BVHMESH: FACE-FIRST MESH GENERATION ON A BOUNDING VOLUME HIERARCHY

Anonymous authors
Paper under double-blind review

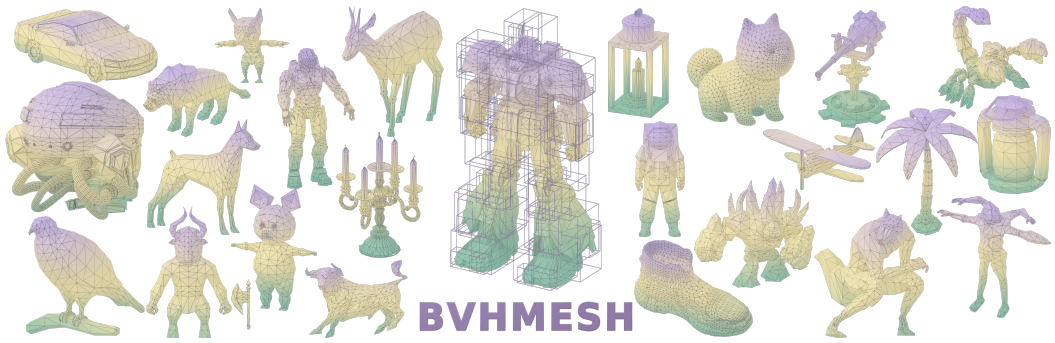

Figure 1: BVHMESH generates artist-style meshes with high geometric fidelity and clean topology.

ABSTRACT

Artist-style meshes are compact and editable, but generating their continuous geometry and irregular connectivity efficiently remains challenging. Field-based methods extract dense tessellations, whereas autoregressive methods directly generate compact meshes but decode sequentially. Recent flow-based methods generate vertices in parallel and then recover faces from predicted connectivity, where edge errors can propagate into invalid surfaces. We introduce BVHMESH as a face-first flow-based framework that decodes triangles in parallel and stitches them into a mesh. To make triangle prediction tractable, we organize triangles in an octree-aligned bounding volume hierarchy (OctBVH). Its nested boxes provide local coordinate frames for coarse-to-fine geometry prediction, while octree alignment allows sparse convolutions to process the hierarchy. A hierarchical VAE and conditional flow matching generate triangle soups from point clouds or images. EdgeMatch then recovers connectivity through learned half-edge matching with explicit checks against non-manifold edges, duplicate faces, and collapsed triangles. On Toys4K, BVHMESH achieves state-of-the-art VAE reconstruction and artist-mesh generation under both point-cloud and image conditioning. In point-cloud generation, its non-manifold edge ratio is zero versus 11.50% for MeshFlow, while inference takes about 15 seconds per mesh even beyond $10{,}000$ faces.

1 INTRODUCTION

Polygonal meshes are the native representation of modern graphics pipelines: they are compact, efficient to render, and support editing, animation, and simulation. Learning to generate such a representation is challenging because meshes couple continuous vertex positions with discrete, irregular topology. The central question is how to generate both efficiently.

Existing approaches face a trade-off between generation efficiency and connectivity quality. Field-based methods generate volumetric fields and extract meshes afterward (Mescheder et al., 2019; Peng et al., 2020; Xiong et al., 2024; Xiang et al., 2025b). These methods scale well to complex geometry, but iso-surface extraction typically produces dense, topology-agnostic tessellations that require simplification and remeshing. Autoregressive methods instead generate compact, artist-style meshes, typically using vertex- or coordinate-level tokens (Siddiqui et al., 2024; Chen et al.,

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.

2025b; Wang et al., 2026b). Improved tokenizations reuse vertices or local patches to shorten the sequence (Chen et al., 2025c; Weng et al., 2025b; Xu et al., 2026) for better efficiency. However, decoding remains sequential, limiting inference efficiency as mesh resolution grows.

A recent family of flow-based mesh generation methods improves efficiency by first generating mesh vertices in parallel then infer pairwise connectivity and recover faces (Li et al., 2026b; Zhao et al., 2026; Long et al., 2026; Wang et al., 2026a;c). The face set is thus a derived quantity that is not predicted or constrained directly. However, pairwise connectivity between vertices alone does not determine valid surface faces: a missing edge can remove several faces, while a spurious edge can create multiple unintended faces. Face recovery may thus amplify connectivity errors into holes, overlapping faces, and non-manifold topology. More expressive topology embeddings or face validation can mitigate these errors, but reliable connectivity recovery remains challenging.

In light of these challenges, we introduce BVHMESH to generate meshes in a *face-first* manner, while maintaining efficiency through parallel decoding with diffusion models. The face-first design allows the model to directly predict triangles, rather than infer them from vertices and edges. Predicting triangles in global coordinates, however, is difficult for generative models. To this end, we draw on bounding volume hierarchies (BVHs), which are widely used in graphics to organize geometric primitives, to support hierarchical generation. Their nested bounding boxes provide progressively finer local coordinate frames, allowing hierarchical quantization to reduce prediction complexity while retaining fine geometric detail. However, BVHs are irregular trees, and existing learning methods, such as sparse convolutions (Wang et al., 2017; Choy et al., 2019; Graham et al., 2018), are not well suited to BVH-based generation. We therefore introduce an octree-aligned BVH representation, OctBVH, enabling a hierarchical VAE to decode triangle geometry from coarse to fine using octree-based sparse convolutions (Wang et al., 2017).

To further reduce connectivity errors, we introduce a novel edge-based approach for stitching the generated triangles. We also incorporate explicit topology checks during stitching to reject connections that would introduce non-manifold edges, duplicate faces, or collapsed triangles. For generation, we fine-tune a two-stage flow-based generative method (Xiang et al., 2025b); During inference, triangles are decoded in parallel, significantly reducing the computational cost associated with sequential face-by-face decoding.

Experiments on the held-out benchmark demonstrate state-of-the-art performance in point-cloud- and image-conditioned artist-mesh generation, as well as VAE mesh reconstruction. Beyond geometric fidelity, our framework better preserves thin structures and openings, with less redundant surface coverage than vertex-first baselines. In point-cloud-conditioned generation, the measured non-manifold edge ratio is zero, compared with 11.50% for MeshFlow (Fig. 3). Generation remains efficient at approximately 15 seconds per mesh, including outputs with more than $10{,}000$ faces, and is roughly an order of magnitude faster than TreeMeshGPT at around $5{,}000$ faces (Fig. 8). Our main contributions are:

- We introduce OctBVH, an octree-aligned BVH with nested local coordinate frames that supports accurate, parallel triangle generation through a hierarchical VAE and flow priors.
- We propose EdgeMatch to recover connectivity through learned one-to-one edge matching with explicit topology checks.
- We demonstrate state-of-the-art artist-mesh generation and reconstruction on Toys4K, with fast inference and improved topological quality, and validate both components through ablations.

2 RELATED WORK

2.1 FIELD-BASED 3D GENERATION AND MESH EXTRACTION

Field-based 3D generative models have achieved high-fidelity shape synthesis across voxel grids, triplanes, and implicit neural functions (Cheng et al., 2023; Gupta et al., 2023; Jun & Nichol, 2023). More recent 3D-native representations, exemplified by 3DShape2VecSet and TRELLIS, use compact VecSets, sparse structured latents, or octrees for scalable, high-quality generation, including on large, open-category datasets (Zhang et al., 2023; 2024; Tencent Hunyuan3D Team, 2025a; Ren et al., 2024; Xiong et al., 2024; Xiang et al., 2025b;a; Wei et al., 2025). Most of these methods obtain polygonal meshes by decoding an intermediate grid field with algorithms such as

Marching Cubes, Dual Contouring, DMTet, or FlexiCubes (Lorensen & Cline, 1987; Ju et al., 2002; Shen et al., 2021; 2023); field-free structured decoders such as TRELLIS.2 also derive connectivity from spatial grid-incidence rules (Xiang et al., 2025a). Consequently, geometric fidelity does not directly translate into production-ready topology. Triangles are placed according to grid resolution and local extraction cases, typically producing dense, irregular tessellations rather than the sparse edge flow and semantically aligned loops found in artist-authored meshes. Feature-aware dual grids preserve sharp creases, but sub-voxel details remain limited by spatial resolution (Xiang et al., 2025a). The extracted meshes therefore offer little direct control over connectivity and commonly require simplification or remeshing before downstream use. Recent topology-aware fields, including TriFlow and Mesh BDF, learn artist-style connectivity as surface signals, with meshes recovered through field-guided extraction (Li et al., 2026a; Song et al., 2026).

2.2 Autoregressive Native-Mesh Generation

Autoregressive methods instead model native mesh elements explicitly. PolyGen established the vertex–face factorization of mesh sequences, and MeshGPT introduced learned geometric tokens for triangle-mesh generation (Nash et al., 2020; Siddiqui et al., 2024). Subsequent methods improve scalability through adjacency-aware traversal, vertex reuse or component decoupling, and hierarchical or patch-based processing including MeshAnything V2 and BPT (Chen et al., 2025b;c; Tang et al., 2025; Weng et al., 2025b; Lionar et al., 2025; Xu et al., 2025; Lin et al., 2026; Kim et al., 2026; Hao et al., 2024; Zhao et al., 2025). FACE combines latent diffusion with an autoregressive decoder that represents each triangle as one token (Wang et al., 2026b). These methods directly produce compact meshes with artist-style tessellation, but all retain an ordered autoregressive stage. Within that stage, each token depends on its predecessors, precluding fully parallel decoding and causing latency to grow with the number of causal steps, which typically increases with mesh complexity. Compressed tokenizations reduce these steps but do not remove the dependency; long contexts remain costly, serialization order affects learning, and early errors can influence subsequent predictions.

2.3 Parallel Native-Mesh Generation and Connectivity

Diffusion and flow-matching methods enable parallel native-mesh generation. Vertex-first models, including LATO, LATO.2, PolyFlow, Nexus, and MeshFlow by Li et al., recover faces from predicted vertex connectivity (Zhao et al., 2026; Long et al., 2026; Wang et al., 2026a;c; Li et al., 2026b). Connectivity errors can therefore yield missing or spurious faces; Nexus additionally validates candidate faces using higher-order features. PolyDiff, MeshCraft, and MeshFlow by Sun et al. instead generate face-level representations or triangle soups (Alliegro et al., 2023; He et al., 2025; Sun et al., 2026). These avoid graph-based face recovery, but shared vertices remain implicit; the latter uses denoising and proximity-based merging, which alone does not guarantee edge-manifoldness. Complementary approaches include differentiable face selection in DMesh and DMesh++ (Son et al., 2024; 2025) and cyclic half-edge connectivity in SpaceMesh, which guarantees edge-manifoldness while deriving faces from vertex neighborhoods (Shen et al., 2024). BVHMesh instead directly generates a spatially hierarchical face set with OctBVH; EdgeMatch stitches its existing half-edges through partial one-to-one matching. Hard checks preserve this face set while preventing non-manifold edges, duplicate indexed faces, and vertex-collapsed faces.

3 Method

BvhMesh generates a triangle mesh in three stages, illustrated in Fig. 2. Two conditional flow-matching models first generate sparse voxels and structured latents (Section 3.4). A hierarchical VAE then decodes latents into an octree-aligned bounding volume hierarchy (OctBVH) whose parent nodes carry bounding boxes and finest-level nodes carry explicit triangles (Section 3.2). Finally, EdgeMatch pairs the half-edges of the resulting triangle soup and merges each matched pair under topological constraints (Section 3.3).

3.1 Face-First Mesh Generation

Recent flow-based mesh generators (Li et al., 2026b; Zhao et al., 2026; Long et al., 2026; Wang et al., 2026a;c) adopt a *vertex-first* pipeline: they generate vertices and topology features, then

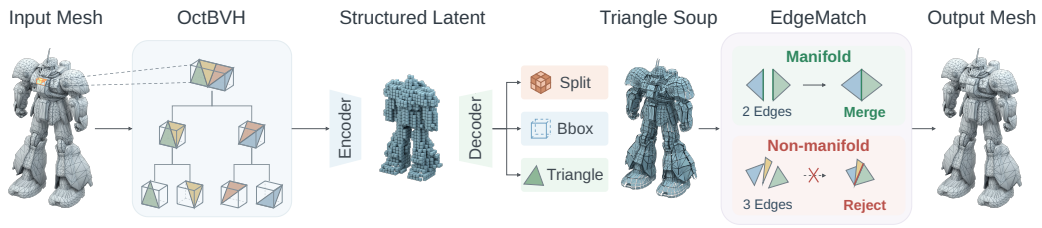

Figure 2: Overview of BVHMESH VAE. The input mesh is represented by octree-aligned bounding volume hierarchy (OctBVH) and encoded into structured latents; the decoder predicts the octree, bounding boxes, and leaf triangles to produce triangle soups, which EdgeMatch stitches through learned one-to-one half-edge matching and topology-constrained merging into the output mesh.

predict pairwise connectivity and recover faces. Variable vertex degrees make connectivity prediction a challenging many-to-many matching problem, and recovering faces from the predicted graph cannot explicitly constrain topology validity. A missing edge can prevent adjacent faces from being recovered, while a spurious edge can form additional triangles with vertices connected to both endpoints, potentially leading to holes or redundant surface coverage.

BVHMESH instead adopts a *face-first* pipeline: it generates explicit triangles, pairs half-edges, and merges corresponding corners into shared vertices. Each predicted triangle already specifies its internal connectivity, leaving the shared-edge relationships between faces to be recovered. Whereas a vertex can have a variable number of neighbors, each half-edge in a manifold mesh has at most one partner. Connectivity recovery thus reduces to a partial one-to-one matching problem, followed by topology checks rejecting merges that would create collapsed or duplicate faces or non-manifold edges. Although errors in geometry and connectivity remain possible, this formulation simplifies connectivity inference by restricting it to constrained matching between half-edges.

Figure 3 shows that vertex-first methods spuriously fill the openings in thin structures such as the Ferris wheel and lattice sphere, whereas BVHMESH preserves these openings and produces a face layout close to the ground truth. We also enlarge the highlighted artifacts and compare redundant area, unsupported area, and non-manifold edge percentage, averaged over the held-out Toys4K dataset. BVHMESH achieves zero non-manifold edges and reduces redundant area in VAE reconstruction to 0.72%, compared with 10.08% for MeshFlow and 19.08% for LATO.

3.2 OCTREE-ALIGNED BVH REPRESENTATION

Predicting triangle coordinates globally requires modeling both overall placement and local shape. We utilize nested BVH boxes to provide progressively finer local coordinate frames, decomposing geometry prediction across scales.

OctBVH construction. We align the BVH with an octree to support octree-based convolutions and attention on its otherwise irregular hierarchy. Given a mesh normalized to $[-1, 1]^3$, we build an octree over triangle centroids. The tree is dense through depth D_{full}; beyond this depth, only occupied cells are retained until $D_{\max}$. Each occupied node at depths D_{bbox} through $D_{\max}$ stores an axis-aligned bounding box enclosing all triangles in its subtree, and each occupied leaf additionally stores one triangle. The octree defines the parent–child structure, while the bounding boxes follow the triangle extents and may extend beyond their corresponding octree cells. At the fine resolution of $2^{D_{\max}}$ cells per axis, nearly all occupied leaves contain a single triangle centroid.

Hierarchical coordinates. Directly regressing continuous coordinates for both bounding boxes and triangles leads to error accumulation through the hierarchy. We combine discrete bounding boxes with continuous triangle coordinates to simplify prediction at each level while preserving fine geometric detail. As illustrated in Fig. 4(a), we quantize each child box within its parent frame using Q bins per axis, rounding lower bounds down and upper bounds up to preserve enclosure of the subtree geometry. At the finest level, triangle vertices use continuous coordinates within their leaf boxes, allowing direct geometric supervision such as normal, edge and area losses.

Hierarchical OctBVH VAE. Our VAE reconstructs the OctBVH hierarchy and its geometric attributes using OCNN and OctFormer blocks (Wang et al., 2017; Wang, 2023). We follow the

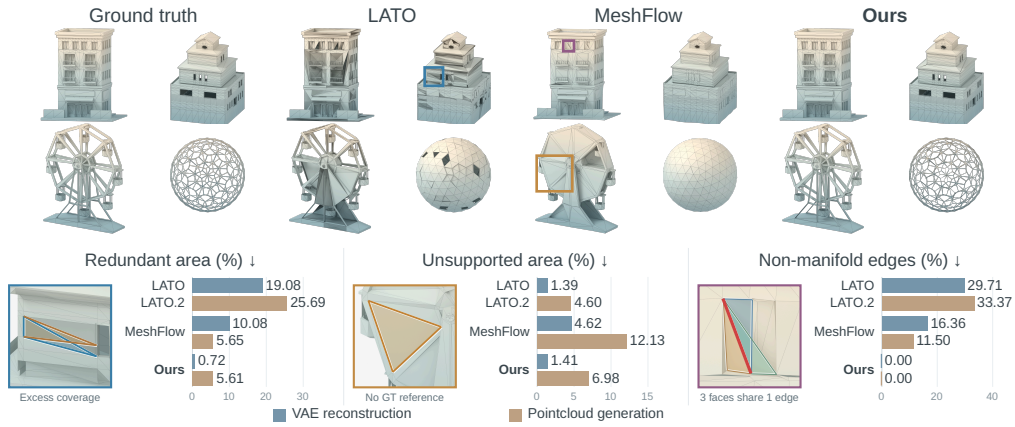

Figure 3: Comparison of face-first and vertex-first methods. Top: VAE reconstructions of held-out examples, with colored boxes marking redundant surfaces, unsupported surfaces, and non-manifold edges. Bottom: enlarged views of the artifact regions and the corresponding quantitative metrics, averaged over the held-out Toys4K dataset. LATO is evaluated only for VAE reconstruction, and LATO.2 only for pointcloud-conditioned generation.

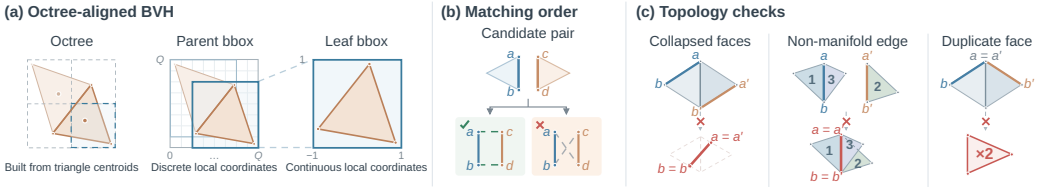

Figure 4: Details of the OctBVH representation and EdgeMatch. (a) An octree built over triangle centroids (left, shown in 2D), quantized bounding-box coordinates in the parent frame (middle), and continuous triangle coordinates in the leaf frame (right). (b) Two scored endpoint correspondences; the check and cross mark the selected and unselected choices. (c) Rejected merges including collapse faces, non-manifold edges, or duplicate faces.

spatial–channel residual shortcuts of DC-AE and TRELLIS.2 (Chen et al., 2025a; Xiang et al., 2025a) to ease optimization under spatial compression. The encoder compresses the OctBVH into structured latents on occupied nodes at 64 resolution. The decoder predicts the eight child-occupancy bits as a 256-way split class, quantized box coordinates with a causal MLP (Lionar et al., 2025), and continuous triangle coordinates with a linear head. We supervise split and box predictions with cross-entropy, and triangle geometry with coordinate, normal, edge-length, and area losses. The full VAE training objective combines the reconstruction terms with a KL regularizer:

$$\mathcal{L} = \mathcal{L}_{\text{split}} + \mathcal{L}_{\text{bbox}} + \mathcal{L}_{\text{tri}} + \lambda_{\text{kl}}\mathcal{L}_{\text{KL}}.$$

3.3 EdgeMatch: Topology-Constrained Stitching

Recovering connectivity from the decoded triangle soups amounts to a partial one-to-one matching among half-edges and merging the corresponding endpoints. Stitching solely based on geometric distances and thresholding can incorrectly merge nearby but separate surfaces. EdgeMatch instead uses a learnable matching head to predict connections and topology checks to reject invalid merges.

Denoising and Matching heads. An OctBVH-based network encodes the input soup once to extract a feature $h_c \in \mathbb{R}^C$ for each triangle corner c, shared by the denoising and matching heads. The denoising head predicts coordinate offsets to refine the triangle geometry before matching. We retrieve candidate half-edge pairs by KNN search on the denoised edge midpoints. A candidate pair of half-edges (a, b) and (c, d) admits two endpoint correspondences (Fig. 4(b)), and we score each correspondence separately with the same network:

$$s_{\parallel} = F_{\text{match}}(h_a, h_c;\, h_b, h_d), \qquad s_{\times} = F_{\text{match}}(h_a, h_d;\, h_b, h_c).$$

Table 1: Pointcloud-conditioned generation on Toys4K. Best results in blue.

Method	CD ↓	HD ↓	NC ↑
MeshAnythingV2	72.68	344.2	0.745
BPT	17.72	112.6	0.902
TreeMeshGPT	18.93	124.2	0.884
DeepMesh	30.47	169.4	0.855
MeshRipple	22.68	144.7	0.875
FastMesh	30.31	173.2	0.830
MeshFlow	11.17	72.6	0.893
LATO.2	9.02	69.7	0.900
Ours	8.29	64.7	0.941

Table 2: VAE reconstruction on Toys4K. Best results in blue.

Method	CD ↓	HD ↓	NC ↑
MeshGPT	28.91	163.2	0.882
TRELLIS	8.55	77.0	0.938
PivotMesh	12.32	56.3	0.901
MeshFlow	8.14	40.5	0.940
LATO	5.66	53.5	0.965
Ours	4.75	25.8	0.972

Here F_{match} is a small neural network that maps the four endpoint features to a score in $[0, 1]$, measuring half-edge compatibility under the specified endpoint correspondence. We use $\max(s_{\parallel}, s_{\times})$ as the pair confidence and retain the higher-scoring correspondence for endpoint merging.

Topology-constrained matching. After candidate filtering, Blossom (Edmonds, 1965) provides initial one-to-one matching proposals. Each proposal is accepted only if merging both endpoint pairs in the current mesh creates no collapsed triangles, duplicate faces, or non-manifold edges (Fig. 4(c)). After matching, each group of merged corners becomes a shared vertex at the mean of their denoised coordinates, and unmatched half-edges remain as boundary edges.

3.4 Generation Pipeline

We follow the two-stage conditional flow-matching architecture of TRELLIS (Xiang et al., 2025b) and fine-tune the models on our mesh dataset. Given an image or pointcloud condition, the first stage generates a sparse voxel structure occupied by triangle centroids, and the second generates OctBVH latents at occupied voxels. The OctBVH VAE decodes these latents into triangles, which EdgeMatch stitches into the output mesh. We use the Hunyuan3D-Omni pointcloud encoder (Tencent Hunyuan3D Team, 2025b) for pointcloud conditioning and DINOv3 (Siméoni et al., 2025) for image conditioning.

4 Experiments

4.1 Datasets

We train all our models on nearly 175K meshes assembled from Objaverse-XL (Deitke et al., 2023), ABO (Collins et al., 2022), HSSD (Khanna et al., 2024), and 3D-FUTURE (Fu et al., 2021). Following prior work (Zhao et al., 2026; Wang et al., 2026a), we evaluate on Toys4K (Stojanov et al., 2021), which is unseen during training. For both training and evaluation, we retain meshes with 100–20,000 faces and at most 5% of triangles exceeding the *one-triangle-per-leaf* budget (Section 3.2).

4.2 VAE Reconstruction

We evaluate mesh reconstruction on Toys4K using our OctBVH VAE followed by EdgeMatch. We report Chamfer Distance (CD, L_1), sampled Hausdorff Distance (HD), and Normal Consistency (NC) , with all CD and HD values scaled by 10^3 throughout the paper. As shown in Table 2, our method achieves the best results on all three geometric metrics. Relative to the strongest baseline for each metric, we reduce CD by 16.1% over LATO and HD by 36.3% over MeshFlow. The qualitative comparison and topology analysis in Section 3.1 further highlight the advantages of our face-first formulation in reducing redundant surface coverage and preserving thin structures.

4.3 Pointcloud-Conditioned Generation

In Table 1, our method achieves the best results on all three metrics, reducing CD by 8.1% and HD by 7.2% relative to LATO.2, the strongest baseline on both distance metrics. Figure 5 highlights

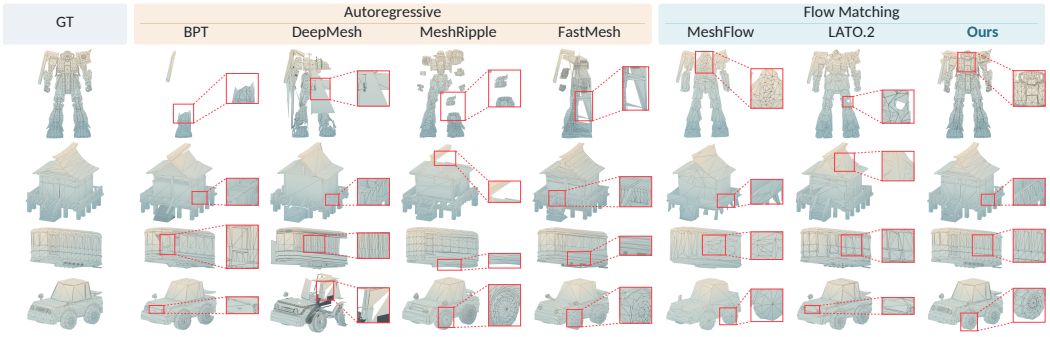

Figure 5: Qualitative comparison of pointcloud-conditioned generation. Insets enlarge regions to show local geometry and connectivity.

Table 3: Quantitative comparison of image-conditioned generation on Toys4K. #Faces is the mean face count per mesh. Best results among artist-mesh outputs are highlighted in light blue; the field-based rows are references rather than competitors.

Method	Geometry			Image agreement		FID ↓	#Faces
	CD ↓	HD ↓	NC ↑	ULIP ↑	Uni3D ↑		
Field-based image-to-3D							
TRELLIS	58.12	278.4	0.759	0.2977	0.2387	9.79	551,609
TRELLIS.2	52.86	252.7	0.776	0.3124	0.2678	8.05	6,605,189
TRELLIS dense mesh → artist mesh							
MeshFlow	60.36	282.1	0.751	0.2820	0.2284	24.57	1,265
BPT	108.50	449.1	0.683	0.2369	0.1935	24.11	2,106
Native topology generation							
LATO.2	58.74	293.2	0.738	0.2647	0.1762	27.66	8,626
Ours	53.54	263.8	0.761	0.2921	0.2314	22.89	5,965

differences in fine geometric detail and local connectivity. MeshFlow simplifies the robot's torso, the building's railings, and the vehicle's front grille. BPT and MeshRipple omit substantial parts of the robot, while DeepMesh leaves the vehicle's cabin incomplete. Our results retain more of these structures and local details.

4.4 Image-Conditioned Generation

In addition to CD, HD, and NC, we report ULIP-Sim and Uni3D-Sim for image–shape agreement and normal-map FID for rendered appearance. Table 3 compares field-based generators, cascades that feed TRELLIS-generated point clouds into artist-mesh generators, and native image-conditioned topology generators.

Among the artist-mesh generators, ours is best on every metric, ahead of both the cascaded methods (MeshFlow, BPT) and the native method LATO.2. In particular, our image–shape agreement improves over the native vertex-first baseline LATO.2 by 10.4% in ULIP-Sim and 31.3% in Uni3D-Sim. Our geometric fidelity is comparable to TRELLIS and TRELLIS.2, with image–shape agreement close to TRELLIS, while using only $5{,}965$ faces per mesh on average, roughly $92\times$ and $1{,}100\times$ fewer, respectively.

Figure 6 shows the qualitative comparison. The TRELLIS column is included for reference only: its tessellation is so dense that the wireframe is nearly opaque. MeshFlow produces a jagged chair-seat boundary and simplified wheel geometry, and LATO.2 produces irregular edge patterns around the chair spindles and hamburger layers. BPT produces incomplete structures including a broken chair spindle, fragmented hamburger layers, and an open bottle boundary. Our results preserve the chair spindles and wheel-rim structure while producing more coherent hamburger layers and more regular edge layouts on the bottle.

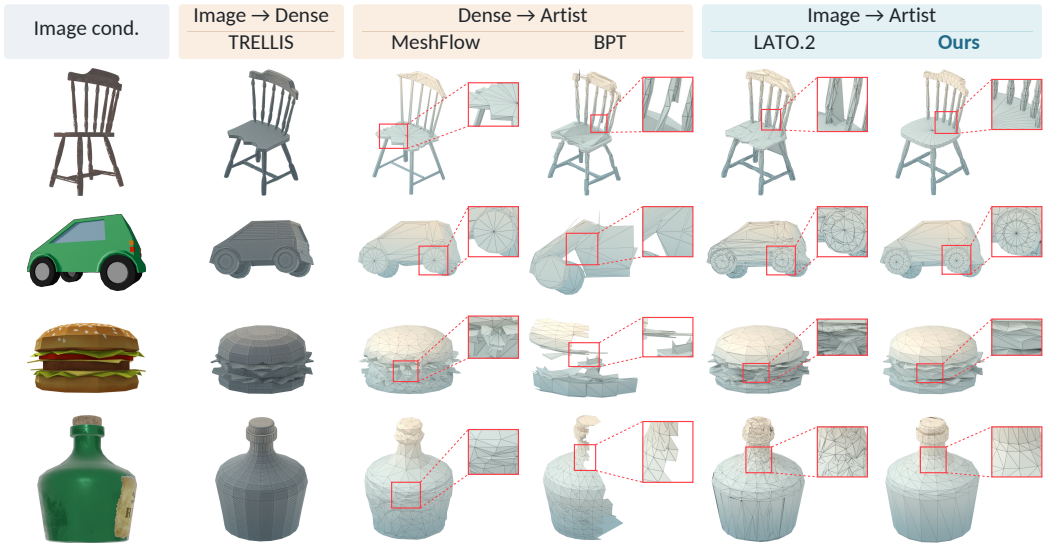

Figure 6: Qualitative comparison of image-conditioned generation. Insets enlarge regions to show local geometry and connectivity.

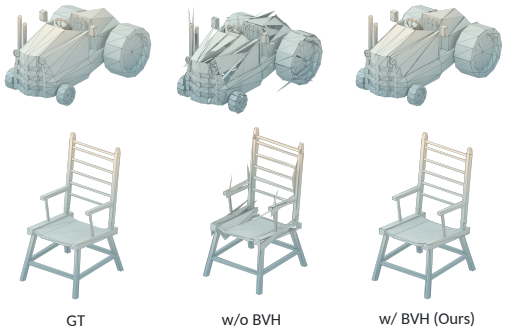

Figure 7: Ablation of OctBVH on VAE reconstruction. Left: ground truth. Middle: w/o OctBVH with discrete global triangle coordinates. Right: our full OctBVH representation.

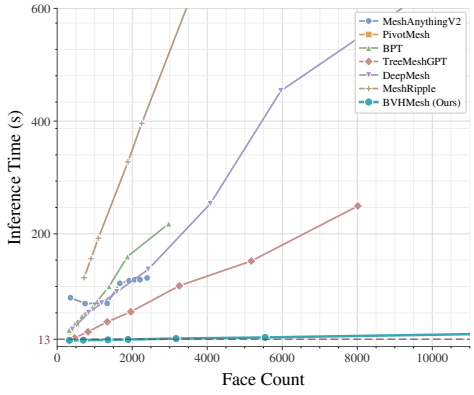

Figure 8: Inference time comparison with autoregressive methods across face counts.

4.5 Ablation Studies

Octree-aligned BVH. Table 4 evaluates two design choices: hierarchical local coordinates and the combination of discrete boxes with continuous triangles. All variants use the same VAE backbone and are evaluated on the decoded triangle soups before EdgeMatch. The variants without OctBVH retain the octree structure but predict global triangle coordinates at leaf nodes; OctBVH instead predicts boxes relative to their parents and triangles within leaf boxes. For discrete triangle coordinates, we use a quantization resolution of 512 in the global frame and 64 in the local leaf-box frame.

Introducing OctBVH improves all three reconstruction metrics over direct global-coordinate prediction. Discretizing bounding boxes further reduces CD from 23.41 to 4.95, supporting the combination of hierarchical local frames and discrete box prediction to simplify learning and mitigate error accumulation across levels. With discrete boxes, replacing discrete triangle coordinates with continuous coordinates enables geometric supervision through normal, edge-length, and area losses and reduces CD from 17.04 to 4.95. Figure 7 illustrates the resulting geometric fidelity: our model reconstructs the tractor's body and the chair's slender supports with substantially fewer surface breaks and spurious spikes than the variant without OctBVH.

EdgeMatch components. Geometric metrics such as CD do not fully capture connectivity errors, and VAE outputs do not retain ground-truth half-edge correspondences. We therefore perturb

Table 5: Ablation of EdgeMatch components. P/R/F1 are reported as percentages. Best results are highlighted in blue.

Variant	$\sigma = 0.004$						$\sigma = 0.012$						$\sigma = 0.020$					
	CD↓	NC↑	HD↓	P↑	R↑	F1↑	CD↓	NC↑	HD↓	P↑	R↑	F1↑	CD↓	NC↑	HD↓	P↑	R↑	F1↑
Baseline	34.55	0.888	242.0	37.4	99.8	47.8	18.80	0.878	126.6	38.1	43.4	33.6	16.20	0.817	88.2	27.1	6.6	8.0
+ One-to-one match	4.42	0.967	23.4	88.6	88.8	88.7	8.36	0.905	46.9	58.6	57.2	57.9	13.80	0.846	69.6	40.1	30.1	34.3
+ Topology	4.36	0.970	17.2	91.5	86.9	89.0	7.59	0.907	36.6	63.4	54.0	58.2	13.44	0.832	68.4	43.3	27.9	33.9
+ Blossom	4.33	0.970	17.0	92.0	88.7	90.2	7.33	0.910	35.4	63.8	56.3	59.7	13.41	0.834	68.1	43.1	28.5	34.3
+ Matching head	4.17	0.974	15.8	96.6	94.9	95.7	6.09	0.939	25.5	81.4	76.7	78.9	11.64	0.875	62.4	66.1	51.5	57.8
+ Denoising head	4.16	0.976	15.9	96.0	94.1	95.0	5.34	0.961	21.9	81.4	75.1	78.0	7.33	0.941	33.1	66.3	57.1	61.2

ground-truth triangle soups while preserving these correspondences to evaluate matching precision and recall directly. Table 5 cumulatively enables EdgeMatch components on the same 200 Toys4K ground-truth meshes at different noise levels. *Baseline* matches half-edges by thresholding geometric distance. For each half-edge, it retrieves $k = 20$ candidates and scores each pair by the sum of Euclidean distances between corresponding endpoints, choosing the endpoint correspondence that minimizes this sum. All pairs with scores below a threshold are welded. + *One-to-one match* accepts pairs greedily in ascending order of the same distance scores, with each half-edge used at most once. + *Topology* adds topology checks before merging, while retaining greedy pairing. + *Blossom* replaces greedy initialization with global one-to-one matching over candidate half-edge pairs. + *Matching head* replaces distance-based scoring with learnable corner embeddings and F_{match} to obtain more accurate matching scores. + *Denoising head* additionally predicts corner offsets, using the refined coordinates for candidate retrieval and averaging merged corners into shared vertices.

The baseline's high recall at low noise is misleading: it accepts many incorrect pairs. One-to-one matching and topology checks reject invalid merges and improve matching precision. The matching head further improves both precision and recall, indicating that learned compatibility better identifies true half-edge partners than geometric distance. The denoising head refines corner positions and is most beneficial at high noise.

Table 4: Ablation of OctBVH representation and coordinate quantization. All metrics are evaluated on triangle soups before EdgeMatch.

Representation	Box coords.	Triangle coords.	CD ↓	HD ↓	NC ↑
w/o OctBVH	–	Discrete	42.94	268.0	0.828
w/o OctBVH	–	Continuous	35.27	235.8	0.706
w/ OctBVH	Continuous	Continuous	23.41	132.5	0.807
w/ OctBVH	Discrete	Discrete	17.04	126.1	0.871
w/ OctBVH (Ours)	Discrete	Continuous	4.95	29.7	0.971

4.6 Inference Efficiency

BVHMesh avoids serial face-by-face decoding and uses a fixed number of flow steps. As shown in Fig. 8, autoregressive inference time increases rapidly with mesh resolution. In contrast, BVHMesh remains nearly constant at around 15 seconds per mesh throughout the tested range, including meshes with more than 10,000 faces. All inference times are measured on an NVIDIA 5090 GPU with batch size 1, covering the complete pipeline from the conditioning input to the final mesh.

5 Conclusion

We presented BVHMesh , a face-first framework for efficient artist-mesh generation conditioned on point clouds or images. OctBVH simplifies triangle prediction through nested local coordinate frames and hierarchical quantization, preserving fine geometry while enabling sparse convolutions and transformers through octree alignment. EdgeMatch simplifies connectivity recovery into learned one-to-one half-edge matching, with explicit topology checks that reject merges creating non-manifold edges, duplicate faces, or collapsed triangles. Experiments demonstrate state-of-the-art reconstruction and conditional artist-mesh generation, with better preservation of thin structures and openings and reduced redundant surface coverage. Parallel triangle decoding keeps inference efficient at approximately 15 seconds per mesh, even for outputs exceeding 10,000 faces.

AI USE STATEMENT

We used GPT-6 to assist with drafting and polishing the manuscript. The authors take full responsibility for all content of this work, including AI-assisted text, claims, code, and data.

REPRODUCIBILITY STATEMENT

The main paper and appendix document our method and experimental setup to support reproducibility. We will release our code and model weights upon acceptance of the paper. Section 3 describes the representation, model architecture, and training objectives, and Section 4.1 specifies the data sources and mesh filtering criteria. Appendices A and B provide additional implementation details, optimization settings, training resources, and inference configurations. Appendix C defines the evaluation metrics and details the surface sampling and rendering protocols.

REFERENCES

Antonio Alliegro, Yawar Siddiqui, Tatiana Tommasi, and Matthias Nießner. PolyDiff: Generating 3D polygonal meshes with diffusion models. *arXiv preprint arXiv:2312.11417*, 2023.

Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. In *ICLR*, 2025a.

Yiwen Chen, Tong He, Di Huang, Weicai Ye, Sijin Chen, Jiaxiang Tang, Zhongang Cai, Lei Yang, Gang Yu, Guosheng Lin, and Chi Zhang. MeshAnything: Artist-created mesh generation with autoregressive transformers. In *ICLR*, 2025b.

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 *ICCV*, 2025c.

Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G. Schwing, and Liang-Yan Gui. SDFusion: Multimodal 3D shape completion, reconstruction, and generation. In *CVPR*, 2023.

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal ConvNets: Minkowski convolutional neural networks. In *CVPR*, 2019.

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F. Yago Vicente, Thomas Dideriksen, Himanshu Arora, Matthieu Guillaumin, and Jitendra Malik. ABO: Dataset and benchmarks for real-world 3D object understanding. In *CVPR*, 2022.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-XL: A universe of 10M+ 3D objects. In *NeurIPS*, 2023.

Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. doi: 10.4153/CJM-1965-045-4.

Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3D-FUTURE: 3D furniture shape with TextURE. *Int. J. Comput. Vis.*, 129:3313–3337, 2021.

Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018.

Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3DGen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023.

Zekun Hao, David W. Romero, Tsung-Yi Lin, and Ming-Yu Liu. Meshtron: High-fidelity, artist-like 3D mesh generation at scale. *arXiv preprint arXiv:2412.09548*, 2024.

Xianglong He, Junyi Chen, Di Huang, Zexiang Liu, Xiaoshui Huang, Wanli Ouyang, Chun Yuan, and Yangguang Li. MeshCraft: Exploring efficient and controllable mesh generation with flow-based DiTs. *arXiv preprint arXiv:2503.23022*, 2025.

Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of Hermite data. *ACM Trans. Graph. (SIGGRAPH)*, 21(3):339–346, 2002.

Heewoo Jun and Alex Nichol. Shap-E: Generating conditional 3D implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

Kenney. Kenney Assets, n.d. URL `https://kenney.nl/assets`. Accessed: 2026-09-26.

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 *CVPR*, pp. 16384–16393, 2024.

Jeonghwan Kim, Yushi Lan, Armando Fortes, Yongwei Chen, and Xingang Pan. FastMesh: Efficient artistic mesh generation via component decoupling. In *3DV*, 2026.

Haoxuan Li, Ziya Erkoç, Daniele Sirigatti, Vladislav Rosov, Lei Li, Angela Dai, and Matthias Nießner. TriFlow: Generating artist-like 3D mesh topology via nearest-vertex vector fields. *arXiv preprint arXiv:2606.20131*, 2026a.

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 *CVPR*, 2026b. 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 *CVPR*, pp. 12706–12718, 2026.

Stefan Lionar, Jiabin Liang, and Gim Hee Lee. TreeMeshGPT: Artistic mesh generation with autoregressive tree sequencing. In *CVPR*, 2025.

Hang Long, Tianhao Zhao, Junkai Lin, Youjia Zhang, Huipeng Guo, Rendong Liang, Jiale Xu, Jozef Hladký, Matthias Nießner, Yuanming Hu, and Wei Yang. LATO.2: Factorized 3D mesh generation with vertex and topology flow. *arXiv preprint arXiv:2607.10623*, 2026.

William E. Lorensen and Harvey E. Cline. Marching Cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH*, 1987.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019.

Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. PolyGen: An autoregressive generative model of 3D meshes. In *ICML*, 2020.

OSM2World contributors. OSM2World, n.d. URL `https://www.osm2world.org/`. Accessed: 2026-09-26.

Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020.

Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. XCube: Large-scale 3D generative modeling using sparse voxel hierarchies. In *CVPR*, 2024.

Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: A hybrid representation for high-resolution 3D shape synthesis. In *NeurIPS*, 2021.

Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. Flexible isosurface extraction for gradient-based mesh optimization. *ACM Trans. Graph. (SIGGRAPH)*, 42(4), 2023. doi: 10.1145/3592430.

Tianchang Shen, Zhaoshuo Li, Marc Law, Matan Atzmon, Sanja Fidler, James Lucas, Jun Gao, and Nicholas Sharp. SpaceMesh: A continuous representation for learning manifold surface meshes. In *SIGGRAPH Asia 2024 Conference Papers*, 2024. doi: 10.1145/3680528.3687634.

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 *CVPR*, 2024.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Sanghyun Son, Matheus Gadelha, Yang Zhou, Zexiang Xu, Ming C. Lin, and Yi Zhou. DMesh: A differentiable mesh representation. In *NeurIPS*, 2024.

Sanghyun Son, Matheus Gadelha, Yang Zhou, Matthew Fisher, Zexiang Xu, Yi-Ling Qiao, Ming C. Lin, and Yi Zhou. DMesh++: An efficient differentiable mesh for complex shapes. In *ICCV*, 2025.

Gaochao Song, Haohan Weng, Luo Zhang, Zibo Zhao, and Shenghua Gao. Mesh BDF: Barycentric dominance field for 3D native mesh generation. *arXiv preprint arXiv:2606.31777*, 2026.

Stefan Stojanov, Anh Thai, and James M. Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *CVPR*, 2021.

Qi Sun, Kiyohiro Nakayama, Jing Nathan Yan, Qixing Huang, Alexander Rush, Leonidas Guibas, Gordon Wetzstein, Jing Liao, and Guandao Yang. MeshFlow: Mesh generation with equivariant flow matching. In *ACM SIGGRAPH Conference Papers*, 2026.

Jiaxiang Tang, Zhaoshuo Li, Zekun Hao, Xian Liu, Gang Zeng, Ming-Yu Liu, and Qinsheng Zhang. EdgeRunner: Auto-regressive auto-encoder for artistic mesh generation. In *ICLR*, 2025.

Tencent Hunyuan3D Team. Hunyuan3D 2.0: Scaling diffusion models for high resolution textured 3D assets generation. *arXiv preprint arXiv:2501.12202*, 2025a.

Tencent Hunyuan3D Team. Hunyuan3D-Omni: A unified framework for controllable generation of 3D assets. *arXiv preprint arXiv:2509.21245*, 2025b.

Chunshi Wang, Haohan Weng, Junliang Ye, Biwen Lei, Yang Li, Zibo Zhao, Zeqiang Lai, Kaiyi Zhang, Yunhan Yang, Zhuo Chen, Chunchao Guo, and Yawei Luo. PolyFlow: Continuous topology embedding flow matching for artist-style mesh generation. *arXiv preprint arXiv:2606.30673*, 2026a.

Hanxiao Wang, Yuan-Chen Guo, Ying-Tian Liu, Zi-Xin Zou, Biao Zhang, Weize Quan, Ding Liang, Yan-Pei Cao, and Dong-Ming Yan. FACE: A face-based autoregressive representation for high-fidelity and efficient mesh generation. *arXiv preprint arXiv:2603.01515*, 2026b.

Hanxiao Wang, Ying-Tian Liu, Yuan-Chen Guo, Qi-Yuan Feng, Zi-Xin Zou, Ding Liang, Biao Zhang, and Yan-Pei Cao. Nexus: Native mesh generation with diffusion. *arXiv preprint arXiv:2607.13563*, 2026c.

Peng-Shuai Wang. OctFormer: Octree-based transformers for 3D point clouds. *ACM Trans. Graph. (SIGGRAPH)*, 42(4), 2023.

Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: Octree-based convolutional neural networks for 3D shape analysis. *ACM Trans. Graph. (SIGGRAPH)*, 36(4), 2017.

Si-Tong Wei, Rui-Huan Wang, Chuan-Zhi Zhou, Baoquan Chen, and Peng-Shuai Wang. OctGPT: Octree-based multiscale autoregressive models for 3D shape generation. *arXiv preprint arXiv:2504.09975*, 2025.

Haohan Weng, Yikai Wang, Tong Zhang, C. L. Philip Chen, and Jun Zhu. PivotMesh: Generic 3D mesh generation via pivot vertices guidance. In *ICLR*, 2025a.

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 *CVPR*, 2025b.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3D generation. *arXiv preprint arXiv:2512.14692*, 2025a.

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 *CVPR*, 2025b.

Bojun Xiong, Si-Tong Wei, Xin-Yang Zheng, Yan-Pei Cao, Zhouhui Lian, and Peng-Shuai Wang. OctFusion: Octree-based diffusion models for 3D shape generation. *arXiv preprint arXiv:2408.14732*, 2024.

Rui Xu, Tianyang Xue, Qiujie Dong, Le Wan, Zhe Zhu, Peng Li, Zhiyang Dou, Cheng Lin, Shiqing Xin, Yuan Liu, Wenping Wang, and Taku Komura. MeshMosaic: Scaling artist mesh generation via local-to-global assembly. *arXiv preprint arXiv:2509.19995*, 2025.

Rui Xu, Dafei Qin, Kaichun Qiao, Qiujie Dong, Huaijin Pi, Qixuan Zhang, Longwen Zhang, Lan Xu, Jingyi Yu, Wenping Wang, and Taku Komura. Strips as tokens: Artist mesh generation with native UV segmentation. *arXiv preprint arXiv:2604.09132*, 2026.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. *ACM Trans. Graph. (SIGGRAPH)*, 2023.

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. CLAY: A controllable large-scale generative model for creating high-quality 3D assets. *ACM Trans. Graph. (SIGGRAPH)*, 43(4), 2024.

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 *ICCV*, 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.

A Additional Method Details

Hierarchical coordinates. We treat each reconstructed parent box as a local coordinate frame, mapping its extent along every axis to $[0, Q]$ with $Q = 64$ cells per axis before encoding its children. For node i, denote the bounds of this parent frame by $\hat{b}^p_{i,\min}$ and $\hat{b}^p_{i,\max}$. At the first bbox depth D_{bbox}, the parent frame is the unit cube $[-1, 1]^3$; at every finer depth, it is the reconstructed box $\hat{b}_{\pi(i)}$ of the parent node. The child box is normalized and discretized within this frame as

$$q_{i,\min} = \left\lfloor Q \frac{b_{i,\min} - \hat{b}^p_{i,\min}}{\hat{b}^p_{i,\max} - \hat{b}^p_{i,\min}} \right\rfloor, \qquad q_{i,\max} = \left\lceil Q \frac{b_{i,\max} - \hat{b}^p_{i,\min}}{\hat{b}^p_{i,\max} - \hat{b}^p_{i,\min}} \right\rceil.$$

When constructing training targets, floor and ceiling snap the minimum and maximum corners outward, respectively, so the reconstructed target box $\hat{b}_i$ conservatively encloses the original geometry. Targets are constructed from coarse to fine, using each reconstructed box $\hat{b}_i$ as its children's frame so their coordinates account for the parent's quantization.

To preserve local detail without further vertex quantization, triangle vertices remain continuous and are represented in their reconstructed leaf box as

$$\tilde{v}_{i,k} = 2 \frac{v_{i,k} - \hat{b}_{i,\min}}{\hat{b}_{i,\max} - \hat{b}_{i,\min}} - 1, \qquad k \in \{1, 2, 3\}.$$

For each axis, if the predicted minimum exceeds the predicted maximum, we swap the two coordinates. Box widths below 10^{-8} are treated as 10^{-8} when converting between local and global coordinates, avoiding division by zero.

OctBVH VAE. We use $(D_{\text{full}}, D_{\text{bbox}}, D_{\max}) = (3, 6, 9)$ to build OctBVH, with a 32-dimensional latent at each occupied node of the depth-6 (64^3) grid and triangle centroids routed at 512^3 resolution. Before constructing the OctBVH, we sort each triangle's corners by their x–y–z coordinates and apply the same permutation to their vertex IDs. When multiple triangle centroids fall in the same leaf, we average the triangles' vertices slot by slot after this sorting to form one representative triangle. Such collisions are very rare due to the high leaf resolution and our dataset filtering.

VAE training losses. *Bounding boxes.* Let h be the VAE's decoder feature and $q = (q^x_{\min}, q^y_{\min}, q^z_{\min}, q^x_{\max}, q^y_{\max}, q^z_{\max})$ the quantized target box, with each coordinate in $\{0, \ldots, Q\}$. The causal MLP predicts these six coordinates in order, conditioning each prediction on h concatenated with learned embeddings of the preceding coordinate tokens. During training, we use teacher forcing to obtain these embeddings from ground-truth coordinates; during inference, we obtain them from previously predicted coordinates. This captures causal dependencies among coordinates within each box while avoiding the cost of global autoregressive generation across boxes. The cross-entropy loss is

$$\mathcal{L}_{\text{bbox}} = -\sum_{j=1}^{6} \log p_\theta(q_j \mid h, q_{<j}).$$

Here, $p_\theta(q_j \mid h, q_{<j})$ is the probability assigned by the causal MLP with parameters θ to the target coordinate token q_j, conditioned on the decoder feature h and preceding coordinate tokens $q_{<j}$.

Triangle coordinates. For the triangle losses below, we write v_k for the ground-truth leaf-local coordinate $\tilde{v}_{i,k}$ defined in Appendix A, omitting the node index, and $\hat{v}_k$ for its prediction, where $k \in \{1, 2, 3\}$. Both are expressed in the same reconstructed ground-truth leaf frame; hats denote predicted quantities throughout these triangle losses. With corner correspondence fixed by preprocessing, the coordinate loss is

$$\mathcal{L}_{\text{mse}} = \sum_{k=1}^{3} \|\hat{v}_k - v_k\|_2^2.$$

Normal loss. Define the unnormalized target normal $n = (v_2 - v_1) \times (v_3 - v_1)$, and define $\hat{n}$ analogously from the predicted corners $\hat{v}_k$. The normal loss is the cosine distance,

$$\mathcal{L}_{\text{normal}} = 1 - \frac{\hat{n}^\top n}{\|\hat{n}\|_2 \|n\|_2}.$$

Edge-length and area losses. Let s_k and $\hat{s}_k$ be the ground-truth and predicted edge lengths, each sorted in ascending order, and let $A = \frac{1}{2}\|n\|_2$ and $\hat{A} = \frac{1}{2}\|\hat{n}\|_2$ be the corresponding areas. Using the smooth L_1 penalty ρ with unit transition threshold, we define

$$\mathcal{L}_{\text{edge}} = \frac{1}{3}\sum_{k=1}^{3}\rho(\hat{s}_k - s_k), \qquad \mathcal{L}_{\text{area}} = \rho(\hat{A} - A).$$

The combined triangle objective is

$$\mathcal{L}_{\text{tri}} = \lambda_{\text{mse}}\mathcal{L}_{\text{mse}} + \lambda_{\text{nrm}}\mathcal{L}_{\text{normal}} + \mathcal{L}_{\text{edge}} + \mathcal{L}_{\text{area}}.$$

KL regularization. The encoder defines a diagonal Gaussian latent posterior $q_\phi(z) = \mathcal{N}(\mu, \text{diag}(\sigma^2))$. We regularize this posterior toward the standard normal prior $p(z) = \mathcal{N}(0, I)$:

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}(q_\phi(z) \,\|\, p(z)) \,.$$

In training, this term is normalized by the latent dimension and averaged over occupied latent nodes.

EdgeMatch. EdgeMatch uses an OctBVH U-Net encoder to extract 64-dimensional corner features shared by the denoising and matching heads. Denoised coordinates are used for candidate retrieval and final vertex averaging. The F_{match} uses the squared feature differences of the two matched endpoint pairs. For the correspondence $(a \leftrightarrow c, b \leftrightarrow d)$, let $u = (h_a - h_c)^2$ and $v = (h_b - h_d)^2$. The scoring function is

$$F_{\text{match}}(h_a, h_c;\, h_b, h_d) = \text{sigmoid}(\text{MLP}(\text{Concat}(u + v,\, |u - v|))) \,.$$

Squares and absolute values are element-wise. The symmetric input makes F_{match} invariant to exchanging the two matched endpoint pairs. Binary cross-entropy supervises the larger logits of endpoint correspondence with the ground-truth half-edge pair label. The denoising head applies $\tanh$ to its output and scales it by 0.05 to obtain corner offsets, trained with an MSE loss. The denoising and matching heads are trained with balanced positive and negative half-edge pairs on ground-truth triangle soups perturbed by Gaussian vertex noise, where the noise level is sampled as $\sigma \sim \mathcal{U}[0.004, 0.020]$ for each shape.

For each half-edge, we retrieve its $k = 20$ nearest neighbours by denoised edge-midpoint distance and retain candidates with confidence above 0.5. Blossom computes a maximum-weight matching on this candidate graph. We first check the pairs selected by Blossom, then the remaining eligible pairs, processing each group from highest to lowest confidence. An incremental union–find structure tests both endpoint merges together against the current vertex groups, rejecting collapsed triangles, non-manifold edges, and duplicate faces before committing a merge. Each half-edge can belong to only one accepted pair; if a pair is rejected, its half-edges can still match other candidates. The final matching may therefore differ from the initial Blossom solution.

B Experimental Details

Dataset curation. We curate our training data from TRELLIS-500K (Xiang et al., 2025b), starting with $450{,}384$ successfully downloaded meshes. We retain meshes with 100–$20{,}000$ faces to focus on compact meshes. We then apply leaf-collision checks to limit locally dense tessellations and triangle aggregation in OctBVH, followed by mesh-quality checks. For the *one-triangle-per-leaf* leaf collision budget, we limit the excess face count and area to 5% of all faces and surface area. Mesh-quality checks require slender-face ratios, the degenerate-face removal ratio, and the non-manifold-edge ratio to be at most 5%, and the vertex-to-face ratio to be at most 1. Table 6 summarizes the cumulative filtering process, which retains $175{,}039$ meshes.

Baselines. We compare VAE reconstruction against MeshGPT (Siddiqui et al., 2024), PivotMesh (Weng et al., 2025a), TRELLIS (Xiang et al., 2025b), LATO (Zhao et al., 2026), and MeshFlow (Li et al., 2026b). For pointcloud-conditioned generation, we compare against autoregressive methods, including MeshAnything V2 (Chen et al., 2025c), PivotMesh (Weng et al., 2025a), BPT (Weng et al., 2025b), TreeMeshGPT (Lionar et al., 2025), DeepMesh (Zhao et al., 2025), MeshRipple (Lin et al., 2026), and FastMesh (Kim et al., 2026), as well as the flow-based MeshFlow (Li et al., 2026b). For image-conditioned generation, we include TRELLIS (Xiang et al.,

Table 6: Training-data filtering. Removed counts meshes excluded at that stage; remaining counts meshes passing all stages up to and including that row.

Filtering	Removed	Remaining
TRELLIS-500K	–	450,384
Face count	161,941	288,443
Leaf collisions	75,075	213,368
Mesh quality	38,329	175,039

2025b) and TRELLIS.2 (Xiang et al., 2025a) as dense-mesh references, cascade baselines that feed point clouds sampled from TRELLIS-generated meshes into MeshFlow (Li et al., 2026b) or BPT (Weng et al., 2025b), and the native image-conditioned topology generator LATO.2 (Long et al., 2026). LATO.2 is concurrent work, and its comparison results are included for reference only.

Conditional flow models. The sparse-structure and structured-latent models follow the conditional flow-matching architecture of TRELLIS, with approximately 1.3B parameters in each denoiser. For pointcloud conditioning, the $40{,}960$ oriented surface samples are encoded into $2{,}048$ conditioning tokens. For image conditioning, we render each training mesh from 16 viewpoints and encode the rendered images with a frozen DINOv3 encoder (Siméoni et al., 2025).

Optimization and inference. All models are optimized with AdamW and a per-GPU batch size of 8. For the OctBVH VAE and EdgeMatch, we use a learning rate of 10^{-4} and weight decay of 10^{-3}; the VAE loss weights are $\lambda_{\mathrm{kl}} = 10^{-6}$, $\lambda_{\mathrm{mse}} = 10$, and $\lambda_{\mathrm{nrm}} = 4$, with unit weights on the split, box, edge, and area terms. The flow models use a learning rate of 10^{-4}, weight decay of 10^{-2}, and condition dropout $p_{\mathrm{uncond}} = 0.1$. The VAE is trained for 1 week on 16 H20 GPUs, EdgeMatch for 3 days on 8 H20 GPUs, and the structure and latent flows for 1 week on 16 and 32 H20 GPUs, respectively. At inference, we use an Euler solver with 25 steps per stage. For pointcloud conditioning, the classifier-free guidance strength is 3.0 in both stages; for image conditioning, it is 2.5 for the structure stage and 3.0 for the latent stage.

C Evaluation Metrics and Protocols

C.1 Geometric Metrics

For geometric evaluation, we sample 10^5 points from each of the predicted and ground-truth mesh surfaces, yielding point sets P and Q, respectively. For each $p \in P$, let $q_p = \arg\min_{q \in Q} \|p - q\|_2$ denote its nearest neighbor in Q, and define p_q analogously for each $q \in Q$. Let $\mathbf{n}_p$ and $\mathbf{n}_q$ denote the corresponding unit surface normals. We report Chamfer Distance (CD), Hausdorff Distance (HD), and Normal Consistency (NC):

$$\mathrm{CD}(P,Q) = \frac{1}{2}\left(\frac{1}{|P|}\sum_{p\in P}\|p-q_p\|_2 + \frac{1}{|Q|}\sum_{q\in Q}\|q-p_q\|_2\right), \tag{1}$$

$$\mathrm{HD}(P,Q) = \max\left\{\max_{p\in P}\|p-q_p\|_2,\ \max_{q\in Q}\|q-p_q\|_2\right\}, \tag{2}$$

$$\mathrm{NC}(P,Q) = \frac{1}{2}\left(\frac{1}{|P|}\sum_{p\in P}\left|\mathbf{n}_p^\top \mathbf{n}_{q_p}\right| + \frac{1}{|Q|}\sum_{q\in Q}\left|\mathbf{n}_q^\top \mathbf{n}_{p_q}\right|\right). \tag{3}$$

CD uses unsquared Euclidean distances, HD is evaluated on the sampled point sets, and NC uses absolute dot products to be invariant to normal orientation. Lower CD and HD and higher NC indicate better geometric agreement.

C.2 Topology and Surface-Area Metrics

Chamfer Distance does not fully characterize connectivity defects or repeated surface coverage, so Fig. 3 additionally reports non-manifold edge percentages and unsupported and redundant area ratios.

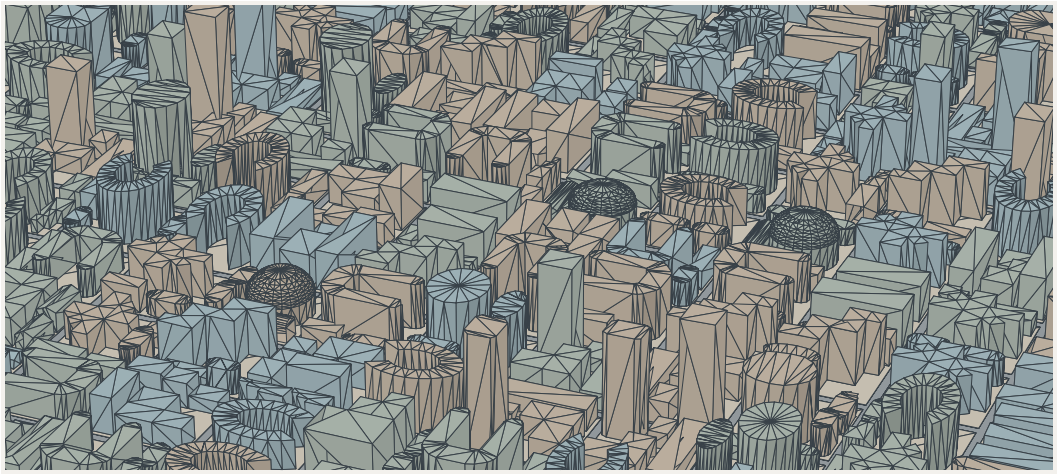

Figure 9: City-level mesh generation from procedural OSM2World scenes.

Non-manifold edge ratio. An edge is *non-manifold* when it is shared by three or more faces; we report the percentage of such edges among all distinct edges.

Unsupported and redundant predicted area. We sample points on each triangle and divide triangle area equally among its samples; overlapping triangles contribute area separately. After normalizing meshes to the unit sphere, predicted and reference samples may match if they are within 0.02 of each other (1% of the sphere diameter). We maximize the total matched area while allowing each sample's area to be used at most once. For example, two predicted triangles covering the same reference patch must share that patch's area budget.

Unsupported is the predicted area with no reference area within this distance. *Redundant* is the predicted area that has nearby reference samples but remains unmatched because their area budget is insufficient. We divide each quantity by the total predicted triangle area and report the resulting percentage, averaged over meshes. Redundant area indicates excess local coverage, and unsupported area indicates predicted surface area without nearby reference support, so lower values are better for both.

C.3 Image-Conditioned Evaluation

Image-agreement metrics. ULIP-Sim and Uni3D-Sim are cosine similarities between an embedding of the generated shape and an embedding of the conditioning image in a joint space, following the protocol of (Wang et al., 2026c). ULIP-Sim uses ULIP-2 PointBERT with a ViT image encoder, and Uni3D-Sim uses Uni3D-g with an EVA-CLIP image encoder. For both, we sample $10{,}000$ surface points from the generated mesh and normalize them to the unit sphere.

Normal-map FID. We render four double-sided normal maps per mesh at 512^2 from an orbit at fixed elevation, extract Inception-v3 features, and compute FID between the ground-truth and generated image sets. Normal-map FID measures rendered appearance under this protocol and does not directly assess mesh connectivity.

D More Results

City- and scene-level mesh generation. Efficient generation at higher face counts also enables BVHMesh to support city- and scene-level content creation through modular generation and assembly. For the city in Fig. 9, we construct procedural source scenes with OSM2World (OSM2World contributors, n.d.), sample a conditioning point cloud for each land parcel, and generate the parcels independently before assembling them into a city. For the scenes in Fig. 10, we use Kenney assets (Kenney, n.d.) as source geometry for point-cloud conditioning and arrange the generated components into a castle and adjoining commercial and suburban neighborhoods. These examples

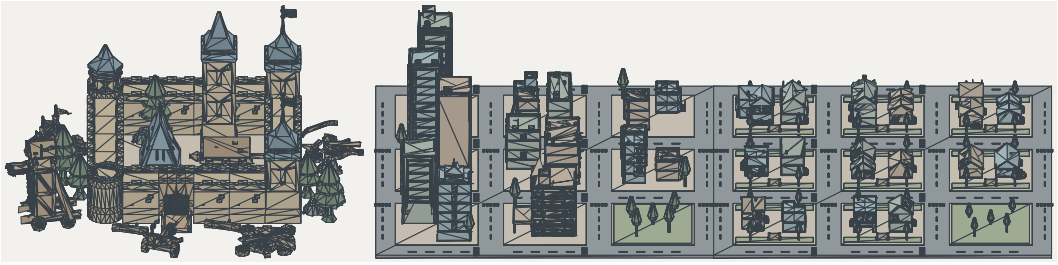

Figure 10: Scene-level mesh generation using Kenney assets for point-cloud conditioning. Generated components form a castle (left) and adjoining commercial and suburban neighborhoods (right).

illustrate how generated meshes can be composed into larger environments with varied building forms and detailed scene elements.

More pointcloud-conditioned comparisons We present more pointcloud-conditioned generation results in Figs. 11 and 12.

More image-conditioned comparisons We present more image-conditioned examples in Fig. 13. MeshFlow and BPT use point clouds sampled from TRELLIS-generated meshes; LATO.2 and our method use image conditioning directly.

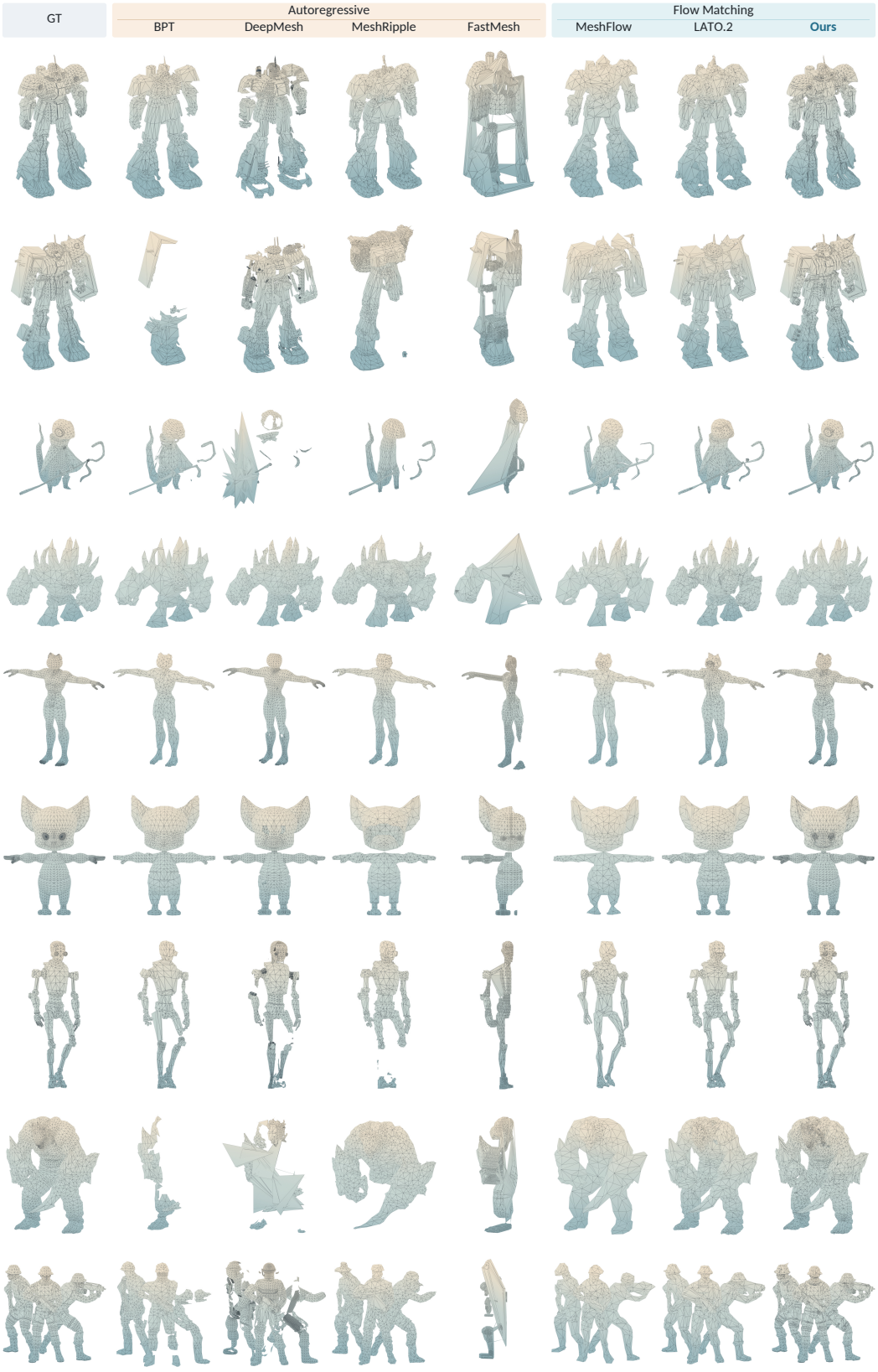

Figure 11: More pointcloud-conditioned generation results.

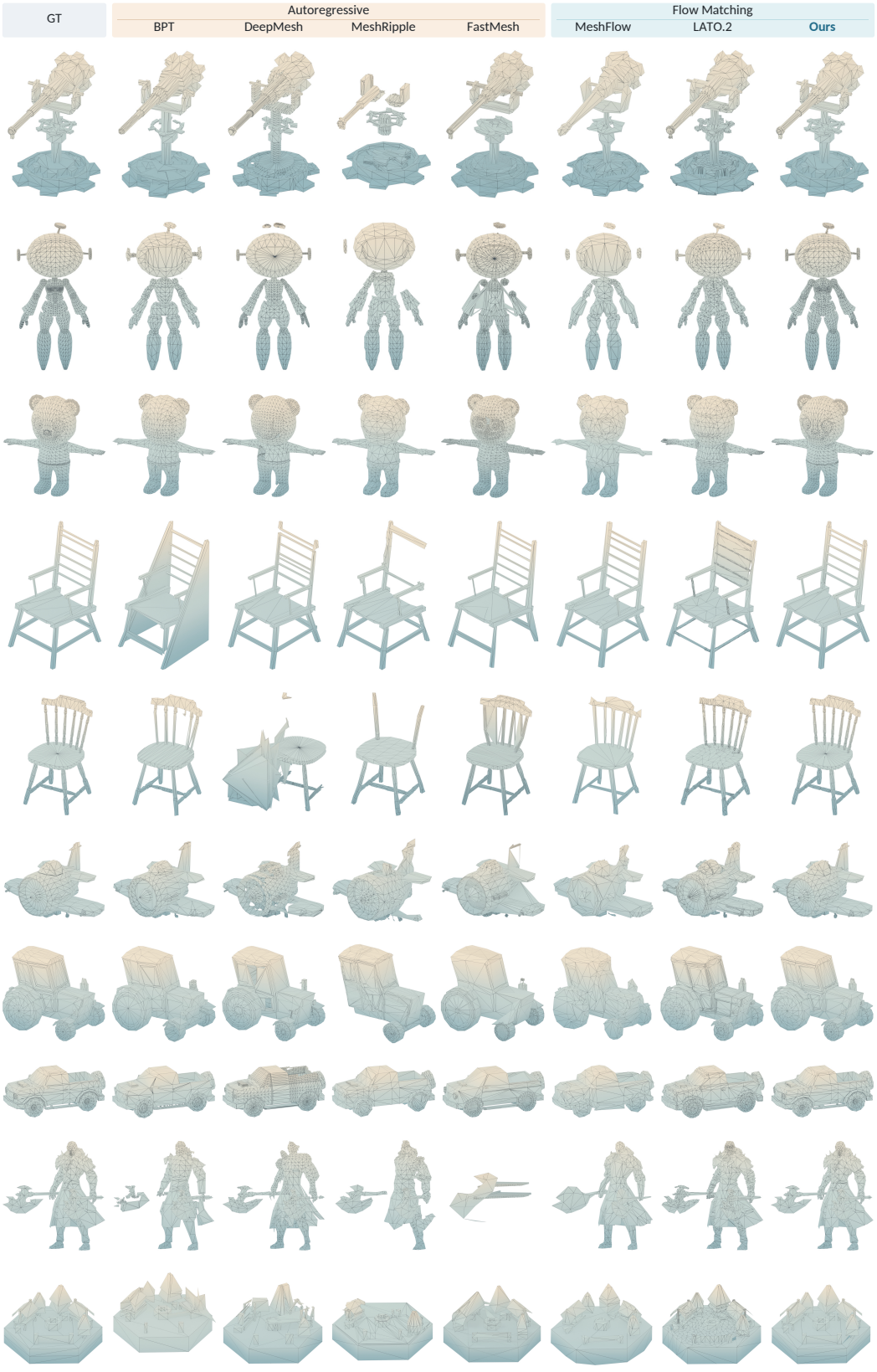

Figure 12: More pointcloud-conditioned generation results.

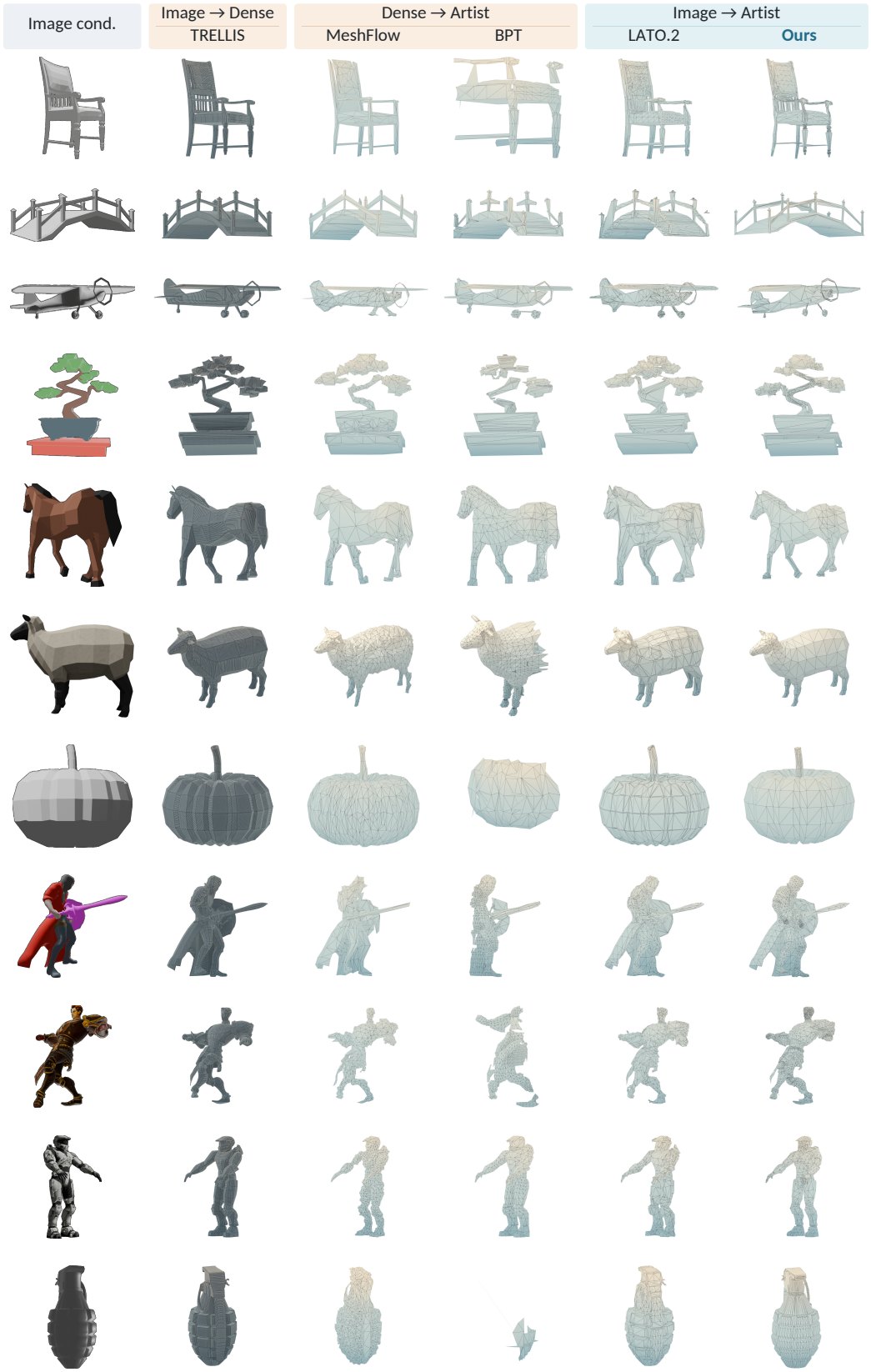

Figure 13: More image-conditioned generation results.