DISCOMESH: EFFICIENT AND FLEXIBLE ARTIST-MESH GENERATION VIA SPARSE COMPUTATION POWERED DISCRETE DIFFUSION

Anonymous authors
Paper under double-blind review



Figure 1: **Artist-mesh generation and part-level editing with DisCoMesh.** DisCoMesh simultaneously generates semantic parts, producing diverse artist meshes with intricate geometry, clean topology, and inter-part coherence. The eagle on the left illustrates part-level editing: the wing highlighted in pink is regenerated while the geometry and topology of the remaining mesh are preserved.

ABSTRACT

Meshes used in games and interactive 3D applications require not only accurate geometry, but also clean topology, fine details, and editable semantic components. However, existing generative methods struggle to produce such assets within a practical computational budget. To address this challenge, we introduce DisCoMesh, a framework based on the view that a multi-part artist mesh is better represented as a set of parts coupled by sparse global relationships rather than as a single global sequence or an ordered sequence of parts. Each semantic part is canonicalized and quantized independently using the full coordinate vocabulary, preserving local geometric detail across parts of different scales. A shared discrete diffusion model then denoises all parts simultaneously, removing the need for an artificial part-generation order. To coordinate simultaneous part generation, we introduce a global interaction module that integrates part placement and global geometric context with sparse token-level communication across parts, preserving coherent connections and spatial relationships. This part-wise formulation enables flexible control over parts, supporting selective part regeneration while preserving the rest of the mesh. Experiments demonstrate that DisCoMesh outperforms existing methods in generation quality, maintains competitive inference speed, and supports diverse downstream applications.

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.

1 INTRODUCTION

Meshes are fundamental assets in games, animation, and interactive 3D applications, where artist-created meshes are valued for their fine-grained geometric details, carefully designed topology, and semantically meaningful parts. Automatically generating such meshes remains challenging.

Artist-mesh generation has made substantial progress through autoregressive modeling of mesh token sequences (Siddiqui et al., 2024; Chen et al., 2024b; 2025b; Weng et al., 2025; Hao et al., 2024). However, scaling these methods to increasingly detailed meshes faces two challenges. First, token-by-token autoregressive decoding requires a sequential inference step for every output token, limiting inference efficiency for long mesh sequences. Second, quantizing the entire mesh in a global coordinate frame leaves geometrically small parts with limited effective resolution, making their fine-grained structures difficult to preserve.

Recent methods based on discrete diffusion (Alliegro et al., 2023; Song et al., 2025) and continuous flow (Li et al., 2026; Zhao et al., 2026) alleviate the first challenge through parallel generation. Their global mesh representations, however, do not explicitly account for the distinct topological patterns and geometric complexity of semantic parts. Part-wise formulations aim to address the second challenge through detailed modeling of individual components. MeshMosaic (Xu et al., 2026) formulates part-wise mesh generation autoregressively, whereas PartDiffuser (Yang et al., 2026) adopts a semi-autoregressive diffusion process that denoises tokens in parallel within each semantic part. Both formulations retain autoregressive dependencies across parts, conditioning each part on previously generated parts.

However, we observe that semantic parts vary in topology and geometry, with no clear natural order for their generation. We therefore view a multi-part artist mesh as a collection of locally intricate parts coupled by global structural relationships. Building on this view, we introduce **DisCoMesh**, a part-aware discrete diffusion framework that simultaneously denoises all semantic parts while maintaining global mesh coherence. Each part is denoised in its own canonical coordinate system under the guidance of its corresponding point cloud. This local representation makes full use of the coordinate vocabulary, allowing even geometrically small parts to retain fine-grained structures.

While part-local denoising captures the topology and geometry of individual parts, maintaining coherence across parts requires global geometric context and direct inter-part communication. We therefore introduce the Global Interaction Module (GIM) to coordinate part generation through global geometric conditioning and sparse inter-part attention. For efficient cross-part communication, the sparse attention combines compressed summaries of all other parts with uncompressed tokens from small sets of content-relevant parts and spatial neighbors. This gives each part access to object-wide context and fine-grained information from selected parts, enabling coherent inter-part generation without dense attention over all mesh tokens.

The part-wise formulation naturally supports flexible control by maintaining a separate denoising state for each semantic part. This allows the same model to selectively regenerate target parts while preserving the geometry and topology of the remaining mesh. Building on this capability, we present an agentic part-level editing workflow that enables localized regeneration of selected semantic parts based on text or image guidance. The same part-wise representation also supports downstream applications such as local level-of-detail control and part composition across different source objects.

To summarize, our main contributions are as follows:

- We introduce DisCoMesh, a discrete diffusion framework that generates multi-part artist meshes through simultaneous part-wise denoising. With each semantic part represented in its own canonical coordinate system, the framework preserves fine-grained geometry and part-specific topological patterns.
- We propose the Global Interaction Module (GIM) that coordinates part generation through global geometric conditioning and sparse inter-part communication, maintaining inter-part coherence without dense attention over all mesh tokens.
- We demonstrate DisCoMesh's favorable balance between generation quality and inference efficiency across different face-count ranges, as well as its flexibility through an agentic part-level editing workflow and diverse downstream applications.

2 Related Work

2.1 Artist-Mesh Generation

Artist-mesh generation has advanced with autoregressive models that learn vertex distributions and connectivity directly from artist-created meshes (Siddiqui et al., 2024; Chen et al., 2024b;a). Subsequent advances in tokenization and architecture improve scalability and structural modeling (Chen et al., 2025b; Weng et al., 2025; Hao et al., 2024; Lionar et al., 2025; Zhao et al., 2025; Song et al., 2026; Lin et al., 2026). Other approaches introduce parallelism through decoupled connectivity prediction (Kim et al., 2026), continuous latent flow (Li et al., 2026; Zhao et al., 2026), and discrete mesh denoising (Alliegro et al., 2023; Song et al., 2025; Yang et al., 2026).

2.2 Discrete Diffusion Models

Discrete diffusion models support parallel generation of categorical data through iterative denoising (Austin et al., 2021; Lou et al., 2024). Masked formulations recover masked tokens using bidirectional context (Shi et al., 2024; Sahoo et al., 2024), with blockwise extensions combining parallel denoising within blocks and autoregressive dependencies across blocks (Arriola et al., 2025). For mesh generation, these models have been adapted to hybrid masked-token prediction and refinement (Song et al., 2025) and denoising within semantic parts (Yang et al., 2026).

2.3 Part-Aware 3D Generation

Part-aware approaches model 3D objects as compositions of individual parts (Lin et al., 2025; Tang et al., 2025; Chen et al., 2025a). FullPart (Ding et al., 2026) further improves local geometric resolution. However, these approaches do not directly model artist-designed face connectivity. For artist meshes, MeshMosaic (Xu et al., 2026) assembles local patches autoregressively, while PartDiffuser (Yang et al., 2026) performs parallel denoising within parts but retains sequential generation across parts. DisCoMesh simultaneously denoises independently quantized part sequences, using part-local modeling and sparse inter-part communication to preserve local topology and inter-part coherence.

3 Preliminaries

Discrete Mesh Representation. Given a triangular mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$ with F faces, we follow Meshtron (Hao et al., 2024) to quantize vertex coordinates and order all vertices lexicographically by (y, z, x), where y denotes the vertical axis. We cyclically rotate each face's vertices to place the smallest first, preserving winding, then sort faces lexicographically by their vertex tuples. Let $\mathcal{A}$ be the coordinate-token vocabulary. We serialize each vertex into three coordinate tokens and concatenate them in face order:

$$\hat{\mathbf{v}}_{i,j} = (\hat{v}^y_{i,j}, \hat{v}^z_{i,j}, \hat{v}^x_{i,j}) \in \mathcal{A}^3, \qquad \mathbf{x} = \left[\hat{\mathbf{v}}_{1,1}, \hat{\mathbf{v}}_{1,2}, \hat{\mathbf{v}}_{1,3}, \ldots, \hat{\mathbf{v}}_{F,1}, \hat{\mathbf{v}}_{F,2}, \hat{\mathbf{v}}_{F,3}\right] \in \mathcal{A}^{9F}. \tag{1}$$

Here, i indexes the ordered faces, and $j \in \{1, 2, 3\}$ indexes the vertices within each face.

Discrete Diffusion Modeling. Given a clean mesh sequence $\mathbf{x}$, we construct a corrupted state $\mathbf{x}_t$ by independently retaining each token with probability α_t or replacing it with a sample from a noise distribution ν (Austin et al., 2021). The token-wise corruption distribution is

$$q_\nu(x_{t,n} \mid x_n) = \alpha_t \, \delta_{x_n}(x_{t,n}) + (1 - \alpha_t) \, \nu(x_{t,n}), \qquad \delta_x(y) = \mathbf{1}[y = x]. \tag{2}$$

We consider two noise distributions. *Mask noise* replaces selected tokens with a dedicated `[MASK]` token, explicitly indicating which tokens are missing. *Uniform noise* instead replaces selected tokens with tokens sampled uniformly from the coordinate-token vocabulary $\mathcal{A}$, yielding a mask-free noisy sequence.

Given the corrupted state $\mathbf{x}_t$, the denoising model predicts clean-token distributions conditioned on the time t and target geometry $\mathbf{c}$, and is trained using cross-entropy:

$$\mathcal{L}_\nu = \mathbb{E}_{\mathbf{x}, \mathbf{c}, t, \mathbf{x}_t} \left[\mathrm{CE}(p_\theta(\cdot \mid \mathbf{x}_t, t, \mathbf{c}), \mathbf{x}) \right]. \tag{3}$$

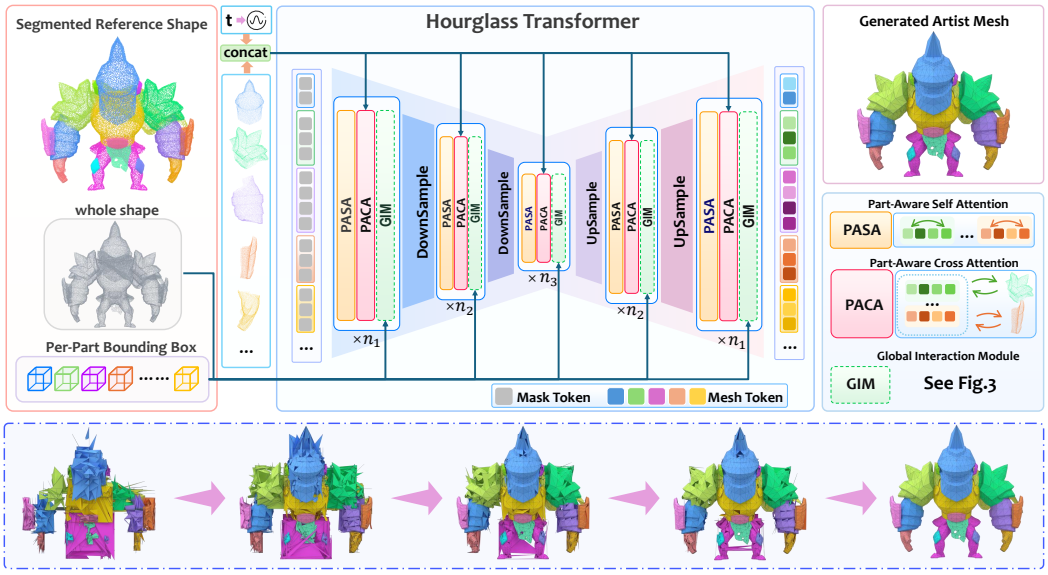


Figure 2: **Overview of DisCoMesh.** A shared Hourglass Transformer simultaneously denoises part sequences. Part-aware self-attention (PASA) captures within-part dependencies, while part-aware cross-attention (PACA) conditions each part on its canonicalized point cloud. The Global Interaction Module (GIM) combines bounding-box and global point-cloud conditioning with sparse inter-part communication. The bottom row shows the simultaneous denoising process across parts.

4 DisCoMesh

DisCoMesh generates artist meshes through simultaneous discrete diffusion over semantic parts. Each part is independently quantized in its canonical coordinate system using 1024 quantization levels per axis, while a global interaction module maintains global coherence. Figure 2 provides an overview of our framework.

4.1 Semantic Decomposition and Part-Wise Representation

Given a reference shape $\mathcal{S}$, we use PartSAM (Zhu et al., 2026) to obtain semantic parts $\{\mathcal{S}_k\}_{k=1}^{N}$. For each part, we extract a point cloud $\mathbf{P}_k$ and its global axis-aligned bounding box $\mathbf{B}_k$. Using $\mathbf{B}_k$, we transform $\mathbf{P}_k$ into a canonicalized point cloud $\widetilde{\mathbf{P}}_k$ and represent the corresponding part mesh in the same local coordinate system. Each part mesh is independently quantized using 1024 quantization levels per axis, preserving the precision of small components despite differences in global scale.

4.2 Model Architecture

We employ a shared hourglass transformer following Meshtron (Hao et al., 2024), with three resolution stages: coordinate, vertex, and face. The network aggregates every three coordinate tokens into a vertex feature and every three vertex features into a face feature. Symmetric upsampling restores coordinate-level resolution.

Semantic parts exhibit distinct local topology and geometry, yet their generation must be coordinated to form a coherent object. We therefore combine part-local modeling with global interaction. Each stage of the hourglass transformer contains multiple blocks equipped with part-aware self-attention (PASA) and part-aware cross-attention (PACA) for part-local modeling. We insert the Global Interaction Module (GIM) after PACA in selected blocks within each stage, interleaving global interaction with part-local processing. GIM incorporates bounding-box and global point-cloud conditioning to capture part layout and overall geometry. It also enables direct communication across part sequences through sparse attention to coordinate generation and promote inter-part coherence.

4.2.1 Part-Local Modeling

We model the topology and target geometry of each semantic part through two complementary attention operations. Part-aware self-attention (PASA) captures dependencies among mesh tokens

within each part by applying self-attention independently to each part sequence:

$$\mathrm{PASA}(\mathbf{x}^{(k)}) = \mathrm{SelfAttn}(\mathbf{x}^{(k)}). \tag{4}$$

This induces a block-diagonal attention pattern over the concatenated sequence.

Part-aware cross-attention (PACA) then conditions each part sequence on features $\mathbf{C}_k$ obtained by concatenating its canonicalized point-cloud features with a timestep embedding:

$$\mathbf{C}_k = \mathrm{Concat}(\mathcal{E}_{\mathrm{pc}}(\widetilde{\mathbf{P}}_k), \mathcal{E}_{\mathrm{time}}(t)), \qquad \mathrm{PACA}(\mathbf{x}^{(k)}, \mathbf{C}_k) = \mathrm{CrossAttn}(\mathbf{x}^{(k)}, \mathbf{C}_k), \tag{5}$$

where $\mathcal{E}_{\mathrm{pc}}$ denotes a pretrained point-cloud encoder (Zhao et al., 2023) and $\mathcal{E}_{\mathrm{time}}$ denotes the timestep embedder.

4.2.2 Global Interaction Module

Part-local modeling captures the local topology and geometric structure of each semantic part, but does not provide whole-shape awareness or fine-grained interaction across parts. To address these limitations, we introduce the Global Interaction Module (GIM), which conditions each part sequence on its global bounding box and the global reference point cloud, supplying the global context absent from the canonicalized part inputs. Beyond this geometric guidance, GIM coordinates part generation through sparse attention, enabling fine-grained communication between part sequences to maintain inter-part coherence without dense attention over all mesh tokens.

Bounding-Box and Global-Geometry Conditioning. The part-local representation describes each part's geometry but omits its position and relative scale within the object. The first component of GIM supplies this spatial information by conditioning each part sequence on its global bounding box. We encode the eight ordered corners $\{\mathbf{r}_{k,j}\}_{j=1}^{8}$ of $\mathbf{B}_k$ into bounding-box tokens $\mathbf{T}_k$:

$$\mathbf{t}_{k,j} = \mathrm{MLP}_{\mathrm{bbox}}(\phi(\mathbf{r}_{k,j})) + \mathbf{e}_j, \qquad \mathbf{T}_k = [\mathbf{t}_{k,1}, \ldots, \mathbf{t}_{k,8}], \tag{6}$$

where ϕ denotes Fourier encoding and $\mathbf{e}_j$ is a learned corner-index embedding. Each part sequence then attends exclusively to its own bounding-box tokens:

$$\mathbf{x}_{\mathrm{bbox}}^{(k)} = \mathrm{CrossAttn}_{\mathrm{bbox}}(\mathbf{x}^{(k)}, \mathbf{T}_k). \tag{7}$$

Bounding boxes specify the spatial extent of individual parts but do not describe the overall geometry of the object. The second component of GIM provides this geometric context through global point-cloud conditioning. The joint sequence $\mathbf{x}_{\mathrm{bbox}}$ then attends to features $\mathbf{C}^{\mathrm{global}}$ from the global reference point cloud $\mathbf{P}^{\mathrm{global}}$:

$$\mathbf{C}^{\mathrm{global}} = \mathcal{E}_{\mathrm{pc}}(\mathbf{P}^{\mathrm{global}}), \qquad \mathbf{x}_{\mathrm{global}} = \mathrm{CrossAttn}_{\mathrm{global}}(\mathbf{x}_{\mathrm{bbox}}, \mathbf{C}^{\mathrm{global}}), \tag{8}$$

where $\mathcal{E}_{\mathrm{pc}}$ is the same point-cloud encoder used in PACA.

Flexible Inter-Part Sparse Attention. Geometric conditioning aligns parts with the reference shape, but direct interaction across parts is also needed to maintain inter-part coherence during denoising. Since dense inter-part attention incurs quadratic cost in total sequence length, we introduce part-aware sparse self-attention (PASSA) as GIM's third component. PASSA combines three complementary routes: compressed global interaction, content-adaptive interaction, and spatial neighbor interaction (Fig. 3). All routes exclude the query part, leaving intra-part dependencies to PASA.

Compressed global interaction. To provide each part with access to the overall generation state at low cost, we pool each source part j along its sequence dimension to obtain compressed tokens $\overline{\mathbf{x}}^{(j)}$. For query part i, we concatenate the compressed tokens of all other parts into a context $\mathbf{C}_i^{\mathrm{cmp}}$:

$$\mathbf{O}_i^{\mathrm{cmp}} = \mathrm{CrossAttn}_{\mathrm{cmp}}(\mathbf{x}^{(i)}, \mathbf{C}_i^{\mathrm{cmp}}). \tag{9}$$

This route provides object-wide context, while the following two routes enable fine-grained communication through uncompressed tokens.

Content-adaptive interaction. To capture fine-grained relationships between relevant parts, the content-adaptive route selects a small set of source parts for interaction through uncompressed tokens. We measure content affinity s_{ij} between the query sequence $\mathbf{x}^{(i)}$ and each source's compressed tokens $\overline{\mathbf{x}}^{(j)}$, and retain the k_r highest-scoring parts:

$$\mathcal{S}_i = \mathrm{TopK}_{j \neq i}(s_{ij}, k_r). \tag{10}$$

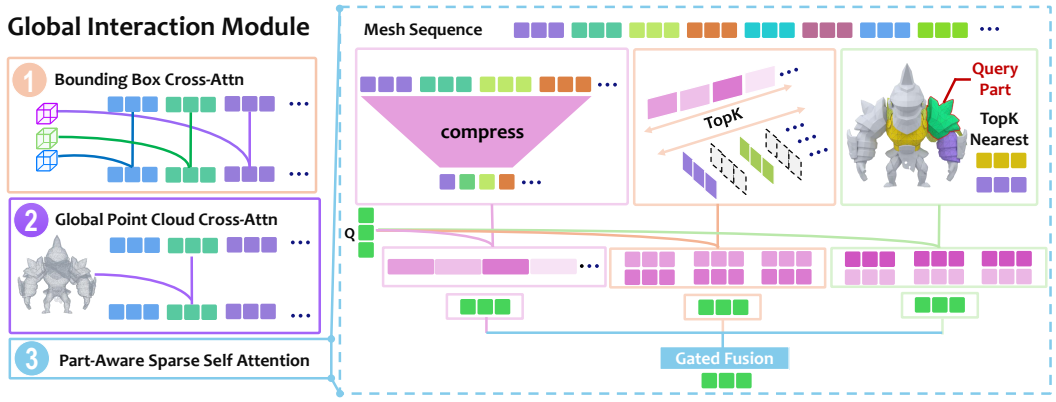


Figure 3: **Global Interaction Module (GIM) and Part-Aware Sparse Self-Attention (PASSA).** Left: GIM combines bounding-box conditioning, global point-cloud conditioning, and PASSA. Right: PASSA integrates compressed global context with uncompressed tokens from content-selected and spatially neighboring parts through learned gated fusion.

The context $\mathbf{C}_i^{\text{content}}$ concatenates the uncompressed tokens of the selected parts in $\mathcal{S}_i$. The query part then attends to this context:

$$\mathbf{O}_i^{\text{content}} = \text{CrossAttn}_{\text{content}}(\mathbf{x}^{(i)}, \mathbf{C}_i^{\text{content}}). \tag{11}$$

This route enables detailed interaction with relevant parts even when they are spatially distant.

Spatial neighbor interaction. Content affinity alone may overlook spatially adjacent parts whose attachment regions require coordination. We therefore select k_n neighboring parts based on bounding-box proximity:

$$\mathcal{N}_i = \text{NeighborTopK}(\mathbf{B}_i, \{\mathbf{B}_j\}_{j \neq i}, k_n). \tag{12}$$

We concatenate the uncompressed tokens of the neighbors in $\mathcal{N}_i$ into a context $\mathbf{C}_i^{\text{neighbor}}$ for cross-attention:

$$\mathbf{O}_i^{\text{neighbor}} = \text{CrossAttn}_{\text{neighbor}}(\mathbf{x}^{(i)}, \mathbf{C}_i^{\text{neighbor}}). \tag{13}$$

This route enables neighboring parts to exchange detailed geometric information around their attachment regions, promoting connectivity and inter-part coherence during generation.

Finally, we fuse the three route outputs using learned per-head gates:

$$\boldsymbol{\alpha}_h = \text{softmax}(\mathbf{a}_h), \quad \mathbf{O}_{i,h} = \alpha_h^{\text{cmp}} \mathbf{O}_{i,h}^{\text{cmp}} + \alpha_h^{\text{content}} \mathbf{O}_{i,h}^{\text{content}} + \alpha_h^{\text{neighbor}} \mathbf{O}_{i,h}^{\text{neighbor}}, \tag{14}$$

where $\mathbf{a}_h \in \mathbb{R}^3$ contains the learnable gate logits for head h.

4.3 Training and Inference

Training Strategy. We adopt a two-stage curriculum learning strategy. To learn rich geometric and topological priors, we first pretrain the model on high-quality single-object meshes without semantic decomposition. We then adapt the model to multi-part generation, introducing bounding-box and global point-cloud conditioning before enabling sparse inter-part attention.

Optimization Objective. We jointly optimize denoising, token-confidence, and intra-part connection objectives (Song et al., 2025). These objectives supervise token prediction and token-confidence estimation for inference-time remasking, and encourage consistent predictions for vertices shared by multiple faces within each part. To encourage inter-part coherence, we introduce an inter-part connection loss $\mathcal{L}_{\text{inter}}$ for vertices shared across parts. The loss measures the Wasserstein distance between the predicted coordinate distributions after they are mapped to a common global frame. The overall objective is:

$$\mathcal{L} = \mathcal{L}_{\text{denoise}} + \mathcal{L}_{\text{conf}} + \mathcal{L}_{\text{intra}} + \lambda_{\text{inter}} \mathcal{L}_{\text{inter}}, \tag{15}$$

where λ_{inter} controls the contribution of inter-part consistency. Detailed formulations of all four objectives are provided in Appendix C.2.

Inference Process. At inference, given a target face budget F_k for part k, we initialize $\mathbf{x}_T^{(k)}$ with $L_k = 9F_k$ mask tokens and concatenate the sequences into $\mathbf{x}_T$, retaining part assignments. Following TSSR's hybrid sampling strategy (Song et al., 2025), we iteratively complete masked tokens, refine coordinate predictions, and remask tokens with low predicted confidence. All parts are denoised simultaneously throughout sampling, with GIM coordinating generation through global conditioning and inter-part communication. After denoising, we map part coordinates back to the global frame using their bounding boxes to obtain the complete mesh.

Table 1: Quantitative comparison on artist meshes and dense meshes. The best results are shown in **bold**, and the second-best results are underlined.

	Artist Meshes					Dense Meshes				
Method	$CD_{\ell_1}\downarrow$	$CD_{\ell_2}(\times10^3)\downarrow$	HD $\downarrow$	$\lvert NC\rvert\uparrow$	EF1 $\uparrow$	$CD_{\ell_1}\downarrow$	$CD_{\ell_2}(\times10^3)\downarrow$	HD $\downarrow$	$\lvert NC\rvert\uparrow$	EF1 $\uparrow$
DeepMesh (Zhao et al., 2025)	0.0107	0.132	0.0562	0.878	0.373	0.0143	0.221	0.0716	0.839	0.109
FastMesh (Kim et al., 2026)	0.0204	0.402	0.0590	0.796	0.131	0.0207	0.424	0.0781	0.760	0.033
MeshMosaic (Xu et al., 2026)	0.0106	0.141	0.0624	0.884	0.429	0.0125	0.170	0.0708	0.855	0.163
MeshRipple (Lin et al., 2026)	0.0127	0.203	0.0673	0.852	0.239	0.0209	0.494	0.1054	0.752	0.034
MeshFlow (Li et al., 2026)	0.0161	0.197	0.0442	0.797	0.082	0.0195	0.266	0.0544	0.754	0.012
TSSR (Song et al., 2025)	0.0090	0.099	0.0513	0.917	0.536	0.0123	0.128	0.0604	0.895	0.111
Ours	**0.0077**	**0.048**	**0.0281**	**0.934**	**0.683**	**0.0100**	**0.079**	**0.0446**	**0.920**	**0.267**

Table 2: Generation time and mesh quality at approximately 8K faces. All timings are measured on a single NVIDIA A800 GPU. The best results are shown in **bold**.

Metric	DeepMesh	FastMesh	MeshMosaic	MeshRipple	MeshFlow	TSSR	**Ours**
$CD_{\ell_1}\downarrow$	0.0114	0.0229	0.0090	0.0157	0.0150	0.0089	**0.0074**
EF1 $\uparrow$	0.336	0.134	0.492	0.159	0.072	0.512	**0.687**
Inf. Time (s) $\downarrow$	479.9	17.6	1213.5	1128.2	**3.6**	894.9	109.0
Faces (K)	7.97	8.13	7.99	7.96	7.74	8.00	8.00

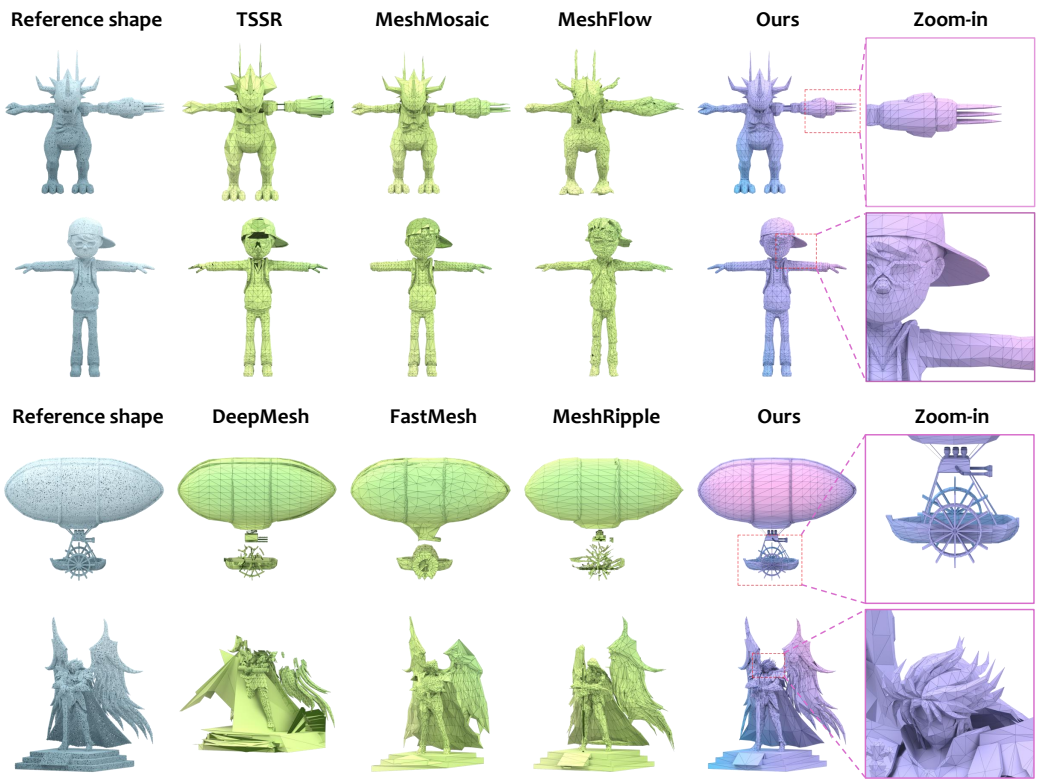


Figure 4: **Qualitative comparisons with baseline methods.** DisCoMesh better preserves intricate geometric structures while maintaining coherent connections between parts.

Table 3: Ablation of bounding-box condition and global point cloud condition in Global Interaction Module. The best results are shown in **bold**.

Variant	$\mathrm{CD}_{\ell_1} \downarrow$	$\mathrm{CD}_{\ell_2}(\times 10^3) \downarrow$	HD $\downarrow$	$\lvert\mathrm{NC}\rvert \uparrow$	EF1 $\uparrow$
w/o BBox Cond.	0.0079	0.054	0.0374	0.910	0.679
w/o Global PC Cond.	0.0079	0.054	0.0373	0.909	0.676
w/o Both Cond.	0.0083	0.072	0.0459	0.905	0.653
Both Cond.	**0.0078**	**0.050**	**0.0359**	**0.911**	**0.682**

Table 4: Ablation of sparse attention and Inter-Part loss. The best results are in **bold**.

Variant	MSVD $\downarrow$
w/o Sparse Attention	0.0038
w/o Inter-Part Loss	0.0037
w/o Both	0.0048
Full Model	**0.0021**

5 Experiments

5.1 Experimental Setup

Datasets. We first pretrain on filtered single-object meshes without semantic decomposition, using G-Objaverse (Zuo et al., 2025; Qiu et al., 2024), HSSD (Khanna et al., 2024), 3D-FUTURE (Fu et al., 2021), and Objaverse-XL (Deitke et al., 2023). Multi-part training then uses PartVerseXL (Ding et al., 2026), PartNeXt (Wang et al., 2025), and G-Objaverse meshes segmented by PartSAM (Zhu et al., 2026). See Appendices A and C.1 for dataset and implementation details.

Baselines and Evaluation Protocol. We compare DisCoMesh with six point-cloud-conditioned baselines on shared artist-mesh and dense-mesh test sets: DeepMesh (Zhao et al., 2025), FastMesh (Kim et al., 2026), MeshMosaic (Xu et al., 2026), MeshRipple (Lin et al., 2026), MeshFlow (Li et al., 2026), and TSSR (Song et al., 2025). We report CD_{ℓ_1}, CD_{ℓ_2}, HD, $|\mathrm{NC}|$, and EF1; see Appendix D for test sets, metric definitions, and evaluation protocols.

5.2 Comparisons

Quantitative Comparisons. Table 1 presents quantitative comparisons on the artist-mesh and dense-mesh test sets. The results demonstrate that DisCoMesh achieves better surface and sharp-edge recovery than competing methods on both test sets.

Inference Efficiency. Table 2 compares generation time and mesh quality at approximately 8K faces, a scale supported by all evaluated methods. All timing measurements are conducted on a single NVIDIA A800 GPU. DisCoMesh achieves a favorable balance between inference speed and generation quality, providing better surface fidelity and sharp-edge recovery than the faster baselines while reducing generation time relative to other competitive methods. Further efficiency comparisons across different face counts are provided in Appendix B.2.

Qualitative Comparisons. Fig. 4 presents qualitative comparisons with the baselines. DisCoMesh better preserves intricate geometric structures while maintaining coherent connections between parts. The zoomed-in views further highlight fine local details and clean mesh topology.

5.3 Ablation Study

We conduct all ablations on a dedicated test set of 200 shapes held out from training.

Global Geometry Conditioning. We ablate bounding-box and global point-cloud conditioning individually and jointly to assess how global context supports part-wise generation, with part-aware sparse attention disabled in all variants. Table 3 shows that combining both conditions improves geometric fidelity over either alone, highlighting their complementary spatial and geometric guidance.

Inter-Part Coherence. We ablate sparse inter-part attention and the inter-part loss individually and jointly, measuring seam alignment using the mean seam vertex distance (MSVD):

$$\mathrm{MSVD}_{ij} = \frac{\sum_{\mathbf{v}\in\mathcal{V}_{ij}} \min_{\mathbf{u}\in\mathcal{V}_{ji}} \|\mathbf{v}-\mathbf{u}\|_2 + \sum_{\mathbf{u}\in\mathcal{V}_{ji}} \min_{\mathbf{v}\in\mathcal{V}_{ij}} \|\mathbf{u}-\mathbf{v}\|_2}{|\mathcal{V}_{ij}| + |\mathcal{V}_{ji}|}, \tag{16}$$

where $\mathcal{V}_{ij}$ and $\mathcal{V}_{ji}$ are the vertex sets on either side of the shared seam between generated parts i and j. Table 4 shows that combining both components reduces MSVD from 0.0048 to 0.0021, highlighting the complementary roles of inter-part communication and consistency supervision in aligning adjacent parts.

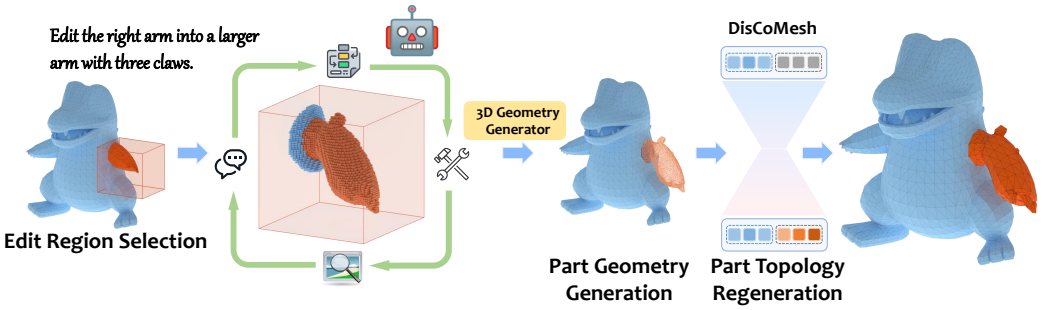


Figure 5: **Overview of agentic mesh editing.** The agent iteratively plans and executes voxel edits, calls a renderer tool to inspect the results, and incorporates user feedback. A 3D geometry generator converts the edited voxels into detailed reference geometry, which guides DisCoMesh in regenerating the selected part while preserving the remaining mesh.

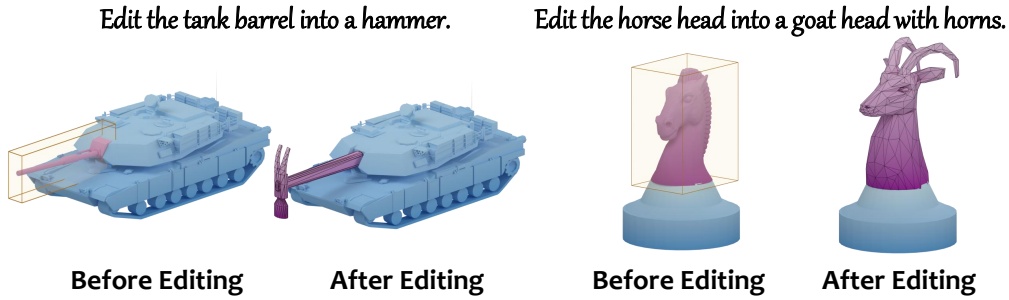


Figure 6: **Agentic mesh editing results.** Text-guided edits transform a tank barrel into a hammer and a horse head into a goat head with horns, while preserving the geometry and topology of the remaining mesh. Purple highlights the target parts, and blue denotes the preserved geometry.

5.4 Flexible Agentic Editing

DisCoMesh's part-wise formulation enables localized regeneration while preserving the rest of the mesh. We use voxel layouts as a directly editable 3D representation for agentic control, without an additional mesh-editing model (Fig. 5).

Given a target part, a user-specified bounding box, and a text or image prompt, the agent examines the target and neighboring parts, including voxelized attachment regions, and creates a coarse voxel layout within the box. It plans and iteratively adds, removes, or reshapes voxels, using rendered previews and user feedback to refine the component's shape and fit with neighboring parts.

The agent then converts the finalized layout into detailed reference geometry using an off-the-shelf 3D generation tool (Xiang et al., 2025). Conditioned on this reference, DisCoMesh performs multi-part **repainting**, regenerating only the selected component while preserving the geometry and topology of all unselected parts. Figure 6 shows diverse part-level edits that maintain coherent attachments to the original shapes.

6 Conclusion

We presented DisCoMesh, which formulates artist mesh generation as simultaneous discrete diffusion over interacting semantic parts. By coupling detailed local modeling with global coordination, DisCoMesh produces intricately detailed artist meshes with coherent inter-part connections, achieving improved geometric fidelity and a favorable balance between generation quality and inference efficiency. The resulting part-level control enables flexible agentic editing while preserving unedited geometry and topology, and supports a broad range of downstream applications.

AI USE STATEMENT

Generative AI tools were used to polish and edit portions of the manuscript. Within our agentic editing workflow, an AI agent creates and modifies voxel layouts to support localized mesh editing. This methodological use is described in the corresponding sections of the paper.

External AI writing assistants were not used to develop the research ideas, construct datasets, or conduct experiments. The authors were responsible for organizing and reporting the experimental results. All AI-assisted text was revised and approved by the authors. We take full responsibility for the final content of this work, including text produced with the assistance of generative AI.

ETHICS STATEMENT

This work aims to improve artist-mesh generation and localized editing for 3D content creation. The study does not involve human participants or identifiable personal data. Our training data are drawn from publicly available 3D datasets, with their sources and processing procedures documented in the paper.

REPRODUCIBILITY STATEMENT

We facilitate reproducibility by documenting our method, data construction, training procedure, and evaluation protocol. Section 4 describes the part-wise representation, model architecture, and training and inference procedures. Appendix A details the data sources and processing pipeline. Appendices C.1 and C.2 provide the training settings and loss formulations, respectively. Appendix D specifies the test sets, metric definitions, sampling procedures, success criteria, and inference-time measurement protocol. Details of the agentic editing workflow are provided in Appendix E. We plan to publicly release our data filtering pipeline, inference code, and model weights.

REFERENCES

Antonio Alliegro, Yawar Siddiqui, Tatiana Tommasi, and Matthias Nießner. PolyDiff: Generating 3d polygonal meshes with diffusion models. *arXiv preprint arXiv:2312.11417*, 2023. doi: 10.48550/arXiv.2312.11417. URL https://arxiv.org/abs/2312.11417.

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2503.09573.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://arxiv.org/abs/2107.03006.

Minghao Chen, Jianyuan Wang, Roman Shapovalov, Tom Monnier, Hyunyoung Jung, Dilin Wang, Rakesh Ranjan, Iro Laina, and Andrea Vedaldi. Autopartgen: Autoregressive 3d part generation and discovery. *arXiv preprint arXiv:2507.13346*, 2025a.

Sijin Chen, Xin Chen, Anqi Pang, Xianfang Zeng, Wei Cheng, Yijun Fu, Fukun Yin, Zhibin Wang, Jingyi Yu, Gang Yu, et al. Meshxl: Neural coordinate field for generative 3d foundation models. *Advances in Neural Information Processing Systems*, 37:97141–97166, 2024a.

Yiwen Chen, Tong He, Di Huang, Weicai Ye, Sijin Chen, Jiaxiang Tang, Xin Chen, Zhongang Cai, Lei Yang, Gang Yu, Guosheng Lin, and Chi Zhang. Meshanything: Artist-created mesh generation with autoregressive transformers, 2024b. URL https://arxiv.org/abs/2406.10163.

Yiwen Chen, Yikai Wang, Yihao Luo, Zhengyi Wang, Zilong Chen, Jun Zhu, Chi Zhang, and Guosheng Lin. Meshanything v2: Artist-created mesh generation with adjacent mesh tokenization. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13922–13931. IEEE, 2025b.

Zhiqin Chen and Hao Zhang. Neural marching cubes. *ACM Transactions on Graphics (Special Issue of SIGGRAPH Asia)*, 40(6), 2021.

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. *arXiv preprint arXiv:2307.05663*, 2023.

Lihe Ding, Shaocong Dong, Yaokun Li, Chenjian Gao, Xiao Chen, Rui Han, Yihao Kuang, Hong Zhang, Bo Huang, Zhanpeng Huang, et al. Fullpart: Generating each 3d part at full resolution. In *International Conference on Learning Representations*, volume 2026, pp. 45856–45876, 2026.

Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129 (12):3313–3337, 2021.

Zekun Hao, David W. Romero, Tsung-Yi Lin, and Ming-Yu Liu. Meshtron: High-fidelity, artist-like 3d mesh generation at scale, 2024. URL https://arxiv.org/abs/2412.09548.

Mukul Khanna, Yongsen Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X Chang, and Manolis Savva. Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16384–16393. IEEE, 2024.

Jeonghwan Kim, Yushi Lan, Armando Fortes, Yongwei Chen, and Xingang Pan. Fastmesh: Efficient artistic mesh generation via component decoupling. In *2026 International Conference on 3D Vision (3DV)*, pp. 1478–1488. IEEE, 2026.

Weiyu Li, Jiarui Liu, Hongyu Yan, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. Craftsman3d: High-fidelity mesh generation with 3d native diffusion and interactive geometry refiner. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5307–5317. IEEE, 2025.

Weiyu Li, Antoine Toisoul, Tom Monnier, Roman Shapovalov, Rakesh Ranjan, Ping Tan, and Andrea Vedaldi. MeshFlow: Efficient artistic mesh generation via MeshVAE and flow-based diffusion transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. doi: 10.48550/arXiv.2606.04621. URL https://arxiv.org/abs/2606.04621. CVPR 2026 Highlight.

Junkai Lin, Hang Long, Huipeng Guo, Jielei Zhang, Jiayi Yang, Tianle Guo, Yang Yang, Jianwen Li, Wenxiao ZHANG, Matthias Nießner, and Wei Yang. Meshripple: Structured autoregressive generation of artist-meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12706–12718, June 2026.

Yuchen Lin, Chenguo Lin, Panwang Pan, Honglei Yan, Yiqiang Feng, Yadong Mu, and Katerina Fragkiadaki. Partcrafter: Structured 3d mesh generation via compositional latent diffusion transformers, 2025. URL https://arxiv.org/abs/2506.05573.

Stefan Lionar, Jiabin Liang, and Gim Hee Lee. Treemeshgpt: Artistic mesh generation with autoregressive tree sequencing. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26608–26617. IEEE, 2025.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848, 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi Zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9914–9925. IEEE, 2024.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4135. URL https://arxiv.org/abs/2406.07524.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3277. URL https://arxiv.org/abs/2406.04329.

Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. Meshgpt: Generating triangle meshes with decoder-only transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Gaochao Song, Zibo Zhao, Haohan Weng, Jingbo Zeng, Rongfei Jia, and Shenghua Gao. Topology-preserved auto-regressive mesh generation in the manner of weaving silk. In *The Fourteenth International Conference on Learning Representations*, 2026.

Kaiyu Song, Hanjiang Lai, Yaqing Zhang, Chuangjian Cai, Yan Pan, Kun Yue, and Jian Yin. Topology sculptor, shape refiner: Discrete diffusion model for high-fidelity 3d meshes generation. *arXiv preprint arXiv:2510.21264*, 2025. doi: 10.48550/arXiv.2510.21264. URL https://arxiv.org/abs/2510.21264.

Stefan Stojanov, Anh Thai, and James M Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1798–1808. IEEE, 2021.

Jiaxiang Tang, Ruijie Lu, Zhaoshuo Li, Zekun Hao, Xuan Li, Fangyin Wei, Shuran Song, Gang Zeng, Ming-Yu Liu, and Tsung-Yi Lin. Efficient part-level 3d object generation via dual volume packing. *arXiv preprint arXiv:2506.09980*, 2025.

Penghao Wang, Yiyang He, Xin Lv, Yukai Zhou, Lan Xu, Jingyi Yu, and Jiayuan Gu. Partnext: A next-generation dataset for fine-grained and hierarchical 3d part understanding. 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. Curran Associates, Inc., 2025. doi: 10.52202/085713-2030. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/579636b69390b41fa2d39b744593bfbe-Paper-Datasets_and_Benchmarks_Track.pdf.

Haohan Weng, Zibo Zhao, Biwen Lei, Xianghui Yang, Jian Liu, Zeqiang Lai, Zhuo Chen, Yuhong Liu, Jie Jiang, Chunchao Guo, Tong Zhang, Shenghua Gao, and C. L. Philip Chen. Scaling mesh generation via compressive tokenization. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11093–11103, 2025. URL https://arxiv.org/abs/2411.07025.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21469–21480. IEEE, 2025.

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

Rui Xu, Tianyang Xue, Qiujie Dong, Le Wan, Zhe Zhu, Peng Li, Zhiyang Dou, Cheng Lin, Shiqing Xin, Yuan Liu, et al. Meshmosaic: scaling artist mesh generation via local-to-global assembly. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20003–20013, 2026.

Yichen Yang, Hong Li, Haodong Zhu, Linin Yang, Guojun Lei, Sheng Xu, and Baochang Zhang. Partdiffuser: Part-wise 3d mesh generation via discrete diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19940–19949, 2026.

Ruowen Zhao, Junliang Ye, Zhengyi Wang, Guangce Liu, Yiwen Chen, Yikai Wang, and Jun Zhu. Deepmesh: Auto-regressive artist-mesh creation with reinforcement learning. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10612–10623. IEEE, 2025.

Tianhao Zhao, Youjia Zhang, Hang Long, Jinshen Zhang, Wenbing Li, Yang Yang, Gongbo Zhang, Jozef Hladký, Matthias Nießner, and Wei Yang. LATO: 3d mesh flow matching with structured TOpology preserving LAtents. *arXiv preprint arXiv:2603.06357*, 2026. doi: 10.48550/arXiv.2603.06357. URL https://arxiv.org/abs/2603.06357.

Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, BIN FU, Tao Chen, Gang YU, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xmxgMij3LY.

Zhe Zhu, Le Wan, Rui Xu, Yiheng Zhang, Honghua Chen, Zhiyang Dou, Cheng Lin, Yuan Liu, and Mingqiang Wei. Partsam: A scalable promptable part segmentation model trained on native 3d data. In *International Conference on Learning Representations*, volume 2026, pp. 132469–132493, 2026.

Qi Zuo, Xiaodong Gu, Yuan Dong, Zhengyi Zhao, Weihao Yuan, Lingteng Qiu, Liefeng Bo, and Zilong Dong. High-fidelity 3d textured shapes generation by sparse encoding and adversarial decoding. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 52–69, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72684-2.

A Dataset Construction and Preprocessing

A.1 Single-Entity Dataset

Data Sources. We assemble the single-entity dataset from publicly available 3D collections, including G-Objaverse (Zuo et al., 2025; Qiu et al., 2024), 3D-FUTURE (Fu et al., 2021), HSSD (Khanna et al., 2024), and Objaverse-XL (Deitke et al., 2023). To obtain a consistent mesh representation for subsequent filtering, we apply scene-node transformations, merge scene geometries, and triangulate polygonal faces.

Geometric Filtering. After mesh standardization, we filter the meshes by face count and geometric quality. We retain meshes containing between 200 and 7,000 triangular faces after cleaning, excluding overly coarse assets while keeping training sequences within the length supported by the single-entity model.

We assess face-area imbalance using the Gini coefficient. For a mesh with F triangular faces and face areas $\{a_i\}_{i=1}^{F}$, we define

$$G_{\text{area}} = \frac{\sum_{i=1}^{F} \sum_{j=1}^{F} |a_i - a_j|}{2F \sum_{i=1}^{F} a_i}. \tag{17}$$

A high G_{area} indicates that a small fraction of faces accounts for a large proportion of the surface area, helping identify meshes with dominant planar artifacts. We additionally record the minimum and maximum face areas, the area proportion occupied by the largest faces, and the fraction of near-zero-area faces to identify concentrated surface coverage and degenerate geometry.

We further examine the minimum and maximum triangle interior angles to identify slender or nearly degenerate faces, whose angles approach 0° or 180°. We also use the vertex-to-face ratio as an indicator of vertex sharing: a high ratio can indicate triangle soups or fragmented meshes with limited connectivity between faces.

Finally, we assess open or fragmented surfaces using the number of boundary loops, the total boundary-edge count, and the size of the largest boundary loop. We also detect large extraneous ground or background planes based on their spatial extent and proportion of the total mesh surface area.

After geometric filtering, the single-entity dataset contains approximately 92K meshes, with the face-count distribution shown in the left panel of Fig. 8.

Representative rejected assets include meshes dominated by large ground planes and those containing numerous open, sheet-like components, identified through face-area imbalance and boundary statistics, respectively (Fig. 7).

A.2 Multi-Part Dataset

Data Sources. We construct the multi-part dataset from PartVerseXL (Ding et al., 2026) and PartNeXt (Wang et al., 2025), both of which provide part-level decompositions of complete 3D shapes. To increase shape diversity, we additionally apply PartSAM (Zhu et al., 2026) to geometrically filtered meshes from G-Objaverse.

Filtering. For PartVerseXL and PartNeXt, we retain shapes with total face counts between 3,000 and 40,000. Each shape contains between 3 and 100 parts, each with more than 100 and at most 7,000 triangular faces. We further filter imbalanced decompositions using Gini coefficients of face areas within parts and of surface area, face count, and bounding-box volume across parts. PartSAM decompositions follow the same total face-count range and per-part upper limit.

After filtering and deduplication, the multi-part dataset contains approximately 11K shapes, with part-count and total face-count distributions shown in the middle and right panels of Fig. 8, respectively.



$G_{\text{area}} = 0.990$ Number of boundary loops: 506

Figure 7: **Examples rejected during geometric filtering.** Left: a large ground plane dominates the surface area, highlighted in orange. Right: numerous open leaf surfaces produce many boundary loops, with boundary edges marked in red.

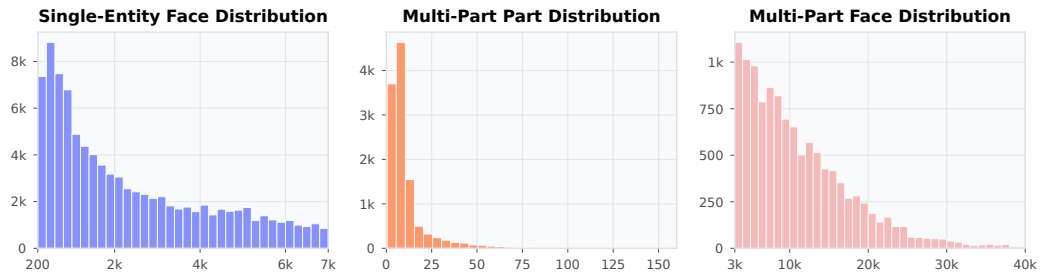


Figure 8: **Dataset statistics.** Left: face-count distribution of single-entity meshes. Middle: part-count distribution of multi-part shapes. Right: total face-count distribution of multi-part shapes.

B Additional Results and Comparisons

B.1 Additional Qualitative Comparisons

DisCoMesh preserves detailed part geometry across a variety of characters and mechanical objects, including small attachments and intricate surface structures. Additional comparisons in Fig. 9 illustrate these advantages over the baselines, with zoomed-in views highlighting fine local details and clean mesh topology.

B.2 Inference Efficiency

DisCoMesh combines high geometric fidelity with efficient inference across different face-count ranges (Fig. 10). It maintains substantially lower inference times than DeepMesh, MeshMosaic, and TSSR, which offer more comparable generation quality. Meanwhile, the CD–EF1 comparison shows better surface and sharp-edge recovery than the faster baselines FastMesh and MeshFlow, highlighting a favorable balance between inference speed, generation quality, and output complexity.

B.3 Level-of-Detail Generation

DisCoMesh remains robust to variations in the specified face budget, preserving both the overall shape and fine geometric details across different face-count ranges. This robustness naturally supports level-of-detail generation within a single model, allowing the same reference shape to be represented with different numbers of faces while retaining consistent geometry. The face budget therefore provides flexible control over mesh complexity for different application requirements. We illustrate this capability with two reference shapes, each generated at four face-budget levels (Fig. 11).

B.4 Scene Generation

DisCoMesh naturally extends to scene generation by treating individual objects as parts of a complete scene, bringing both asset-level and scene-level generation into a unified part-wise framework. DisCoMesh can simultaneously generate approximately 100 objects within a single scene, highlighting its scalability, versatility, and generalization from individual assets to complex scenes (Fig. 12).

B.5 Diverse Downstream Applications

The explicit part-wise representation of DisCoMesh enables a range of downstream applications through direct manipulation of individual part sequences. Our framework supports part editing, addition, removal, part-level retopology, and composition across different source objects, covering both part-level geometric and topological editing and structural recombination (Fig. 13).

For part editing, a selected component is regenerated from an updated geometric condition while the remaining parts are preserved and provide object context. Part addition augments an object with newly generated components, whereas part removal directly removes selected components from the representation. Part-level retopology regenerates a target component under a different face budget while keeping the other part sequences unchanged, enabling local control over mesh complexity. Finally, part composition combines geometric conditions from different objects and simultaneously regenerates the assembled part set as a coherent shape.

C Training Details

C.1 Implementation Details

Our model contains approximately 1B parameters. We train the model on 32 NVIDIA A800 GPUs with a global batch size of 256. The pretrained point-cloud encoder remains frozen throughout training.

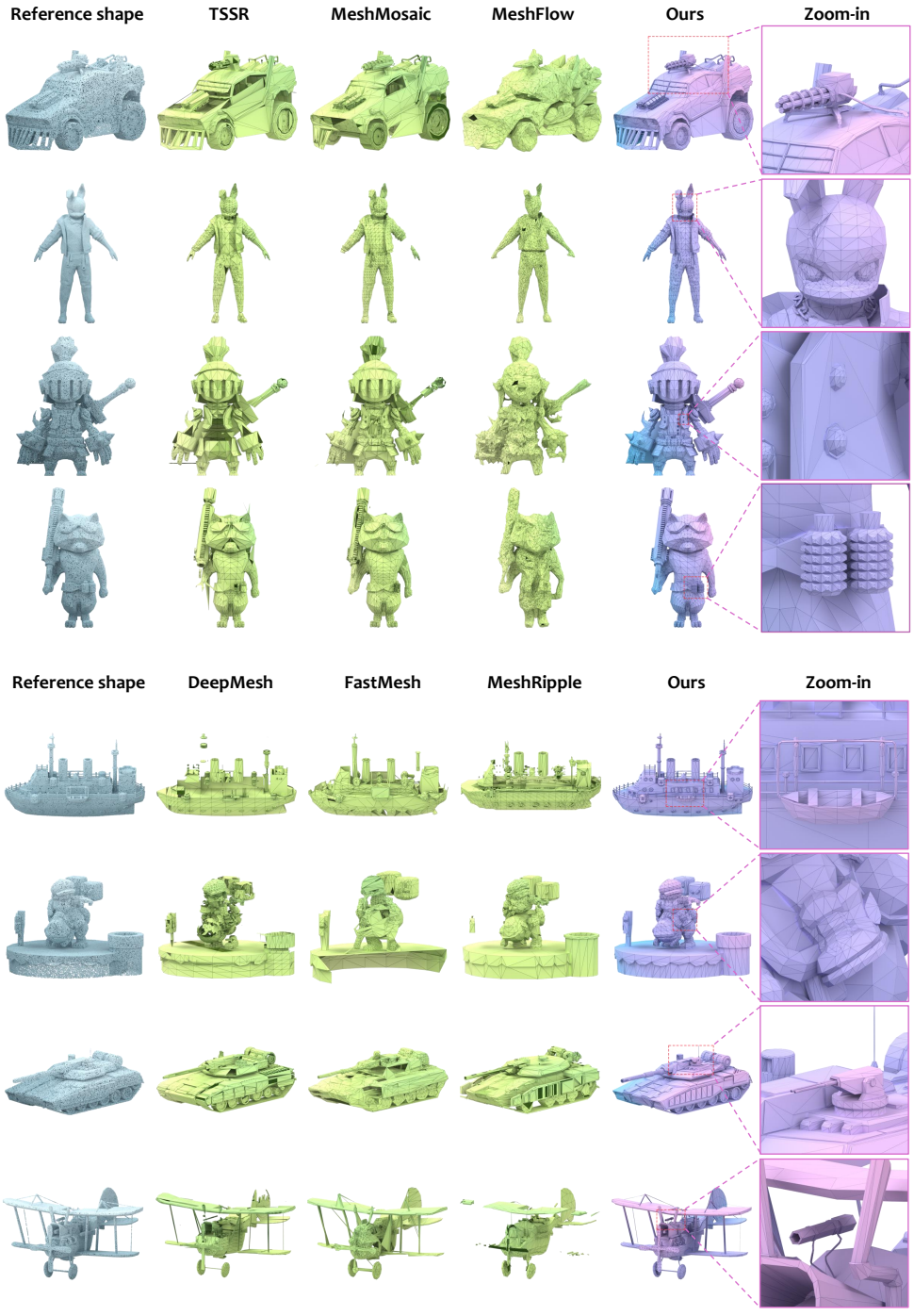
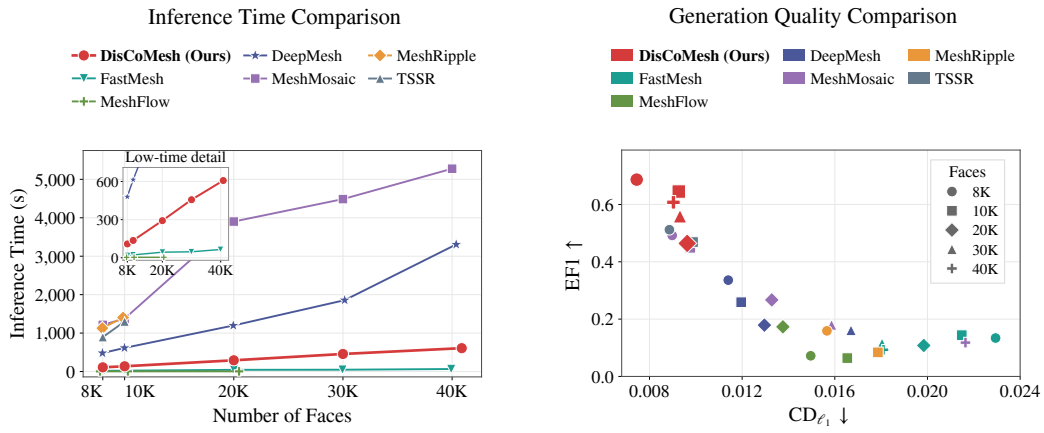


Figure 9: **Additional qualitative comparisons with baseline methods.** DisCoMesh preserves detailed geometry and coherent part structures across diverse shapes. Zoomed-in views highlight fine local details and clean mesh topology.



(a) Inference time

(b) Generation quality

Figure 10: **Inference efficiency across face-count ranges.** (a) Inference time versus output face count. (b) Surface and sharp-edge recovery measured by CD_{ℓ_1} and EF1. Colors denote methods, while marker shapes in (b) distinguish face-count ranges.

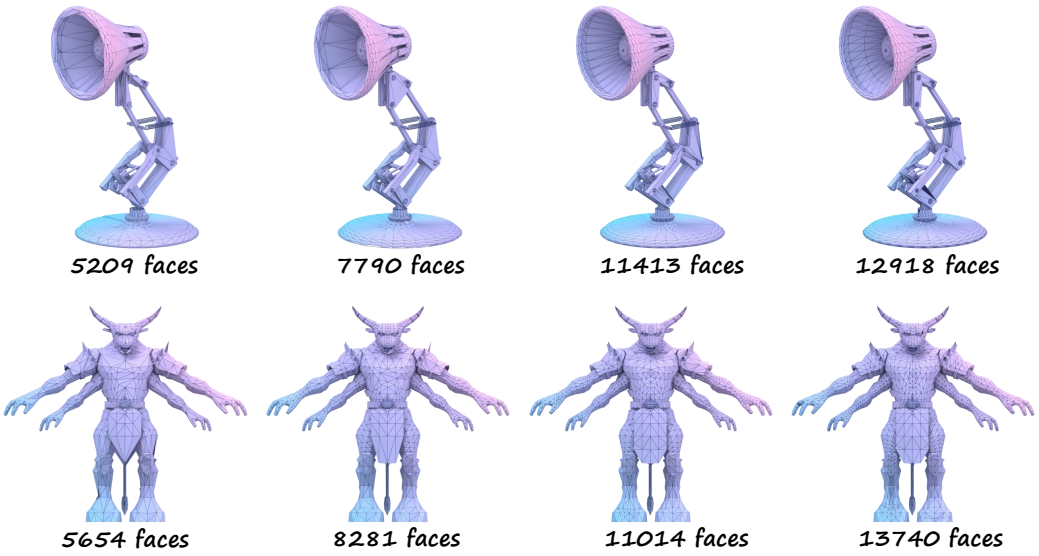


Figure 11: **Level-of-detail generation through face-budget control.** Each row shows the same reference shape generated at four face-budget levels, preserving the overall shape and fine geometric details across different face counts.

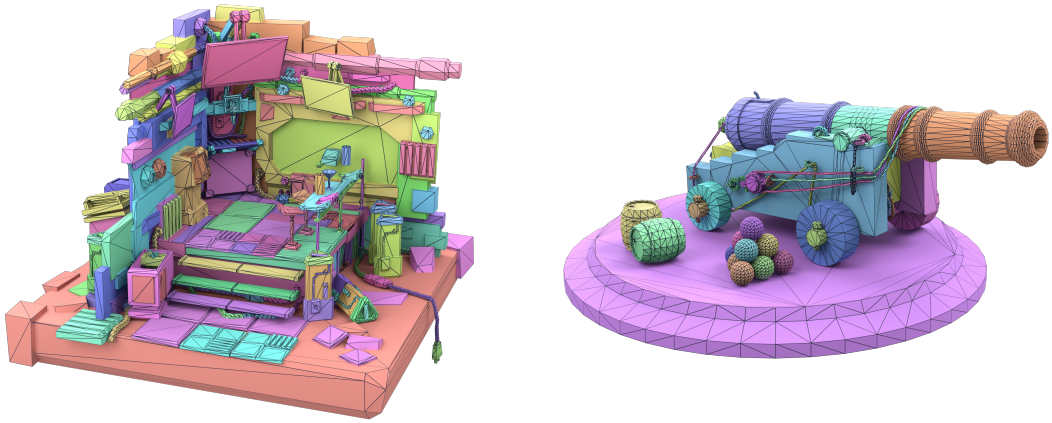


Figure 12: **Scene generation with DisCoMesh.** Our part-wise framework naturally extends to complete scenes, enabling simultaneous generation of approximately 100 objects.

We first pretrain the model on single-object meshes without semantic decomposition for 10K steps, followed by 4K steps of multi-part training. The second stage initializes the backbone from the pretrained checkpoint and introduces newly initialized GIM parameters. Within this stage, we first enable bounding-box and global point-cloud conditioning, followed by sparse inter-part attention.

We use the AdamW optimizer with a weight decay of 0.01, $(\beta_1, \beta_2) = (0.9, 0.999)$, and $\epsilon = 10^{-8}$. Both stages use the same learning-rate schedule. We linearly warm up the learning rate to 10^{-4} over the first 100 steps of each stage and keep it constant for the remaining steps.

C.2 Training Objectives

We provide the formulations of the four training objectives. Following TSSR (Song et al., 2025), we use denoising, token-confidence, and intra-part connection objectives. We additionally introduce an inter-part consistency objective to encourage agreement at shared interfaces across parts.

Denoising Supervision. Let $\mathbf{x}$ denote the clean token sequence, and let $\mathbf{p}^{\text{mask}}$ and $\mathbf{p}^{\text{refine}}$ denote the predicted token distributions during masked prediction and refinement, respectively. The denoising loss supervises both passes using cross-entropy:

$$\mathcal{L}_{\text{denoise}} = \text{CE}(\mathbf{p}^{\text{mask}}, \mathbf{x}) + \text{CE}(\mathbf{p}^{\text{refine}}, \mathbf{x}). \tag{18}$$

Both terms are averaged over valid coordinate tokens.

Token-Confidence Supervision. The confidence predictor estimates token correctness to guide remasking during inference. For each token $\hat{x}_n$ obtained by argmax after refinement, we supervise its predicted confidence using whether it matches the ground-truth token x_n:

$$y_n = \mathbf{1}[\hat{x}_n = x_n], \qquad \mathcal{L}_{\text{conf}} = \text{BCE}(\mathbf{q}, \mathbf{y}), \tag{19}$$

where $\mathbf{q}$ denotes the predicted correctness probabilities, $\mathbf{y}$ contains the correctness labels, and BCE denotes binary cross-entropy averaged over valid coordinate tokens. The confidence head is trained using detached backbone features.

Intra-Part Connection Consistency. The face-wise representation contains repeated occurrences of vertices shared by multiple faces. We encourage consistent coordinate logits across these occurrences during masked prediction. Let $\mathcal{G}$ denote the collection of shared-vertex groups within individual parts. For each group $G \in \mathcal{G}$, let $\mathbf{z}_r^a$ denote the predicted logits for occurrence r along

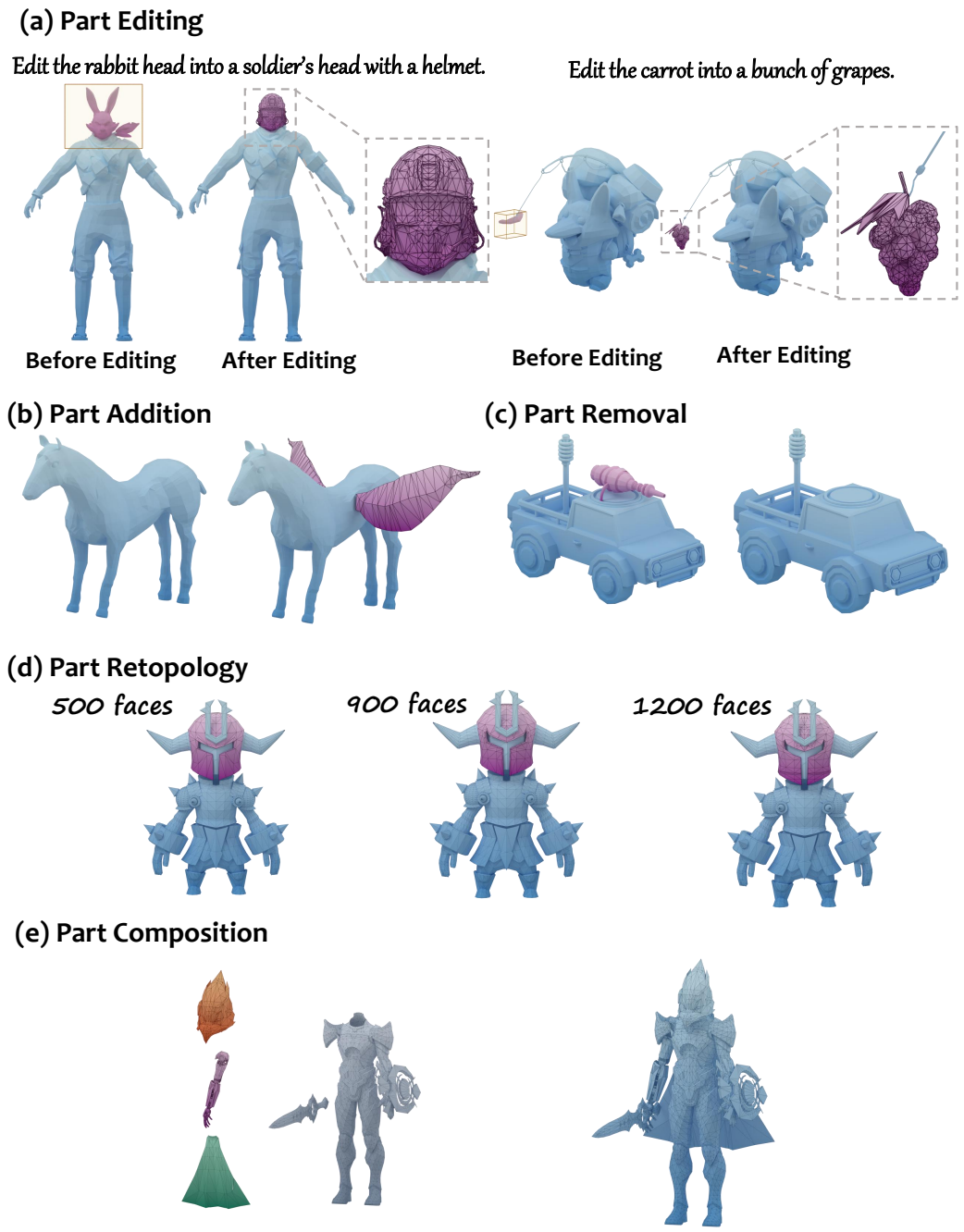


Figure 13: **Diverse downstream applications supported by DisCoMesh.** (a) Part editing regenerates selected components from updated geometric conditions. (b) Part addition introduces new components. (c) Selected components are removed while the remaining mesh is preserved. (d) Part-level retopology varies the selected component's face budget across 500, 900, and 1,200 faces while preserving the remaining parts. (e) Part composition combines an incomplete base with parts from different source objects. In (a)–(c), each example shows the input on the left and the result on the right. Purple highlights the affected parts in (a)–(d); distinct colors identify source parts in (e).

axis a. The consensus logits are

$$\overline{\mathbf{z}}_G^a = \frac{1}{|G|} \sum_{r \in G} \mathbf{z}_r^a. \tag{20}$$

The intra-part connection loss penalizes deviations from this consensus:

$$\mathcal{L}_{\text{intra}} = \frac{1}{|\mathcal{G}|} \sum_{G \in \mathcal{G}} \frac{1}{3|G|} \sum_{a \in \{x,y,z\}} \sum_{r \in G} \|\mathbf{z}_r^a - \overline{\mathbf{z}}_G^a\|_1. \tag{21}$$

Each group contains occurrences from the same part. Cross-part consistency is enforced separately by $\mathcal{L}_{\text{inter}}$ in the global coordinate system, as described below.

Inter-Part Consistency. To encourage coherent connections across parts, we align the predicted coordinate distributions of shared interface vertices. Since each part predicts coordinates in its own canonical space, we first map these distributions to a common global coordinate system. The following construction is applied separately to masked prediction and refinement.

For a shared vertex v, let $p_{v,i}^a(u)$ denote the predicted probability mass at local bin center $u \in [-1, 1]$ along axis $a \in \{x, y, z\}$ in part i. Using the part's bounding box $\mathbf{B}_i$, we map each local bin center to its global coordinate while preserving its probability mass:

$$g = c_i^a + \frac{s_i}{2} u, \qquad P_{v,i}^a(g) = p_{v,i}^a(u), \tag{22}$$

where c_i^a is the bounding-box center along axis a, s_i is the cube side length used for canonicalization, and $P_{v,i}^a$ is the resulting discrete distribution over global coordinates.

For a vertex v shared by parts i and j, we measure the discrepancy between their predicted global coordinate distributions as

$$d_{v,ij} = \sum_{a \in \{x,y,z\}} W_1(P_{v,i}^a, P_{v,j}^a), \tag{23}$$

where W_1 denotes the one-dimensional Wasserstein distance.

For each prediction pass, the inter-part consistency loss averages these discrepancies across shared interface vertices, encouraging agreement between parts at their interfaces. The final inter-part objective averages the two passes:

$$\mathcal{L}_{\text{inter}} = \frac{\mathcal{L}_{\text{inter}}^{\text{mask}} + \mathcal{L}_{\text{inter}}^{\text{refine}}}{2}. \tag{24}$$

D Evaluation Details

D.1 Evaluation Data

We evaluate on an artist-mesh test set and a dense-mesh test set. The artist-mesh test set combines approximately 300 meshes from our G-Objaverse test split with approximately 700 meshes from Toys4K (Stojanov et al., 2021). We construct the dense-mesh test set using TRELLIS (Xiang et al., 2025) to generate approximately 250 meshes from reference images provided in the official repositories of image-to-3D methods, including TRELLIS, TRELLIS.2 (Xiang et al., 2026), and Craftsman (Li et al., 2025).

D.2 Metrics and Evaluation Protocol

Conditioning Inputs. All methods are evaluated under point-cloud conditioning. Each baseline uses the number of input points specified by its official implementation.

Normalization and Sampling. We center the reference bounding box at the origin and isotropically scale its longest side to one, so that it lies within $[-0.5, 0.5]^3$. The same transformation is applied to the generated mesh. We sample 30K points uniformly over the surface of each mesh using triangle-area-weighted sampling. Each sampled point inherits the unit normal of its triangle. For each reference mesh, the same sampled points are used across methods.

Surface Metrics. Let $\mathcal{P}$ and $\mathcal{G}$ denote the sampled point sets of the generated and reference meshes, respectively. For a point $\mathbf{u}$ and a nonempty point set $\mathcal{Q}$, we define the nearest-neighbor distance and correspondence as

$$d(\mathbf{u}, \mathcal{Q}) = \min_{\mathbf{v}\in\mathcal{Q}} \|\mathbf{u}-\mathbf{v}\|_2, \qquad \pi_{\mathcal{Q}}(\mathbf{u}) = \arg\min_{\mathbf{v}\in\mathcal{Q}} \|\mathbf{u}-\mathbf{v}\|_2. \tag{25}$$

The two Chamfer distances sum the directional means:

$$\begin{aligned} \mathrm{CD}_{\ell_1} &= \frac{1}{|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{P}} d(\mathbf{p},\mathcal{G}) + \frac{1}{|\mathcal{G}|}\sum_{\mathbf{g}\in\mathcal{G}} d(\mathbf{g},\mathcal{P}), \\ \mathrm{CD}_{\ell_2} &= \frac{1}{|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{P}} d(\mathbf{p},\mathcal{G})^2 + \frac{1}{|\mathcal{G}|}\sum_{\mathbf{g}\in\mathcal{G}} d(\mathbf{g},\mathcal{P})^2. \end{aligned} \tag{26}$$

Here, CD_{ℓ_1} uses Euclidean distances, whereas CD_{ℓ_2} uses squared Euclidean distances. We report CD_{ℓ_2} multiplied by 10^3 in the tables.

The Hausdorff distance is the maximum nearest-neighbor distance over both directions:

$$\mathrm{HD} = \max\{\max_{\mathbf{p}\in\mathcal{P}} d(\mathbf{p},\mathcal{G}), \max_{\mathbf{g}\in\mathcal{G}} d(\mathbf{g},\mathcal{P})\}. \tag{27}$$

Let $\mathbf{n}(\mathbf{u})$ denote the unit normal at a sampled point $\mathbf{u}$. Absolute normal consistency averages the two directional mean absolute normal dot products:

$$\begin{aligned} |\mathrm{NC}| = &\frac{1}{2|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{P}} |\mathbf{n}(\mathbf{p})^\top \mathbf{n}(\pi_{\mathcal{G}}(\mathbf{p}))| \\ &+ \frac{1}{2|\mathcal{G}|}\sum_{\mathbf{g}\in\mathcal{G}} |\mathbf{n}(\mathbf{g})^\top \mathbf{n}(\pi_{\mathcal{P}}(\mathbf{g}))|. \end{aligned} \tag{28}$$

Edge F-Score. We evaluate sharp-edge recovery using EF1 following the official NMC (Chen & Zhang, 2021) implementation. We independently sample 100K surface points from each mesh. Sharp-edge points are identified from neighboring sample pairs whose Euclidean distance is at most 0.004 and whose absolute normal dot product is below 0.2. All edge thresholds are defined in the same normalized coordinate frame.

Let $\mathcal{E}_P$ and $\mathcal{E}_G$ denote the extracted sharp-edge point sets of the generated and reference meshes. For nonempty edge sets, edge precision P_e and recall R_e are computed using the matching threshold $\tau_e = 0.005$:

$$\begin{aligned} P_e &= \frac{1}{|\mathcal{E}_P|}\sum_{\mathbf{p}\in\mathcal{E}_P} \mathbf{1}[d(\mathbf{p},\mathcal{E}_G) < \tau_e], \\ R_e &= \frac{1}{|\mathcal{E}_G|}\sum_{\mathbf{g}\in\mathcal{E}_G} \mathbf{1}[d(\mathbf{g},\mathcal{E}_P) < \tau_e]. \end{aligned} \tag{29}$$

EF1 is their harmonic mean:

$$\mathrm{EF1} = \frac{2P_eR_e}{P_e+R_e}, \tag{30}$$

with $\mathrm{EF1} = 0$ when $P_e + R_e = 0$. Empty edge sets are handled following the official NMC implementation.

Result Aggregation. To prevent extreme errors from failed generations from dominating the average quality scores, we evaluate geometric quality on successful outputs. For all methods, a generation is considered successful if it produces a valid mesh with $\mathrm{HD} \le 0.15$. We report the mean quality metrics over successful cases, together with the success rate over the full test set.

Inference Time. All timing measurements are conducted on a single NVIDIA A800 GPU. We measure inference time as the elapsed time required to generate a mesh from the prepared input. The measurement covers the mesh generation process and excludes input preparation, mesh repair, and file export. We apply the same timing scope to all evaluated methods.

E Details of the Agentic Editing Workflow

Agent and editing state. We use GPT-5.6 Sol as the underlying model for our editing agent. Given a selected part, a target bounding box, and a text or image request, the editing environment maintains an editable target occupancy and read-only context voxels representing neighboring parts and their attachment regions. Both use a 64^3 grid within the target bounding box. The agent receives a shared instruction, tool definitions, the current state summary, and rendered observations, and returns structured tool calls for execution.

Interactive editing workflow. The agent uses `edit_voxels` to add or remove target voxels through coordinate selections or geometric primitives, then calls `render_voxels` to inspect the updated layout together with its context. After each rendering, the agent evaluates the current shape and its attachment to neighboring parts, then decides whether to revise the voxels or finalize the layout. This edit–render–evaluate cycle incorporates optional user feedback or direct voxel edits. The environment validates tool operations and keeps the context immutable. Once the layout is satisfactory, `finalize_voxels` performs attachment-preserving cleanup and validation; any remaining violations are returned for another editing iteration. After successful finalization, the agent calls `generate_reference_mesh` to obtain detailed reference geometry, followed by `run_multipart_repaint` to regenerate the selected part with DisCoMesh while preserving the remaining mesh.

Agent instruction summary. The shared instruction directs the agent to construct a coarse part shape that follows the editing request and forms compatible attachments to neighboring geometry. The agent must modify only the target occupancy within the bounding box, keep the context unchanged, maintain target connectivity, and execute edits through the provided tools. It evaluates rendered proposals for semantics, scale, orientation, proportions, and attachment, incorporating user feedback when available.

Voxel editing and rendering. Target and context occupancies are stored as binary NumPy arrays. Voxel additions and removals use Boolean operations, with SciPy supporting connected-component analysis. Finalization preserves attachment anchors, removes non-anchor target voxels that collide with the context outside the attachment region, and discards components disconnected from the anchors. Blender produces fixed multi-view renderings with distinct colors for the target and context.

Reference geometry generation. We use the Stage-2 generation procedure of TRELLIS (Xiang et al., 2025) to generate detailed reference geometry from the finalized voxel layout, conditioned on the text or image editing request. The resulting mesh provides geometric guidance for subsequent DisCoMesh repainting.

Multi-part repainting. The reference mesh provides the updated target point-cloud condition, including attachment anchors, and determines the target bounding box. We combine it with the unchanged non-target components and resample the assembled geometry to update the global point-cloud condition. DisCoMesh then corrupts and regenerates only the selected part sequence, while all other sequences remain uncorrupted and provide whole-object context. Editing states, tool calls and outputs, and user feedback are recorded for inspection and replay.

F Discussion

F.1 Mesh Representation and Parallel Generation

DisCoMesh adopts the face-wise coordinate representation used in Meshtron (Hao et al., 2024) and TSSR (Song et al., 2025), encoding each triangle with nine coordinate tokens. Each token position corresponds to a specific vertex coordinate within a face. This explicit layout is well suited to discrete diffusion: coordinates can be corrupted and refined simultaneously while retaining their association with individual faces.

Compressive tokenizations such as BPT (Weng et al., 2025) and DeepMesh (Zhao et al., 2025) shorten mesh sequences through patch-level vertex reuse and block-index compression. Their de-

coding relies on preceding tokens to recover vertex positions and face connectivity. During parallel denoising, these preceding tokens may themselves be masked or revised, altering the geometric interpretation of the tokens that depend on them. The face-wise representation adopted in DisCoMesh avoids these decoding dependencies and supports simultaneous refinement of all face coordinates. Developing more compact face encodings that retain this property is a promising direction for further improving the efficiency of parallel mesh generation.

F.2 Artist-Mesh Generation Paradigms

Artist-mesh generation requires both reliable generation of complete shapes and faithful preservation of geometric details and face structures. We discuss autoregressive generation, continuous flow, and discrete diffusion along these two dimensions, with quantitative results summarized in Table 5. Success rate (SR) is defined as the percentage of test cases that produce a valid mesh with $\mathrm{HD} \leq 0.15$. Geometric metrics are averaged over successful generations.

Table 5: **Generation robustness and geometric fidelity.** SR measures the percentage of test cases producing a valid mesh with $\mathrm{HD} \leq 0.15$. Geometric metrics are averaged over successful generations. The best results are shown in **bold**, and the second-best results are underlined.

Paradigm	Method	SR ↑	CD_{ℓ_1} ↓	HD ↓	EF1 ↑
Autoregressive	DeepMesh	77.3%	0.0107	0.0562	0.373
Autoregressive	MeshMosaic	74.0%	0.0106	0.0624	0.429
Autoregressive	MeshRipple	71.6%	0.0127	0.0673	0.239
AR + parallel connectivity	FastMesh	86.8%	0.0204	0.0590	0.131
Continuous flow	MeshFlow	**99.6%**	0.0161	<u>0.0442</u>	0.082
Discrete diffusion	TSSR	90.1%	<u>0.0090</u>	0.0513	<u>0.536</u>
Discrete diffusion	**DisCoMesh (Ours)**	<u>98.5%</u>	**0.0077**	**0.0281**	**0.683**

Autoregressive methods (Zhao et al., 2025; Xu et al., 2026; Lin et al., 2026) can generate artist meshes with detailed geometry. However, autoregressive decoding uses earlier token predictions as fixed context for subsequent generation, leaving earlier decoding errors uncorrected. In our evaluation, these methods achieve success rates ranging from 71.6% to 77.3%, indicating remaining difficulties in consistently generating complete, geometrically faithful meshes.

MeshFlow (Li et al., 2026) generates continuous mesh latents in parallel through rectified flow, achieving a success rate of 99.6% and an HD of 0.0442, yet its EF1 is only 0.082. FastMesh (Kim et al., 2026), which combines autoregressive vertex generation with parallel connectivity prediction, achieves an 86.8% success rate but an EF1 of only 0.131. These results show that reliable recovery of the overall shape does not necessarily ensure faithful preservation of sharp features. Our qualitative comparisons further show that MeshFlow struggles to reproduce the highly non-uniform vertex distributions and carefully designed topology of artist-created meshes (Figs. 4 and 9).

Discrete diffusion supports parallel generation and iterative correction while directly modeling explicit mesh faces. During this iterative denoising process, mesh tokens exchange information across the entire sequence through bidirectional attention, without the restriction to preceding tokens imposed by autoregressive decoding. TSSR (Song et al., 2025) achieves a 90.1% success rate and an EF1 of 0.536, demonstrating strong sharp-feature recovery alongside reliable generation. DisCoMesh further reaches a 98.5% success rate, approaching the success rate of MeshFlow while achieving the best CD, HD, and EF1 among the evaluated methods. These results support simultaneous discrete generation as an effective approach to preserving intricate artist-mesh geometry while maintaining high generation success rates.