UnfoldArt: Zero-Shot Recovery of Full Articulated 3D Objects from Text or Image

Mohamed El Amine Boudjoghra[1] **Ivan Laptev**[2] **Angela Dai**[1]

[1]Technical University of Munich
[2]Mohamed Bin Zayed University of Artificial Intelligence



Figure 1: **UnfoldArt** generates articulated 3D objects from text or a single image, recovering external parts and interior geometry revealed by articulation. Hierarchical agents compare global and local observations and use a generated probe video to refine joint predictions. Guided video exposes occluded geometry, which 3D latent inpainting reconstructs into separate part and body meshes.

Abstract

Articulated 3D objects are essential for interactive environments in embodied AI, robotics, and virtual reality, but reconstructing their structure and motion from sparse observations remains challenging. Existing approaches are constrained by limited supervised data or insufficient priors to recover articulation, hidden geometry, and internal object structure. We present the first debate-driven agentic approach for articulated 3D object reconstruction from text or image inputs, combining hierarchical, debate-based reasoning with a video generative prior for articulation modeling. High-level agents reason about object semantics and motion using knowledge from vision-language and video models, while low-level agents estimate articulation parameters and interaction points; together, they engage in structured debate to resolve ambiguities in structure and motion. To enable reliable generation, including interior object structures, from only text or image inputs, we introduce a video generative prior that synthesizes plausible object motions and exposes occluded interiors and geometry that cannot be inferred from a single static view. By combining agentic reasoning with a video generative prior, our approach jointly infers articulation and reconstructs complete 3D articulated objects, producing high-fidelity geometry, internal structure, and motion-consistent states beyond directly observed surfaces.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Articulated 3D objects support interaction in embodied AI, robotic simulation, and virtual reality. Generating them from text or image inputs would enable the creation of complex, manipulable scenes in which each object exposes its external form and the hidden interior structures that become visible as its components are articulated. Realizing this requires jointly recovering an object's part decomposition, articulation parameters, and the occluded structure revealed through motion.

Most efforts have focused on training supervised models Cao et al. [2025], Liu et al. [2025a] directly on articulated 3D datasets such as PartNet-Mobility Xiang et al. [2020], Mo et al. [2019], Chang et al. [2015]; these models achieve strong in-domain results, but their generalization is limited by the small scale of the available ground-truth data (on the order of 2k objects). Existing ground-truth data also typically models only exterior structure and omits interior components.

A more recent line of work reduces this reliance on articulated supervision by exploiting pre-trained generative priors Chen et al. [2025], Le et al. [2025], Lu et al. [2025], yet each such method depends on an input or resource that our single-image, single-state setting does not assume: multi-view observations across articulation states Chen et al. [2025], part meshes retrieved from a fixed asset library Le et al. [2025], or a video prior fine-tuned on articulation-specific data Lu et al. [2025].

We use pretrained vision-language models (VLMs) and video generative models as sources of articulation evidence and combine their predictions through agent debate. VLMs predict parts and plausible motion types, and video models generate the corresponding motion. The generated videos provide visual evidence about articulation parameters and reveal geometry that is occluded in a static view. Two signals guide articulation in our pipeline: agreement between global and local agents on articulation parameters, and a frozen video prior, which we use first as motion evidence in the debate and then to expose occluded geometry. The 3D, image, and video priors are general-purpose pretrained models that we use without articulation-specific training.

We propose a debate-driven agentic approach for articulated 3D object reconstruction from text or image inputs. From the input prompt or image, we first obtain a TRELLIS2 Xiang et al. [2025a] mesh as an initial 3D estimate of the object, which we use to localize parts and estimate joints. A hierarchy of agents then divides labor between global and local reasoning: a Decomposer identifies parts and their plausible motions, a Grounder translates this into a 2D segmentation of each part, and an Articulator predicts joint type, axis, and pivot for each part. The Articulator's output triggers a two-round debate between the global and local agents that first exploits their disagreement and then grounds the proposed motions in freely generated video to resolve articulation ambiguity. After the two debate rounds, the agreed-upon articulation conditions a guided video inpainting pass that exposes the hidden geometry behind each movable part, which we reconstruct into an interactable URDF with generated interior geometry.

Our contributions are:

- The first debate-driven agentic approach to articulated 3D reconstruction; the agents compare whole-object and part-level observations, then use a generated probe video to refine joint parameters.
- Reconstruction of articulated objects with generated interior geometry via 3D latent inpainting driven by articulation-conditioned generated video.

2 Related Work

3D shape and part generation. Recent 3D generative models generate static geometry from text or image inputs Zhang et al. [2023, 2024], Zhao et al. [2025], Xiang et al. [2025b]; the largest are trained on repositories such as Objaverse Deitke et al. [2022, 2023]. Part-aware generators synthesize objects from an image as sets of parts Tang et al. [2026], Yang et al. [2025], Lin et al. [2025], and X-Part Yan et al. [2025] decomposes a given mesh into complete parts with a diffusion model. Segmentation methods learn 3D part features Ma et al. [2026], Yang et al. [2024], Liu et al. [2025b], and SegviGen Li et al. [2026a] predicts part-indicative colors with a pretrained 3D generative model. We use TRELLIS2 Xiang et al. [2025a] for the initial mesh and focus on articulation and interior geometry.

Articulation prediction for existing geometry. Qian et al. [2022] recover 3D planar articulation from internet videos, and A3VLM Huang et al. [2024] fine-tunes a vision-language model on articulated datasets such as PartNet-Mobility ($\sim$2K objects) to predict actionable articulation. Particulate Li et al. [2026b] predicts parts, kinematic structure, and joint parameters for a static mesh in a single forward pass. These methods articulate given geometry; we also generate the part geometry and interiors from a single image or text.

Articulated reconstruction from observations. Per-instance optimization methods fit neural or Gaussian representations to observations of an object in multiple articulation states Liu et al. [2023], Weng et al. [2024], Wu et al. [2026] or in a single video Song et al. [2024], recovering part geometry and motion; A-SDF Mu et al. [2021] learns a category-level implicit model with separate shape and articulation codes. Feed-forward methods learn from articulated datasets: CAGE Liu et al. [2024] and SINGAPO Liu et al. [2025a] generate articulated part layouts and retrieve part meshes, URDFormer Chen et al. [2024] predicts URDFs from real-world images, MeshArt Gao et al. [2025] generates articulated meshes part by part, and PhysX-Anything Cao et al. [2025] fine-tunes a VLM to produce simulation-ready assets. Digital cousins Dai et al. [2024] instead build interactive scenes from a single image by retrieving similar assets. SPARK He et al. [2026] learns a part prior from PartNet-Mobility and refines joints from a generated open-state image through differentiable kinematics and rendering. We use no articulated training data.

Generative priors for articulation. Articulate-Anything Le et al. [2025] uses a VLM in an actor-critic loop to assemble articulated objects from parts retrieved from a 3D asset library, and Articulate AnyMesh Qiu et al. [2025] uses VLMs with visual prompting to segment parts and construct joints for arbitrary rigid meshes. Articraft Zhou et al. [2026] formulates articulated asset generation as writing programs against a domain-specific SDK with an LLM agent. Multi-agent debate Du et al. [2023] improves language-model reasoning by having model instances propose and debate their answers over multiple rounds; our agents debate from different views of the same object.

Articulated generation from sparse observations. FreeArt3D Chen et al. [2025] uses a pretrained static 3D diffusion model Xiang et al. [2025b] as a shape prior: given a few images of an object in different articulation states, it optimizes part geometry, texture, and joints through score distillation. DreamArt Lu et al. [2025] works from a single image, reconstructing part-segmented meshes with an image-to-3D model and optimizing joints from articulation videos of a fine-tuned video diffusion model. We start from text or a single image and obtain the multi-state observations that FreeArt3D takes as input from a frozen video model.

3 Method

Given an unposed image y_i or text prompt y_t, we reconstruct an articulated 3D object with generated interior geometry, output as a URDF $\mathcal{U} = (\{M_i\}, \{J_i^\star\}, \mathcal{K})$ comprising part meshes, joints, and a kinematic tree. An overview of our approach is shown in Fig. 2. We resolve articulation through a structured global-local debate and use a frozen video prior to provide motion evidence and expose occluded geometry. The video prior thus serves two roles: a *probe* video V_i^{probe} for joint refinement in the debate (Sec. 3.2), and a *guided* video V_i^{art} for reconstructing geometry exposed by part motion (Sec. 3.4).

We first reconstruct an initial 3D estimate of the object M with TRELLIS2 Xiang et al. [2025a] and re-render it through Canny-conditioned Flux Labs et al. [2025] to obtain a photorealistic image I aligned with M, which serves as input to the VLM-based agents and video generation. A hierarchy of VLM agents then divides labor between global semantic reasoning and local geometric prediction: the agents propose movable parts, segment them in 2D, and estimate an initial articulation $\tilde{J}_i$ for each part i. They then refine this estimate to $J_i^{(1)}$ in the first debate round and to the final $J_i^\star$ in the second. Finally, $J_i^\star$ conditions a guided per-part video pass whose final frame drives a 3D latent-inpainting stage that separates movable part and static body meshes.

3.1 Hierarchical Agentic Reasoning

We distribute articulation estimation across three agents:

$$\{(\ell_i, \tilde{\tau}_i)\} = \mathcal{D}(I), \qquad m_i = \mathcal{G}(I, \ell_i), \qquad \tilde{J}_i = \mathcal{A}(I|_{m_i}, M|_{m_i}). \tag{1}$$

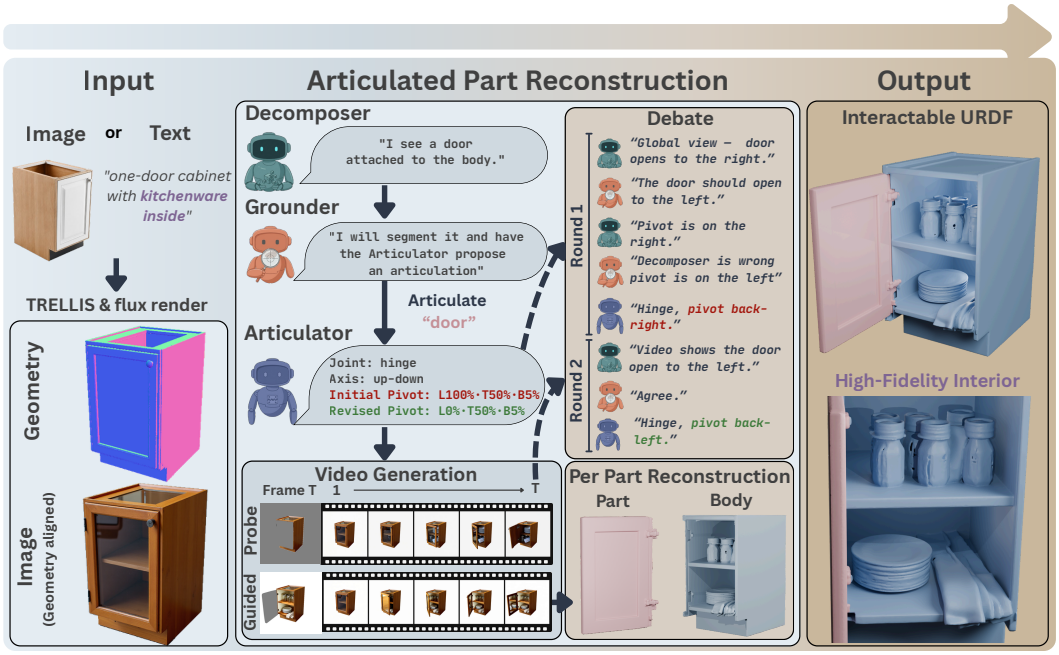

Figure 2: **Method overview.** Given an unposed image or a text prompt as input, we generate a TRELLIS2 Xiang et al. [2025a] mesh and an aligned Flux Labs et al. [2025] render, giving the agents 3D geometry and a photorealistic image for articulation reasoning. A hierarchy of three agents divides labor between global and local reasoning: a *Decomposer* reasons globally over parts and motion, a *Grounder* chooses a segmentation strategy, and an *Articulator* predicts the joint type, axis, and initial pivot. A two-round global-local debate then refines the prediction: the first round exploits whole-object–part disagreement, and the second grounds it in a freely generated video of the proposed motion, revising the pivot from initial (*red*) to final (*green*). Conditioned on the articulation agreed upon in the debate, a guided video pass then drives each part through its motion, anchored to an end-frame mask representing the most articulated state. The exposed interior geometry is then reconstructed with each part to form an interactable URDF.

The *Decomposer* $\mathcal{D}$ proposes part labels ℓ_i and coarse motion types $\tilde{\tau}_i$ from the image I. The *Grounder* $\mathcal{G}$ converts the Decomposer's analysis into a 2D mask m_i by prompting a segmentation model. This step is necessary because many parts are not text-addressable; for example, the chained links of a humanoid cannot be queried by name the way a *door* or *drawer* can. We therefore adapt the prompt to the object: text-recoverable parts are queried by class, while complex articulated objects are segmented by prompting an image generative model with a generic *"movable parts"* prompt. Finally, the *Articulator* $\mathcal{A}$ predicts the initial joint $\tilde{J}_i$ from a local crop $I|_{m_i}$ and the corresponding mesh region $M|_{m_i}$.

In the debate, the three agents take on distinct roles: $\mathcal{D}$ argues globally, $\mathcal{G}$ critiques locally, and $\mathcal{A}$ judges. The Decomposer $\mathcal{D}$ argues from the full-object image and carries authority on whole-object structure, while the Grounder $\mathcal{G}$, which sees the full image during segmentation, is restricted in the debate to the local crop and its fine-grained cues such as hinges and seams. The debate uses the different scopes of $\mathcal{D}$ and $\mathcal{G}$ (Sec. 3.2): a disagreement between them marks a joint estimate for revision. The Articulator $\mathcal{A}$ makes the initial local proposal $\tilde{J}_i$ and then resolves the disagreement between $\mathcal{D}$ and $\mathcal{G}$. Joint type follows from part semantics, which both views capture, whereas the axis and pivot depend on cues that only one view contains: the whole-object layout or the local hardware. The debate therefore targets the axis and pivot. Auxiliary LLMs manage the remaining generation steps (App. 6.8).

3.2 Articulation Estimation via Agent Debate

The debate first compares the agents' global and local estimates in Round 1, then uses generated motion to refine the resulting joint hypothesis in Round 2. We start from the initial Articulator

proposals $\tilde{J}_i = (\tau_i, a_i, p_i, [0, \theta_i^{\max}])$ for each part i, with joint type $\tau_i \in \{\text{revolute}, \text{prismatic}\}$, axis a_i, pivot p_i, and range $\theta_i^{\max}$. The Articulator does not emit these directly: it reports qualitative descriptors (rotation orientation, movement direction, pivot location, and interaction hardware), which a deterministic post-processing step maps to (τ_i, a_i, p_i). The range $\theta_i^{\max}$ is held at a fixed per-type default during the debate and recovered for the full object at URDF export.

Round 1: global-local disagreement as evidence. An LLM prompted with the global image I sometimes returns a different answer than when prompted with a local crop $I|_{m_i}$, because the two views provide different context: on a cabinet with two symmetric doors and no visible knobs, the global agent infers opposing motion from shape symmetry, whereas the local agent, lacking context, may guess wrong; when visible hardware identifies the hinge side, the local view can resolve the ambiguity. Concretely, $\mathcal{G}$ flags any local contradiction (e.g., a handle on the proposed pivot edge, as in Fig. 2) and $\mathcal{D}$, as the whole-object authority, reconsiders its call in light of the flag; $\mathcal{A}$ then consolidates the exchange into $J_i^{(1)} = \mathcal{A}(\tilde{J}_i, c_{\mathcal{D}}, c_{\mathcal{G}})$, where $c_{\mathcal{D}}$ and $c_{\mathcal{G}}$ are the agents' critiques. The exchange is asymmetric: $\mathcal{G}$ may flag local evidence but cannot override $\mathcal{D}$ on whole-object identity (which door of a pair, which row of a stack).

Round 2: generated video as evidence. To check for incorrect joints that both agents may accept after Round 1, we generate a probe video V_i^{probe} with a frozen video prior Jiang et al. [2025], conditioned only on I and generated once per part. A VLM quality agent inspects a five-frame strip of V_i^{probe} and emits two signals for the debate: a quality judgment that gates the axis optimization of Sec. 3.3, and a motion prior describing how the part moves. Conditioned on this prior, $\mathcal{D}$ (full last frame) and $\mathcal{G}$ (crop) judge whether the part opened naturally and, when it did, propose a revised pivot from the observed swing. A strict-consensus rule adopts the video-grounded $J_i^\star$ only when the quality check passes, both sides find the pose plausible, and (for hinges) they agree on the swing direction. If any check fails, we retain $J_i^{(1)}$.

3.3 Axis Refinement from Probe Video

For accepted generations, we track points sampled on $M|_{m_i}$ across V_i^{probe} with CoTracker Karaev et al. [2024a,b], and fit a Rodrigues rotation (revolute) or shared translation (prismatic), minimizing

$$\mathcal{L} = \mathcal{L}_{\text{rep}}(\hat{y}, y) + \lambda_{\text{geo}}\, \mathcal{R}_{\text{geo}}(a, p, \hat{a}_{\text{LLM}}). \tag{2}$$

Here $\mathcal{L}_{\text{rep}}$ is the Huber reprojection error and $\mathcal{R}_{\text{geo}}$ combines smoothness, monotonicity, and an agreement term pulling the fitted axis toward the quality agent's motion prior $\hat{a}_{\text{LLM}}$. We return the refined axis $\hat{a}$ to the debate, while the agents determine the pivot because its fitted position is more sensitive to track noise.

3.4 Articulation-Conditioned Video for Interior Recovery

The agreed articulation $J_i^\star$ lets us generate the geometry occluded behind each movable part. We cast part-body separation as an inpainting problem in 3D latent space: we assign part and body labels from motion and rest-pose evidence and inpaint the unresolved boundary. We derive the labeling from the part's articulation sweep, leaving a boundary shell for inpainting. To obtain the geometry for this assignment, we synthesize a video that drives the part through its motion and exposes the interior beneath it.

Articulation-conditioned video. We synthesize a guided video V_i^{art} explicitly conditioned on $J_i^\star$ that also reveals interior structures. Starting from I, we remove all parts to obtain a body-revealing image I^{body} using Flux, exposing previously hidden interior geometry such as cabinet cavities and drawer slots. For each part i we transport its mesh by $J_i^\star$ and project to 2D, yielding a mask m_i^{open} that represents the most articulated state and conditions the final video frame. Each part receives a guided video, anchored on m_i at the first frame and m_i^{open} at the last, with intermediate frames free for the video model to fill in:

$$V_i^{\text{art}} = \text{WAN}(I^{\text{body}}, m_i, m_i^{\text{open}}), \quad \text{mask}(F_T) = m_i^{\text{open}}, \tag{3}$$

where F_T is the final frame of V_i^{art}. WAN-VACE propagates the rest-pose part's texture, fine details (e.g., a door knob), and geometry through intermediate frames, so that, when generation succeeds, F_T depicts the original part displaced along the predicted motion. Before using the articulated final frame for part segmentation and reconstruction, we check its quality: an LLM scores F_T for part integrity

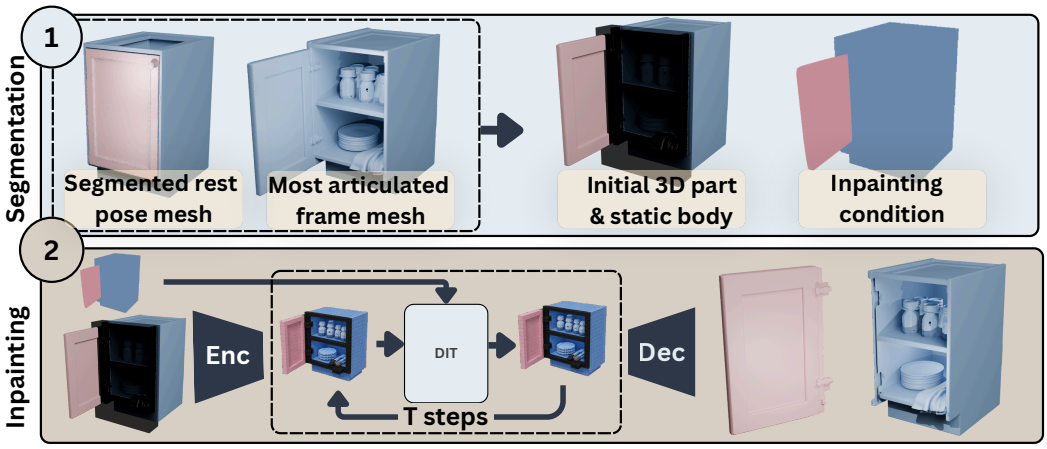

Figure 3: **Part reconstruction via 3D latent inpainting.** For each part (pink), we use the rest-pose mesh and the mesh reconstructed from the maximum articulation frame of the guided video V_i^{art}. Together with the part's articulation sweep, these meshes identify part (pink) and body (blue) regions in the underlying voxel grid. The resulting initial 3D segmentation leaves voxels near the part-body boundary ambiguous (black). After rendering this partial assignment to 2D as the inpainting condition (right), we encode it into latents and denoise using RePaint to jointly resolve the part-body boundary and complete the previously occluded interior to yield separate part and body meshes.

and motion plausibility, and generations where the part has drifted or disintegrated are rejected and re-run.

Initial 3D part segmentation. To lift the part and interior geometry in V_i^{art} to 3D, we reconstruct a TRELLIS2 mesh M_i^{open} from its articulated final frame F_T, then label each voxel as part, body, or unknown. The joint trajectory $J_i^\star$ indicates where the part moves during articulation, and the body-revealing image I^{body} indicates where the body sits at rest. Concretely, we lift the 2D part mask m_i along $J_i^\star$ and sweep it from rest to maximum articulation, yielding the voxel volume Ω_i^{sweep} that the moving part traverses; we also lift I^{body} into 3D to obtain Ω^{body}. Voxels of M_i^{open} in Ω_i^{sweep} belong to the part and voxels in Ω^{body} belong to the body, with ambiguity only along the boundaries of these sets. We mark this boundary shell as unknown, producing a binary mask w with $w(j) = 1$ for committed voxels and $w(j) = 0$ for the shell, where j indexes voxels of M_i^{open}.

Resolving the boundary with latent inpainting. The committed part and body regions form a known latent z_{t-1}^{known} paired with w, and we denoise the latent grid z using RePaint Lugmayr et al. [2022]:

$$z_{t-1} = w \odot z_{t-1}^{\text{known}} + (1 - w) \odot z_{t-1}^{\text{unknown}}, \tag{4}$$

where z_{t-1}^{known} contains the known voxel latents with noise added to match the current step, z_{t-1}^{unknown} comes from the normal denoising step on z_t, and $\odot$ is the element-wise product. At each step, the part and body anchors are re-injected and information propagates into the unknown shell as the latent denoises. Once denoising completes, we group the voxels by similarity to the per-region anchor means in latent space and decode each group into a separate mesh; processing one part-body pair at a time reduces this grouping to a single binary partition. The resulting body and part meshes, together with $\{J_i^\star\}$ and $\mathcal{K}$, form the final URDF $\mathcal{U}$.

4 Results

4.1 Experimental Setup

Datasets. We evaluate on three datasets, all containing only revolute and prismatic joints. The first is PartNet-Mobility Xiang et al. [2020], Mo et al. [2019], Chang et al. [2015], using the SINGAPO Liu et al. [2025a] test split (7 classes, 77 objects), which is in-domain for the supervised baselines. The second is an Objaverse-Household split from Objaverse-animated Deitke et al. [2022, 2023] (12 classes, 53 objects) of household objects that are semantically close to PartNet-Mobility and have

Table 1: **Comparison with state of the art** on an Objaverse-OOD split (10 classes, 40 objects), an Objaverse-Household split (12 classes, 53 objects), and PartNet-Mobility (7 classes, 77 objects). RS/AS denote rest and articulated states. Since FreeArt3D requires multi-state images, we supply additional inputs from our video-synthesis step. SINGAPO and PhysX-Anything are supervised on PartNet-Mobility. Side labels group methods as Sup. (supervised) and Zero-shot.

	Method	RS-dgIoU (↓)	AS-dgIoU (↓)	RS-dcDist (↓)	AS-dcDist (↓)	RS-dCD (↓)	AS-dCD (↓)	AOR (↓)	Axis Err (↓)	Pivot Err (↓)	Joint Acc% (↑)
Objaverse-OOD (10 classes)											
Sup.	SINGAPO Liu et al. [2025a]	1.049	1.049	0.171	0.197	0.154	0.201	0.004	53.5°	0.306	37.6
Sup.	PhysX-Anything Cao et al. [2025]	1.191	1.193	0.519	0.539	0.455	0.525	0.006	71.5°	0.627	18.8
Zero-shot	FreeArt3D Chen et al. [2025]	**0.768**	**0.769**	0.156	0.164	0.133	0.152	**0.000**	85.0°	0.437	28.6
Zero-shot	Articulate-Anything Le et al. [2025]	1.387	1.388	0.242	0.264	0.158	0.227	0.011	67.0°	0.193	47.8
Zero-shot	**Ours**	1.044	1.049	**0.121**	**0.154**	**0.044**	**0.111**	0.008	**25.5°**	**0.103**	**64.2**
Objaverse-Household (12 classes)											
Sup.	SINGAPO Liu et al. [2025a]	1.213	1.215	0.243	0.360	0.177	0.353	0.005	39.8°	0.30	48.5
Sup.	PhysX-Anything Cao et al. [2025]	1.134	1.139	0.314	0.621	0.236	0.401	0.011	36.8°	0.46	19.9
Zero-shot	FreeArt3D Chen et al. [2025]	**0.926**	**0.927**	0.183	0.406	0.248	0.388	0.13	25.0°	0.34	**58.4**
Zero-shot	Articulate-Anything Le et al. [2025]	1.178	1.183	0.216	0.477	0.136	0.346	**0.001**	23.5°	0.33	57.5
Zero-shot	**Ours**	0.955	0.960	**0.175**	**0.244**	**0.103**	**0.188**	0.024	**17.3°**	**0.17**	57.5
PartNet-Mobility (7 classes)											
Sup.	SINGAPO Liu et al. [2025a]	**0.570**	**0.579**	**0.076**	**0.119**	**0.028**	**0.049**	**0.001**	**1.4°**	**0.022**	50.6
Sup.	PhysX-Anything Cao et al. [2025]	0.953	0.958	0.177	0.281	0.089	0.195	0.006	21.6°	0.193	34.7
Zero-shot	FreeArt3D Chen et al. [2025]	0.818	0.820	0.127	0.187	0.315	0.360	0.018	67.5°	0.41	33.3
Zero-shot	Articulate-Anything Le et al. [2025]	0.949	0.953	0.145	0.207	0.057	0.115	**0.001**	7.5°	0.36	72.3
Zero-shot	**Ours**	0.655	0.662	0.103	0.154	0.032	0.078	0.006	8.0°	0.15	**77.5**

no long kinematic chains, but carry more complex geometry, higher part counts, and part motions less common in PartNet-Mobility (e.g., back-hinged chest lids, pedal-actuated bins). The third is an Objaverse-OOD split from Objaverse-animated (10 classes, 40 objects) whose categories lie far outside curated articulation datasets, including helicopters, robotic arms, and humanoid figures, many of which also exhibit chained kinematics. Full category lists are in the appendix.

Baselines. We compare with four state-of-the-art articulated object generation methods, spanning supervised and zero-shot regimes. SINGAPO Liu et al. [2025a] is a diffusion-based method trained on PartNet-Mobility that retrieves part meshes from a fixed database. PhysX-Anything Cao et al. [2025] fine-tunes a VLM for articulation prediction and uses TRELLIS Xiang et al. [2025b] for part generation. Articulate-Anything Le et al. [2025] is zero-shot, running a frozen VLM in an actor-critic loop that assembles part meshes from the PartNet-Mobility asset library; we run it from the same single image as our method. FreeArt3D Chen et al. [2025] is also zero-shot, optimizing static 3D diffusion priors via SDS, but requires multi-view observations across articulation states, with no single-image mode; we supply them from frames of our guided WAN-VACE video. FreeArt3D also receives our predicted joint types.

Table 2: **Ablations** (PartNet-Mobility). **(A)** joint prediction; the bottom row is the full configuration used in Tab. 1. **(B)** part reconstruction.

(A) Joint prediction

Variant	Axis Err (°) ↓	Pivot Err ↓	Joint Acc (%) ↑
Debate rounds (Llama 4, SAM3 segmentation)			
Decomposer only (global)	9.29	0.255	79.6
Grounder only (local)	15.71	0.244	77.5
+ Round 1 (disagreement)	9.44	0.206	79.6
+ Round 2 (video evidence)	8.27	0.206	79.6
Segmentation backend (Llama 4, full debate)			
SAM3	8.27	0.206	79.6
Flux 2 Klein 9B	18.00	0.225	58.8
Flux 2	14.52	0.278	59.3
Gemini	9.84	0.236	77.1
Full config (Sonnet 4.6, Gemini, debate)			
Ours (full)	8.0	0.15	77.5

(B) Part reconstruction

Variant	RS-dgIoU↓	RS-dcDist↓	RS-dCD↓
Probe video + SegviGen Li et al. [2026a]	0.922	0.166	0.06
Guided video + inpainting (Ours)	**0.655**	**0.103**	**0.03**

Metrics. We adopt evaluation metrics from SINGAPO Liu et al. [2025a] and FreeArt3D Chen et al. [2025], covering geometry, articulation, and structural plausibility. For geometry, we report SINGAPO's part-level distances over Hungarian-matched parts: bounding-box gIoU error (dgIoU), part-centroid distance (dcDist), and per-part Chamfer distance (dCD), all lower-is-better, in the rest state (RS-) and averaged across articulated states (AS-), including generated interior geometry. We also report SINGAPO's average overlapping ratio (AOR), which detects unrealistic sibling-part collisions. dgIoU and AOR depend on the predicted part

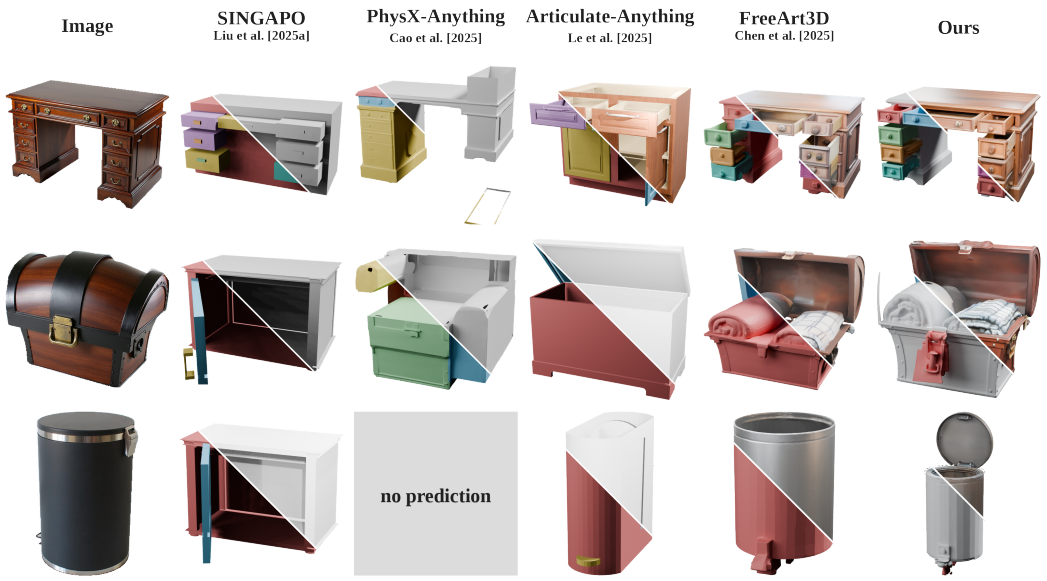

Figure 4: **Qualitative results on Objaverse-Household.** In these examples of Objaverse-Household objects, the outputs of the supervised baselines (PhysX-Anything, SINGAPO) deviate from the input geometry. FreeArt3D follows the input shape with less fine-scale detail and, in some examples, does not separate parts. Our results show separate movable parts and generated interior geometry.

decomposition, since the number of predicted parts determines the matched boxes and the sibling pairs evaluated by AOR. For articulation, following FreeArt3D, we report joint-axis direction error (in degrees) and joint-pivot error (axis-to-axis distance for revolute joints). For each method, we average axis and pivot errors over its correctly typed joints. Since FreeArt3D takes joint types as input, we also report joint-type accuracy (Joint Acc) after Hungarian matching.

Implementation. Local generation and reconstruction run on one NVIDIA A100 GPU. We use Claude Sonnet 4.6 for our reasoning agents and video quality screener, and Gemini Team et al. [2023] for segmentation because it masks both text-addressable parts and the chained parts in our Objaverse splits. To evaluate the segmentation choice, we compare backends under a fixed Llama 4 configuration in Tab. 2. We generate 41 frames with WAN-VACE, set $\lambda_{\text{geo}} = 0.2$, and run the debate for two rounds.

4.2 Comparison with State of the Art

Quantitative results. Table 1 reports performance across the three sets. On Objaverse-OOD, whose categories and chained kinematics lie outside the supervised baselines' training data, our method has the lowest part-distance and articulation errors and the highest joint-type accuracy. Compared with SINGAPO and PhysX-Anything, we attribute this to the absence of category-specific priors in our pipeline. FreeArt3D estimates articulation from the supplied video frames; on chained objects, the generated motion of individual links is often small or inconsistent, which can limit its part separation. Our agents estimate the initial joints from the image, so these estimates do not depend on generated motion alone. Objaverse-Household shows the same pattern for part distances and joint parameters, and our joint-type accuracy is on par with FreeArt3D. On PartNet-Mobility, SINGAPO has the lowest geometry, axis, and pivot errors, which is consistent with its training on this dataset and its retrieval of part meshes from it; our method has the highest joint-type accuracy. FreeArt3D has lower dgIoU on both Objaverse splits and the lowest AOR on Objaverse-OOD.

Qualitative results. Figures 4 and 5 compare with state of the art on Objaverse-Household and PartNet-Mobility. In the illustrated Objaverse-Household examples, PhysX-Anything produces cabinet-like geometry and SINGAPO produces coarse part shapes. FreeArt3D follows the input shape but loses fine-scale detail and sometimes fails to separate parts. The PartNet-Mobility examples show opened parts and internal components; Fig. 8 (appendix) shows generated interior structures such as

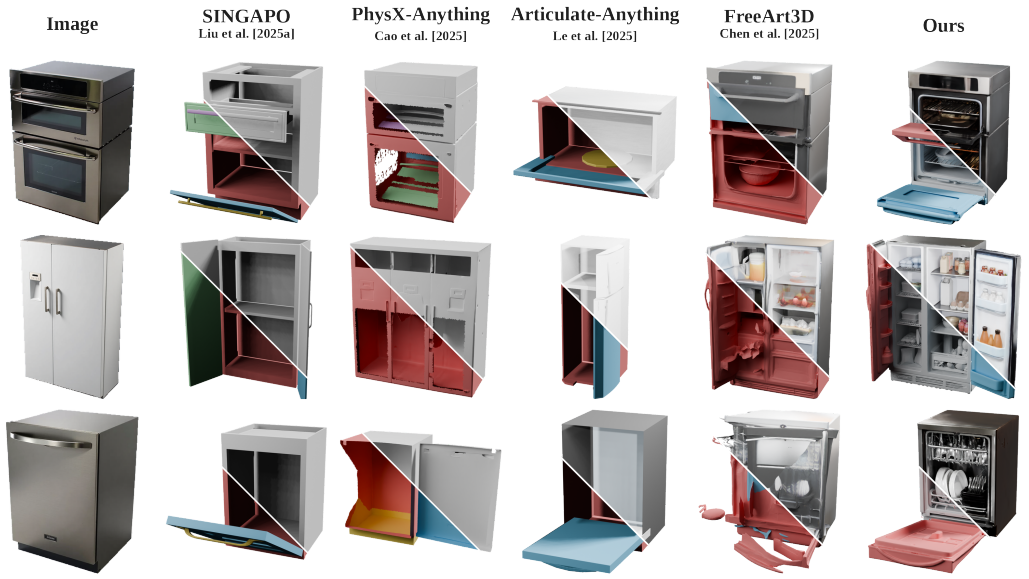

Figure 5: **Qualitative results on PartNet-Mobility.** The examples show reconstructed parts and interiors for categories included in the supervised baselines' training data.

racks and shelving. Figure 6 illustrates the text-to-image-to-articulated-3D path using text-generated images of a drawbridge, robotic mechanisms, a lamp, and a gate. Each row shows a reconstructed object in successive articulation states, with colors identifying its parts; additional examples appear in App. 6.1.

4.3 Ablation Study

Agent debate vs. single-agent prompting. Tab. 2(A) compares single-agent prompting against our debate, with the reconstruction stage fixed and Llama 4 as the backbone for all variants. The two single-agent variants have different errors: the global view alone (Decomposer) mislocates pivots, and the local crop alone (Grounder) has the highest axis error. We interpret this as the two views carrying complementary cues: the full image constrains the axis through the object's layout, while the crop shows the hardware that locates the pivot. Round 1 combines both, lowering pivot error below either single agent while keeping axis error close to the Decomposer's. Round 2 lowers axis error further. In this round, the probe-video tracks refine the axis (Sec. 3.3), and the agents may revise the pivot or keep the Round-1 estimate.

Segmentation backend. With Llama 4 and the full debate fixed, SAM3 Carion et al. [2026] gives the most accurate joints among the four backends on PartNet-Mobility (Tab. 2(A)), followed by Gemini. PartNet-Mobility parts such as doors and drawers can be queried by name, which suits SAM3's text-prompted segmentation. The two Flux2 variants have lower joint-type accuracy, which we attribute to their indirect segmentation through detected joint regions (App. 6.6). The full configuration (Sonnet 4.6, Gemini) is listed separately in Tab. 2(A).

Articulation-guided reconstruction. Tab. 2(B) ablates our reconstruction stage. To evaluate the combined effect of articulation guidance on video generation and inpainting, we replace guided generation and latent inpainting with free (conditioned only on I) WAN-VACE Jiang et al. [2025] video and SegviGen Li et al. [2026a] surface segmentation. The guided configuration lowers all three rest-state errors. We attribute this to the shared articulation: the end-frame mask places the part where the joint moves it, and the sweep labels most voxels before inpainting, whereas surface segmentation infers the part-body boundary from color alone.

Limitations. The pipeline uses several VLM queries, a probe video, and a guided video for each part, before retries; in the SAM3-based timing study, reconstruction averaged 73.4 minutes per object on one NVIDIA A100. Our approach also focuses on parts that move, and so does not further



Figure 6: **Text-to-articulated 3D results.** Each input image in the left column is generated from a text prompt. The remaining columns show successive articulation states of the reconstructed 3D objects, with colors indicating parts.

decompose static regions into their finer-scale semantic components. In addition, very small or low-amplitude articulations, such as flat buttons, are difficult to capture through a video prior, since the corresponding motion is too subtle to manifest reliably in generated video.

5 Conclusion

We presented the first debate-driven agentic approach to articulated 3D object reconstruction from text or image inputs, resolving articulation ambiguity through a structured global-local debate that exploits disagreement between whole-object and part-level views and adjudicates it against freely generated video. A guided video inpainting pass conditioned on the agreed articulation then exposes the geometry occluded behind each movable part, yielding interactable URDFs with articulation and generated interior structures. Our evaluation covers household objects with part motions less common in PartNet-Mobility and objects with chained kinematics, such as robotic arms.

Acknowledgements

This work was supported by the ERC Starting Grant SpatialSem (101076253).

References

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.

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, 2026. URL `https://arxiv.org/abs/2511.16719`.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Chuhao Chen, Isabella Liu, Xinyue Wei, Hao Su, and Minghua Liu. Freeart3d: Training-free articulated object generation using 3d diffusion. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pages 1–13, 2025.

Zoey Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Karthikeya Vemuri, Alan Wu, Dieter Fox, and Abhishek Gupta. Urdformer: A pipeline for constructing articulated simulation environments from real-world images. *arXiv preprint arXiv:2405.11656*, 2024.

Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Automated creation of digital cousins for robust policy learning. *arXiv preprint arXiv:2410.07408*, 2024.

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. *arXiv preprint arXiv:2212.08051*, 2022.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663*, 2023.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Daoyi Gao, Yawar Siddiqui, Lei Li, and Angela Dai. Meshart: Generating articulated meshes with structure-guided transformers. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 618–627. IEEE, 2025.

Yumeng He, Ying Jiang, Jiayin Lu, Yin Yang, and Chenfanfu Jiang. Spark: Sim-ready part-level articulated reconstruction with vlm knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7233–7243, 2026.

Siyuan Huang, Haonan Chang, Yuhan Liu, Yimeng Zhu, Hao Dong, Peng Gao, Abdeslam Boularias, and Hongsheng Li. A3vlm: Actionable articulation-aware vision language model. *arXiv preprint arXiv:2406.07549*, 2024.

Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17191–17202, 2025.

Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proc. arXiv:2410.11831*, 2024a.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *Proc. ECCV*, 2024b.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space, 2025. URL https://arxiv.org/abs/2506.15742.

Long Le, Jason Xie, William Liang, Hung-Ju Wang, Yue Yang, Yecheng Jason Ma, Kyle Vedder, Arjun Krishna, Dinesh Jayaraman, and Eric Eaton. Articulate-anything: Automatic modeling of articulated objects via a vision-language foundation model. In *International Conference on Learning Representations*, volume 2025, pages 17578–17602, 2025.

Lin Li, Haoran Feng, Zehuan Huang, Haohua Chen, Wenbo Nie, Shaohua Hou, Keqing Fan, Pan Hu, Sheng Wang, Buyu Li, and Lu Sheng. Segvigen: Repurposing 3d generative model for part segmentation. *arXiv preprint arXiv:2603.16869*, 2026a.

Ruining Li, Yuxin Yao, Chuanxia Zheng, Christian Rupprecht, Joan Lasenby, Shangzhe Wu, and Andrea Vedaldi. Particulate: Feed-forward 3d object articulation. In *CVPR*, 2026b.

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

Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. Paris: Part-level reconstruction and motion analysis for articulated objects. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 352–363. IEEE, 2023.

Jiayi Liu, Hou In Ivan Tam, Ali Mahdavi-Amiri, and Manolis Savva. Cage: Controllable articulation generation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17880–17889. IEEE, 2024.

Jiayi Liu, Denys Iliash, Angel Chang, Manolis Savva, and Ali Mahdavi Amiri. Singapo: Single image controlled generation of articulated parts in objects. In *International Conference on Learning Representations*, volume 2025, pages 97511–97532, 2025a.

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 *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9704–9715. IEEE, 2025b.

Ruijie Lu, Yu Liu, Jiaxiang Tang, Junfeng Ni, Yuxiang Wang, Diwen Wan, Gang Zeng, Yixin Chen, and Siyuan Huang. Dreamart: Generating interactable articulated objects from a single image. *arXiv preprint arXiv:2507.05763*, 2025.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 11451–11461. IEEE, 2022.

Changfeng Ma, Yang Li, Xinhao Yan, Jiachen Xu, Yunhan Yang, Chunshi Wang, Zibo Zhao, Zhuo Chen, Yanwen Guo, and Chunchao Guo. P 3-sam: Native 3d part segmentation. In *European Conference on Computer Vision*, pages 195–211. Springer, 2026.

Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-sdf: Learning disentangled signed distance functions for articulated shape representation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12981–12991. IEEE, 2021.

Shengyi Qian, Linyi Jin, Chris Rockwell, Siyi Chen, and David F Fouhey. Understanding 3d object articulation in internet videos. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1589–1599. IEEE, 2022.

Xiaowen Qiu, Jincheng Yang, Yian Wang, Zhehuan Chen, Yufei Wang, Tsun-Hsuan Wang, Zhou Xian, and Chuang Gan. Articulate anymesh: Open-vocabulary 3d articulated objects modeling. *arXiv preprint arXiv:2502.02590*, 2025.

Chaoyue Song, Jiacheng Wei, Chuan Sheng Foo, Guosheng Lin, and Fayao Liu. Reacto: Reconstructing articulated objects from a single video. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5384–5395. IEEE, 2024.

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. *Advances in Neural Information Processing Systems*, 38:27115–27137, 2026.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Yijia Weng, Bowen Wen, Jonathan Tremblay, Valts Blukis, Dieter Fox, Leonidas Guibas, and Stan Birchfield. Neural implicit representation for building digital twins of unknown articulated objects. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3141–3150. IEEE, 2024.

Di Wu, Liu Liu, Zhou Linli, Anran Huang, Liangtu Song, Qiaojun Yu, Qi Wu, and Cewu Lu. Reartgs: Reconstructing and generating articulated objects via 3d gaussian splatting with geometric and motion constraints. *Advances in Neural Information Processing Systems*, 38:102889–102915, 2026.

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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

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

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21469–21480. IEEE, 2025b.

Xinhao Yan, Jiachen Xu, Yang Li, Changfeng Ma, Yunhan Yang, Chunshi Wang, Zibo Zhao, Zeqiang Lai, Yunfei Zhao, Zhuo Chen, et al. X-part: high fidelity and structure coherent shape decomposition. *arXiv preprint arXiv: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 preprint arXiv:2411.07184*, 2024.

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 *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pages 1–12, 2025.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024.

Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv preprint arXiv:2501.12202*, 2025.

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 preprint arXiv:2605.15187*, 2026.

6 Appendix

6.1 Additional Text-to-Articulated 3D Results

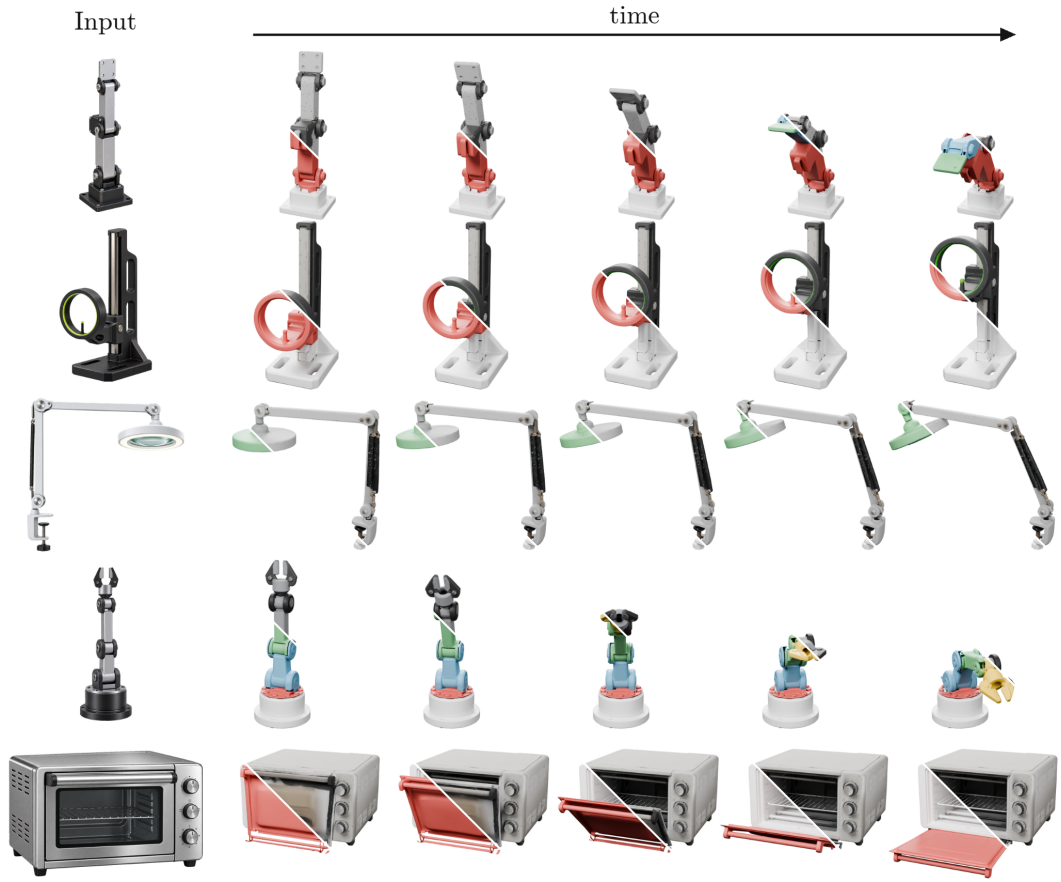

Figure 7: **Additional text-to-articulated 3D results.** Each left-column image is generated from text. The remaining columns show successive articulation states of the reconstructed objects, with colors indicating parts.

6.2 Robustness and Runtime

We evaluate robustness and runtime using TRELLIS2 with SAM3 for segmentation and Claude Sonnet 4.6 as the reasoning model.

6.2.1 Evaluation protocol

Geometry is evaluated per Hungarian-matched part: dgIoU is bounding-box gIoU error, dcDist is centroid distance, and dCD is Chamfer distance; RS denotes rest state, AS averages articulated configurations, and AOR measures sibling-part overlap. Axis direction error and revolute axis-to-axis pivot distance use each method's correctly typed joints; joint-type accuracy is reported separately.

6.2.2 Video reliability

We annotated all 201 free probe videos on PartNet-Mobility for motion plausibility, blind to the Round-2 decision. Failures included part deformation and detachment, particularly on small parts such as handles.

Table 3: Probe-video audit and Round-2 detection of implausible generations.

Measure	Value
Implausible videos	32.8%
Detection precision	90.0%
Detection recall	81.8%
Implausible but accepted / all videos	6.0%

6.2.3 Sensitivity to joint and mask errors

We test ten successful objects selected for zero clean axis error, with a clean joint accuracy of 88.9%. On this subset, three clean runs yield RS-dCD 0.0110, 0.0095 and 0.0116, with mean 0.0107 and standard deviation 0.0011.

Axis and pivot perturbations. We perturb joints after Round 2 so that the perturbations affect only the subsequent reconstruction. Tilt is measured from the ground-truth axis, and pivot shift is a percentage of the object bounding-box diagonal.

Table 4: Joint perturbations on ten successful objects. Clean-run standard deviation: 0.0011.

Axis tilt	RS-dCD↓	Pivot shift	RS-dCD↓
0° (clean)	0.0107	0% (clean)	0.0107
5°	0.0100	5%	0.0106
10°	0.0226	10%	0.0108
20°	0.0367	20%	0.0265
30°	0.0584	30%	0.0238
45°	0.0541	—	—

Partial mask degradation. After SAM segmentation, we partition each mask into 100 k-means patches and drop a random fraction before joint prediction and reconstruction. This procedure tests partial pixel loss while preserving part identity; on this subset, the reported aggregate joint metrics remain unchanged through 50% dropout. The pipeline operates on the convex hull of the lifted mask, which gives the inpainting stage full coverage of the part and of its articulation sweep; random patch dropout leaves this hull unchanged, so the joint is unaffected and only the reconstruction varies.

Table 5: Mask-patch dropout on the same ten objects; clean-run RS-dCD standard deviation is 0.0011.

Dropped	RS-dCD↓	Axis Err (°)↓	Pivot Err↓	Joint Acc (%)↑
0%	0.0107	0.0	0.092	88.9
10%	0.0125	0.0	0.092	88.9
20%	0.0088	0.0	0.092	88.9
30%	0.0084	0.0	0.092	88.9
50%	0.0094	0.0	0.092	88.9

6.2.4 Debate statistics

Rates are over all evaluated joints. Disagreement is the fraction of joints on which the Decomposer and Grounder return different reads when queried independently. Correction is the fraction on which the joint leaving a round is closer to the ground truth in axis or pivot than the joint entering it, and failure is the fraction on which it is farther. On PartNet-Mobility, disagreement, correction and failure are 13.9%, 10.0% and 0.0% in Round 1, and 14.4%, 14.9% and 0.0% in Round 2. On Objaverse, they are 9.1%, 21.7% and 0.7% in Round 1, and 31.9%, 53.2% and 0.0% in Round 2.

6.2.5 Choice of models

We chose TRELLIS2 over SAM3D because SAM3D is trained to generate a single object: the revealed interiors are usually cluttered, with shelves holding several items, drawer contents and compartment dividers, and SAM3D returns the container without them, whereas TRELLIS2 generates the whole

revealed volume including its contents, which is what the 3D latent inpainting stage consumes (Sec. 3.4).

We chose WAN-VACE over LTX-Video and HunyuanVideo because it accepts a mask and inpaints inside it, which the guided pass uses to steer the part to its predicted end state; LTX-Video and HunyuanVideo expose no equivalent mask conditioning, and their closest proxy, conditioning on a warped end frame, fixes appearance rather than the region the part must occupy. We therefore compare the three generators without 3D guidance on a common subset of 66 objects. Using each generator for free generation, as in the probe video, worsens RS-dgIoU, RS-dcDist and RS-dCD to 0.928, 0.157 and 0.055 with WAN, 0.818, 0.151 and 0.056 with LTX-Video, and 0.915, 0.150 and 0.058 with HunyuanVideo. The three generators perform similarly, so the gain comes from the mask conditioning rather than from the choice of generator.

6.2.6 Segmentation-failure audit

We match cached predictions from all 77 PartNet-Mobility objects to ground-truth parts by IoU, without rerunning the pipeline or injecting corruption. The audit identifies overlapping failure classes in 63 objects, with counts and metrics reported below.

Table 6: Segmentation failure definitions.

Class	Definition
Missing part	A ground-truth part has no matched prediction.
Merged instances	One prediction spans multiple ground-truth parts.
Split part	Multiple predictions cover one ground-truth part.
Wrong region	A matched pair has IoU in $[0.25, 0.5)$.

Table 7: Segmentation-failure subsets in the SAM3 run; classes overlap and do not isolate causal effects. N counts objects.

Class	N	RS-dCD↓	Joint Acc (%)↑	Axis Err (°)↓	Pivot Err↓
Missing part	20	0.0627	78.3	14.5	0.248
Merged instances	13	0.0610	88.6	0.7	0.105
Split part	18	0.0358	78.0	15.0	0.260
Wrong region	59	0.0385	75.1	10.1	0.165

6.2.7 Runtime

On one NVIDIA A100 over 77 PartNet-Mobility objects, the SAM3 configuration averages 73.4 minutes per object. Fixed work costs 7.3 minutes, with approximately 25.3 minutes per movable part; entries below are rounded independently.

Table 8: Stage-level runtime in the SAM3 study.

Stage	Scope	min/part	min/object
Initial mesh (TRELLIS2)	Object	—	7.0
Flux render	Object	—	0.3
Agents and debate	Part	0.8	2.1
Free video generation	Part	7.0	18.3
Guided video generation	Part	7.0	18.3
Articulated mesh reconstruction	Part	7.0	18.3
3D latent inpainting	Part	3.5	9.1
Total			73.4

Table 9: Runtime comparison on PartNet-Mobility; Articulate-Anything was not included in this timing study.

Method	Training-free	min/object
SINGAPO	No	0.6
PhysX-Anything	No	16.6
FreeArt3D	Yes	179.6
Ours	Yes	73.4

6.2.8 Out-of-distribution evaluation

The Objaverse-OOD split contains 40 objects from ten categories, selected by class coverage and part count before evaluating the methods. Table 1 reports the results and Table 11 lists the OOD categories.

6.3 Additional Implementation Details

Image-to-mesh correspondence. We render the initial mesh from a known camera and use Canny-conditioned Flux to obtain an aligned image, allowing image-space masks and tracked motion to be mapped into the mesh coordinate frame.

Joint fitting. For accepted probe videos, we fit 3D motion to 2D tracks using Huber reprojection loss and geometric regularization (Sec. 3.3). The Huber loss limits the influence of large reprojection residuals during fitting, after which we combine the refined axis with the debate pivot to obtain the final joint.

Voxel assignment. We use the articulation sweep to assign part voxels and the body-revealing reconstruction to assign body voxels, leaving the boundary unknown. RePaint preserves the assigned regions while filling the unknown shell, after which latent similarity separates the part and body (Sec. 3.4).

Automatic routing. For text-addressable parts, the Grounder emits kind names and the tool layer assigns position-ordered identifiers to the returned instance masks. Other parts use generative segmentation, whose part colors are lifted to 3D and refined by inpainting, without manual correction.

Backbones. The additional studies use TRELLIS2 and SAM3; the main configuration uses Gemini for segmentation and Claude Sonnet 4.6 for reasoning.

Debate protocol. The debate uses fixed agent roles, two rounds and structured output schemas. Round 1 resolves global–local disagreement, while Round 2 accepts video evidence only after quality and agreement checks; the prompts and resolver rules are provided below.

Coordinate convention. Agents reason in image coordinates, where the up-down, left-right, and back-front directions correspond to the y, x, z axes of the mesh M.

6.4 Evaluation Categories

Tables 10 and 11 list the categories and per-category object counts of our two Objaverse-animated evaluation splits (Sec. 4.1).

6.5 Interior Diversity

Figure 8 shows additional examples of generated interior structures.

Table 10: **Objaverse-Household split** (12 classes, 53 objects). Household categories semantically close to PartNet-Mobility, without long kinematic chains. The first 4 categories match the supervised baselines' training categories.

Category	#Obj	Category	#Obj
In baselines' training categories			
Refrigerator	4	Microwave oven	3
Desk / table	4	Dresser / drawers	8
Other household categories			
Treasure chest	11	Stove / range	3
Box / crate / case	6	Door	3
Cabinet / wardrobe	6	Small appliance	2
Trash bin	2	Faucet	1

Table 11: **Objaverse-OOD split** (10 classes, 40 objects), featuring object categories far outside curated articulation datasets, many of which also show chained kinematics.

Category	#Obj	Category	#Obj
Helicopter	7	Multi-legged creature	4
Aircraft / spacecraft	3	Sci-fi pod	5
Robotic arm	4	Wind turbine	4
Humanoid robot	5	Animal	3
Humanoid figure	2	Vehicle / machinery	3

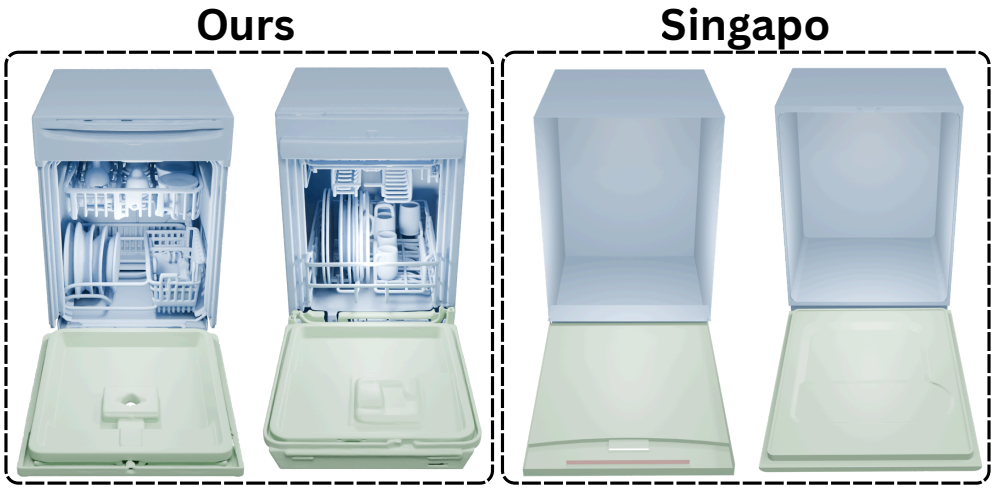

Figure 8: **Diversity of interior geometry.** Generated interior structures include geometry hidden from view in the input image.

6.6 FLUX-based Joint Detection

FLUX 2 is an open-weight alternative to the closed-source Gemini segmentation backend. Gemini segments chained parts when prompted with a generic "movable parts" instruction. In our experiments, FLUX 2 does not segment parts reliably when prompted directly, so we use it as a joint detector: we detect joint regions in 2D, lift them to 3D, remove the joint region to split the parts, cluster the remaining surface with DBSCAN, and inpaint the region between parts. Figure 9 illustrates joint detection, lifting, and part extraction.

Joint detection via saturation. We prompt FLUX to generate images in which candidate joint regions are rendered with high saturation. Running connected-component analysis on the resulting 2D saturation maps yields per-view joint proposals.

Lifting to 3D. The 2D joint regions are lifted into 3D by aggregating consistent components across views, producing a coarse 3D mask of the joint region.

Part extraction. We remove the joint region from the object and cluster the remaining surface points with DBSCAN. The resulting clusters serve as the noisy part segmentation that is subsequently refined by our inpainting stage.

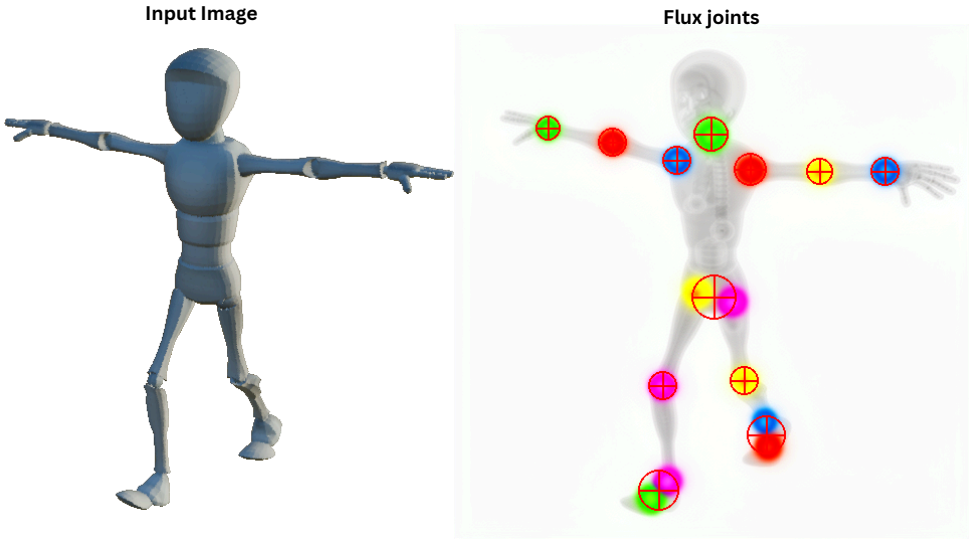

Figure 9: **Noisy part segmentation from FLUX joint detection.** FLUX generates images with high-saturation joint regions; connected components on the saturation maps are lifted to 3D to localize the joint. After removing the joint region, DBSCAN clustering on the remainder yields the noisy part segmentation used as input to our inpainting stage.

6.7 Agent Prompts

The prompts below are grouped by the method stage they implement. Output schemas and minor formatting rules are abbreviated for readability.

6.7.1 Hierarchical agents (§3.1)

Decomposer (Stage 1). The Decomposer consumes the rendered image I and proposes parts and coarse motion guesses, producing the initial part hypothesis used to drive segmentation. It returns part kinds and counts, such as "three drawers"; instance identifiers are assigned after segmentation.

Decomposer system prompt

You are looking at a photograph of a 3D object. Your job is to give a short, high-level read of what kinds of parts this object has, roughly how each kind moves, and roughly how the kinds fit together.

This is a rough first pass. You are not labelling individual instances and you are not counting them precisely. Somebody else will look at the 3D mesh and pick out each specific instance later.

How to think about it. Think in kinds, not instances. If you see three stacked drawers, you say "drawers — multiple, stacked vertically, they slide straight out toward the viewer". You do not say "drawer_0, drawer_1, drawer_2". Keep it rough — you are pointing at categories and gestures, not measurements.

What you must produce. A single object-level note with: `object_type` (a short natural name — "wooden dresser", "ceramic teapot"); `parts` (a list of part kinds, each with `part_type`, `description`, a `motion` block containing `moves`, `direction`, and `reasoning`, and a `count_hint` like "one", "multiple stacked vertically"); and a `hierarchy_sketch` (one or two sentences on how the kinds fit together).

Do not pin down exact pivots. Do not mention axes, angles, degrees, or percentages — that level of detail is for later steps.

Rules. Plain everyday language only — no X/Y/Z, no coordinates, no technical terms. Use "left/right/forward/back/up/down" — nothing more technical. Do not mint ids like "drawer_0". Do not invent kinds you cannot clearly see. If a kind does not move, set `moves` to false and `direction` to null.

Grounder (Stage 2). The Grounder receives the Decomposer's kind list and selects a segmentation route per kind: a text-query route for text-queryable parts (doors, drawers, handles), or a separate route for chained kinematics that text queries cannot resolve (humanoid figures, robot arms, spider legs). With the default Gemini backend, chained parts are segmented by prompting Gemini to color the movable parts; the joint-driven route in the prompt below applies to the Flux2 backend (App. 6.6). The complement of the part masks defines the body. In the prompt below, "SAM" denotes the segmentation backend generically; our default backend is Gemini, with SAM3 as an alternative (Tab. 2).

Grounder system prompt

You decide what SAM should look for in the input photograph. The Decomposer has already told you what kinds of parts the object has, roughly how many of each, and how each kind tends to move. Your job is to turn that into a list of plain kind-level phrases for SAM, plus a motion note per kind.

SAM runs once with all of your phrases together, returning one mask per instance — if there are three drawers and you say `"drawer"`, SAM returns three drawer masks. The tool layer mints one part id per mask, ordered by position.

Writing good SAM phrases. Use the bare kind name (`door`, `drawer`, `handle`, `knob`, `wheel`, `shelf`, `lid`). *No positional qualifiers* — never "top drawer", "upper door", "left handle". SAM does not understand positional words. Add a descriptor only when the class name is ambiguous (`cabinet door`, `fridge door`). The body always falls out as the complement of the part masks — never list it as a SAM phrase.

When to use the joint-driven route instead of SAM. The decision is about *kinematic chaining*. Use SAM when every moving part pivots directly on the body — doors, drawers, lids, wheels, knobs on cabinets, fridges, dressers, microwaves, suitcases. Use joints when a part's pivot lives on another moving part: humanoid figures (foot → shin → thigh → hip), articulated robot arms, spider/octopus/crab legs, quadrupeds, excavators, articulated lamps. SAM masks alone cannot recover per-link pivots in chains.

Output. `sam_kinds` (one entry per kind, with `sam_prompt`, `part_type`, `motion_guess`), an optional `joints_kinds` for chained kinds, a `body_motion` note, and a `reasoning` field.

Rules. Plain language only. Stay close to the Decomposer's note — if it names kinds you cannot see in the photo, flag it in `reasoning` rather than silently dropping or inventing kinds. Do not refine motion — copy from the Decomposer's note. A later step handles precise pivots and rotations.

Articulator (Stage 3). The Articulator predicts the initial joint $\tilde{J}_i$ from a local crop around each part. It uses visible hardware to estimate the pivot: find the visible interaction point, place the pivot on the opposite edge, and check the pivot against the visible hardware. Because the Articulator uses a local crop, whole-object identity questions (which door of a pair, which row of a stack) are deferred to the Decomposer in the debate.

Articulator system prompt

You are looking at a close-up crop of one part of a 3D object. Your job is to say where the pivot sits inside this crop — the point the part hinges or pulls away from. The crop has direction labels drawn on its edges (left/right, top/bottom, FRONT/BACK).

Your scope — local-crop detail only. You are the LOCAL-DETAIL agent. Reason from what is directly visible in this crop: handle position, visible hinge gap or shadow, seam direction, how the red-outlined edges sit inside the body. A downstream WHOLE-OBJECT critic sees the full image and is responsible for any question that depends on the object as a whole ("which door of a pair", "which row of drawers"). You MUST NOT answer those questions. Anchor pivot calls on local evidence only.

Chain of reasoning — anchored on the visible hardware. Reason in this sequence and state each step explicitly:

Step 1 — Find the hardware in the rest crop. Identify the visible interaction point (handle / knob / pull / bar) and name which edge it sits on: `right`, `left`, `top`, `bottom`, or `front face / centred`.

Step 2 — The pivot is on the OPPOSITE edge. Apply the table:

Hardware sits on... (closed)	Pivot edge → percentage
`right` edge	`left` edge → `from_left` ≈ 0%
`left` edge	`right` edge → `from_left` ≈ 100%
`top` edge (oven door, drop-front)	`bottom` edge → `from_top` ≈ 100%
`bottom` edge (flip-up cabinet)	`top` edge → `from_top` ≈ 0%
`front` centred (slider)	no hinge — all three ≈ 50%
`front` edge (chest lid)	`back` edge → `from_back` ≈ 0%

Step 3 — Motion sanity-check. When the part opens, the handle traces an arc and sweeps TOWARD the hinge. If `movement_direction` and `pivot_estimate` are incompatible (handle on right but `from_left` ≈ 100%), redo step 1.

Output. `pivot_estimate` (`from_left`, `from_top`, `from_back` as percentages), `rotation_orientation` (∈ vertical, horizontal, depth, none-slides), `movement_direction`, `interaction_hardware`, and `reasoning` that explicitly walks through steps 1→2→3.

Rules. Plain everyday language. No X/Y/Z, no coordinates, no degree numbers. Always read FRONT/BACK labels — a pivot near BACK signals a rear attachment.

6.7.2 Round 1 debate: prior-based global/local critique (§3.2)

Decomposer (Round 1). The Decomposer critiques the Articulator's $\tilde{J}_i$ from the global perspective, with access to the full-object image and the Stage-1 kind-level read. It is the highest authority in the debate: the Grounder is a local validator and cannot override the Decomposer on whole-object identity.

Decomposer Round 1 system prompt

You are the Decomposer in a structured debate about how one specific articulated part moves. **You are the HIGHEST authority in this debate.** The Grounder validates local crop-level details (visible hinge gap, handle position) but cannot override your whole-object call — they only see a zoomed crop, no whole-object context, no kind-level prior.

You are shown: (1) the FULL object image with a RED CONTOUR around the debate-target part — *the Grounder does not see this image*, only the zoomed crop; (2) the part crop; (3) your own earlier prior; (4) the Articulator's proposal; (5) the Grounder's latest argument.

Your scope — whole-object identity is yours alone. You are the SOLE authority on: which door of a symmetric pair this is; which row/column of a grid of drawers; which side of the body the part sits on; what kind of object this is. The Grounder has been explicitly told not to answer these. You OWN these calls. Pin every `from_left`/`from_top`/`from_back` percentage to a whole-object observation first ("in the full image the red contour is on the right door of a pair, therefore the hinge is on the right edge").

Reasoning chain. Apply the same hardware → pivot table as Stage 3, but anchored on the FULL-object image. Highlight at least one whole-object observation the Grounder cannot make from the crop alone ("looking at the full cabinet I can see this is the right door of a pair, so the hinge must run on the right edge"). Stand behind your prior unless the Grounder raises a concrete LOCAL visual point that reframes your read — a Grounder claim about whole-object identity is not a legitimate challenge.

Output. `motion_position`, `pivot_position`, `interaction_position`, `response_to_articulator` (with concrete field-level edits and target values), and `response_to_other_side`, all in full sentences.

Rules. Plain everyday language. Do not converge out of politeness — if kind-level convention says the Articulator is wrong, keep saying so. Do not propose 3D axes or rotation matrices.

Grounder (Round 1). The Grounder critiques $\tilde{J}_i$ from the local perspective, with access only to the zoomed part crop. Its role is strictly validation: it can flag concrete local contradictions (handle visible on the same edge as the proposed pivot) but cannot override the Decomposer on object identity or kind-level convention.

Grounder Round 1 system prompt

You are the Grounder in a structured debate. **Your role is strictly local validation.** The Decomposer is the highest authority — it has the full-object image AND the canonical Stage-1 kind-level read. You check whether the Decomposer's whole-object call is consistent with what the zoomed crop shows pixel-level. You are NOT here to override the Decomposer on motion / pivot / object-type / which-side-of-pair — those are out of your scope.

You are shown: (1) the part crop with a RED CONTOUR; (2) your own earlier prior; (3) the Articulator's proposal; (4) the Decomposer's latest argument.

Validation behaviour. If the Decomposer's pivot call places the hinge on edge X, check the crop: is the visible handle on the OPPOSITE edge? Is the visible hinge gap or shadow on edge X? If yes, you are validating — say so. If you spot a concrete LOCAL contradiction (handle on the same edge as the proposed pivot), surface it as a flag for the Decomposer to reconsider — frame it as a local observation, NOT a counter-claim about the object type.

Do not make whole-object identity claims of your own ("this is the left door", "this is a dishwasher"). The crop alone cannot support those. Do not propose a different pivot edge if the Decomposer's call is consistent with local evidence — rubber-stamping is a valid output. Apply the same hardware → pivot table as Stage 3 to corroborate or flag, never to override.

Output. `motion_position`, `pivot_position`, `interaction_position`, `response_to_articulator` (only dispute fields the crop lets you judge — do not dispute whole-object identity claims), and `response_to_other_side` (defer on whole-object identity claims and say so).

Rules. Plain everyday language. If you read the crop differently from the Decomposer on a strictly local point, hold your ground and explain why. Do not propose 3D axes or rotation matrices.

Round 1 resolver (Articulator-as-judge). The Articulator combines the agents' critiques to obtain the Round-1 consensus $J_i^{(1)}$ that Round 2 then tests against video evidence. The decision rule is asymmetric: the Decomposer wins by default on whole-object questions; the Grounder can flag local contradictions but cannot override the Decomposer on whole-object identity.

Round 1 resolver system prompt

You are a neutral moderator who has just read a short structured debate about one articulated part. The bundle contains: `articulator_output` (the proposed pivot/rotation/movement/hardware); `grounder_prior` and `decomposer_prior` (each side's INITIAL no-peek read); `grounder_rounds` and `decomposer_rounds` (per-round arguments). Your job is to consolidate the debate into one concrete refinement strategy the Articulator will act on.

Authority asymmetry. *The Decomposer is the HIGHEST authority. The Grounder is for validation only.* The Decomposer wins by default on motion / pivot / kind / object-type / which-side-of-pair / whole-object structure — it saw the full-object image AND carries the Stage-1 kind-level read. The Grounder is a local-validator: its only job is checking whether the Decomposer's call is consistent pixel-level with the crop. The Grounder NEVER overrides the Decomposer on whole-object identity — it cannot tell which door of a pair, what kind of object, or how this kind opens.

When the Grounder's local-validation flags a real inconsistency (e.g. "Decomposer says hinge bottom but crop shows handle on bottom"), surface it in `refinement_strategy` as a Grounder-flagged concern — but resolution still goes through the Decomposer's framework. Do NOT default to "articulator_correct" when the Articulator's whole-object identity disagrees with the Decomposer — the Articulator also only saw the crop.

Output. `grounder_final_motion`, `decomposer_final_motion`, `critics_agree_on_motion`, `motion_matches_articulator`, `consensus` ($\in$ articulator_correct, articulator_wrong_on_motion, articulator_wrong_on_pivot, articulator_wrong_on_hardware, mixed, none), `refinement_strategy` (a full-sentence field-by-field instruction naming the fields to edit and target values), `strongest_dispute`, and `debate_summary`.

Rules. Plain everyday language. `refinement_strategy` must name fields and target values — generic complaints are not useful.

6.7.3 Kinematic tree assembly

Structurer. The Structurer combines per-instance articulation notes, mesh adjacency, and the Decomposer's hierarchy sketch to produce the kinematic tree $\mathcal{K}$.

Structurer system prompt

You figure out how the instances of this object actually fit together, now that every instance has been isolated on the mesh and has its own articulation note. You do not see the photograph — you work from written notes on each instance, adjacency information, and a rough hierarchy sketch from the Decomposer.

What you are given. (1) All instance records: name, kind, description, face count, motion inherited from the Decomposer's pass, and the per-instance articulation note (movement direction, rotation, hidden-geometry judgement). (2) Adjacency information: for each pair of instances, whether they touch, and how large the shared border is (none / small / medium / large). (3) The rough hierarchy sketch from the Decomposer (one or two sentences like "drawers sit inside the body; handles sit on the front of drawers").

Your job. For each instance, determine: `parent_id` (which instance it is attached to), `children` (which instances are directly attached and can move relative to it), and `adjacency_ids` (which other instances share a border zone).

How to reason. An instance with many faces and several neighbours is usually a structural body or frame. An instance with few faces and one or two neighbours is usually a leaf (drawer, handle, door). Use the rough sketch as a strong guide but trust the adjacency information — if it contradicts the sketch, describe the conflict in your reasoning. The articulation note tells you how an instance moves relative to its attachment: a part that slides straight out should have a parent behind or around it; a part that swings sideways should have a parent along its hinge edge. Handles are almost always children of the part they sit on. Drawers are children of the body. Siblings share a parent — they may or may not touch each other.

Output. One entry per instance with `part_id`, `parent_id`, `children`, `adjacency_ids`, and `reasoning`.

Rules. Plain spatial words in reasoning ("large shared border", "sits inside", "attached to the top of"). No X/Y/Z, no pixel coordinates, no face index lists. If you disagree with the rough hierarchy sketch or any articulation note, say so clearly. Do not try to change how any instance moves — you only describe how they fit together.

6.7.4 Round 2 debate: video-grounded refinement (§3.2)

Per-side Round 2 debate. In Round 2, the Decomposer (full last frame) and the Grounder (cropped last frame) each inspect the most articulated frame of the probe video V_i^{probe} independently. Both agents use the same prompt, with the input image and framing adjusted to their respective views. Informed by the quality agent's motion prior, each side judges whether the part opened naturally and, when it did, proposes a revised pivot from the observed swing in the same closed-state codebook from Round 1, using the Round-1 consensus $J_i^{(1)}$ as the starting estimate.

Round 2 per-side debate system prompt

You are looking at the final frame of a video the system generated for one articulated part — the SAME part the Round-1 debate discussed in its CLOSED state. What is new is video evidence of WHICH WAY THE PART OPENED. A quality agent has already screened the clip and given you a coarse motion read; treat it as a prior, not as ground truth. You revise the pivot in the same closed-state codebook (`from_left` / `from_top` / `from_back` percentages), driven by the visible open-state swing.

Step 1 — Naturalness. Decide if the open pose is physically coherent for this kind of part. Hinge: did the part rotate cleanly about a single fixed edge to a sensible angle, body intact, no penetration or fragmentation? Slider: did it translate cleanly without rotating or changing size? If unnatural → `verdict` = keep_prior. If too cropped or blurry to tell → `verdict` = inconclusive.

Step 2 — Pivot proposal (HINGE ONLY). The hinge stays fixed on ONE rest-pose edge while the part swings around it. Identify which rest-pose edge stays put and apply the swing-to-pivot table: swing left → `from_left` $\approx 0\%$; swing right → `from_left` $\approx 100\%$; swing up → `from_top` $\approx 0\%$; swing down → `from_top` $\approx 100\%$; swing forward → `from_back` $\approx 0\%$; swing backward → `from_back` $\approx 100\%$. The two non-hinge axes are $\approx 50\%$; use multiples of 10%. For SLIDERS, skip Step 2 and set `revised_pivot`: null. You are PROPOSING a pivot from the observed motion, not merely confirming Round 1 — if the swing contradicts the Round-1 pivot, say so and give the corrected value.

Anchoring. Everything references the part's REST (closed) bbox edges, never the open-pose position.

Output. `open_pose_plausible`, `open_pose_evidence` (one sentence on what the frame shows), `open_swing_direction`, `revised_pivot`, `verdict` ($\in$ accept_refined, keep_prior, inconclusive), and `verdict_reasoning`.

Rules. Plain everyday language. The pivot encoding must match the Round-1 closed-state codebook. The frame is your single piece of evidence — describe what it actually shows.

Round 2 resolver. The resolver applies a strict-consensus rule: the final articulation $J_i^\star$ adopts the video-grounded revision only when the clip was usable and both sides judge the open pose plausible and (for hinges) agree on the swing direction. When these conditions hold, the update combines the optimized axis from §3.3 with the consensus pivot; otherwise, the resolver retains the Round-1 consensus $J_i^{(1)}$.

Round 2 resolver system prompt

You are a neutral judge consolidating the Round-2 debate that revised the pivot using the visible open-swing direction in the generated end frame. Both sides assessed the frame INDEPENDENTLY using the SAME closed-state codebook from Round 1, each informed by the quality agent's motion prior.

The bundle has: `quality_agent` (the usability verdict and coarse motion read); `motion_type` ($\in$ hinge, slider, unknown); `grounder` (LOCAL viewer's verdict from the crop); `decomposer` (GLOBAL viewer's verdict from the full frame). Each verdict carries `open_pose_plausible`, `open_swing_direction`, `revised_pivot`, `verdict`.

Decision rule — strict consensus. Default is keep_prior. Set `consensus` = accept_refined IFF the quality agent marked the clip usable AND both verdicts are accept_refined AND (for hinge) both reported the same `open_swing_direction` (one of right/left/up/down/forward/backward, NOT unclear). For slider, swing direction is irrelevant. Set `consensus` = keep_prior if the clip was unusable or either side returned keep_prior; `consensus` = inconclusive otherwise.

When accept_refined on a hinge with matching swing, compute `agreed_revised_pivot` by AVERAGING the two sides' `revised_pivot` percentages component-wise and rounding to the nearest 10%. For sliders, `agreed_revised_pivot` = null.

Output. `grounder_verdict`, `decomposer_verdict`, `sides_agree`, `consensus`, `agreed_open_swing_direction`, `agreed_revised_pivot`, `refinement_strategy`, and `debate_summary`.

Rules. Plain everyday language. Use the same integer-percent codebook as Round 1. The strict-consensus rule is mandatory — do not promote accept_refined unless the clip was usable AND both sides voted accept_refined AND (slider, OR hinge with matching non-unclear swing).

6.8 Helper LLMs

Beyond the three reasoning agents, we use auxiliary LLMs to manage the generation flow: when to invoke the debate, what to render during video generation, and when a generated frame is reliable enough to reconstruct from. The optional routing described below can bypass debate or video generation; the reported configuration in Sec. 4.1 uses two debate rounds.

6.8.1 Orchestrator

The Orchestrator decides how much of the pipeline to run for a given object. It selects between the full debate and a Decomposer-only pass based on whether the articulation is ambiguous, and it skips video generation for objects with no occluded interior, recovering the 3D parts directly from the static lifting in that case.

6.8.2 Inpainter

The Inpainter produces the text prompts that condition WAN and Flux, describing the geometry that should become visible during generation, such as a cabinet cavity or a drawer slot once the part is removed.

6.8.3 Orientation LLM

We use an LLM to identify the front-facing direction of each part from the rendered viewpoint, resolving the part's orientation relative to the camera so that subsequent axis and pivot predictions are expressed in a consistent frame.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state our two main contributions, a debate-driven agentic pipeline with three hierarchical agents (Sec. 3) and a dual use of video generative priors for articulation evidence and interior recovery (Sec. 3), both validated by the experiments in Sec. 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: discussed in the paper

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: we provide details on the LLM we used, the GPU, optimization parameter, and datasets used.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Code, agent prompts, and the lists of evaluation samples will be released with documentation. The datasets used (PartNet-Mobility and Objaverse) are already publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Sec. 4.1 and the appendix give the models, segmentation backends, number of video frames, λ_{geo}, number of debate rounds, and evaluation splits. The method is training-free, so no training details apply.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not report error bars because each evaluation run requires multiple VLM queries and video generations per object, making repeated runs across the full 170-object benchmark computationally prohibitive. Three repeated runs on a ten-object subset give an RS-dCD standard deviation of 0.0011 (App. 6.2.3); we did not measure run-to-run variation on the full benchmark.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: App. 6.2.7 reports the GPU type and per-stage runtime of our method and the runtimes of SINGAPO, PhysX-Anything, and FreeArt3D; Articulate-Anything was not included in the timing study.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics. The work involves no human subjects, uses publicly available datasets (PartNet-Mobility, Objaverse) under their respective licenses, and poses no foreseeable risks of harm or misuse beyond those of standard 3D generative modeling.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: This work is foundational research on articulated 3D object reconstruction, with primary applications in embodied AI, robotic simulation, and virtual reality. We identify no direct path to negative societal impact beyond the general considerations applicable to 3D generative modeling.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release any new pretrained models, generative models, or scraped datasets. Our method composes existing publicly available components (TRELLIS2, Flux, WAN, and a VLM) at inference time, and we evaluate on existing public datasets (PartNet-Mobility and Objaverse).

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All existing assets are credited via citation and used in accordance with their terms. We use the PartNet-Mobility Xiang et al. [2020] and Objaverse Deitke et al. [2022] datasets, and the models TRELLIS2 Xiang et al. [2025a], Flux Labs et al. [2025], WAN-VACE Jiang et al. [2025], SAM 3 Carion et al. [2026], Gemini Team et al. [2023], Claude Sonnet 4.6 and Llama 4.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We will release the evaluation samples, i.e., the lists of PartNet-Mobility and Objaverse objects in each test split (the SINGAPO test split of PartNet-Mobility, Objaverse-Household and Objaverse-OOD), with documentation of how they were selected. The objects themselves remain available from the original datasets.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: VLMs are central to our agentic pipeline. We use Claude Sonnet 4.6 for all three agents (Decomposer, Grounder, Articulator) and the debate stages in the main configuration and Llama 4 in the ablations, described in Sec. 3 with the agent prompts in the appendix, with output schemas and minor formatting rules abbreviated.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.