DAEDALUS: VERIFIER-GUIDED AGENTIC REFINEMENT FOR PART-STRUCTURED 3D GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generating 3D objects as editable mesh parts is essential for animation and interactive modeling, yet producing a decomposition that matches user intent remains challenging. Users must specify not only which parts to generate, but often their spatial extents, while incorrect controls or generated geometry require further correction. We present Daedalus, an agentic framework that connects semantic planning, controllable neural part generation, and verification-guided local refinement. Given a reference image, a part schema, and a whole-object geometric prior, an agent reviews semantic masks on mesh renderings, and geometric tools fit part bounding boxes from the selected surfaces. A mixed-condition generator then synthesizes parts using heterogeneous image, text, bbox, and voxel controls, supporting both individual instances and instance sets as semantic targets. Vision-language-conditioned pretraining establishes the geometric backbone, which is adapted to part generation with shared whole-object context. Verification guides targeted condition revision or part regeneration; candidates are accepted only when checks improve, while accepted non-target geometry is preserved. Experiments on 122 PartObjaverse objects demonstrate improved part-level fidelity under spatial conditioning and benefits from agentic bbox refinement. Qualitative results further illustrate prompt-dependent decompositions and downstream articulation.

1 INTRODUCTION

Generating 3D objects as semantically meaningful mesh parts is essential for editing, animation, and simulation (Lin et al., 2025; Zhu et al., 2026; Zhou et al., 2026): downstream applications operate on individual components, not merely an object's overall surface. Recent part-aware generators have made substantial progress toward this goal (Chen et al., 2025a; Lin et al., 2025; Yang et al., 2025b). Yet producing a decomposition that matches a user's intent remains challenging. The same vehicle may need its wheels grouped as one editable unit or represented as separate instances, each with a specified spatial extent. Customization therefore concerns both *what constitutes a part* and *where its geometry should lie*. CubePart enables semantic part control, while MultiCube additionally conditions synthesis on explicit per-part bounding-box (bbox) controls (Zhu et al., 2026; Pun et al., 2026). Yet specifying a valid 3D layout is itself a modeling task(Feng et al., 2023): users may know which parts they want without knowing their precise spatial extents or how to distinguish repeated instances. MultiCube also supports layouts proposed by a large language model (LLM), but proposing a layout does not ensure its consistency with a reference object or the generated geometry. The challenge is thus not only to generate meshes from spatial controls, but to construct reliable controls from user intent and revise them when generation reveals a mismatch.

Recent agentic 3D systems show that LLMs can plan object structure, execute modeling tools, and revise results using feedback (Zhang et al., 2025b; Zhou et al., 2026; Wang et al., 2026; Lin et al., 2026). These capabilities suggest a way to automate part-layout construction, but using them for controllable neural part generation raises two challenges.(Sun et al., 2025; Huang et al., 2024) First, the system must connect part-level decisions to the generator: which instance a request refers to, where it belongs, and which part or condition should change after a failure. Neural feedback loops such as Interact3D address object composition (Shan et al., 2026); our setting requires these decisions within a jointly generated object while preserving its accepted parts. Second, planning a

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

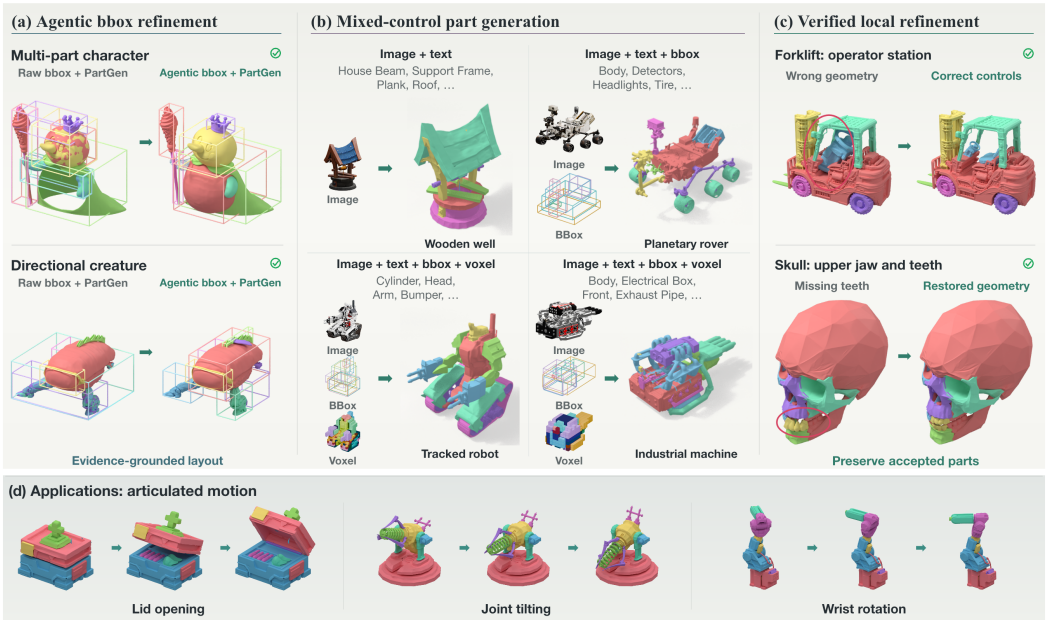

Figure 1: **Daedalus combines agentic planning, controllable part generation, and verified refinement.** **(a)** Evidence-grounded refinement of agent-initialized part bounding boxes. **(b)** Part generation with mixed image, text, bbox, and voxel conditions. **(c)** Verified local refinement while preserving accepted parts. **(d)** Articulated motion enabled by the generated parts.

structure does not provide the geometry needed to realize it. Procedural agents can produce precise hard-surface assemblies, but detailed freeform geometry remains less naturally expressed through their construction operations (Lin et al., 2026). This motivates a division of labor: an agent plans and revises the part structure, while a learned 3D generator synthesizes the geometry.

To address these challenges, we introduce Daedalus, an agentic framework for compositional 3D generation. Given a reference image, a target part schema, and a whole-object geometric prior, it constructs the part layout and refines the generated meshes in a *plan–generate–verify–refine* loop. The agent reviews semantic masks on mesh renderings, and geometric tools use the selected surfaces to derive bboxes. The layout is thus a revisable intermediate state, rather than an input the user must specify accurately in advance. After synthesis, verification distinguishes faulty controls from geometry and assembly errors and guides the choice of a corrective tool. The same generator regenerates selected parts while retaining accepted geometry; only candidates that pass verification replace the previous result. To execute these plans, our mixed-condition generator supports both an individual instance, such as one wheel, and an instance set, such as all wheels, as a semantic target. Different targets can use different available text, image, bbox, and voxel controls in the same generation process. We train with both target granularities and varying condition subsets. Each part processes its own controls, while geometric features exchanged with other parts and the whole object coordinate the assembly. Together, these components connect the agent's part-level decisions to controllable neural synthesis and targeted refinement.

Our contributions are threefold:

1. We introduce an agentic framework for compositional 3D generation that constructs and revises part layouts from semantic intent and geometric evidence, connecting layout planning to joint neural synthesis and verified local refinement while preserving accepted parts.

2. We develop a unified part generator that supports instance-level and set-level semantic targets with heterogeneous per-part controls, coordinating their geometry through shared whole-object context.

3. We demonstrate improved part-level geometric fidelity and competitive whole-object quality on PartObjaverse under spatial conditioning, and showcase flexible control and downstream articulation with the generated parts.

2 Related Work

Part-structured 3D generation. Early structured generators model part hierarchies and relations (Li et al., 2017; Mo et al., 2019; Jones et al., 2020; Gao et al., 2019), while part-factorized implicit and diffusion representations enable local manipulation (Hertz et al., 2022; Koo et al., 2023; Nakayama et al., 2023). Recent work extends part-aware synthesis and reconstruction to more diverse objects. Part123 (Liu et al., 2024) reconstructs part-aware surfaces from multi-view segmentation, whereas PartGen and HoloPart explicitly complete part geometry from image regions or mesh segments (Chen et al., 2025a; Yang et al., 2025a). PartCrafter jointly denoises separate part latents, while AutoPartGen generates parts autoregressively (Lin et al., 2025; Chen et al., 2025b). Other representations emphasize efficient volume packing (Tang et al., 2025), unified geometry–segmentation latents (He et al., 2026), full-resolution per-part grids (Ding et al., 2025), or completion and refinement in exploded and assembled configurations (Sun et al., 2026). These advances establish strong geometric backbones. We build on joint generation, focusing on how semantic requests become revisable controls and how diagnosed failures lead to targeted regeneration.

Semantic grounding and spatial control. Part localization draws on complementary mechanisms: rendered-view label lifting (Liu et al., 2023; Zhou et al., 2023), distilled or learned part feature fields (Yang et al., 2024; Liu et al., 2025), and native 3D point-prompted segmentation (Ma et al., 2025). Such surface evidence is useful for conditioning, but does not itself recover complete part geometry. Generative decomposition therefore incorporates image masks, localized features, and spatial prompts (Yang et al., 2025b; Yan et al., 2025; Zhang et al., 2025a). CubePart (Zhu et al., 2026) introduces open-vocabulary schemas for whole-to-part generation; MultiCube (Pun et al., 2026) combines part-specific text and boxes across whole-object and part synthesis. DreamPartGen (Yu et al., 2026) couples geometry and appearance with relational semantics. We instead emphasize the reliability and granularity of the controls: an agent reviews semantic mask, and triangle-level grounding with 3D instance assignment determines their boxes. Coupled target and condition sampling supports both an individual instance and an instance set within a semantic slot.

Agentic 3D authoring and refinement. Agentic modeling makes 3D representations executable and revisable. ShapeCraft and Articraft generate structured shape programs (Zhang et al., 2025b; Zhou et al., 2026); aDSL and Procedura connect program construction to verifiable constraints and assembly relations (Wang et al., 2026; Lin et al., 2026). Beyond program synthesis, ViSculpt performs localized viewport-guided mesh edits (Pang et al., 2026), and MUSE tracks persistent requirements while preserving satisfied scene content (Xu et al., 2026). Neural regeneration also admits feedback: Interact3D revises image guidance to correct object interactions (Shan et al., 2026), and X-Part already resamples selected parts while retaining others (Yan et al., 2025). Our refinement operates over a persistent neural part state, separating condition revision from mesh re-extraction and target-part regeneration. Accepted non-target parts supply noise-matched context to the same joint generator, while typed evidence governs minimal-target routing and transactional acceptance. The distinction is this diagnosis-to-execution interface with explicit preservation and rollback, rather than agent feedback or selective resampling alone.

3 Method

Our goal is to generate a compositional 3D object from a reference image I, an ordered semantic schema $S = (s_1, \ldots, s_N)$ with N part targets, and a whole-object mesh W providing geometric context. The output is a sequence of part meshes $P = (P_1, \ldots, P_N)$ that follow the schema and assemble in a common coordinate frame. For $i \in \{1, \ldots, N\}$, target s_i specifies either one instance (e.g., a front wheel) or an instance set (e.g., all wheels). Thus, one output mesh may contain several connected components. We obtain W from an upstream image-to-3D model or supply it externally.

The framework comprises mesh-grounded layout planning (Section 3.1), mixed-condition part generation (Section 3.2), and verified local refinement (Section 3.3); see Figure 2.

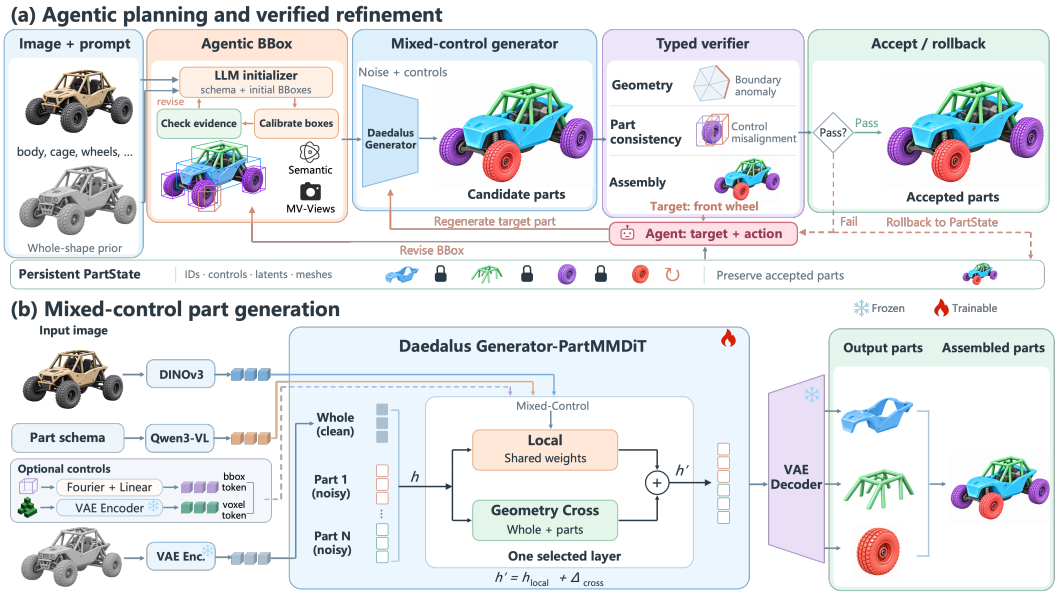

Figure 2: **Schematic overview of Daedalus. (a)** Agentic layout planning and verified local refinement. **(b)** Mixed per-part conditioning with shared whole-shape context.

3.1 Mesh-grounded Semantic Conditioning

Semantic proposals and agentic mask repair. The schema specifies which parts to generate, but not their 3D extents. MultiCube can obtain bbox layouts from an LLM (Pun et al., 2026); however, predicting coordinates directly requires resolving scale, orientation, and occlusion without explicit surface measurements. We instead ground each target in the geometry of W. After normalizing the mesh, we render RGB images, depth, and triangle-ID buffers from calibrated orthographic views. SAM3.1, the updated release of SAM 3 (Carion et al., 2025; Meta, 2026), proposes masks for each target's text query, retaining modifiers such as *front-left*. A pretrained vision-language model (VLM) agent reviews whether the masks contain the requested part and the correct number of instances. It selects valid candidates, removes unrelated components, or requests SAM3.1 to retry with a tighter 2D bbox prompt. SAM3.1 determines pixel boundaries; the agent decides which evidence to retain or revise.

From masks to 3D instances. We next associate the selected image regions with mesh triangles. Let v index a view, p a foreground pixel, $F_v(p)$ its visible triangle ID, and $M_i^v(p) \in \{0,1\}$ the accepted mask for target i. For a mesh triangle f visible in view v, its mask support is

$$r_{if}^v = \frac{\left|\{p : F_v(p) = f,\ M_i^v(p) = 1\}\right|}{\left|\{p : F_v(p) = f\}\right|}, \tag{1}$$

where $|\cdot|$ denotes cardinality. We aggregate these observations across views to identify the supporting surface of each target; invisible triangles contribute no observation. A mask may cover several instances of the same category. For example, if a mask contains both front and rear wheels, we use their 3D positions and the schema's directional labels to assign them to the corresponding targets. If the target is instead *all wheels*, we retain both. Assignment separates competing instances of the same category without forbidding legitimate nested parts.

Bbox fitting and correction. After filtering isolated outlier triangles, let X_i be the supporting vertex positions for target i. For nonempty X_i, we fit an axis-aligned bbox

$$\widehat{B}_i = \left[\min_{x \in X_i} x - \delta_i \mathbf{1},\ \max_{x \in X_i} x + \delta_i \mathbf{1}\right], \tag{2}$$

where extrema are coordinate-wise, $\delta_i \geq 0$ is a margin bounded by object scale and part type, and $\mathbf{1}$ is the three-dimensional all-ones vector. Thus, bbox coordinates come from the mesh rather than

language-model predictions. Geometric checks flag missing support, inconsistent instance ordering, and excessive bbox extent. These flags identify what needs review: missing or contaminated masks call for candidate selection or a segmentation retry; confused instances call for reassignment using their 3D positions. The geometric tools then refit the affected bbox from the revised support. A bbox is retained only when its support and identity checks pass; insufficient evidence leaves the target unresolved rather than triggering arbitrary expansion.

3.2 Mixed-condition Part Generation

Controls and geometric representation. The generator combines semantic, visual, and spatial evidence for each target. We denote its available controls by $C_i = (T_i, I, B_i, V_i)$: a schema-contextualized text description T_i, the shared reference image I, a bbox $B_i \in \mathbb{R}^6$ from Section 3.1 (minimum and maximum coordinates), and optional voxelized mesh-surface support V_i providing coarse structure within that bbox. Missing conditions are masked out independently for each part; voxel controls are used only with a compatible bbox. Frozen DINOv3 (Siméoni et al., 2025) and Qwen3-VL (Qwen Team, 2025) encode images and text, respectively; trainable projections map their features into the generator's token space. Bboxes use Fourier embeddings of their center, size, face centers, and corners, with learned role embeddings. A shared, frozen shape variational autoencoder (VAE), with encoder E and decoder D, represents each mesh as L latent vectors of dimension d (Zhang et al., 2023). It maps the whole mesh to $z_W = E(W) \in \mathbb{R}^{L\times d}$ and each ground-truth training part P_i^{gt} to $z_i = E(P_i^{\mathrm{gt}})$; optional voxel-surface controls use the same encoder followed by a separate projection. The whole mesh, parts, and spatial controls share the object coordinate frame.

Vision-Language-Conditioned Geometric Generator. We train the generator in two successive stages: vision-language-conditioned 3D pretraining and mixed-condition part generation. In Stage 1, the generator learns whole-object geometry with visual and textual conditioning from an image I and a whole-object description T_W. The transformer predicts the flow velocity of a noisy whole-object latent from the available condition tokens using flow matching (Lipman et al., 2023). In Stage 2, we initialize the weight-shared local transformer from the pretrained generator and fine-tune it jointly with the newly introduced part-conditioning and cross-group modules. This extends the image–text interface to mixed per-part image, text, bbox, and voxel controls. The whole-object latent z_W now serves as clean geometric context rather than a prediction target; only part latents are noised and supervised. This transfers the whole-object geometric prior to controllable part generation.

Part-local control with shared geometric context. We build on the local and cross-part processing of PartCrafter and CubePart (Lin et al., 2025; Zhu et al., 2026). Where MultiCube's part stage combines text and bbox conditions (Pun et al., 2026), our interface allows different parts to receive different subsets of text, image, bbox, and voxel evidence in one forward pass. We organize tokens into $N + 1$ groups: the whole object ($i = 0$) and the requested parts ($i = 1, \ldots, N$). Weight-shared local blocks process each group with its own conditions, while cross-group blocks exchange only geometry tokens. Let H_i^ℓ contain group i's condition and geometry tokens at layer ℓ, and let $H_{i,Z}^\ell$ be its geometry subsequence. At selected layers, the two branches act in parallel:

$$H_i^{\ell+1} = \mathrm{Local}_\ell(H_i^\ell) + \mathrm{Scatter}_i\big[\Delta_\ell\big(\mathrm{Concat}_{j=0}^{N} H_{j,Z}^\ell\big)\big]. \tag{3}$$

Here Local_ℓ is the local transformer block, Δ_ℓ is a cross-group residual block, and Concat concatenates geometry sequences. $\mathrm{Scatter}_i$ places group i's residual in its geometry rows and zeros in its condition rows; other layers use only the local block. Direct controls therefore stay attached to their target, while geometry features carry information across parts. The whole group receives the full schema and clean z_W, but no image, bbox, or voxel controls. Its hidden features evolve within the network; the sampler updates only part latents.

Training for different targets and available controls. We vary both *what a part target contains* and *which controls describe it*. For an annotated group of repeated instances, training selects either one instance or the union of the instances—for example, one wheel or all wheels. The target mesh, text, and bbox are changed together so that they describe the same selection; the group occupies one target slot in either case. Next, we sample an allowed condition subset independently for each part. Thus, one part can receive text and image features while another also receives a bbox or voxel support. The reference image is encoded once, but access to its features is selected per part.

Voxel controls always accompany a compatible bbox; bbox-only examples receive small boundary perturbations to model imperfect spatial inputs. Joint condition dropout is used to support classifier-free guidance (Ho & Salimans, 2022).

Flow training and mesh decoding. Both stages use a linear flow-matching path, with z_W as the target in pretraining and z_i in part adaptation. For each object, all parts share a noise level $\sigma \in [0, 1]$, with independent standard-normal noise matrices ϵ_i:

$$x_{i,\sigma} = (1 - \sigma)z_i + \sigma\epsilon_i, \qquad v_i^* = \epsilon_i - z_i, \tag{4}$$

where $x_{i,\sigma}$ is the noisy part latent and v_i^* its target velocity. We minimize the mean squared velocity error (Appendix A). The whole latent is a condition, not a prediction target. At inference, we jointly integrate part latents from $\sigma = 1$ to 0 and decode them with D; adaptive mesh extraction is detailed in Appendix B.

3.3 Verification-guided Local Refinement

After generation, a verifier checks mesh validity, agreement with the available controls, and assembly consistency to identify parts requiring revision. Accepted controls, latents, meshes, and sampling noise are stored in `PartState` throughout refinement.

Targeted correction. The controller selects an affected part and a corrective action from the verifier's report. Suspected extraction artifacts trigger mesh re-extraction; localization errors trigger evidence-guided bbox revision and regeneration; geometry defects trigger resampling with the existing controls. Inconsistent voxel controls are disabled after bbox changes. The agent selects actions, while geometric tools and the trained generator execute them; it does not directly edit mesh vertices. Corrections remain limited by the available evidence: semantic assignment errors require upstream review, and bbox overlap alone does not establish a mesh intersection.

Preservation and acceptance. We reuse the same generator for local resampling, following known-content conditioning and selective part generation (Lugmayr et al., 2022; Yan et al., 2025). Non-target latents provide context at each noise level, constructed from their accepted values and fixed noise using the flow path in Equation (4). Only target latents are updated and decoded, leaving accepted non-target meshes unchanged. A candidate replaces the saved state only when it reduces the reported error without worsening the available validity and assembly checks; otherwise, the system retains the previous result. Refinement stops when checks pass, improvement stalls, or the allotted budget is exhausted.

4 Experiments

We evaluate the fidelity of generated part meshes and their response to semantic and spatial controls.

4.1 Experimental Setup

Datasets. Following CubePart (Zhu et al., 2026), we evaluate part-mesh generation on PartObjaverse (Yang et al., 2024; Deitke et al., 2023). We retain objects with fewer than eight parts and further filter for geometrically complete assets, yielding 122 evaluation objects.

Baselines. We compare against PartCrafter (Lin et al., 2025), PartPacker (Tang et al., 2025), OmniPart (Yang et al., 2025b), and CubePart (Zhu et al., 2026). These methods expose different interfaces: PartCrafter and PartPacker generate parts from images; OmniPart uses image-region guidance; and CubePart decomposes a whole mesh according to a text part schema. We evaluate Daedalus with image and text controls, then with additional bboxes, and finally with bboxes and voxel support. The first two settings decompose an input whole mesh; the voxel-conditioned setting directly generates parts without whole-mesh input. Inputs therefore differ across settings and methods.

Metrics. We evaluate part and whole-object geometry using Chamfer distance (CD) and F-score. CD sums bidirectional mean nearest-neighbor distances without squaring; F-score uses a threshold

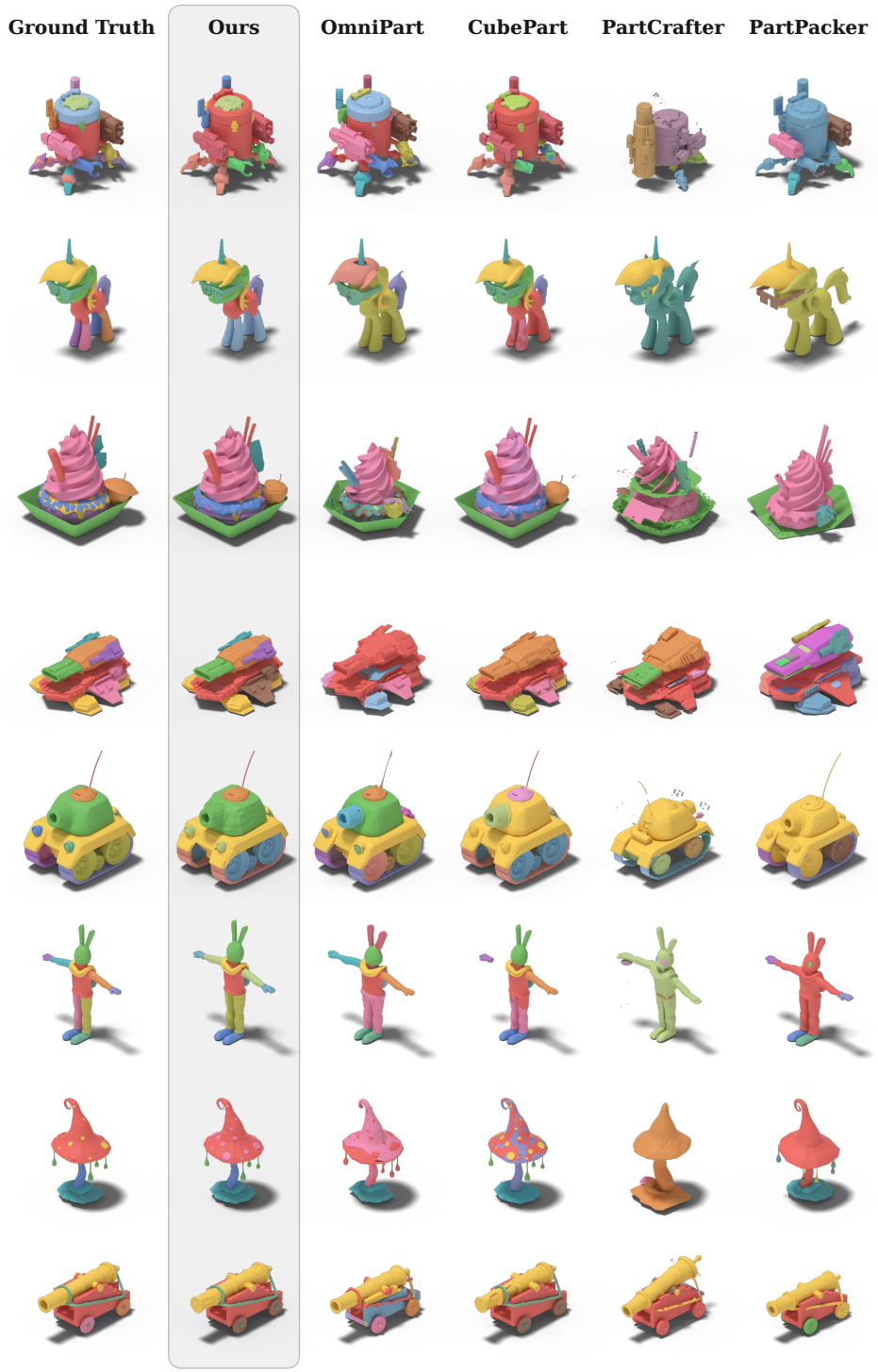

Figure 3: **Qualitative comparison of part-mesh generation.** Each row presents an object with its reference decomposition and the assembled outputs of Daedalus, OmniPart, CubePart, PartCrafter, and PartPacker, from left to right. The examples cover articulated characters, vehicles, and stylized props with varied part structures. Our results preserve distinct components and fine geometry, including the rabbit's hands and limbs and the mushroom's hanging details. Some baseline outputs merge neighboring components, omit small parts, or exhibit fragmented surfaces despite recovering the overall shape. Colors identify separate mesh parts rather than surface textures; color identities need not correspond across methods.

of 0.1. We additionally report Box IoU between normalized whole-object bboxes, which measures global extent agreement rather than part localization. Sampling, normalization, matching, and aggregation are detailed in Appendix E.

Implementation Details. We implement our framework in PyTorch and train it in two stages. Training data, conditioning schedules, optimization settings, and compute costs are provided in Appendix A.

4.2 Experimental Results

Quantitative comparison. Table 1 reports geometric fidelity under the listed conditioning configurations. With image, text, and bbox controls, Daedalus obtains a part-level CD of 0.062 and an F-score of 0.941, compared with 0.336 and 0.701 for CubePart, the strongest external baseline on these metrics in the table. This configuration also achieves the highest Box IoU (0.969) and the lowest holistic CD (0.043) among the reported settings. Without explicit spatial controls, our image–text configuration obtains a lower part-level CD than the external baselines, but its part-level F-score remains below CubePart's. The gains are therefore not uniform across conditioning settings and metrics.

Table 1: **Quantitative evaluation for multi-part generation.** We compute Chamfer distance and F-scores at part and holistic levels, alongside bounding box agreement at the holistic level. Best and second-best values across the listed settings are highlighted in **red** and **blue**, respectively. Our first two settings use whole-mesh input; the voxel setting directly generates parts without it. Inputs differ across methods; see the text for their conditioning interfaces.

Method	Input	Part-Level		Holistic-Level		
		CD ↓	F-score ↑	Box IoU ↑	CD ↓	F-score ↑
PartCrafter (Lin et al., 2025)	Image	0.626	0.262	0.699	0.191	0.661
PartPacker (Tang et al., 2025)	Image	0.586	0.338	0.788	0.132	0.799
OmniPart (Yang et al., 2025b)	Image + masks	0.434	0.475	0.861	0.101	0.897
CubePart (Zhu et al., 2026)	Mesh + text	0.336	0.701	0.958	**0.049**	0.975
Daedalus (Ours)	img + prompt	0.312	0.675	**0.967**	0.050	**0.977**
	img + prompt + bbox	**0.062**	**0.941**	**0.969**	**0.043**	**0.987**
	img + prompt + bbox + voxel	**0.060**	**0.969**	0.937	0.053	**0.987**

Qualitative comparison. Figure 3 compares the assembled, part-colored meshes with their reference decompositions. The examples highlight both surface quality and the separation of neighboring components. Further discussion is provided in Appendix F.

4.3 Ablation Study

Effect of input modalities. The three Daedalus configurations in Table 1 compare different control settings. Adding bboxes to image–text conditioning reduces part-level CD from 0.312 to 0.062 and increases F-score from 0.675 to 0.941, highlighting the benefit of explicit part localization. Figure 4(a) illustrates these effects: bbox conditioning separates the wings from adjacent body regions in the winged character, while spatial controls clarify the rabbit's limb boundaries and reduce fragments near its base. The voxel-conditioned direct-generation setting yields smoother local surfaces, with part-level CD of 0.060 and F-score of 0.969, but lowers holistic Box IoU from 0.969 to 0.937 and raises holistic CD from 0.043 to 0.053. Unlike the first two settings, it has no whole-mesh input to anchor the overall shape. This may allow greater variation in global proportions and surface placement, explaining the weaker holistic CD and Box IoU despite strong part-level fidelity. The comparison therefore reflects different conditioning regimes, not an isolated effect of adding voxels.

To strengthen textual control, Stage 1 trains the geometric generator with both visual and textual conditioning, which is transferred to part generation in Stage 2. Figure 4(b) shows how different part-label prompts control the decomposition of the same input image. For example, the robot can be decomposed into body and wheels, or into finer components such as the head and arms; the

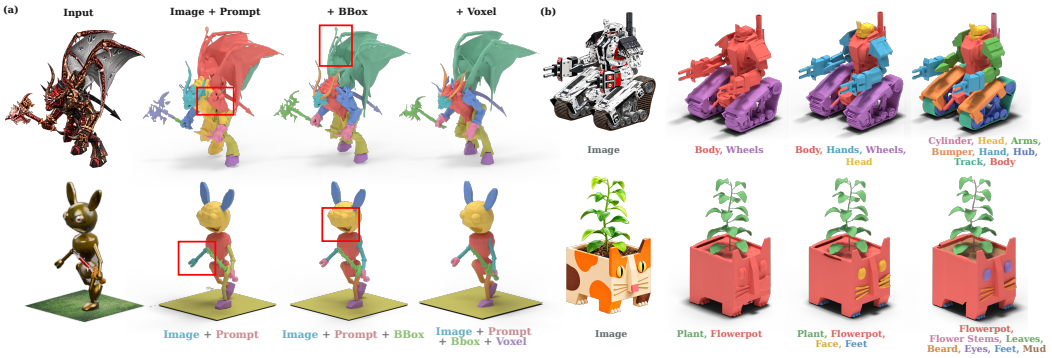

Figure 4: **Generation under different control settings.** **(a)** Image–text conditioning, with additional bbox or bbox–voxel controls. The voxel setting uses direct generation without whole-mesh input. **(b)** Different part-label prompts produce decompositions with different semantic components and levels of granularity for the same input image. Colors distinguish output parts.

flowerpot example similarly separates additional details when requested. These results illustrate control over both part semantics and decomposition granularity.

Effect of bbox refinement. Table 2 compares predicted and agentically refined bboxes as conditions for part generation. Refinement reduces part-level CD from 0.1211 to 0.1129 and increases F-score from 0.8152 to 0.8376. It also improves holistic CD from 0.0567 to 0.0520, F-score from 0.9522 to 0.9619, and Box IoU from 0.9442 to 0.9537. The consistent gains support refining spatial conditions before generating the parts, benefiting both individual components and the assembled shape. GT-bbox results from Table 1 are included for reference.

Table 2: **Effect of agentic bbox refinement on part generation.** All settings use image, text, and bbox conditions. GT-bbox results from Table 1 are included for reference. Box IoU measures whole-object bbox agreement.

BBox condition	Part-Level		Holistic-Level		
	CD ↓	F-score ↑	Box IoU ↑	CD ↓	F-score ↑
Predicted	0.1211	0.8152	0.9442	0.0567	0.9522
Agentically refined	**0.1129**	**0.8376**	**0.9537**	**0.0520**	**0.9619**
Ground truth (reference)	0.0620	0.9410	0.9690	0.0430	0.9870

5 Conclusion

We presented Daedalus, a framework that connects agentic planning with controllable neural part generation. The agent grounds semantic requests in a revisable part layout, while a whole-guided generator synthesizes geometry under mixed image, text, bbox, and voxel controls. Verification supports targeted correction while preserving accepted parts. Experiments on PartObjaverse demonstrate strong part-level fidelity under spatial conditioning and improved generation with agentically refined bboxes. The resulting part structure also supports downstream editing and articulation.

Limitations and future work. Our framework depends on the quality of the whole-object prior and semantic grounding, making missing geometry, occlusion, and ambiguous part identities challenging. Verification relies on available geometric checks and evidence rather than guaranteeing correct semantics, watertightness, or collision-free assembly. Repeated rendering and regeneration also add computational cost. Future work will investigate more reliable grounding and geometric verification, and learning from refinement trajectories to reduce repeated sampling and improve correction efficiency.

AI USE STATEMENT

Generative AI was used to polish the language of the manuscript.

ETHICS STATEMENT

This work studies controllable generation of part-structured 3D objects. We do not identify ethical concerns specific to the proposed method or experiments.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide training configurations, architecture details, conditioning and refinement procedures, and evaluation protocols in Appendices A–E. Section 4 specifies the evaluation subset and conditioning settings used in the reported comparisons.

REFERENCES

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. *arXiv*, abs/2511.16719, 2025.

Minghao Chen, Roman Shapovalov, Iro Laina, Tom Monnier, Jianyuan Wang, David Novotný, and Andrea Vedaldi. Partgen: Part-level 3d generation and reconstruction with multi-view diffusion models. In *CVPR*, 2025a.

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

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023.

Lihe Ding, Shaocong Dong, Yaokun Li, Chenjian Gao, Xiao Chen, Rui Han, Yihao Kuang, Hong Zhang, Bo Huang, Zhanpeng Huang, Zibin Wang, Dan Xu, and Tianfan Xue. Fullpart: Generating each 3d part at full resolution. *arXiv*, abs/2510.26140, 2025.

Weixi Feng, Wanrong Zhu, Tsu-Jui Fu, Varun Jampani, Arjun R. Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. In *NeurIPS*, 2023.

Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. SDM-NET: deep generative network for structured deformable mesh. *ACM Trans. Graph.*, 38, 2019.

Xufan He, Yushuang Wu, Xiaoyang Guo, Chongjie Ye, Jiaqing Zhou, Tianlei Hu, Xiaoguang Han, and Dong Du. Unipart: Part-level 3d generation with unified 3d geom-seg latents. In *CVPR*, 2026.

Amir Hertz, Or Perel, Raja Giryes, Olga Sorkine-Hornung, and Daniel Cohen-Or. SPAGHETTI: editing implicit shapes through part aware generation. *ACM Trans. Graph.*, 41, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv*, abs/2207.12598, 2022.

Ian Huang, Guandao Yang, and Leonidas J. Guibas. Blenderalchemy: Editing 3d graphics with vision-language models. In *ECCV*, 2024.

R. Kenny Jones, Theresa Barton, Xianghao Xu, Kai Wang, Ellen Jiang, Paul Guerrero, Niloy J. Mitra, and Daniel Ritchie. Shapeassembly: learning to generate programs for 3d shape structure synthesis. *ACM Trans. Graph.*, 39, 2020.

Juil Koo, Seungwoo Yoo, Minh Hieu Nguyen, and Minhyuk Sung. SALAD: part-level latent diffusion for 3d shape generation and manipulation. In *ICCV*, 2023.

Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao (Richard) Zhang, and Leonidas J. Guibas. GRASS: generative recursive autoencoders for shape structures. *ACM Trans. Graph.*, 36, 2017.

Youtian Lin, Yikang Yang, Zhanpeng Hu, Mengqi Zhou, Feihu Zhang, Xun Cao, Jiaheng Liu, and Yao Yao. Procedura: Agentic 3d modeling with procedural control. *arXiv*, abs/2608.26238, 2026.

Yuchen Lin, Chenguo Lin, Panwang Pan, Honglei Yan, Yiqiang Feng, Yadong Mu, and Katerina Fragkiadaki. Partcrafter: Structured 3d mesh generation via compositional latent diffusion transformers. In *NeurIPS*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

Anran Liu, Cheng Lin, Yuan Liu, Xiaoxiao Long, Zhiyang Dou, Hao-Xiang Guo, Ping Luo, and Wenping Wang. Part123: Part-aware 3d reconstruction from a single-view image. In *SIGGRAPH*, 2024.

Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In *CVPR*, 2023.

Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *ICCV*, 2025.

Andreas Lugmayr, Martin Danelljan, Andrés Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022.

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*, abs/2509.06784, 2025.

Meta. SAM 3.1 release notes. Official software release, March 27, 2026, 2026. URL `https://github.com/facebookresearch/sam3/blob/main/RELEASE_SAM3p1.md`.

Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J. Mitra, and Leonidas J. Guibas. Structurenet: hierarchical graph networks for 3d shape generation. *ACM Trans. Graph.*, 38, 2019.

George Kiyohiro Nakayama, Mikaela Angelina Uy, Jiahui Huang, Shi-Min Hu, Ke Li, and Leonidas J. Guibas. Difffacto: Controllable part-based 3d point cloud generation with cross diffusion. In *ICCV*, 2023.

Bo Pang, Jiaqi Pan, Xiaocheng Zhang, Jiacheng Xu, Guoping Wang, and Peng-Shuai Wang. Visculpt: Visual-centric agentic geometry editing. *arXiv*, abs/2608.24169, 2026.

Ava Pun, Kangle Deng, Yiheng Zhu, Jun-Yan Zhu, Maneesh Agrawala, and Tinghui Zhou. Multicube: Compositional 3d generation with part-level semantic and spatial control. *arXiv*, abs/2608.20448, 2026.

Qwen Team. Qwen3-vl technical report. *arXiv*, abs/2511.21631, 2025.

Hui Shan, Keyang Luo, Ming Li, Sizhe Zheng, Yanwei Fu, Zhen Chen, and Xiangru Huang. Interact3d: Compositional 3d generation of interactive objects. *arXiv*, abs/2603.16085, 2026.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun 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*, abs/2508.10104, 2025.

Chunyi Sun, Junlin Han, Weijian Deng, Xinlong Wang, Zishan Qin, and Stephen Gould. 3d-gpt: Procedural 3d modeling with large language models. In *3DV*, 2025.

Wanhu Sun, Zhongjin Luo, Heliang Zheng, Jiahao Chang, Chongjie Ye, Huiang He, Shengchu Zhao, Rongfei Jia, and Xiaoguang Han. Ei-part: Explode for completion and implode for refinement. *arXiv*, abs/2603.14021, 2026.

Jiaxiang Tang, Ruijie Lu, Max 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. In *NeurIPS*, 2025.

Rui-Huan Wang, Si-Tong Wei, Jia-Qi He, Heng-Yi Wei, Baoquan Chen, and Peng-Shuai Wang. adsl: Agentic 3d creation via joint agent-program design. *arXiv*, abs/2608.17975, 2026.

Ruijie Xu, Xinnan Zhu, Jiayu Ying, Daoguo Dong, Yuzhou Ji, and Xin Tan. MUSE: agentic 3d scene authoring via memory-grounded incremental requirement satisfaction. *arXiv*, abs/2606.14168, 2026.

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*, abs/2509.08643, 2025.

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*, abs/2411.07184, 2024.

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*, abs/2504.07943, 2025a.

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. In *SIGGRAPH Asia*, 2025b.

Tianjiao Yu, Xinzhuo Li, Muntasir Wahed, Jerry Xiong, Yifan Shen, Ying Shen, and Ismini Lourentzou. Dreampartgen: Semantically grounded part-level 3d generation via collaborative latent denoising. *arXiv*, abs/2603.19216, 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.*, 42, 2023.

Longwen Zhang, Qixuan Zhang, Haoran Jiang, Yinuo Bai, Wei Yang, Lan Xu, and Jingyi Yu. BANG: dividing 3d assets via generative exploded dynamics. *arXiv*, abs/2507.21493, 2025a.

Shuyuan Zhang, Chenhan Jiang, Zuoou Li, and Jiankang Deng. Shapecraft: LLM agents for structured, textured and interactive 3d modeling. *arXiv*, abs/2510.17603, 2025b.

Matt Zhou, Ruining Li, Xiaoyang Lyu, Zhaomou Song, Zhening Huang, Chuanxia Zheng, Christian Rupprecht, Andrea Vedaldi, and Shangzhe Wu. Articraft: An agentic system for scalable articulated 3d asset generation. *arXiv*, abs/2605.15187, 2026.

Yuchen Zhou, Jiayuan Gu, Xuanlin Li, Minghua Liu, Yunhao Fang, and Hao Su. Partslip++: Enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation. *arXiv*, abs/2312.03015, 2023.

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. In *SIGGRAPH*, 2026.

SUPPLEMENTARY MATERIAL

This supplement details the training configuration, generator architecture, mesh-grounded conditioning, local refinement, and evaluation protocol. It complements the method and experiments in the main paper.

A ADDITIONAL TRAINING DETAILS

Two-stage training. Stage 1 pretrains a vision-language-conditioned 3D generator on approximately 4 million public 3D objects for 200K steps with a batch size of 512, progressively increasing image resolution from 512×512 to 768×768. Stage 2 initializes the weight-shared local transformer from this generator and jointly fine-tunes it with the new part-conditioning and cross-group modules for mixed-condition part generation. We train for 100K steps with a batch size of 112 at 768×768 on approximately 500K part-annotated objects from public datasets and internal game assets, with validation splits strictly held out. The image, text, and shape encoders remain frozen; the generator and condition projections are trained.

Resolution and conditioning schedules. In Stage 1, image resolution increases from 512×512 to 576×576, 640×640, 704×704, and 768×768 at steps 5K, 15K, 25K, and 35K, respectively. The image-conditioning probability is linearly increased to 30%, while the text-conditioning probability decreases from 100% to 70%. Stage 2 uses a fixed resolution of 768×768 and progressively increases the conditioning probabilities for images from 30% to 50%, bounding boxes from 10% to 30%, and voxels from 5% to 15%. Voxel conditioning requires a compatible bbox, so the available modalities are not sampled as unconstrained independent choices. All conditions are jointly dropped with probability 5% for classifier-free guidance.

Optimization and compute. Across both stages, AdamW uses a learning rate of 10^{-4} and weight decay of 0.01, with gradient norm clipping at 1.0. We use BF16 mixed precision and maintain an exponential moving average (EMA) of model weights with decay 0.9999. Stage 1 and Stage 2 require approximately 18,800 and 10,800 GPU-hours, respectively.

Flow-matching objective. Using the noisy part latents and target velocities defined in Equation (4), let $v_{\theta,i}$ be the velocity for target i predicted by the joint generator with parameters θ. We minimize

$$\mathcal{L} = \mathbb{E}\left[\frac{1}{NLd}\sum_{i=1}^{N} \|v_{\theta,i}(x_{1:N,\sigma}, \sigma, z_W, C_{1:N}) - v_i^*\|_F^2\right], \tag{5}$$

where $1{:}N$ indexes all part targets, $\|\cdot\|_F$ is the Frobenius norm, and the expectation covers training objects, target and condition choices, noise, and noise levels. The whole latent is a condition, not a prediction target.

B GENERATOR ARCHITECTURE AND CONTROL INTERFACE

Geometry representation and backbone. The shared shape VAE represents each whole object or part using $L = 1024$ latent vectors of dimension $d = 64$. During generation, all part meshes retain their positions and scales in the object coordinate frame; they are not independently recentered. The 768×768 configuration uses 35 transformer layers with hidden dimension 1536, 12 attention heads, and a SwiGLU feed-forward dimension of 4096. Geometry-only cross-group blocks are inserted at zero-indexed layers 0, 8, 17, and 26. As in Equation (3), the local and cross-group branches read the same layer input. The cross-group residual is added only to geometry-token rows, not directly to condition-token rows.

Image, text, and bbox tokens. The frozen DINOv3 encoder provides image features, and the frozen Qwen3-VL text encoder provides semantic features; trainable projections map both into the transformer space. At 768×768, the image patch grid has $48 \times 48 = 2304$ tokens. Each part receives its own schema-contextualized text and selected controls. An axis-aligned bbox is stored as minimum and maximum 3D coordinates. Its encoding uses 16 roles: the center, the size vector, six

face centers, and eight corners. Each three-dimensional vector is encoded with 10-frequency Fourier features, projected, and augmented with a learned role embedding. The size vector describes extent rather than a surface location. Optional voxel-surface controls are encoded by the shared shape VAE and a separate condition projection.

Semantic targets and missing conditions. A semantic slot specifies an output target, not necessarily one connected component. For example, *wheels* may denote a union of wheel instances, whereas *front-left wheel* denotes one instance. Training selects the target geometry, text, and spatial controls together so that their granularity agrees. Condition availability is then selected per slot. Image features can be computed once and made available to different slots without requiring every part to use the same modalities. Bbox-only training examples receive bounded boundary perturbations; bbox–voxel pairs remain geometrically consistent.

Whole-conditioned and direct generation. The first two configurations in Table 1 use a whole mesh as geometric context. Its latent remains clean and is neither updated by the sampler nor included in the part-prediction loss. Its hidden features nevertheless evolve inside the transformer and participate in cross-group interaction. The voxel-conditioned configuration in that table instead generates parts without a whole-mesh input. Thus, its spatial evidence comes from the provided controls rather than a complete reference surface. The comparison changes both condition availability and the presence of whole-object context; it is not a controlled voxel-only ablation.

Mesh extraction. The decoder supports global field sampling and adaptive local sampling. For adaptive mesh extraction, we first locate each part on a coarse grid, then sample its decoded field more densely within that region. This allocates finer spatial sampling to small parts without changing their generated latents. The resulting coordinates remain in the common object frame. Consequently, re-extraction can address discretization artifacts but cannot recover a part missing from the generated field.

C Details of Mesh-grounded Bbox Construction

Coordinate and identity consistency. The whole mesh, camera poses, triangle-ID buffers, and fitted boxes share one normalized coordinate frame. Each schema entry has a stable part identity throughout candidate selection, geometric fitting, and generation. Directional terms are interpreted in the declared object frame rather than reassigned independently in each view. If orientation or instance identity is ambiguous, the target remains unresolved until additional evidence is available.

Agent decisions versus geometric measurements. The segmentation model proposes pixel masks from the target text. The agent checks semantic identity and instance coverage, selects candidates, removes unrelated components, or requests segmentation within a tighter image-space box. Accepted masks are associated with visible mesh triangles using the triangle-ID buffers and the support ratio in Equation (1). The agent therefore selects and revises evidence; it does not directly predict final mesh vertices or fit unconstrained 3D coordinates. Occluded triangles supply no direct mask observation, so this procedure does not imply complete amodal segmentation.

Repeated instances and fitting. Same-category targets are assigned jointly using their spatial support and directional relations. A mask containing several wheels may support multiple single-instance targets, while a set-level target retains the union of the requested instances. Exclusivity applies to competing same-category instances, not indiscriminately to nested semantic parts. Outlying triangles are filtered before coordinate-wise extrema are used to fit the bbox in Equation (2). Checks cover finite ordered coordinates, nonempty geometric support, schema agreement, instance ordering, and excessive extent. A failed check triggers evidence review or reassignment rather than unconditional box expansion.

Traceability and initialization variants. We retain mask candidates, accepted masks, triangle assignments, fitted boxes, and revision records so that a localization error can be traced to its evidence. The earlier LLM-initialized box path is distinct from the mask-grounded construction described above: it predicts initial spatial coordinates and requires their alignment to the mesh frame. These

paths share the downstream generator interface but not the same initialization assumptions. Likewise, manual overrides must be distinguished from automatic decisions when interpreting results.

D Local Refinement and State Preservation

Diagnosis and action scope. The verifier distinguishes mesh health, control consistency, and assembly evidence. Mesh-health checks include empty outputs, non-finite coordinates, degenerate faces, boundary or non-manifold edges, and fragmentation. Boundary edges alone do not establish an erroneous hole: open surfaces and multi-component semantic targets may be intentional. Similarly, bbox overlap is only a coarse candidate for assembly review, not proof of mesh intersection. The response depends on the source of the failure:

1. **Extraction artifact:** re-extract the existing latent before changing the generated geometry.
2. **Localization error:** revise the bbox from geometric evidence and regenerate the affected target. Disable stale voxel controls if they are inconsistent with the revised box.
3. **Generation error:** retain valid controls and resample the target with a new noise realization.
4. **Semantic or relation ambiguity:** request upstream review, or select the smallest coupled target set justified by the available evidence. An unresolved upstream request is not counted as a successful repair.

An unintended opening caused by insufficient field-sampling resolution may be addressed by re-extraction. If the defect persists in the decoded field, target regeneration is required; neither action guarantees a watertight result without subsequent verification.

Noise-matched context for local resampling. Let $\mathcal{T} \subseteq \{1, \dots, N\}$ be the selected target slots and $\mathcal{K} = \{1, \dots, N\} \setminus \mathcal{T}$ the preserved slots. For each $j \in \mathcal{K}$, let z_j^* be its accepted clean latent and ϵ_j^* its saved, fixed noise. At the current noise level $\sigma \in [0, 1]$, its context state is

$$x_{j,\sigma}^{\mathrm{ctx}} = (1 - \sigma) z_j^* + \sigma \epsilon_j^*, \qquad j \in \mathcal{K}. \tag{6}$$

The joint model receives the complete ordered part sequence, with target states and these context states in their original slots. The sampler updates only slots in $\mathcal{T}$; context states are reconstructed at the next noise level using Equation (6). At the end of sampling, only target candidates are decoded. Accepted non-target meshes are reused directly, rather than regenerated and compared for approximate similarity.

Candidate acceptance and rollback. `PartState` stores accepted controls, clean latents, fixed noise, meshes, and their provenance. A candidate is produced separately from this state. Acceptance requires improvement of the diagnosed target error, no regression in available validity and assembly checks, and unchanged non-target artifacts. Hash checks protect saved non-target meshes, latents, noise, and metadata; they establish preservation of these artifacts, not semantic correctness or physical compatibility. If accepted, the target's controls, latent, noise, and mesh are committed together. Otherwise, the candidate is rejected and the previous accepted state remains active. In particular, a failed bbox revision does not leave a new box attached to an old accepted mesh. Refinement stops when the checks pass, improvement stalls, or the configured budget is exhausted; unresolved issues remain recorded.

Scope of verification. The implemented loop supports local neural execution and transactional acceptance. Semantic and mask scores, as well as evidence of actual triangle-level collision or self-intersection, currently depend on externally supplied structured observations where available. Upstream schema and mask revision requests are routed but are not yet fully connected to automatic callbacks. Thus, the loop does not establish a general guarantee of watertightness, semantic correctness, or collision-free assembly.

E Detailed Evaluation Protocol

Evaluation objects and conditioning regimes. We report both part-level and whole-object geometry, since a plausible assembled shape does not necessarily imply an accurate decomposition. The evaluation subset contains 122 geometrically complete PartObjaverse objects with fewer than eight annotated parts. The three generator configurations and the external baselines use the interfaces described in Section 4. In particular, whole-mesh-conditioned decomposition and voxel-conditioned direct generation provide different geometric evidence and should not be interpreted as identical-input comparisons.

Sampling, alignment, and distance. We sample 2,048 points per part and 8,192 per assembled object. Predictions and references are orientation-aligned and independently normalized to a longest bbox edge of 2. For predicted and reference point sets X and Y, respectively, Chamfer distance is

$$\mathrm{CD}(X,Y) = \frac{1}{|X|}\sum_{x\in X}\min_{y\in Y}\|x-y\|_2 + \frac{1}{|Y|}\sum_{y\in Y}\min_{x\in X}\|y-x\|_2, \tag{7}$$

where $|X|$ and $|Y|$ are point counts and $\|\cdot\|_2$ is Euclidean distance. The two directional means are summed, not averaged, and the distances are not squared. For F-score, precision is the fraction of predicted points within distance 0.1 of a reference point, and recall is the corresponding fraction of reference points covered by predictions. F-score is their harmonic mean, with value zero when both are zero.

Part correspondence and aggregation. Parts are matched by normalized semantic names for CubePart and Daedalus, and greedily by CD for the other baselines. For CubePart and Daedalus, exact name-set agreement is checked before evaluation. Other baselines use greedy matching in reference-part order: each reference part selects the remaining prediction with the smallest CD, breaking distance ties by F-score. In the metric routine, an unmatched reference part receives CD $= 1$ and F-score $= 0$. Extra predicted parts are excluded from part-level averages but remain in the assembled prediction for whole-object metrics. Part scores are averaged within each object; all metrics are then averaged over the common valid evaluation set.

Interpretation of Box IoU. Box IoU is the intersection volume divided by the union volume of the axis-aligned bboxes of the normalized assembled prediction and reference. It measures agreement of whole-object extents, not overlap between individual predicted and reference part boxes. Independent normalization also means that this metric does not measure absolute scale or translation recovery. Good part-level geometry and a lower holistic Box IoU can therefore coexist when assembled proportions differ.

F Additional Qualitative Discussion

Figure 3 compares part-mesh generation across methods. For instance, Daedalus retains the rabbit character's hands and limbs as distinct components, while the mushroom example preserves the cap, stem, and hanging details. Several baseline outputs instead merge these structures or lose small components; some also exhibit fragmented surfaces. These examples illustrate why whole-object similarity alone is insufficient to characterize part generation.