federico Tombari (eds.), *Computer Vision – ECCV 2026*, pp. 494–514, Cham, 2026. Springer Nature Switzerland. ISBN 978-3-032-36839-3.

Sung-Hoon Yoon, Minghan Li, Gaspard Beaudouin, Congcong Wen, Muhammad Rafay Azhar, and Mengyu Wang. Splitflow: Flow decomposition for inversion-free text-to-image editing. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 153207–153225. Curran Associates, Inc., 2025. doi: 10.52202/085713-5124. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/e124f1547f7ac87e33d348b827d4291b-Paper-Conference.pdf.

Biao Zhang, Jiapeng Tang, Matthias Nießner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Trans. Graph.*, 42(4), July 2023. ISSN 0730-0301. doi: 10.1145/3592442. URL https://doi.org/10.1145/3592442.

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 Trans. Graph.*, 43(4), July 2024. ISSN 0730-0301. doi: 10.1145/3658146. URL https://doi.org/10.1145/3658146.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068.

Yang Zheng, Mengqi Huang, Nan Chen, and Zhendong Mao. Pro3d-editor: A progressive-views perspective for consistent and precise 3d editing. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 129917–129941. Curran Associates, Inc., 2025. doi: 10.52202/085713-4326. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/bc6a1f968f8b1dae3e880f3f723d7d46-Paper-Conference.pdf.

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 (CVPR)*, pp. 14387–14397, June 2026.

Zijun Zhou, Yingying Deng, Xiangyu He, Weiming Dong, and Fan Tang. Multi-turn consistent image editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15792–15801, October 2025.

A Additional Benchmark Details

ComplexEdit3D-Bench contains 100 editing samples, including 20 with source assets from GSO (Downs et al., 2022) and 80 with source assets from Objaverse-XL (Deitke et al., 2023). Each complex editing instruction contains two to four editing requests. The MLLM decomposes it into atomic editing instructions and groups those that can be clearly specified from the same viewpoint. For the selected samples, two or three views suffice to clearly specify all requested changes. Of these, 80 samples use two view-specific conditions and 20 use three.

B Evaluation Protocols

For image-based evaluation, we render the edited assets from eight viewpoints, with an elevation of 15° and azimuth intervals of 45°.

CD. We sample 20,480 points from each source and edited mesh in the shared canonical coordinate system, discard points inside the union of the 3D editing masks, and apply voxel downsampling with a voxel size of 0.01. We then compute the mean bidirectional, non-squared Euclidean Chamfer distance (Fan et al., 2017).

PSNR, SSIM, and LPIPS. We evaluate source preservation using PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) on source and edited renderings from eight views. For each view, we compute PSNR from the mean squared error outside the projected editing mask and average the spatial SSIM and LPIPS maps over the same region. We then average each metric across views.

Local DINO-I. We use DINOv2 (Oquab et al., 2024) to evaluate alignment between the edited asset and the target images. However, each target image reflects only the edits assigned to its view, whereas the final asset incorporates all requested edits. Whole-image comparison may therefore penalize valid edits specified by other conditions. To address this, we compute Local DINO-I within each condition's projected editing mask. Specifically, we encode complete target images and corresponding edited renderings using DINOv2 and compute cosine similarities between spatially corresponding patch features. We average these similarities over patches overlapping the mask and then across condition views.

CLIP-T. We compute CLIP (Radford et al., 2021) cosine similarity between a global description of the desired edited asset and each of the eight edited renderings, then average across views.

C Implementation Details

We implement our method on TRELLIS (Xiang et al., 2025) and conduct experiments on a single NVIDIA A800 80GB GPU. Following Nano3D (Ye et al., 2026), we use a 25-step sampling schedule with $n_{\max} = 15$, $n_{\min} = 0$, and $n_{\text{avg}} = 5$ for joint sparse-structure editing. Here, n_{avg} is the number of noise samples used to average the FlowEdit velocity difference at each step. The source and target CFG scales are set to 1.5 and 5.5, respectively. For TGSW's independent editing and restoration trajectories, we use the same schedule but set $n_{\text{avg}} = 1$ to reduce the cost of spatial-weight estimation. We share noise samples across views during joint editing and reuse the same noise sequence across TGSW editing and restoration runs. We set $\tau^{\text{tgsw}} = 0.5$ and $\tau^{\text{dgr}} = \tau^{\text{dvaf}} = 1$. For SLat generation, we use 25 sampling steps and a CFG scale of 5.0. The disagreement-weight estimation window is set to $K^{\text{dgr}} = 3$ steps for DGR and $K^{\text{dvaf}} = 5$ steps for DVAF.

D Additional Experiments

D.1 Experiments on TRELLIS.2

To evaluate applicability beyond TRELLIS, we implement both ConcordFlow and FlowEdit (Kulikov et al., 2025) on TRELLIS.2 (Xiang et al., 2026). We apply FlowEdit sequentially under the view-specific conditions to perform multi-round editing, whereas ConcordFlow jointly integrates the

view-specific conditions in a single round. We report source preservation and condition alignment in Tab. 5. ConcordFlow outperforms multi-round FlowEdit across all five metrics on TRELLIS.2, supporting its applicability beyond the original TRELLIS backbone.

Table 5: Comparison of multi-round FlowEdit and single-round ConcordFlow on TRELLIS.2.

Method	Source Preservation				Condition Alignment
	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	CLIP-T ↑
FlowEdit (Kulikov et al., 2025)	0.018	21.33	0.918	0.090	0.244
ConcordFlow	**0.009**	**26.05**	**0.950**	**0.047**	**0.249**

D.2 Global-Text Baseline

Following recent 3D editing methods (Li et al., 2026; Ye et al., 2026; Zhou et al., 2026; Lim et al., 2026), ConcordFlow builds upon an image-conditioned 3D generator. Alternatively, complex editing instructions can be handled directly using a text-conditioned generator. We therefore construct a global-text baseline using text-conditioned TRELLIS (Xiang et al., 2025), with a source description and a global target description covering all requested edits as the conditions for single-round FlowEdit. Tab. 6 compares the global-text baseline with ConcordFlow. ConcordFlow achieves a higher CLIP-T score, indicating better overall alignment with the editing instructions in this comparison.

Global textual conditions provide less explicit spatial and appearance guidance for preserving source content than view-specific image pairs, which may allow unintended changes beyond the requested edits. Fig. 5 provides a qualitative comparison. In this robot example, the global-text baseline changes the torso to blue rather than the head and modifies the arm structure. ConcordFlow better balances the requested edits with the preservation of unrelated source content.

Table 6: CLIP-T comparison of ConcordFlow and the global-text baseline on TRELLIS.

Method	Global Text	ConcordFlow
CLIP-T ↑	0.242	**0.255**

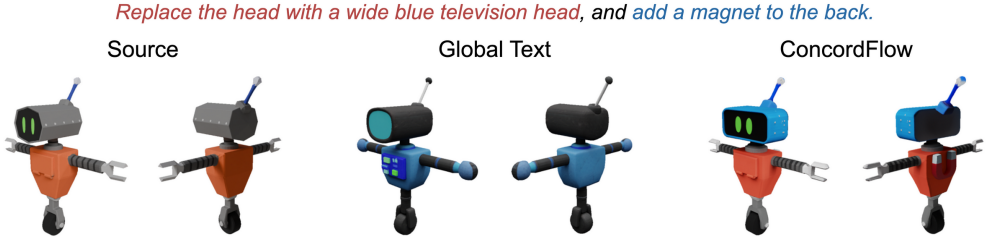


Figure 5: Qualitative comparison with the global-text baseline.

D.3 Additional Component Ablations

Tab. 7 reports additional ablations with no components enabled, each component enabled individually, and all three enabled. Components are disabled as in the main-paper ablations. When all three are disabled, we uniformly average per-condition FlowEdit velocity differences during sparse-structure editing and target-conditioned velocities during SLat generation. Each component alone improves Local DINO-I and CLIP-T over this uniform-fusion configuration, while enabling all three achieves the highest alignment scores.

Disabling all three components achieves better source-preservation scores. Differences in CD, PSNR, and SSIM are relatively small, while enabling TGSW or DGR leads to a noticeable increase in LPIPS. This increase may partly reflect LPIPS features whose receptive fields cross the evaluation-mask boundaries, allowing changes inside the masks to influence scores outside them.

Table 7: Additional component ablations of ConcordFlow on ComplexEdit3D-Bench.

Sparse Structure		SLat	Source Preservation				Condition Alignment	
TGSW	DGR	DVAF	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	L. DINO-I ↑	CLIP-T ↑
–	–	–	0.010	27.01	0.956	0.033	0.667	0.235
✓	–	–	0.010	26.65	0.954	0.041	0.730	0.247
–	✓	–	0.010	26.72	0.954	0.043	0.744	0.247
–	–	✓	0.010	27.00	0.956	0.035	0.681	0.244
✓	✓	✓	0.010	26.51	0.954	0.045	0.767	0.255

D.4 Effect of Restoration in TGSW

We compare Edit-only in Fig. 6, which computes spatial weights from editing displacement alone, with TGSW, which uses both editing and restoration displacements. Fig. 6 shows that independent editing from View 2 removes the rear wall, although its instruction only requests removal of the interior blocks and entrance steps. Edit-only assigns a high contribution to View 2 in this region, and the rear wall is also missing in its joint result. The unintended rear-wall removal is not reflected in the target image, so swapping the image conditions does not explicitly specify its reversal. This may weaken the consistency between editing and restoration displacements, reducing View 2's weight in this region. In contrast, TGSW reduces this contribution, and the displayed joint result retains the rear wall. The weight comparison illustrates how restoration trajectories help distinguish incidental displacement from the intended editing influence.

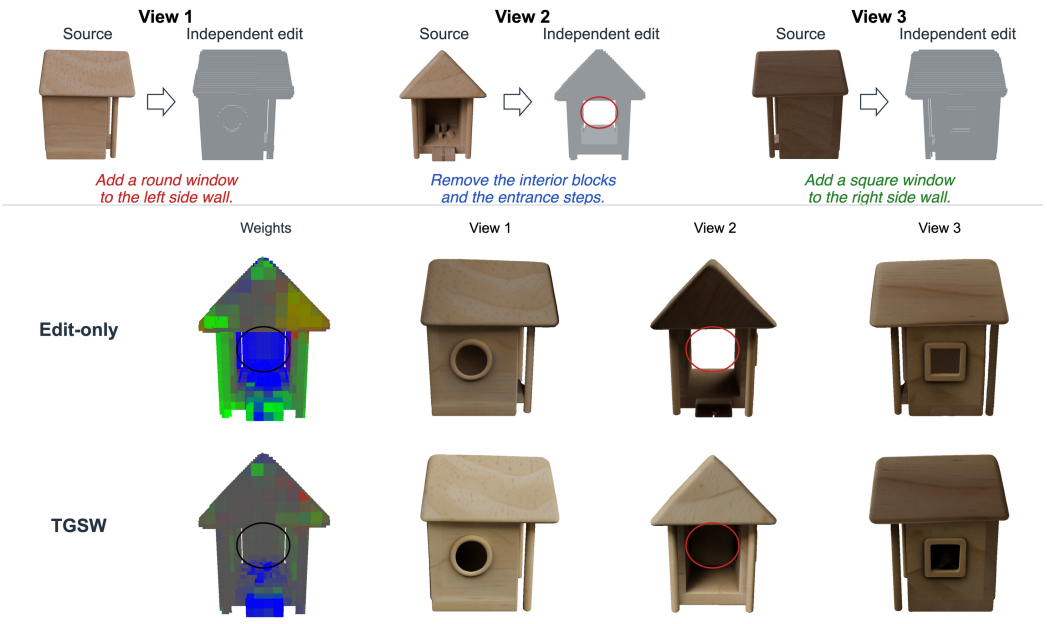


Figure 6: Restoration analysis of TGSW: independent edits (top), spatial weights and joint results (bottom). Red, blue, and green encode contributions from Views 1–3. Circles highlight the rear wall, lost with Edit-only but retained in the TGSW result.

D.5 Analysis of DVAF

Fig. 7 visualizes edited results and spatial condition weights for a two-view girl example and a three-view astronaut example. In the girl example, disagreement-based weighting produces black hair across views, but also changes the color of the broom bristles, which are unrelated to the requested edit. After propagating disagreement from visible to occluded voxels, the hair weights become more spatially consistent and the bristle color is better preserved. Adding visibility fusion further shifts

the weights on the rear-facing broom bristles toward the second condition, while retaining the first condition's dominant contribution to the hair edit. These weight maps help relate local appearance changes to the contributions of different conditions.

In the astronaut example, disagreement-based weighting produces the requested smile and medical kit, but introduces blue bands across the chest. The result with disagreement propagation reduces these bands, while the Full result retains the smile, headset, and medical kit. The three-condition weight maps highlight how contributions vary across the face, backpack, and medical kit.

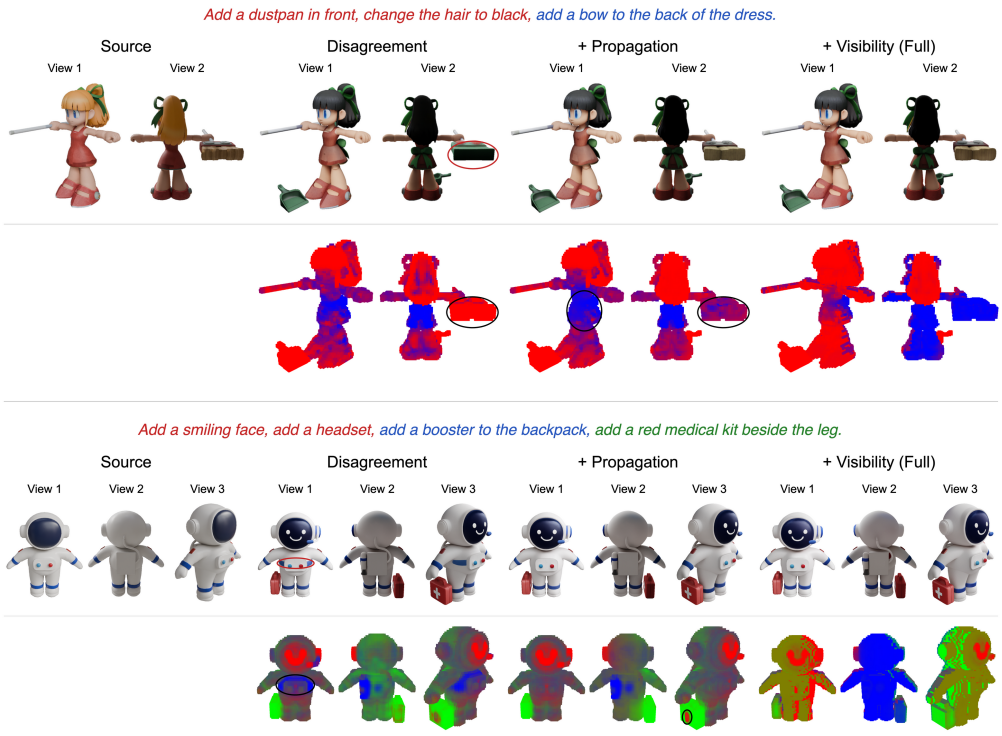


Figure 7: Qualitative analysis of DVAF on two- and three-view examples. Each example pairs edited results with spatial condition weights on the corresponding voxel structure. Red and blue denote Views 1 and 2; green additionally denotes View 3 in the astronaut example. Mixed colors encode relative contributions, with purple and gray indicating equal weights for two and three conditions, respectively.

D.6 Sensitivity to the Disagreement Estimation Window

We evaluate how many initial sampling steps are used to estimate disagreement-based weights in DGR and DVAF. For DGR, we compute and use the current weights during the first K^{dgr} joint editing steps and reuse their average at each latent location for subsequent steps. For DVAF, we compute the disagreement weights during the first K^{dvaf} SLat sampling steps. For subsequent steps, we use their average at each active voxel to compute the entropy and fused DVAF weights. Tab. 8 reports two parameter sweeps: we vary $K^{\mathrm{dgr}} \in \{1, 2, 3, 5, 10, 15\}$ with $K^{\mathrm{dvaf}} = 5$, and vary $K^{\mathrm{dvaf}} \in \{1, 2, 3, 5, 10, 15, 20, 25\}$ with $K^{\mathrm{dgr}} = 3$. Each setting is evaluated on the same 100 samples.

Tab. 8 shows that performance varies modestly with the estimation window. Across the tested settings, Local DINO-I ranges from 0.757 to 0.767 when varying K^{dgr}, and from 0.760 to 0.767 when varying K^{dvaf}. Source-preservation metrics remain nearly unchanged, while CLIP-T varies between 0.253 and 0.255. These results suggest that performance is not highly sensitive to the estimation window within the tested range.

Fig. 8 reports mean velocity disagreement over sampling steps, separately for the full spatial domain and regions inside and outside the evaluation masks. For DGR, the mean inside the evaluation masks peaks at step 3 and then decreases; for DVAF, all three means decrease throughout sampling. This decrease in magnitude may weaken the effect of disagreement-based reweighting at a fixed temperature, motivating the reuse of weights estimated during early steps.

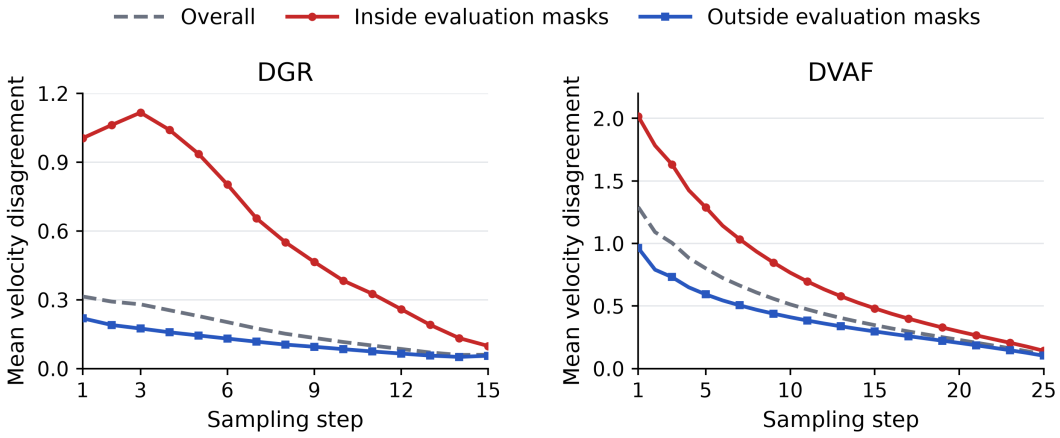


Figure 8: Mean velocity disagreement over sampling steps for DGR and DVAF, measured over the full spatial domain and inside and outside the evaluation masks.

Table 8: Sensitivity to the number of initial steps used for disagreement-based weight estimation. Each block varies one module's estimation window while fixing the other.

K^{dgr}	K^{dvaf}	CD ↓	PSNR ↑	SSIM ↑	LPIPS ↓	L. DINO-I ↑	CLIP-T ↑
Varying DGR with $K^{\text{dvaf}} = 5$							
1	5	0.010	26.50	0.953	0.045	0.757	0.254
2	5	0.010	26.53	0.953	0.045	0.762	0.254
3	**5**	0.010	26.51	0.954	0.045	0.767	0.255
5	5	0.010	26.50	0.953	0.045	0.764	0.253
10	5	0.010	26.56	0.954	0.045	0.761	0.254
15	5	0.010	26.56	0.954	0.045	0.761	0.254
Varying DVAF with $K^{\text{dgr}} = 3$							
3	1	0.010	26.49	0.953	0.045	0.760	0.254
3	2	0.010	26.50	0.954	0.045	0.763	0.255
3	3	0.010	26.50	0.954	0.045	0.764	0.255
3	**5**	0.010	26.51	0.954	0.045	0.767	0.255
3	10	0.010	26.51	0.954	0.045	0.764	0.253
3	15	0.010	26.51	0.954	0.045	0.765	0.253
3	20	0.010	26.50	0.954	0.045	0.763	0.253
3	25	0.010	26.50	0.954	0.045	0.762	0.253

E Additional Qualitative Results

Fig. 9 and Fig. 10 present additional qualitative results of ConcordFlow with two or three view-specific conditions, covering addition, removal, and replacement. Each example shows the source and edited assets from corresponding viewpoints, with different colors distinguishing the editing requirements.

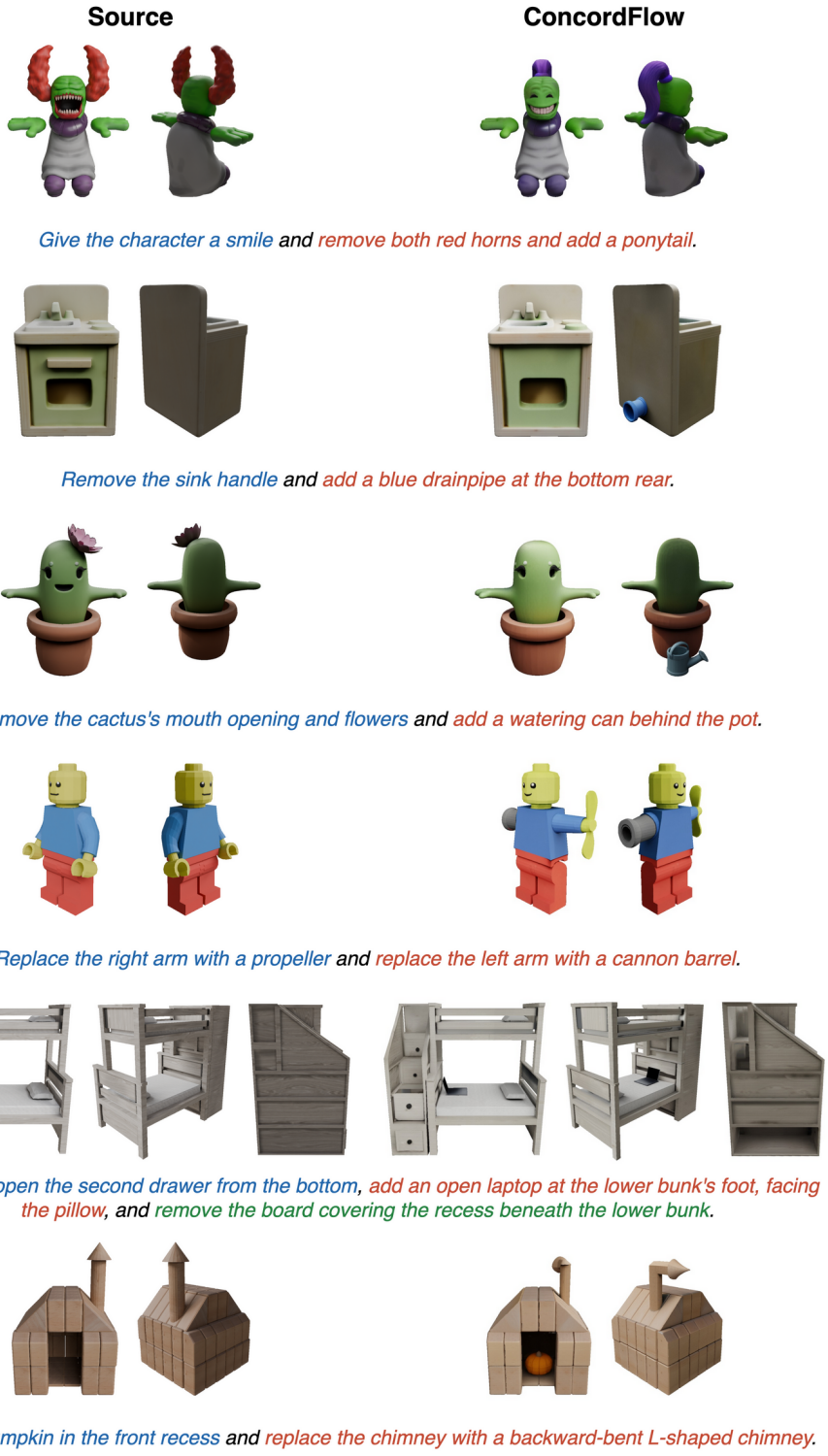


Figure 9: Additional qualitative results of ConcordFlow.



Figure 10: Additional qualitative results of ConcordFlow (continued).