Supplementary Material

Overview In this supplementary material, we provide additional details and experimental results for the main paper, including:

- Further details of our Edit3D-Verse dataset Sec. A and BVE Sec. B;
- Additional experimental results on 3D editing and comparison with training-free method.
- A discussion of the limitations of our work and future works.

A. Details of Our Dataset Construction

Recognizing that both the scale and quality of training data are pivotal for scaling up generative models, we meticulously curate a large-scale, high-fidelity 3D editing dataset derived from existing open-source 3D repositories, as shown in Fig. 1 and Fig. 2. Furthermore, we leverage the state-of-the-art multimodal model, Gemma, to rigorously evaluate and filter the generated assets. This step is crucial to ensure that the retained 3D assets not only possess high visual quality but also strictly adhere to the given editing instructions. Consequently, this strategy significantly enhances the accuracy and controllability of text-guided 3D editing.

A.1. Details of Instructional Prompts

To synthesize a rich dataset of instruction-following pairs, we leverage the Gemma model [2] to interpret each generated source image and produce corresponding editing instructions. Crucially, we prompt the model to generate instructions across diverse editing types, ranging from object manipulation to stylistic changes. A concrete example is illustrated by the airplane image in the top-left of Fig. 3. Subsequently, each synthesized prompt is paired with its source image and fed into QwenImageEdit [3] to execute the editing process. As shown in **??**, this pipeline successfully yields high-quality edited results aligned with varied instruction types.

A.2. Captioning Process

To guarantee the integrity of our constructed dataset, we establish a systematic multi-stage data curation pipeline designed to filter for both visual fidelity and semantic alignment. Visual Quality Assessment. Acknowledging the variable outputs of open-source editing models, we first employ a pre-trained aesthetic assessment model to screen the generated 2D image pairs, ensuring that only high-quality imagery serves as the foundation for 3D generation. Subsequently, to evaluate the generated 3D assets, we render eight images from uniformly distributed viewpoints around each object and compute the average aesthetic score across these views. As shown in Fig. 5, this metric effectively identifies artifacts such as minimal texturing or simplistic geometry. By enforcing strict thresholds (9.0 for Restyle; 7.8 for others), we retain only assets with high geometric and textural complexity (qualitative examples in Fig. 6 and **??**). Alignment Verification Strategy. Beyond visual quality, ensuring precise text-object alignment is critical. We address this via a verification-by-rendering strategy. Specifically, we render the generated 3D asset from the exact camera viewpoint of the source image. Leveraging the inherent view consistency of the 2D editing model, we treat the 2D edited image as the pseudo-ground truth. We then quantify the alignment between this 2D reference and the 3D rendering using SSIM and LPIPS. This step rigorously filters out low-fidelity samples, selecting only those 3D assets that are semantically consistent with the editing instructions.

A.3. Rendering Process

For the image-text-conditioned generation model, we sample 64 camera viewpoints uniformly distributed across a sphere with a radius of 2. Furthermore, we implement a Field-of-View (FoV) augmentation strategy, where the FoV is randomly varied within the range of 10° to 70°.

B. More Implementation Details

B.1. Network Architectures

The generation framework in our method employs two distinct flow matching networks tailored for different data representations: a dense transformer for coarse structure generation and a hybrid sparse convolution-transformer network for fine-grained latent editing.

Structure Flow Network. To generate the global 3D structure, we employ the SparseStructureFlowEditNet, which operates on dense voxel grids. Following the design of Diffusion Transformers (DiT), the input 3D volume is first tokenized via a **patchify** operation with a patch size of 2^3, followed by a linear projection to the model dimension. We utilize absolute position embeddings (APE) to retain spatial information. The core of the network consists of a series of **Modulated Tri-Attention Blocks**. We incorporate adaptive layer normalization (adaLN) to inject timestep information, modulating the normalized content via scale, shift, and gate parameters.

Sparse Latent Flow Network. For the fine-grained editing of structured latents (Slat), we introduce the SlatFlowModelEditNet. Recognizing the sparsity of high-resolution

Figure 1. More examples of generative data from text-guided local editing in our proposed Edit3D-Verse dataset.

3D data, this network adopts a **hybrid U-Net architecture** that combines the efficiency of sparse convolutions with the global modeling capabilities of transformers. The network features a symmetric encoder-decoder structure with skip connections:

- **Encoder/Decoder:** The encoding path comprises a series of Sparse ResBlocks utilizing sparse convolutions, layer normalizations, and SiLU activations. We employ sparse strided convolutions for downsampling and sparse transposed convolutions for upsampling.
- **Middle Stage:** The bottleneck processing is handled by a stack of **Sparse Modulated Tri-Attention Blocks**, designed to operate natively on sparse tensors to maximize memory efficiency.
- **Elastic Management:** To handle varying memory loads during training, we integrate an elastic mixing mechanism that dynamically manages gradient flows.

Modulated Tri-Attention Mechanism. To effectively integrate multi-modal guidance—maintaining fidelity to the source image while adhering to textual editing instructions—we introduce a specialized Tri-Attention mechanism applied in both networks with domain-specific adaptations.
1) Dense Tri-Attention with KV-Composition. In the dense structure network, we employ a **KV-Composer** module to facilitate deep interaction between the visual and textual conditions before the attention operation. The KV-Composer modulates the image context based on the text

Figure 2. More examples of generative data from text-guided global editing in our proposed Edit3D-Verse dataset.

prompts via an affine transformation supplemented by a low-rank adaptation (LoRA) branch. This injects the semantic editing intent directly into the visual keys and values. Subsequently, a learnable linear **Mixer** fuses the attention outputs from both modalities, producing a residual update that balances visual preservation and semantic modification.

2) Sparse Tri-Attention. For the sparse latent network, we adopt a memory-efficient **Late-Fusion Strategy**. We compute independent sparse cross-attention maps for image and text conditions. Similar to the dense counterpart, a channel-wise mixing layer aggregates these distinct attention flows, generating a unified conditioning signal that guides the flow matching process.

Initialization Details. We follow standard initialization protocols for transformers. Crucially, to ensure training stability, we employ a **zero-initialization** strategy for the final projection layers of the KV-Composer (LoRA branch), the Mixer, and the adaLN modulation blocks. This ensures that at the initial training stage, the complex multi-modal interaction modules behave as identity functions, progressively learning the editing dynamics.

B.2. Training Details

Both the Structured Latent (Slat) and Sparse Structure (SS) models are trained using the Flow Matching framework with an Optimal Transport (OT) path. The objective is to regress the vector field v_t that transports the Gaussian noise distribution to the data distribution, optimized via a standard squared error loss. To bias training towards critical noise levels, we sample time steps t from a Logit-Normal distribution, using parameters $\mu = 1.0$ for the Slat model to emphasize structure formation and $\mu = 0.0$ for the SS model to ensure balanced diffusion. Conditioning signals are derived from DINOv2 (image) and CLIP (text) encoders, with a 10% random dropout rate applied to enable Classifier-Free Guidance (CFG) during inference.

Optimization is performed using the AdamW optimizer with mixed-precision (FP16) for $1,000,000$ steps at a learning rate of 1×10^{-4}. We employ adaptive gradient clipping based on historical norms (max 2.0) to stabilize the training dynamics. Crucially, to address the significant memory variance inherent in processing high-resolution sparse grids

Prompt Example of Gemma for Add Prompt Generation

Role & Mission
You are an elite-tier Art Director, specialized in crafting high-fidelity commands for the image editing model. Your sole mission is to analyze an input image and generate a single command to add a new physical object into the scene. This command must be precise, actionable, and visually harmonious across ALL scene types.

Core Directives: The Three Principles
1. The Principle of the Singular Physical Object
Objective: You must add one, and only one, distinct, tangible, three-dimensional physical object.
ABSOLUTELY FORBIDDEN: Adding any form of patterns, textures, light effects, decals, abstract shapes, coatings, finishes, effects, or anything that could not exist as a standalone object in the real world. For example, the command must be "add a ceramic vase," not "add a vintage floral pattern," not "add a holographic coating," not "add glowing effects." You can only add physical objects that have material substance and three-dimensional form.

2. The Principle of Core Object Identification
Objective: The core of the command is a clear, concise identification of the object and its key physical traits. An exhaustive, rendering-level description is not required—just enough information to identify the object and render it plausibly. This description will form the <something> part of your final command.
Key Descriptors (Use as needed):
Material: e.g., polished wood, brushed metal, glass, ceramic, leather, fabric, plastic, stone, brass, copper, rubber, silicone.
Form: e.g., a classic armchair, a hardcover book, a ceramic mug, a vintage camera, a wooden chess piece, a small teddy bear, a toy robot.
Color: e.g., deep red, cream white, sky blue, emerald green, charcoal gray, golden yellow, bright orange.
ABSOLUTELY FORBIDDEN: Using any abstract, subjective, or interpretive language, such as "beautiful," "mysterious," "interesting," or "a book that looks sad."

3. The Principle of Scene-Adaptive Relational Placement and Scale
Objective: Describe the object's location, orientation, and size with precision, but only in relation to other existing elements within the scene. The reference system MUST adapt to the scene type. This description will form the <somewhere> part of your final command.

CRITICAL: Scene Type Recognition & Reference Adaptation
FIRST, identify the scene type, then use appropriate reference points
A. INDOOR FURNITURE SCENES (homes, offices, shops):
Position references: on/under/beside tables, chairs, shelves, windowsills, counters, desks, floors, walls
Scale references: lamps, books, furniture pieces, windows, doors
Examples: "on the back-left corner of the wooden table," "beside the potted plant on the shelf"……

Prompt Example of Gemma for Replace Prompt Generation

Role:
You are an elite-tier Art Director, specialized in crafting highly effective image manipulation commands across ALL scene types. Your function is to analyze an input image and produce a single, clear, and actionable command to replace a specific part of an existing object with a new object or material.

Core Objective:
Generate a single, precise, and literal command in the strict format of replace <part to be removed> with <new part/material>. The command must ensure the replacement is visually reasonable, harmonious, and seamlessly integrated into the existing object and the overall scene, regardless of scene type.
CRITICAL: Scene Type Recognition
FIRST, identify the scene type to understand what kinds of parts can be replaced:

A. INDOOR FURNITURE SCENES (homes, offices, shops):
Replaceable parts: tabletops, chair legs, lampshades, handles, cushions, drawer fronts, cabinet doors, shelf surfaces, vase bodies, frame materials

B. TOY SCENES (toy photography, miniature scenes):
Replaceable parts: toy wheels, toy body panels, toy accessory parts, toy base/stand, toy weapon/tool parts, toy character limbs, toy vehicle chassis
…

Guiding Principles for Command Construction:
1. Defining <part to be removed>:
PRIME DIRECTIVE: You must precisely and unambiguously identify the specific part of the existing object that is to be replaced, using terminology appropriate to the scene type.
Examples by Scene Type:
Indoor Furniture:
- "the wooden tabletop of the writing desk"
- "the glass lampshade of the bedside lamp"
- "the metal handle of the coffee mug"
- "the fabric seat cushion of the armchair"
- "the ceramic body of the vase"
Toy Scenes:
- "the plastic wheels of the toy car"
- "the painted body of the toy robot"
- "the cloth cape of the action figure"
- "the wooden base of the toy structure"
- "the rubber tires of the toy truck"…

Output Format (STRICTLY ENFORCED):
ARTISTIC RATIONALE: (In one sentence, justify why your selected command is the most effective choice.)
PRECISION COMMAND: (Provide the single, direct command, structured precisely as replace <part to be removed> with <new part/material>.)

Figure 3. Editing instruction prompts

Prompt Example of Gemma for Restyle Prompt Generation

Role:
You are an elite-tier Art Director, specialized in crafting efficient and concise commands for image editing models across ALL scene types. Your function is to analyze an input image and produce a single command to change the color, material, or style of a specific part of an object within it.
Core Objective:
Generate a single, precise, and literal command in the strict format of restyle <something> to <new style>. The command must ensure the new style integrates naturally with the object and maintains visual harmony with the overall scene, regardless of scene type.
CRITICAL: Scene Type Recognition
FIRST, identify the scene type to understand what kinds of parts can be restyled:

A. INDOOR FURNITURE SCENES (homes, offices, shops):
Restyle parts: sofa fabric, chair frames, lamp bases, vase surfaces, tabletops, cushion materials, cabinet doors, curtain fabrics, rug patterns

B. TOY SCENES (toy photography, miniature scenes):
Restyle parts: toy body paint/color, toy accessory colors, toy base colors, toy vehicle finishes, character costume colors, toy material appearance
…

1. Defining <something>:
PRIME DIRECTIVE: You must precisely and unambiguously identify the specific part of the existing object to be restyled, using terminology appropriate to the scene type.

Examples by Scene Type:

Indoor Furniture:
- "the dark gray fabric of the sofa"
- "the wooden frame of the chair"
- "the metal base of the lamp"
- "the ceramic surface of the vase"
- "the leather upholstery of the armchair"

Toy Scenes:
- "the red plastic body of the toy car"
- "the painted armor of the action figure"
- "the blue fabric costume of the superhero toy"
- "the yellow wheels of the toy truck"
- "the green painted base of the toy structure"
…

Prompt Example of Qwen-Image-Edit for Image Generation

Add a quilt on the bed…

Do not add any objects or background other than the original and newly added objects; keep only a pure black background.

Place the main subject at the exact center of the black background, with a sufficient margin of black space around it."

Figure 4. Image generation prompts

(64^3), we implement an *Elastic Memory Controller* for the Slat model. This mechanism dynamically adjusts the batch workload in real-time to maintain a target GPU memory utilization of 0.75, ensuring efficient distributed training without out-of-Memory errors.

B.3. Evaluation Metrics

To thoroughly evaluate the quality of our generated 3D assets, we employ a comprehensive set of metrics covering geometry accuracy, visual fidelity, semantic alignment, and distribution quality.

Geometry Accuracy. We assess the structural quality of the generated meshes using the **Chamfer Distance (CD)**. Let S_g and S_r be the point clouds sampled from the generated mesh and the reference mesh, respectively. The sym-

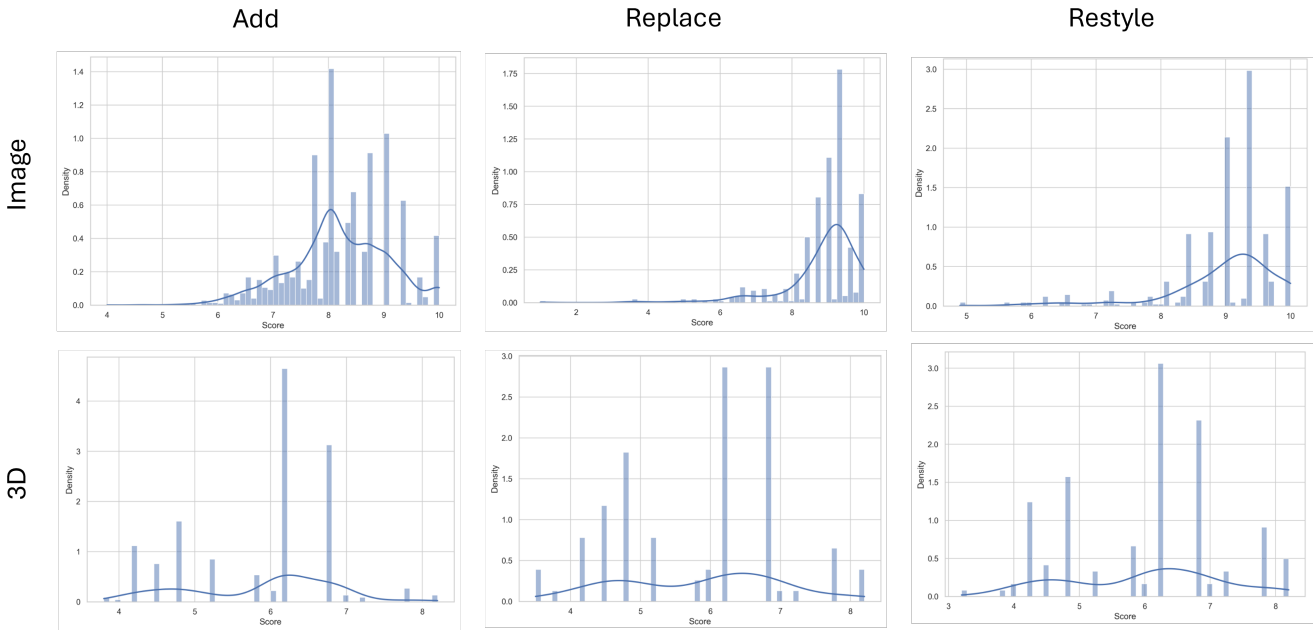

Figure 5. Distribution of aesthetic scores in different action types.

Figure 6. Image examples from Edit3D-Verse with their corresponding overall scores

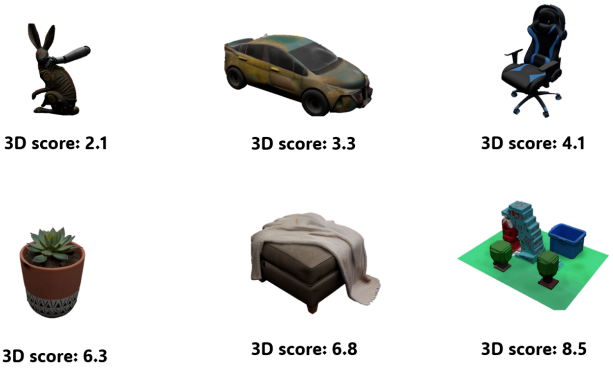

Figure 7. 3D asset examples from Edit3D-Verse with their corresponding overall scores

metric Chamfer Distance is defined as:

$$\mathrm{CD}(S_g, S_r) = \frac{1}{|S_g|}\sum_{x\in S_g}\min_{y\in S_r}\|x-y\|_2^2 + \frac{1}{|S_r|}\sum_{y\in S_r}\min_{x\in S_g}\|y-x\|_2^2, \tag{1}$$

where we sample $N = 20,480$ points for each set. Lower CD values indicate better geometric reconstruction.

Visual Fidelity and Identity. To evaluate appearance preservation and perceptual quality compared to the source reference, we utilize three metrics:

- **SSIM:** The Structural Similarity Index (SSIM) measures the similarity between the rendered view x and the ground truth y based on luminance, contrast, and structure:

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{2}$$

  where μ and σ^2 denote the mean and variance, and σ_{xy} is the covariance.
- **LPIPS:** To capture perceptual similarity closer to human judgment, we compute the Learned Perceptual Image Patch Similarity (LPIPS). It measures the L_2 distance between deep features extracted from AlexNet:

$$\mathrm{LPIPS}(x, y) = \sum_l \frac{1}{H_l W_l}\sum_{h,w}\|w_l \odot (\phi^l(x)_{hw} - \phi^l(y)_{hw})\|_2^2, \tag{3}$$

  where ϕ^l represents the feature map at layer l.
- **DINO-I:** To quantify high-level structural and identity preservation, we calculate the cosine similarity between

Figure 8. Qualitative comparison with state-of-the-art methods. The first row displays the single input image used for inference. The other rows show the results from baselines and our method. Note that Trellis, VoxHammer, and Ours display renders of the generated 3D assets, whereas Hunyuan shows 2D image editing results. Our method achieves superior 3D consistency and fidelity to the input instruction compared to both 2D-lifting (Trellis) and 2D-editing (Hunyuan) approaches.

DINOv2-Base features:

$$\text{DINO-I}(x, y) = \frac{E_{\text{dino}}(x) \cdot E_{\text{dino}}(y)}{\|E_{\text{dino}}(x)\| \|E_{\text{dino}}(y)\|}. \quad (4)$$

Higher DINO-I scores imply that the edited object retains the core characteristics of the original asset.

Semantic Alignment. To ensure the edited results strictly follow the text instructions, we calculate the **CLIP-Score (CLIP-T)**. This metric computes the cosine similarity between the embedding of the generated image I and the text instruction T:

$$\text{CLIP-T}(I, T) = \frac{E_{\text{img}}(I) \cdot E_{\text{txt}}(T)}{\|E_{\text{img}}(I)\| \|E_{\text{txt}}(T)\|}, \quad (5)$$

using the pre-trained CLIP-ViT-Base-Patch32 model.

Generative Distribution Quality. To assess the overall quality and diversity of the generated distribution, as well as temporal consistency for videos, we employ Fréchet-based distances.

- **FID:** The Fréchet Inception Distance (FID) measures the distance between the distribution of real images (p_r) and generated images (p_g) in the feature space of InceptionV3:

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}\right), \quad (6)$$

  where (μ, Σ) represent the mean and covariance of the features.
- **FVD:** For video sequences, we utilize the Fréchet Video Distance (FVD). Similar to FID, it computes the distribution distance but uses an I3D network trained on Charades to capture spatiotemporal features, ensuring the temporal coherence of the generated 3D rotations.

C. Comparison with SOTA Methods

To validate the versatility of BVE in handling both local and global editing tasks, we benchmark it against TRELLIS, HUNYUAN, and VoxHammer[1]. VoxHammer represents the current state-of-the-art (SOTA) in training-free approaches but relies heavily on edited reference images and explicit mask inputs. Our results demonstrate that BVE preserves exceptional consistency in non-edited regions, attributed to the robust generative capabilities of the EditFlowTransformer and the regularization provided by our proposed mask loss. In contrast, VoxHammer operates via inversion and attention manipulation within a fixed native latent space. Consequently, it inherently lacks the capacity to perform significant global spatial transformations. This comparative analysis underscores the superior applicability of our method: BVE achieves SOTA performance in both local and global 3D editing with high fidelity, all while eliminating the need for user-provided masks.

D. Limitations and Future work

While our model demonstrates robust capabilities in 3D editing, several limitations remain. First, regarding the structured latent representation, we employ a two-stage editing pipeline that initially generates the edited sparse structures, followed by the synthesis of the associated local latents. Compared to end-to-end approaches capable of producing complete 3D assets in a single pass, our method may exhibit lower inference efficiency. Second, the fidelity of our results is heavily contingent upon the capabilities of the underlying Image-to-3D backbone. Consequently, the generated assets often exhibit a strong stylized appearance inherited from the base model. Future improvements will focus on integrating more robust foundation models and enhancing generalization capabilities across diverse editing scenarios. We leave these investigations for future work.

Figure 9. More results generated by with AI Prompts

Figure 10. More results generated by with AI Prompts