SHAPEMIND: NATIVE 3D UNDERSTANDING AND GENERATION OF SEMANTICALLY STRUCTURED 3D ASSETS

Anonymous authors
Paper under double-blind review



Figure 1: **ShapeMind: think before you generate.** ShapeVLM interprets images and coarse geometry to infer semantic parts, spatial bounds, and applicable kinematics. Semantic PartGen uses the image and inferred part descriptions and bounds to generate separate low-poly meshes. Colors distinguish parts, wireframes reveal mesh topology, and green markers indicate joint axes.

ABSTRACT

Production-ready 3D assets demand more than visual fidelity: they require semantically structured geometry for editing, composition, and interaction in applications such as embodied simulation. We present ShapeMind, a framework that generates low-poly 3D assets with explicit semantic part structure and, where applicable, kinematic structure. Following the principle of *think before you generate*, ShapeMind couples ShapeVLM for 3D semantic understanding with Semantic PartGen for native mesh generation. ShapeVLM combines visual evidence with coarse geometry encoded as compact text within the existing vocabulary of a vision-language model (VLM). This design reuses pretrained semantic and commonsense knowledge for 3D structural reasoning without introducing a heterogeneous modality. The resulting understanding of object composition and kinematics provides the structural basis for Semantic PartGen to synthesize high-quality low-poly meshes organized into meaningful parts. Its part-aware vertex and topology generation jointly models the object while preserving each part as an independently editable mesh. To support this conditional generation task, we curate 300K semantically annotated low-poly assets. Experiments demonstrate state-of-the-art results in both 3D object semantic understanding and structurally coherent part geometry generation.

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

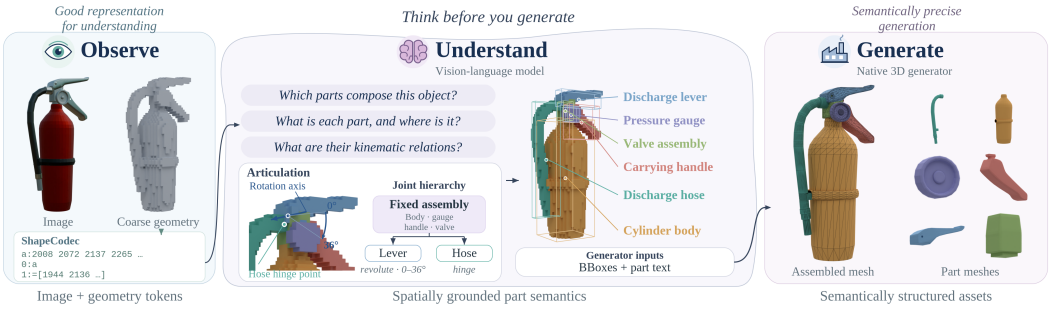

Figure 2: Overview of ShapeMind. ShapeCodec encodes coarse geometry as text for VLM reasoning about semantic parts, spatial extents, and kinematic relationships. Part description and spatial bounds then condition native 3D generation of semantically organized part meshes. Voxel colors illustrate part correspondences for visualization.

1 Introduction

The structure encoded in a generated 3D asset determines how its geometry can be edited, composed, and manipulated. Supporting these operations requires geometric components with explicit semantic identities and coherent spatial relationships, together with kinematic structure for movable parts (Zhu et al., 2026; Cao et al., 2025). These requirements arise across content creation, interactive environments, and embodied simulation, where downstream workflows depend on access to meaningful object components. Practical workflows also place constraints on geometric complexity, favoring compact triangle meshes that retain this explicit part structure.

Meeting these requirements depends on the guidance available during geometry synthesis. Image-to-3D models produce detailed surfaces (Xiang et al., 2024), yet an image alone does not fully specify how an object should be decomposed. PartCrafter and PartPacker generate separable components without explicit semantic guidance (Lin et al., 2025; Tang et al., 2025). OmniPart and X-Part add masks, boxes, or segmentation features, but omit part-level text descriptions (Yang et al., 2025b; Yan et al., 2025). CubePart introduces textual part schemas, although ambiguous spatial names can lead to swapped or misplaced components (Zhu et al., 2026). These methods extract meshes from geometric fields, so meeting low-poly budgets can require additional simplification or remeshing.

A useful generation interface should connect semantic intent with spatial structure. Text describes a part's meaning and desired geometric attributes; 3D boxes specify its placement and extent while enabling interactive layout control. Together, they make explicit semantic and structural information that visual evidence alone may leave ambiguous. This motivates **ShapeMind** and its central principle, *think before you generate*: infer an object's semantic organization before using it to guide native mesh synthesis (Figure 2).

The understanding stage must ground semantic knowledge in the geometry of the specific object. VLM-based asset generators demonstrate the potential of semantic and physical reasoning from images (Cao et al., 2025; 2026), but a single view leaves occluded geometry uncertain. Point-cloud-conditioned models introduce learned 3D–language interfaces (Xu et al., 2023; Wang et al., 2026b), whose training depends on scarce aligned supervision (Guo et al., 2025). Our **ShapeVLM** presents coarse geometry as compact text within the VLM's existing vocabulary, reusing pretrained semantic and commonsense knowledge for 3D structural reasoning. Its **ShapeCodec** representation encodes voxel contours with collinear collapse and template reuse, exposing 3D evidence through the native image–text interface. Two-turn supervision links structural predictions to requested part geometry, grounding semantic identities, spatial extents, and applicable kinematics without introducing a separate geometry encoder.

This inferred structure becomes an actionable specification for **Semantic PartGen**. Part descriptions and boxes complement the image, allowing semantic requirements and spatial layout to condition generation jointly. Building on native vertex–topology generation (Wang et al., 2026a), Semantic PartGen first generates part vertices and then their connectivity. Cross-part attention coordinates the object while preserving each component as a separate low-poly mesh. The resulting representation supports independent editing and composition; inferred kinematics remain associated with the parts for interaction where applicable.

To support this conditional generation task, we curate 300K semantically annotated low-poly assets. Experiments on PartNet-Mobility (Xiang et al., 2020) and PartObjaverse-Tiny demonstrate leading results in 3D understanding and conditional part-mesh generation, complemented by qualitative evaluation on PhysX-Bench (Cao et al., 2026).

Our contributions are:

- **Native 3D structural understanding:** We introduce ShapeCodec, a compact contour-based representation that expresses 3D geometry within a VLM's existing text vocabulary, together with ShapeVLM, a two-turn supervised VLM coupling semantic structure prediction with part geometry reconstruction.
- **Native multi-part representation and generation:** We introduce octree forest with part-local topology latent, designed for native generation of multi-part mesh. Based on these representations, we propose a tri-stream diffusion transformer, guided by the conditions infered by ShapeVLM.
- **Performance and generalization:** Experiments across multiple benchmarks demonstrate state-of-the-art performance in 3D semantic understanding together with part-mesh generation.

2 Related Works

3D understanding. Part segmentation localizes an object's components: SAMPart3D (Yang et al., 2024) distills visual features for multi-granularity decomposition, while P3-SAM (Ma et al., 2025) and PartSAM (Zhu et al., 2025) learn promptable segmentation from native 3D data. Beyond segmentation, PointLLM (Xu et al., 2023) and ShapeLLM (Qi et al., 2024) connect point-cloud encoders with language models for semantic understanding. ArtLLM (Wang et al., 2026b) predicts part layouts and joints from point clouds to guide part generation; SIMART (Zhang et al., 2026) jointly predicts decomposition and kinematics using sparse VQ-VAE geometry tokens. ShapeVLM instead presents coarse geometry as ordinary text, enabling image–geometry reasoning through the VLM's existing vocabulary without a separate learned 3D encoder.

Multi-part 3D generation. Part-aware generators recover complete components that can be edited independently. HoloPart (Yang et al., 2025a) completes segmented surfaces using local geometry and global context. PartCrafter (Lin et al., 2025) jointly denoises part latents with hierarchical attention, while PartPacker (Tang et al., 2025) organizes parts into complementary volumes for image-conditioned generation. For explicit structural control, OmniPart (Yang et al., 2025b) plans part boxes from 2D masks, and X-Part (Yan et al., 2025) combines box prompts with point-wise semantic features. CubePart (Zhu et al., 2026) specifies the desired decomposition through open-vocabulary part names. ShapeMind jointly uses part descriptions and 3D bounds to express semantic requirements and constrain spatial layout. Text specifies a component's identity and geometric attributes, while boxes resolve its spatial extent and provide direct control over the layout.

Native mesh generation. Native mesh models directly generate vertices and connectivity. PolyGen (Nash et al., 2020) models vertex and face sequences, while MeshGPT (Siddiqui et al., 2023) learns discrete geometric embeddings for autoregressive triangle generation. MeshAnything (Chen et al., 2024b) adds shape conditioning for artist-like meshing, and BPT (Weng et al., 2024) compresses sequences through block indexing and patch aggregation to support larger meshes. Moving beyond sequential generation, LATO (Zhao et al., 2026) uses structured topology-preserving latents with flow matching, while Nexus (Wang et al., 2026a) separates octree-based vertex diffusion from continuous topology generation. Following the vertex–topology factorization of Nexus, Semantic PartGen jointly conditions mesh synthesis on images, part descriptions, and spatial bounds, with cross-part attention coordinating the generation of semantically organized low-poly components.

3 Method

We consider image-conditioned generation of low-poly 3D assets with explicit semantic part structure and, where applicable, kinematic structure. ShapeMind addresses this task through **ShapeVLM** for geometry-grounded semantic understanding and **Semantic PartGen** for native mesh synthesis

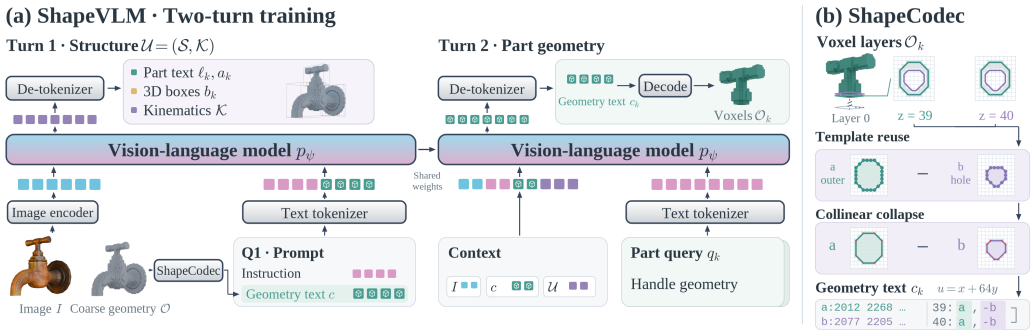

Figure 3: **ShapeVLM.** (a) Two-turn training predicts semantic structure followed by the requested part's geometry. (b) ShapeCodec represents voxel slices through signed contours, template reuse, and collinear collapse. Geometry is serialized using ordinary text tokens.

(Figure 2). Given an image I and coarse object geometry G_0, which can be generated by an image-to-3D model, we discretize the geometry into a filled occupancy grid $\mathcal{O}$ and serialize it as a text token sequence c. ShapeVLM predicts a structured understanding of the object, $\mathcal{U} = (\mathcal{S}, \mathcal{K})$, comprising a semantic part interface $\mathcal{S}$ and kinematic information $\mathcal{K}$:

$$\widehat{\mathcal{U}} \sim p_\psi(\mathcal{U} \mid I, c), \qquad \widehat{\mathcal{M}} \sim p_{\theta,\phi}(\mathcal{M} \mid I, \widehat{\mathcal{S}}, N). \tag{1}$$

Here, ψ denotes the ShapeVLM parameters; θ and ϕ denote Semantic PartGen's vertex and topology generation parameters, and N controls the object-level vertex budget. For an object with n_p parts, the output is $\mathcal{M} = \{M_k = (V_k, F_k)\}_{k=1}^{n_\mathrm{p}}$, where V_k and F_k are the vertex and face sets of part k.

Structured object understanding. The semantic part interface specifies the per-part conditions for generation, while kinematic information describes how parts move:

$$\mathcal{S} = \{(b_k, \ell_k, a_k)\}_{k=1}^{n_\mathrm{p}}, \qquad \mathcal{K} = \{(P_j, \pi_j, \tau_j, \eta_j)\}_{j=1}^{J}. \tag{2}$$

Each $b_k \in \mathbb{R}^6$ specifies the minimum and maximum corners of an axis-aligned box; ℓ_k and a_k give the part's semantic label and textual description. For kinematic group j, $P_j \subseteq \{1, \ldots, n_\mathrm{p}\}$ identifies parts moving together, π_j its parent, τ_j its joint type, and η_j its applicable motion parameters, such as axis, pivot, and range. All spatial quantities share an object frame. Semantic PartGen is conditioned on $\mathcal{S}$, with coordinates mapped to its normalized frame; $\mathcal{K}$ is associated with the generated parts as motion metadata.

3.1 SHAPEVLM

ShapeVLM predicts structured object understanding from an image and a textual encoding of coarse geometry. ShapeCodec defines this encoding, and a two-turn training objective grounds the predicted semantic parts in their voxel geometry (Figure 3).

3.1.1 SHAPECODEC

ShapeCodec balances compactness with ease of geometric interpretation (Figure 3(b)). It retains explicit boundary points that describe shape, reducing sequence length through straight-edge simplification and reuse of repeated contours. This keeps key spatial evidence accessible, with reconstruction requiring line interpolation and region filling.

Surface occupancy and filling. We normalize the input geometry and discretize its surface on a 64^3 grid. Generated or scanned geometry may have unreliable interiors; our objective is to infer object structure from its external shape. We therefore use 3D flood fill to fill enclosed empty cells, producing occupancy $\mathcal{O}$ while preserving cavities connected to the exterior. Filling also reduces internal boundaries and fragmented slice contours, improving encoding efficiency.

Signed contour encoding. We represent each horizontal slice $\mathcal{O}_z$ by ordered closed contours. An outer contour adds its enclosed region, while a hole contour subtracts its region; nested contours are applied from outside to inside. Their signs therefore specify how to reconstruct a slice, including islands within holes. The slice index supplies z, and each boundary point (x, y) is stored as $u = x + 64y$. Within a contour, *collinear collapse* retains only segment endpoints and direction changes, including reversals on thin protrusions. Thus, the encoded points explicitly describe the shape's boundary, while line interpolation and contour filling recover the remaining occupied voxels.

Template reuse. Contours repeated across slices share a template. Similar contours reference that template with exact boundary-point additions and removals; unique contours are written directly. Together with collinear collapse, this removes redundancy while permitting exact reconstruction of $\mathcal{O}$. Appendix A.1 provides the matching rules and encoding and decoding pseudocode.

Encoding efficiency. We compare against run-length encoding (RLE), which represents consecutive occupied indices by start–length pairs and reuses slice templates as in PhysX-Omni (Cao et al., 2026). Table 1 reports object-averaged token counts and their ratios to ShapeCodec on 200 objects from each of four datasets. All methods use the same tokenizer; RLE also uses flood fill. Sampling, serialization, and ablation details are given in Appendix A.3.

Table 1: Geometry tokenization on 800 objects. Relative token counts are normalized by ShapeCodec.

Method	Geometry tokens↓	Relative tokens (×)↓
Raw voxel xyz	205,540.6	33.89
RLE + flood fill	9,272.9	1.53
Ours w/o flood fill	8,513.0	1.40
Ours w/o collinear collapse	13,525.3	2.23
Ours w/o template reuse	9,240.8	1.52
Ours	**6,064.8**	**1.00**

3.1.2 Geometry-Supervised Structural Learning

We fine-tune Qwen3-VL-8B-Instruct (Bai et al., 2025) using two-turn conversations (Figure 3(a)). The first turn already predicts the required understanding $\mathcal{U} = (\mathcal{S}, \mathcal{K})$ from image I and geometry sequence c. To require finer geometric understanding than boxes and labels alone, the second turn asks for the voxel geometry of a specified part. Given a query q_k, the model produces its ShapeCodec sequence c_k in the shared object frame:

$$p_\psi(\mathcal{U}, c_k \mid I, c, q_k) = p_\psi(\mathcal{U} \mid I, c)\, p_\psi(c_k \mid I, c, \mathcal{U}, q_k). \tag{3}$$

This more demanding reconstruction task encourages the model to interpret the geometry encoded in c and associate it with individual semantic parts, providing auxiliary supervision for first-turn structural prediction. We train on assistant responses with teacher forcing; optimization and conversation sampling are detailed in Appendix A.2. At inference, only the first turn is needed: $\widehat{\mathcal{S}}$ conditions mesh generation, and $\widehat{\mathcal{K}}$ supplies motion metadata.

3.2 Semantic PartGen

Semantic PartGen generates separate triangle meshes in a shared object frame from an image I, a semantic part interface $\mathcal{S} = \{(b_k, \ell_k, a_k)\}_{k=1}^{K}$ generated by ShapeVLM, and an object-level vertex-count condition N. Each part has an axis-aligned box $b_k \in \mathbb{R}^6$, a label ℓ_k, and an appearance description a_k. Building on Nexus Wang et al. (2026a), we first generate vertices $V = \{V_k\}$ and then connectivity latents $Z = \{Z_k\}$:

$$p(V, Z \mid I, \mathcal{S}, N) = p_\theta(V \mid I, \mathcal{S}, N)\, p_\phi(Z \mid V, I, \widetilde{\mathcal{S}}(V)), \qquad F_k = \mathcal{R}\big(D(Z_k, V_k)\big). \tag{4}$$

Here, $\widetilde{\mathcal{S}}(V)$ replaces input boxes with boxes fitted to the generated vertices, retaining the text. The frozen mesh VAE decoder D and face recovery algorithm in Wang et al. (2026a) $\mathcal{R}$ recover the triangles F_k, and eventually $M_k = (V_k, F_k)$.

3.2.1 Native Multi-Part Representation

Octree forest for vertices. Each part's quantized vertices define an octree in the common object frame; together, the trees form an octree forest. Occupancy indicates a mesh vertex. At depth d, let $\mathcal{P}_d$ collect active parent nodes with their positions and part identities, and let O_d collect their eight-channel child-occupancy vectors. For trees of depth L, generation factorizes as

$$p_\theta(V \mid I, \mathcal{S}, N) = \prod_{d=0}^{L-1} p_\theta(O_d \mid \mathcal{P}_d, d, I, \mathcal{S}, N). \tag{5}$$

Starting from one root per part, the model denoises O_d and expands occupied children to form $\mathcal{P}_{d+1}$; final leaves determine the vertices. Vertices from different parts retain separate identities even when they occupy the same grid cell. N guides the total vertex count without enforcing an exact cardinality.

Part-local topology. The topology model denoises a connectivity latent at each generated vertex, conditioned on its fixed position, the fitted part boxes, part text, and image. Its training targets come from the pretrained mesh VAE of Nexus. The frozen decoder produces edge and face features

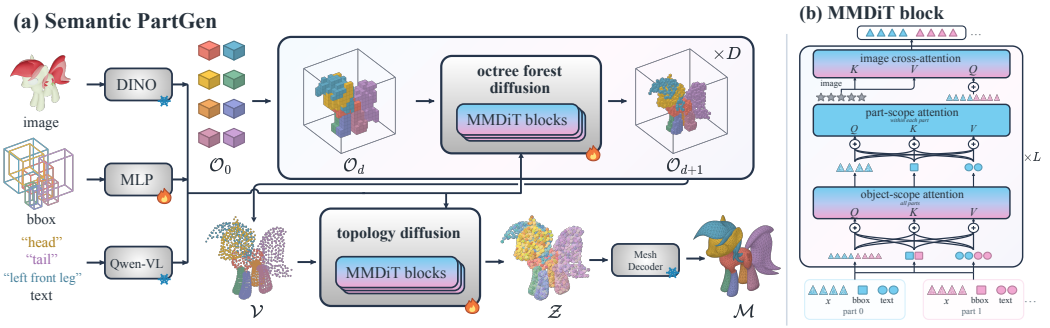

Figure 4: **Semantic PartGen.** (a) Two diffusion models generate part vertices and connectivity latents. A frozen mesh decoder recovers separate triangle meshes. (b) Tri-stream blocks couple geometry, box, and text features through joint attention within and across parts; image features enter through cross-attention.

$e_{k,i}, f_{k,i}$ at each vertex. Spacetime-distance and area scores s_E, s_F recover

$$\begin{aligned} E_k &= \{(i,j) : i < j,\ s_E(e_{k,i}, e_{k,j}) > 0\}, \\ F_k &= \{(i,j,r) \in \mathcal{C}(E_k) : s_F(f_{k,i}, f_{k,j}, f_{k,r}) > 0\}, \end{aligned} \tag{6}$$

where $\mathcal{C}(E_k)$ contains closed triples $i < j < r$ of predicted edges. Score calculation and recovery is restricted to each part, so faces cannot connect different parts, while the generation networks can exchange information across parts.

3.2.2 Conditional Tri-Stream Diffusion Transformer

Tri-stream architecture. The vertex and topology generators share the architecture in Figure 4, with separate weights. Each generator processes noisy geometry, part boxes, and part text in three streams. Following MMDiT Shin et al. (2025), the streams use modality-specific parameters and exchange information through joint attention. Box and text features evolve alongside geometry, while only the geometry stream predicts denoised outputs.

Scoped attention and image conditioning. At each layer, modality-specific projections produce Q^m, K^m, V^m for geometry, box, and text streams, $m \in \{x, b, \text{text}\}$. Joint attention operates within each part or across the object Zhu et al. (2026), followed by image cross-attention:

$$\begin{aligned} \widetilde{H} &= \mathrm{SA}_s([Q^m]_m, [K^m]_m, [V^m]_m), \quad s \in \{\text{part}, \text{obj}\}, \\ H^x_{\text{out}} &= \widetilde{H}^x + \mathrm{CA}\Big(Q^I(\widetilde{H}^x), K^I(C_I), V^I(C_I)\Big) \end{aligned} \tag{7}$$

Here, $[\cdot]_m$ denotes token concatenation among modalities and C_I contains frozen image features. Geometry features are initialized from noisy targets x_t, and the head predicts clean targets $\widehat{x}_0$ after the final layer. Box and text streams bypass image cross-attention. Normalization, positional encoding, and the residual and feed-forward operations preceding cross-attention are omitted for brevity.

Training. Both generators use flow matching between clean targets and Gaussian noise, implemented as clean-target prediction with a velocity-space loss. We first pretrain on segmented meshes conditioned on boxes and images, then add a newly initialized text stream and fine-tune on a curated semantically annotated dataset introduced in 4.2. The mesh VAE and the image and text feature extractors remain frozen. Implementation and training details appear in Appendix B.

4 Experiments

Our experiments test *think before you generate* through structural understanding and mesh generation from predicted part interfaces. We separately evaluate native generation under ground-truth conditions and ablate the representation, supervision, and generation architecture.

4.1 Semantic Part Mesh Dataset Curation

Pretraining dataset. We curate approximately 1 million low-poly objects with 2–32 parts from sources including Objaverse-XL and TexVerse. Mesh hierarchies and connected components define geometric partitions, paired with rendered images and part boxes for pretraining without text.

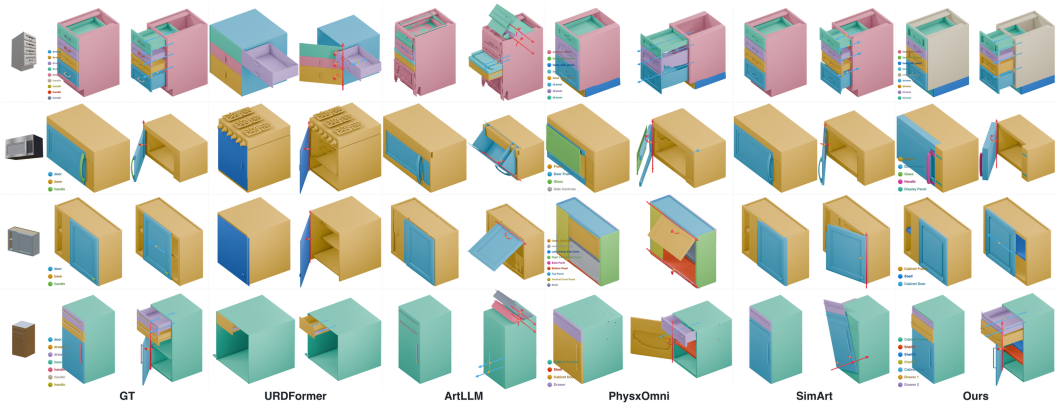

Figure 5: **PartNet-Mobility comparisons.** Paired views show part organization and motion; colors identify parts and arrows indicate joints.

Semantic adaptation dataset. We select approximately 300K assets through part-count and VLM-based quality filtering. Following CubePart (Zhu et al., 2026), paired textured views and color-coded part masks with IDs and contours guide VLM annotation of part labels and geometric descriptions for fine-tuning.

4.2 Implementation Details

ShapeVLM. We fine-tune Qwen3-VL-8B-Instruct on Artiverse (Iliash et al., 2026), PhysX-3D, PhysX-Mobility (Cao et al., 2025), and PhysXVerse (Cao et al., 2026), mixing structure-only and two-turn records at 1:1. Images and coarse geometry are near-aligned, with boxes and kinematics transformed consistently. Training details appear in Appendix A.2.

Semantic PartGen. Vertex and topology generators are trained separately on the datasets in Section 4.1. Architecture, optimization, and sampling settings appear in Appendix B.

4.3 Comparison

4.3.1 Native 3D Understanding and Generation

Evaluation setup. Following ArtLLM (Wang et al., 2026b), we evaluate on 77 PartNet-Mobility (Xiang et al., 2020) objects. ShapeVLM receives an image and voxelized ground-truth geometry; its predicted part text and boxes condition Semantic PartGen, with predicted kinematics attached to the generated parts. Baselines include URDFormer (Chen et al., 2024a), ArtLLM, SIMART (Zhang et al., 2026), and PhysX-Omni (Cao et al., 2026). URDFormer covers 66 objects within its five training categories. PhysX-Bench (Cao et al., 2026) provides qualitative evaluation on more diverse objects.

Metrics and protocol. We report box mIoU, joint-type accuracy, axis/pivot errors, range IoU, graph accuracy, and generated-part CD/F@0.01, averaged per object. SIMART's geometry scores are omitted because it segments its input mesh. Timing covers first-turn network inference only.

Results and Analysis. ShapeVLM achieves the best localization, joint-type, and graph accuracy, and its predicted interface yields the best reported part-geometry scores (Table 2). Figure 5 shows distinct movable components with coherent spatial and motion relationships, connecting structural understanding to mesh synthesis. On PhysX-Bench (Figure 6), the flying object and handheld example exhibit plausible finer decompositions and motion predictions beyond the ground-truth annotations.

4.3.2 Native Part Generation

Evaluation setup. We evaluate conditional generation on 60 PartObjaverse-Tiny objects (368 parts) (Yang et al., 2024) with at most eight parts, following CubePart (Zhu et al., 2026). Semantic PartGen receives an image, ground-truth boxes, labels and descriptions, and a ground-truth-derived vertex count, isolating generation from ShapeVLM prediction errors. Baselines are PartPacker (Tang et al., 2025), PartCrafter (Lin et al., 2025), PatchAlign3D (Hadgi et al., 2026) + HoloPart (Yang et al., 2025a), OmniPart (Yang et al., 2025b), and CubePart.

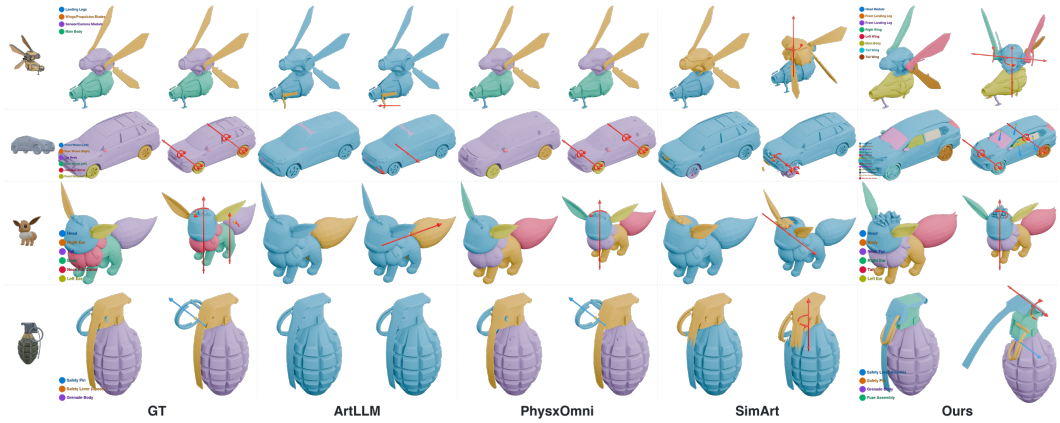

Figure 6: **Qualitative comparisons on PhysX-Bench.** Diverse object shapes test semantic decomposition and kinematic reasoning beyond the household objects in Figure 5. Colors distinguish parts; paired views visualize the assets and their predicted articulation.

Table 2: 3D understanding and generated part geometry on PartNet-Mobility (URDFormer: five categories). Best/second-best values across all listed methods are bold/underlined; ties share a rank.

Method	mIoU↑	Type Acc.↑	Axis Err.↓	Pivot Err.↓	Range IoU↑	Graph Acc.↑	Time (s)↓	Part-CD $\times 10^3$ ↓	Part-F @.01↑
URDFormer	0.1332	0.7909	0.4301	0.3522	<u>0.8216</u>	0.5000	–	102.181	0.0086
ArtLLM	0.5895	0.7644	0.4812	0.1794	**0.8743**	0.5844	–	29.755	0.3415
SIMART	0.6543	0.8377	0.4150	0.1954	0.5518	<u>0.7792</u>	–	–	–
PhysX-Omni	0.4146	0.8737	0.3504	0.1822	0.7235	0.5455	–	43.058	0.1694
ShapeVLM + RLE	<u>0.8055</u>	<u>0.9005</u>	<u>0.2499</u>	0.0714	0.7546	0.7532	<u>21.40</u>	<u>17.822</u>	0.3584
ShapeVLM (T1-only)	0.7623	0.8968	0.2618	<u>0.0627</u>	0.7882	0.7273	–	19.813	<u>0.3667</u>
ShapeVLM	**0.8140**	**0.9328**	**0.1870**	**0.0495**	0.8050	**0.8052**	**16.81**	**17.480**	**0.3842**

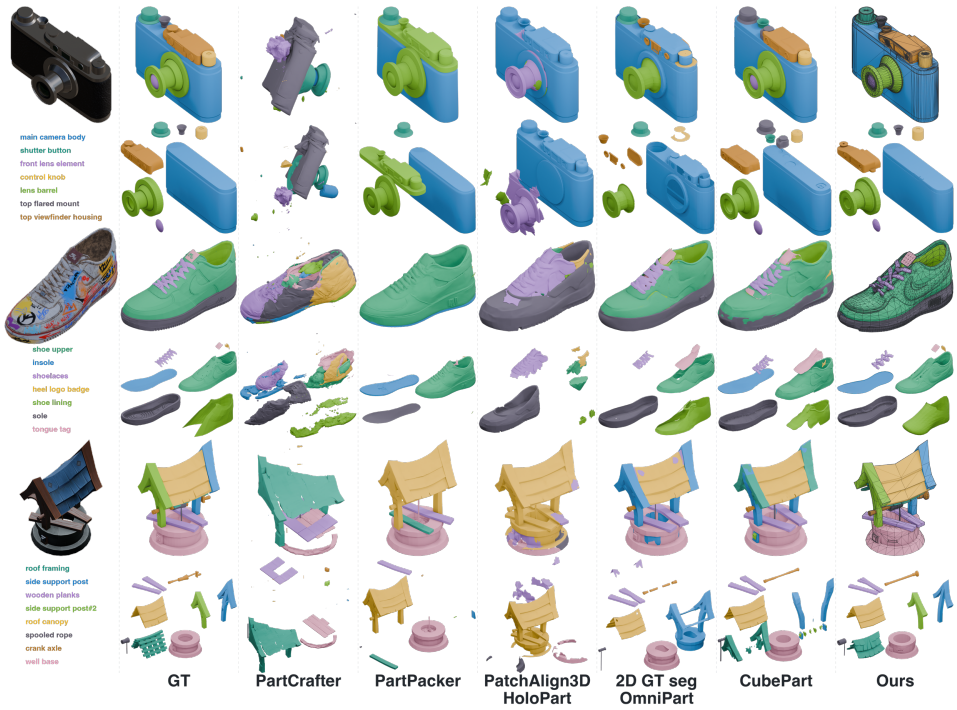

Figure 7: Qualitative comparison using the same source images under each method's input settings. Colors indicate part correspondences; assembled and exploded views show geometry and decomposition.

Metrics and protocol. We report object- and part-level Chamfer distance (CD↓) and F-score at threshold 0.1 (F↑), averaged per object. PartPacker uses the image; PartCrafter also receives the ground-truth part count. Other baselines use HY2.1 meshes generated from the same image by Hunyuan3D-2.1, with part labels for PatchAlign3D + HoloPart and CubePart, or oracle 2D part masks for OmniPart. Oracle 3D bbox variants supply ground-truth boxes and align the input mesh to the ground-truth pose; oracle mesh variants supply the ground-truth whole-object mesh and omit object-level scores. Metric and correspondence details appear in Appendix B.2.

Results and Analysis. Semantic PartGen leads all reported object and part metrics (Table 3), including both part metrics against every oracle-box and oracle-mesh variant. Against OmniPart with oracle meshes and masks, Part-CD improves from 0.03444 to 0.02518 and Part-F from 0.91252 to 0.96148. Figure 7 shows fewer merged or missing components and part geometries with higher fidelity; wireframes reveal high quality low-poly topology, while other isosurface based baselines generate dense polygons.

Table 3: Part generation on PartObjaverse-Tiny. Baseline comparisons use objects with at most eight parts; ablations use the full set. Best/second-best values are bold/underlined.

Method	Obj-CD↓	Obj-F@0.1↑	Part-CD↓	Part-F@0.1↑	Mesh input
PartObjaverse-Tiny: #parts ≤ 8					
PartPacker	0.05897	0.83799	0.18596	0.49301	N/A
PartCrafter	0.13769	0.51498	0.33219	0.15962	N/A
PatchAlign3D + HoloPart	0.04583	0.90408	0.15261	0.55532	HY2.1
w/ Oracle 3D bbox	<u>0.04355</u>	<u>0.91478</u>	0.05911	0.81670	HY2.1
w/ Oracle mesh	–	–	0.12290	0.63602	GT
Oracle Seg-mask + OmniPart	0.04539	0.90443	0.08423	0.74559	HY2.1
w/ Oracle 3D bbox	0.04393	0.91118	0.06441	0.83367	HY2.1
w/ Oracle mesh	–	–	<u>0.03444</u>	<u>0.91252</u>	GT
CubePart	0.04403	0.90863	0.16135	0.65204	HY2.1
w/ Oracle mesh	–	–	0.12677	0.77461	GT
Ours	**0.02925**	**0.97344**	**0.02518**	**0.96148**	N/A
PartObjaverse-Tiny: full set					
Ours w/o cross-part attention	0.033410	0.955581	0.025337	0.954996	N/A
Ours w/o pretraining	0.038244	0.935025	0.029639	0.939115	N/A
Ours w/o text stream†	0.030928	0.965386	0.022902	0.963225	N/A
Ours	**0.028983**	**0.972984**	**0.020757**	**0.971042**	N/A

4.4 Ablation

ShapeVLM. ShapeCodec reduces geometry-token length. Compared with RLE, ShapeVLM improves localization, joint-type accuracy, and generated-part geometry with lower inference latency (Tables 1 and 2). Two-turn supervision improves over T1-only training; inference still uses only first-turn predictions.

Semantic PartGen. Removing cross-part attention introduces overlaps and gaps; omitting pretraining degrades detail and image alignment; removing text confuses an occluded dagger with a hammer (Table 3, Figure 8). These ablations support cross-part coordination, geometry pretraining, and semantic conditioning.



Figure 8: Semantic PartGen ablations.

5 Conclusion

We presented ShapeMind, a framework for semantically structured 3D asset generation following *think before you generate*. ShapeVLM reasons about semantic and kinematic structure using coarse geometry encoded as text, providing part descriptions and boxes that guide Semantic PartGen's native low-poly mesh generation. Experiments demonstrate strong 3D understanding and state-of-the-art part generation, with qualitative evidence of generalization to diverse objects. These results support semantic structure as an effective interface between VLM reasoning and native 3D synthesis.

6 AI USE STATEMENT

This section discloses the use of large language models (LLMs) as general-purpose assistance tools in this research. LLMs were used to edit and polish portions of the manuscript for clarity and readability. We also used AI tools to assist with data processing and organization, including semantic part labeling. The authors reviewed and revised all AI-assisted text, and conducted rigorous manual checks of the processed data, semantic annotations, and experimental outputs. The authors take full responsibility for the final manuscript, claims, methods, code, and results, including content produced with the assistance of generative AI tools.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, et al. Qwen3-VL Technical Report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Ziang Cao, Fangzhou Hong, Zhaoxi Chen, Liang Pan, and Ziwei Liu. PhysX-Anything: Simulation-Ready Physical 3D Assets from Single Image. *arXiv preprint arXiv:2511.13648*, 2025. URL https://arxiv.org/abs/2511.13648.

Ziang Cao, Yinghao Liu, Haitian Li, Runmao Yao, Fangzhou Hong, Zhaoxi Chen, Liang Pan, and Ziwei Liu. PhysX-Omni: Unified Simulation-Ready Physical 3D Generation for Rigid, Deformable, and Articulated Objects. *arXiv preprint arXiv:2605.21572*, 2026. URL https://arxiv.org/abs/2605.21572.

Qiuyu Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Dieter Fox, and Abhishek Gupta. URDFormer: A Pipeline for Constructing Articulated Simulation Environments from Real-World Images. In *Proceedings of Robotics: Science and Systems*, 2024a. doi: 10.15607/RSS.2024.XX.124. URL https://www.roboticsproceedings.org/rss20/p124.html.

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. *arXiv preprint arXiv:2406.10163*, 2024b. URL https://arxiv.org/abs/2406.10163.

Zilu Guo, Hongbin Lin, Zhihao Yuan, Chaoda Zheng, Pengshuo Qiu, Dongzhi Jiang, Renrui Zhang, Chun-Mei Feng, and Zhen Li. PiSA: A Self-Augmented Data Engine and Training Strategy for 3D Understanding with Large Models. *arXiv preprint arXiv:2503.10529*, 2025. URL https://arxiv.org/abs/2503.10529.

Souhail Hadgi, Bingchen Gong, Ramana Sundararaman, Emery Pierson, Lei Li, Peter Wonka, and Maks Ovsjanikov. PatchAlign3D: Local Feature Alignment for Dense 3D Shape Understanding. *arXiv preprint arXiv:2601.02457*, 2026. URL https://arxiv.org/abs/2601.02457.

Denys Iliash, Jiayi Liu, Egor Fokin, Qirui Wu, Ali Mahdavi-Amiri, Manolis Savva, and Angel X. Chang. Artiverse: A Diverse and Physically Grounded Dataset for Articulated Objects. *arXiv preprint arXiv:2605.24403*, 2026. URL https://arxiv.org/abs/2605.24403.

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. *arXiv preprint arXiv:2506.05573*, 2025. URL https://arxiv.org/abs/2506.05573.

Changfeng Ma, Yang Li, Xinhao Yan, Jiachen Xu, Yunhan Yang, Chunshi Wang, Zibo Zhao, Yanwen Guo, Zhuo Chen, and Chunchao Guo. P3-SAM: Native 3D Part Segmentation. *arXiv preprint arXiv:2509.06784*, 2025. URL https://arxiv.org/abs/2509.06784.

Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. PolyGen: An Autoregressive Generative Model of 3D Meshes. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. URL https://proceedings.mlr.press/v119/nash20a.html.

Zekun Qi, Runpei Dong, Shaochen Zhang, Haoran Geng, Chunrui Han, Zheng Ge, Li Yi, and Kaisheng Ma. ShapeLLM: Universal 3D Object Understanding for Embodied Interaction. *arXiv preprint arXiv:2402.17766*, 2024. URL https://arxiv.org/abs/2402.17766.

Joonghyuk Shin, Alchan Hwang, Yujin Kim, Daneul Kim, and Jaesik Park. Exploring Multimodal Diffusion Transformers for Enhanced Prompt-based Image Editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19492–19502, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Shin_Exploring_Multimodal_Diffusion_Transformers_for_Enhanced_Prompt-based_Image_Editing_ICCV_2025_paper.html.

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. *arXiv preprint arXiv:2311.15475*, 2023. URL https://arxiv.org/abs/2311.15475.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, et al. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025. URL https://arxiv.org/abs/2508.10104.

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. URL https://arxiv.org/abs/2506.09980.

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*, 2026a. URL https://arxiv.org/abs/2607.13563.

Penghao Wang, Siyuan Xie, Hongyu Yan, Xianghui Yang, Jingwei Huang, Chunchao Guo, and Jiayuan Gu. ArtLLM: Generating Articulated Assets via 3D LLM. *arXiv preprint arXiv:2603.01142*, 2026b. URL https://arxiv.org/abs/2603.01142.

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. *arXiv preprint arXiv:2411.07025*, 2024. URL https://arxiv.org/abs/2411.07025.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A SimulAted Part-Based Interactive ENvironment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. URL https://arxiv.org/abs/2003.08515.

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. *arXiv preprint arXiv:2412.01506*, 2024. URL https://arxiv.org/abs/2412.01506.

Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. PointLLM: Empowering Large Language Models to Understand Point Clouds. *arXiv preprint arXiv:2308.16911*, 2023. URL https://arxiv.org/abs/2308.16911.

Xinhao Yan, Jiachen Xu, Yang Li, Changfeng Ma, Yunhan Yang, Chunshi Wang, Zibo Zhao, Zeqiang Lai, Yunfei Zhao, Zhuo Chen, and Chunchao Guo. X-Part: High Fidelity and Structure Coherent Shape Decomposition. *arXiv preprint arXiv:2509.08643*, 2025. URL https://arxiv.org/abs/2509.08643.

Yunhan Yang, Yukun Huang, Yuan-Chen Guo, Liangjun Lu, Xiaoyang Wu, Edmund Y. Lam, Yan-Pei Cao, and Xihui Liu. SAMPart3D: Segment Any Part in 3D Objects. *arXiv preprint arXiv:2411.07184*, 2024. URL https://arxiv.org/abs/2411.07184.

Yunhan Yang, Yuan-Chen Guo, Yukun Huang, Zi-Xin Zou, Zhipeng Yu, Yangguang Li, Yan-Pei Cao, and Xihui Liu. HoloPart: Generative 3D Part Amodal Segmentation. *arXiv preprint arXiv:2504.07943*, 2025a. URL https://arxiv.org/abs/2504.07943.

Yunhan Yang, Yufan Zhou, Yuan-Chen Guo, Zi-Xin Zou, Yukun Huang, Ying-Tian Liu, Hao Xu, Ding Liang, Yan-Pei Cao, and Xihui Liu. OmniPart: Part-Aware 3D Generation with Semantic Decoupling and Structural Cohesion. *arXiv preprint arXiv:2507.06165*, 2025b. URL https://arxiv.org/abs/2507.06165.

Chuanrui Zhang, Minghan Qin, Yuang Wang, Baifeng Xie, Hang Li, and Ziwei Wang. SIMART: Decomposing Monolithic Meshes into Sim-ready Articulated Assets via MLLM. *arXiv preprint arXiv:2603.23386*, 2026. URL https://arxiv.org/abs/2603.23386.

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. URL https://arxiv.org/abs/2603.06357.

Yiheng Zhu, Kangle Deng, Jean-Philippe Fauconnier, Inaki Navarro, Daiqing Li, Ava Pun, Yinan Zhang, Peiye Zhuang, Xiaoxia Sun, Maneesh Agrawala, Kiran Bhat, and Tinghui Zhou. CubePart: An Open-Vocabulary Part-Controllable 3D Generator. *arXiv preprint arXiv:2605.28763*, 2026. URL https://arxiv.org/abs/2605.28763.

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. *arXiv preprint arXiv:2509.21965*, 2025. URL https://arxiv.org/abs/2509.21965.

A SHAPEVLM IMPLEMENTATION AND SHAPECODEC DETAILS

A.1 SHAPECODEC ENCODING AND DECODING

Preprocessing and representation. Geometry is normalized to a shared 64^3 grid; boxes and motion parameters use the same coordinate transform. For meshes, we deterministically sample 200,000 surface points and quantize their positions. Our implementation also applies the view-dependent azimuth alignment used by the image–geometry training pairs before normalization. Three-dimensional flood fill marks empty cells reachable from the grid boundary and fills the remaining enclosed cells. This preprocessing defines $\mathcal{O}$ and does not recover the original continuous surface or guarantee closure of gaps in incomplete scans. The codec is lossless relative to $\mathcal{O}$, not relative to the unfilled surface geometry.

Within each slice, we trace outer and hole contours in nesting order. A positive contour adds its boundary and enclosed region; a negative contour subtracts its hole region. This ordered construction preserves a solid island nested inside a hole. Point indices use $u = x + 64y$, with $x = u \bmod 64$ and $y = \lfloor u/64 \rfloor$; the slice record supplies z. Template definitions and slice records use ordinary text characters and the unchanged Qwen3-VL tokenizer.

Exact collinear collapse. Let $d_1 = p_i - p_{i-1}$ and $d_2 = p_{i+1} - p_i$ be consecutive integer steps along a contour. We retain p_i if $d_1 \times d_2 \neq 0$ or $d_1 \cdot d_2 \leq 0$. This preserves both corners and reversals, including one-voxel-wide protrusions. Integer line rasterization between retained points restores the dense contour. Collapse is applied only when writing contours, so matching and edit computation use dense boundary-point sets.

Template selection and exact edits. Identical boundary sets share a template. Other contours are grouped by boundary overlap $|B \cap T| / \max(|B|, |T|)$, with threshold 0.70. Repeated identical contours retain their common template; other groups use a representative minimizing total serialized boundary-edit length. For groups larger than 60 contours, the implementation approximates this selection with the contour of median boundary size. Given a contour boundary B and a template boundary T, the edits are $\Delta^+ = B \setminus T$ and $\Delta^- = T \setminus B$. The decoder recovers $B = (T \cup \Delta^+) \setminus \Delta^-$ and retraces it to restore contour order. Thus, a similar template does not approximate or replace the target shape: all differences are explicitly retained.

Each occurrence is written either as a template reference with edits or as an inline contour, whichever requires fewer characters. This cost comparison follows geometric matching; it is not the matching criterion itself. The implementation additionally removes unused templates and refines template assignments when doing so shortens the complete serialized representation. Template definitions are emitted before first use. The decoder nevertheless collects all definitions before reconstructing slices, allowing forward references in model predictions. Algorithms 1 and 2 summarize the lossless procedure; optional lossy simplification is disabled.

Algorithm 1 ShapeCodec encoding

Input: filled occupancy grid $\mathcal{O}$. **Output:** geometry text c.
1. Extract slice contours (z, σ, B) in nesting order; σ denotes addition or subtraction.
2. Construct reusable templates T by matching dense contour boundaries.
3. Represent each B inline or by a template reference (T, Δ^+, Δ^-),
   where $\Delta^+ = B \setminus T$ and $\Delta^- = T \setminus B$ preserve the exact boundary.
4. Apply collinear collapse to template and inline boundaries; pack pivots as $u = x + 64y$.
5. Serialize template definitions and ordered signed layer records into c.
6. **Return** c.

Algorithm 2 ShapeCodec decoding

Input: valid geometry text c. **Output:** occupancy grid $\mathcal{O}$.
1. Parse all template definitions and ordered slice records.
2. Unpack template points and rasterize segments to recover dense boundaries.
3. Initialize $\mathcal{O}$ to empty.
4. **For each** slice z and its entries in their recorded order:
   If the entry is inline: unpack its contour points.
   Otherwise: recover $B = (T \cup \Delta^+) \setminus \Delta^-$ and retrace its contour.
   Rasterize the contour and fill its enclosed region R.
   Add R to $\mathcal{O}_z$ for a positive sign; subtract R for a negative sign.
5. **Return** $\mathcal{O}$.

A.2 SHAPEVLM TRAINING IMPLEMENTATION

We fine-tune Qwen3-VL-8B-Instruct, freezing its visual encoder and updating its visual merger and language model. Structure-only (T1) and two-turn (T12) records are mixed at a 1:1 record ratio. Each T12 part query independently extends the annotated first-turn response, rather than conditioning on a predicted response during training. The target part geometry remains in the whole-object coordinate frame. For conversation tokens s, the training objective is

$$\mathcal{L}_{\mathrm{VLM}} = -\frac{1}{\sum_t m_t} \sum_t m_t \log p_\psi(s_t \mid I, s_{<t}), \tag{8}$$

where m_t selects assistant-response tokens and excludes padding and unknown, null-valued targets. T1 records supervise structured understanding; T12 records supervise both understanding and part geometry through teacher forcing. Geometry is ordinary text in the instruction/response stream; no geometry-specific vocabulary or encoder is introduced.

The first response serializes semantic part descriptions and boxes $\mathcal{S}$ together with kinematic groups $\mathcal{K}$ as structured JSON. Motion parameters depend on joint type: revolute joints include an axis, pivot, and angular range, while prismatic joints include a translation direction and range. At inference, the first response suffices for the ShapeMind pipeline. The second-turn reconstruction objective supplies additional geometric supervision during training; it is not an intermediate mesh-generation stage.

A.3 CODEC EVALUATION PROTOCOL

We select 200 training objects from each of Artiverse, PhysX-3D, PhysX-Mobility, and PhysXVerse, for 800 objects in total. Within each dataset, the mean percentile rank of part count and mesh face count defines four complexity strata. We sample 50 objects per stratum, randomizing category order and object selection to cover diverse categories. This is a balanced diagnostic sample, not an estimate of the original category-frequency distribution. Each object is evaluated once at view 0; all methods use the same selected objects, normalization, and surface voxelization. No filtering by sequence length or codec performance is applied.

Variants and reconstruction checks. Raw voxel xyz lists one occupied voxel per line as three space-separated integers, in increasing (z, y, x) order. RLE uses the same filled occupancy as ShapeCodec, with slice-template similarity threshold 0.50. The no-flood-fill variant omits only the initial three-dimensional interior filling; it retains the two-dimensional region filling needed to decode contours. The no-collinear-collapse variant retains dense boundary points in serialization and its length estimates. The no-template-reuse variant writes every contour inline while retaining collinear collapse. Every encoded object is decoded and checked for exact equality with its own input occupancy grid.

Token counts and relative cost. We tokenize geometry text with the Qwen3-VL-8B-Instruct tokenizer, excluding instructions, image tokens, chat wrappers, and output annotations. Let $L_{i,m}$ denote the token count of method m for object i. We report its mean token count and relative cost normalized by ShapeCodec:

$$\overline{L}_m = \frac{1}{800} \sum_{i=1}^{800} L_{i,m}, \qquad R_m = \frac{\overline{L}_m}{\overline{L}_{\mathrm{ShapeCodec}}}. \tag{9}$$

ShapeCodec therefore has relative cost $1.00\times$; larger values indicate higher token usage. Ratios are computed from unrounded means. The raw xyz row represents the filled voxel grid. The no-flood-fill row encodes unfilled surface occupancy, so its relative cost includes the effect of this preprocessing change; it is not a lossless representation of the filled grid.

B PartGen Implementation and Evaluation Details

B.1 Implementation Details

Conditioning and part identity. The vertex and topology generators use separate weights. Frozen DINOv3 (Siméoni et al., 2025) and Qwen3-VL (Bai et al., 2025) provide image and part-text features, respectively. Part text comprises semantic labels and available appearance descriptions. Shared part identifiers associate geometry, box, and text tokens with the same component, while spatial encodings locate geometry in a common object frame. Geometry and boxes are transformed consistently to align with the conditioning view.

Training and data preparation. Both generators are pretrained on segmented low-poly meshes with images and part boxes, then adapted to semantic conditioning using annotated assets. The mesh VAE and image and text encoders remain frozen. Designated held-out objects are excluded from training. Training data preparation filters very small mesh components and merges identically captioned segments during semantic adaptation; evaluation retains the fixed part-instance annotations. Condition dropout supports classifier-free guidance at inference.

Geometry targets and mesh readout. Vertex training samples an octree depth and predicts child-occupancy targets; positive denoised channels select children during generation. Quantization merges coincident vertices within each part. Connectivity training uses standardized vertex latents from the frozen mesh VAE, conditioned on reference vertices and boxes. At inference, these are replaced by generated vertices and fitted boxes. Predicted latents are unstandardized before part-local decoding, and the decoder assigns triangle winding after the face readout in Eq. 6.

B.2 Evaluation Protocol

Benchmark and conditioning. We use PartObjaverse-Tiny (Yang et al., 2024) with fixed instance-level annotations. The full evaluation set contains 197 objects (2,335 parts) with 2–32 parts; the remaining three objects have 35, 40, and 83 parts. Baseline comparisons use the shared subset of 60 objects (368 parts) with at most eight parts. Repeated labels remain separate instances. Each object uses the fixed view-9 render. PartGen receives the image, annotated labels and appearance descriptions, and ground-truth part boxes in stage 1. The object-level vertex-count condition is derived from the reference mesh under a fixed generation budget. The reference mesh is used to construct these conditions, rather than supplied as a whole-object mesh input. In stage 2, the topology generator uses the generated vertices and fitted part boxes, retaining the same part text and image. Connectivity is decoded per part without additional mesh cleanup.

Metrics and correspondence. We sample 10,000 points uniformly by triangle area from each complete mesh and, separately, from each part. For point sets A and B, we use unsquared Euclidean Chamfer distance,

$$\mathrm{CD}(A,B) = \frac{1}{2}\left(\frac{1}{|A|}\sum_{a\in A}\min_{b\in B}\|a-b\|_2 + \frac{1}{|B|}\sum_{b\in B}\min_{a\in A}\|b-a\|_2\right). \tag{10}$$

F-score is the harmonic mean of precision and recall under a nearest-neighbor distance threshold of 0.1. Distances are in object coordinates with the reference object's longest bounding-box edge normalized to 2. Parts are never normalized independently. Obj-CD and Obj-F evaluate the union of part surfaces; Part-CD and Part-F first average the part scores within an object and then average over objects, so each object has equal weight.

For PartGen, generated slots are compared with their original instance IDs, without permutation matching or post-hoc pose fitting. All 197 objects have valid object and part scores. For external methods, we use their saved evaluation protocol: each complete predicted object is centered and normalized to longest extent 2; one whole-object yaw is selected by object CD from 32 candidate yaws for the baselines. The same transformation is applied to every part, without ICP or per-part fitting. CubePart fixes unique-label correspondences and uses minimum-CD assignment within repeated-label groups. Other external methods use Hungarian matching by part CD. Table 3 reports matched-part averages for these methods: unmatched parts do not contribute to its part columns. These geometric scores therefore do not by themselves measure completeness of the requested decomposition.

Oracle baseline variants. Oracle 2D segmentation uses ground-truth part-instance masks projected into the conditioning view. Oracle 3D boxes are ground-truth axis-aligned part boxes; these supplied-box runs also align the HY2.1 input mesh to the reference pose by a whole-object yaw search before generation. This input alignment is separate from the output alignment used for metric

computation. Oracle-mesh variants replace the HY2.1 generated mesh with the ground-truth whole-object mesh while retaining the corresponding method's part conditions. They provide reference object geometry, rather than the ground-truth part meshes as outputs. Object-level scores for these variants are omitted in the comparison table; their part scores assess the resulting decomposition and part reconstruction.

C MORE RESULTS



Figure 9: More results generated by Semantic PartGen.



Figure 10: More results generated by ShapeVLM followed by Semantic PartGen.