INSIDEART: BENCHMARKING ARTICULATED 3D GENERATION WITH USABLE INTERIORS

Anonymous authors
Paper under double-blind review

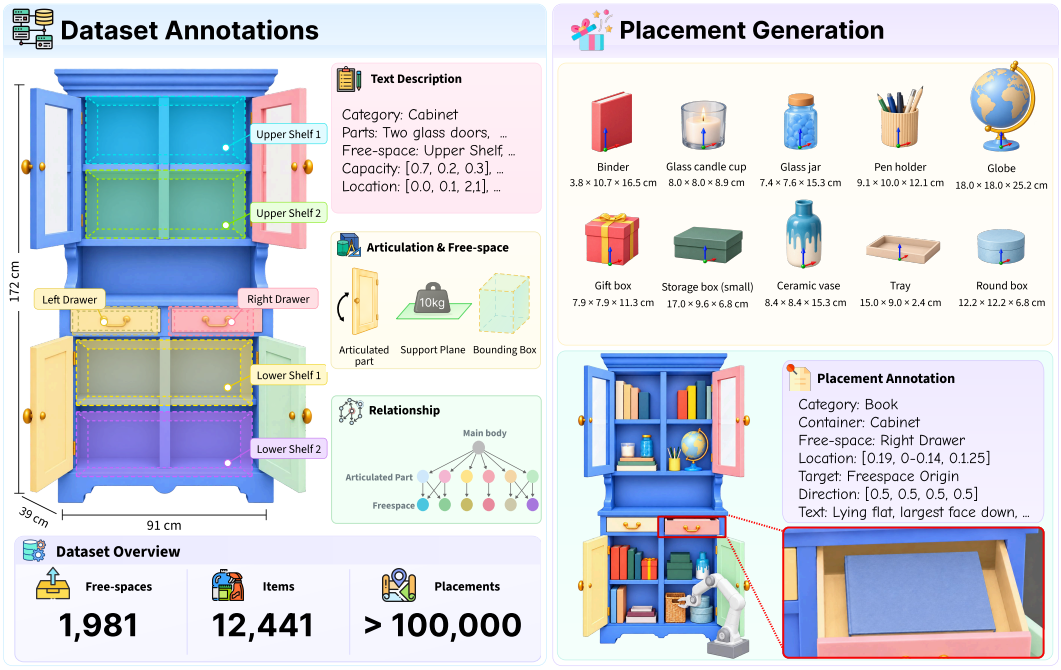


Figure 1: **InsideArt captures articulated interiors and corresponding accessibility.** It links movable parts to support surfaces, free-space regions, and access paths, with over 100,000 placements across 654 containers, 1,981 free-space regions, and 12,441 items. The benchmark supports image-conditioned articulated generation and language-guided interior synthesis and placement, providing a foundation for downstream advanced embodied manipulation tasks.

ABSTRACT

Usable articulated containers require accessible interior space, not just accurate exterior geometry and joints. We introduce **InsideArt**, a dataset and benchmark linking articulated 3D containers to support surfaces, interior free space, ordered and alternative access paths, and language-described item arrangements. Our data-construction pipeline combines vision–language proposals, human verification, geometric constraints, and rigid-body simulation to produce over **100,000** placement scenes across 654 containers with 1,981 annotated free-space regions. The dataset supports three complementary tasks: image-conditioned articulated container generation, language-guided interior synthesis, and language-guided item placement. We evaluate interior-volume recovery and access relations beyond surface similarity, and define a placement protocol covering containment, collision freedom, stability, and orientation compliance. Reconstruction and free-space evaluations reveal that accurate surfaces can coexist with missing interior regions and incorrect access relations, highlighting the need to evaluate functionality beyond appearance and joint accuracy. Our dataset, benchmark, and source-code will be made publicly available.

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

1 INTRODUCTION

Many everyday interactions with furniture concern what is *inside* it. Retrieving a book from a cabinet drawer requires a usable compartment, a physically plausible item placement, and a motion that provides access. Simulating such interactions therefore requires more than movable furniture: containers must have usable interiors populated with accessible items. Geometry, articulation, and item arrangements jointly determine their functionality.

Existing resources address these aspects largely separately. PartNet-Mobility and GAPartNet emphasize parts and joints (Xiang et al., 2020; Geng et al., 2023), while 3D-FRONT focuses on furniture layouts (Fu et al., 2021). Interactive environments such as AI2-THOR, ProcTHOR, and RoboCasa populate scenes using fixed asset libraries and procedural placements (Kolve et al., 2017; Deitke et al., 2022; Nasiriany et al., 2024). Recent articulated generators recover parts and kinematics (Cao et al., 2026), but plausible exteriors and joints do not ensure usable compartments. This motivates generating articulated containers with interior geometry and language-guided item arrangements.

We use a closed-state image as the visual reference, with language specifying hidden interior structure and guiding placement. Three challenges arise. First, unobserved interiors must be synthesized consistently with both the visible exterior and language-specified partitions and compartments. Second, interior regions must be associated with the parts that expose them, including access dependencies beyond the kinematic tree: a door may need to open before a drawer extends, or either door may expose a shared compartment. Third, item arrangements must satisfy containment, collision avoidance, support, and orientation constraints alongside instructed spatial relations. These challenges require linked supervision and evaluation beyond surface reconstruction and joint accuracy.

We introduce **InsideArt**, a dataset and benchmark comprising 654 articulated containers from PartNet-Mobility and Artiverse, $1{,}981$ free-space regions, $12{,}441$ interior items, and over $100{,}000$ placements across three difficulty levels. Its annotations connect multi-state observations and structural descriptions with interior regions, access paths, item poses, support relations, and placement language. Our construction pipeline combines vision–language proposals, human verification, geometric constraints, and rigid-body simulation. Humans verify annotations and accept or reject automatically generated layouts rather than manually arranging items.

Beyond establishing the benchmark, we evaluate and analyze existing methods on image-conditioned articulated generation, language-guided interior synthesis, and language-guided placement, distinguishing image-only from text-assisted inputs. The results expose complementary limitations: the evaluated generators reach at most 0.66 free-space instance F1 and 0.65 access-topology F1 (Section 5.2). For placement, only 10% of VLMPose's hard scenes are collision-free, whereas GPT-6-Astra is largely collision-free but places fewer than 30% of items within $5\,\text{cm}$ of the instructed positions (Section 5.3). These findings distinguish geometric plausibility from accessibility and instruction compliance. To this end, our major contributions are summarized as follows:

- **A dataset of articulated container interiors.** To the best of our knowledge, InsideArt is the first dataset to jointly annotate interior free space, kinematic access paths, and language-described multi-item placements within articulated containers.
- **A verified data-construction pipeline** combining vision–language proposals, human verification, geometric processing, and constrained automatic placement.
- **An interior-aware benchmark and empirical study.** We define complementary generation and placement tasks, evaluate existing methods, and analyze their limitations in interior recovery, accessibility, physical feasibility, and instruction compliance.

2 RELATED WORK

Indoor scene synthesis. Early work on indoor scene generation relied on procedural rules and optimization with hand-crafted priors on object co-occurrence and ergonomics (Yu et al., 2011; Fisher et al., 2012). With the availability of large-scale scene datasets such as 3D-FRONT (Fu et al., 2021), learning-based methods became dominant: autoregressive transformers such as SceneFormer (Wang et al., 2021) and ATISS (Paschalidou et al., 2021) generate object-by-object layouts,

while diffusion-based approaches such as DiffuScene (Tang et al., 2024), LEGO-Net (Wei et al., 2023), and PhyScene (Yang et al., 2024a) denoise full scene configurations, with PhyScene additionally enforcing physical plausibility and interactivity-aware guidance. More recently, large language models have been used to drive open-vocabulary scene synthesis, either by predicting layouts directly, as in LayoutGPT (Feng et al., 2023), or by orchestrating retrieval and constraint solving, as in Holodeck (Yang et al., 2024b) and InstructScene (Lin & Mu, 2024). However, all of these methods place *static* assets: the generated furniture cannot be opened, and the scenes are not directly usable for interaction-rich simulation. Our work addresses exactly this gap by generating interiors whose objects carry part-level articulated structure.

Articulated object modeling and generation. Datasets such as PartNet-Mobility with the SAPIEN simulator (Xiang et al., 2020) and GAPartNet (Geng et al., 2023) enabled learning-based modeling of articulated objects. One line of work reconstructs articulation from observations: Ditto (Jiang et al., 2022) and PARIS (Liu et al., 2023) infer joint parameters and part geometry from before/after interaction states, while URDFormer (Chen et al., 2024) and Articulate-Anything (Le et al., 2025) convert images or meshes into simulation-ready URDF descriptions. Another line generates articulated objects directly: CAGE (Liu et al., 2024) generates controllable part-level geometry and joints from a category and structure specification, and SINGAPO (Liu et al., 2025) produces articulated assets from a single image. These methods focus on the geometry and kinematics of individual articulated assets. In contrast, our benchmark links articulated containers to their contents through interior free space, access conditions, and spatial and physical constraints on interior item placement.

Interactive environments for embodied AI. Simulation platforms such as AI2-THOR (Kolve et al., 2017), iGibson (Shen et al., 2021), and Habitat (Szot et al., 2021) provide interactive scenes, but their environments are manually authored and therefore limited in scale and diversity. ProcTHOR (Deitke et al., 2022) showed that procedurally generated houses substantially improve embodied agent generalization, and RoboCasa (Nasiriany et al., 2024) demonstrated the value of generative kitchen environments for robot manipulation learning. These systems, however, rely on fixed asset libraries with pre-authored articulation and procedural placement rules. Instead, this paper proposes a data generation pipeline that enables scalable creation of diverse 3D articulated assets with simulation-ready interiors beyond the coverage of any fixed asset library.

3 Dataset Overview

InsideArt is a dataset and benchmark for generating articulated 3D containers together with their interior structure. Each articulated container, such as a cabinet, is populated with interior items such as trays, glass jars, and gift boxes. Alongside the 3D assets, InsideArt provides images of each container in multiple articulation states (e.g. closed and opened), a language description of its structure, images of the items placed inside, and the spatial relationships between those items. It supports tasks including, but not limited to: (a) image-conditioned 3D articulated object generation; (b) language-guided synthesis of interior geometry; and (c) language-guided placement of items inside containers. This section describes the data format and statistics. The data-generation pipeline is introduced in Section 4, and Section 5 benchmarks the tasks listed above.

3.1 Dataset Schema

Each data sample links a container record to a placement record. The former describes the container's articulation, interior regions, and access conditions; the latter places items in target regions. Figure 2 illustrates the container (C1–C4) and placement (P1–P5) schemas of a real example.

Container schema. The container record comprises four field groups:

C1. *Identity and scale:* identifier, category, scale, and metric dimensions.

C2. *Articulation and states:* joint identifiers, parent links, types, axes, limits, closed/open values, and action descriptions.

C3. *Interior regions:* region identifiers, names, 3D free-space boxes, and dimensions.

C4. *Motion attachment and access:* the moving part associated with each region, alternative ordered opening sequences, and access descriptions.

Placement schema. The placement record comprises five field groups:

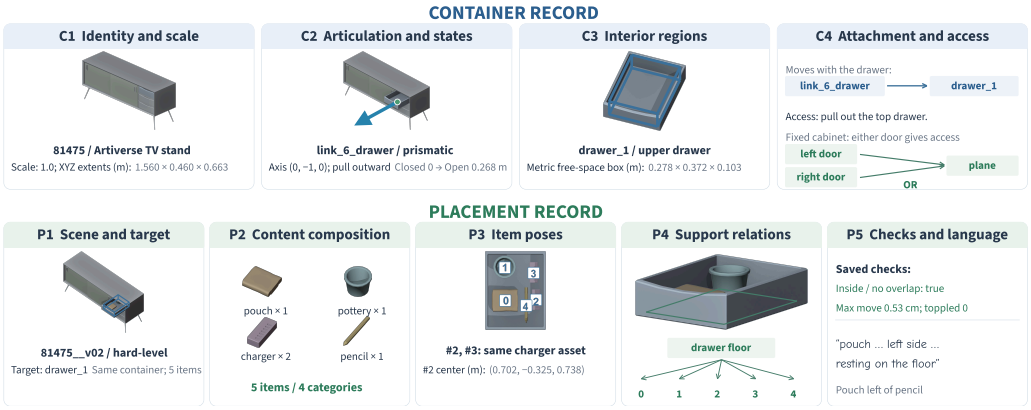
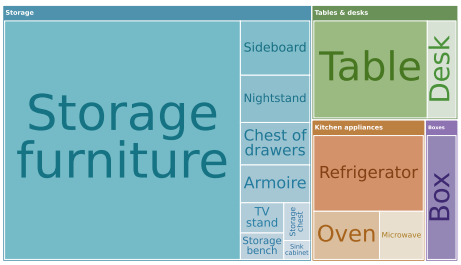
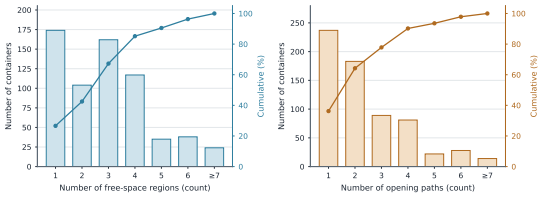


Figure 2: **InsideArt schema illustrated by a real example.** Top: container records (C1-C4) of a TV stand. Bottom: its five-item upper-drawer placement record (P1-P5). Better viewed zoomed in.



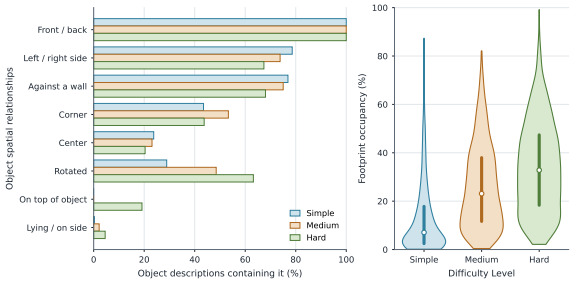
(a) Container categories



(b) Free-space regions and opening paths per container



(c) Items per semantic group

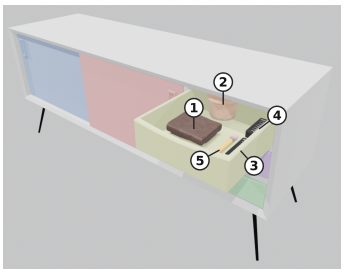


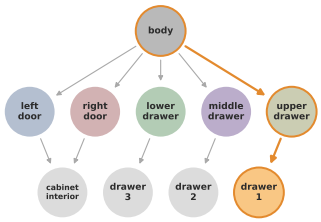
(d) Spatial relations and area occupancy of placements

Figure 3: **InsideArt statistics.** (a) Container categories, with area proportional to the numbers, grouped by function. (b) Distribution of containers by the number of available spaces and opening paths. (c) Number of items across 16 semantic groups, covering 198 item categories. (d) Difficulty levels of placements. Left: spatial relationships used at each difficulty level. Right: the percentage of the available placement area occupied by items.

P1. *Scene and target:* scene identifier, difficulty level, container identifier, and target region.
P2. *Content composition:* food/non-food class, layout type, and instance counts by item category.
P3. *Item instances and poses:* instance indices, asset identifiers, sources, categories, scales, up-axis conventions, metric dimensions, final positions, and rotations.
P4. *Support relations:* the supporting entity (floor or another item) and stacking layer.
P5. *Validation and language:* recorded displacement, toppling, and post-settling containment; per-item descriptions, spatial relations, and a scene summary.

Repeated items may share an asset but retain distinct instance indices and poses. Open-state observations and reference poses serve as supervision or evaluation labels, not default inputs to closed-image generation in the benchmark. Please refer to Appendices I.1 and I.3 for more details.





Access: pull out the upper drawer (free space *drawer 1*).

1. pouch: left side, mid-depth, against the left wall;
2. pottery: back-left corner, against the left wall;
3. battery charger: right side, mid-depth, against the right wall;
4. battery charger: back-right corner, against the right wall;
5. pencil: front-right corner, lying on its side.

All items rest on the drawer floor.

Figure 4: A placement scene in a TV stand with five items in the upper drawer (with colored articulated part). Left: the opened container and placed items. Middle: the access graph from articulated parts to free spaces. Right: the access requirement and item position descriptions.

3.2 Dataset Statistics

Containers. InsideArt contains 654 articulated containers from PartNet-Mobility and Artiverse, covering 16 categories with storage furniture (444), tables and desks (94), kitchen appliances (90), and boxes (26) (Figure 3a,b). 85.2% of containers have at most four regions, with 1,981 free-space regions in total. For distinct opening paths, 64.2% have one or two and 90.2% have at most four.
Items. The dataset includes 12,441 placeable items spanning food, tableware, stationery, textiles, and storage accessories. Category, scale, and canonical orientation annotations support compatible selection and placement in a metric frame(Figure 3c). Appendix A shows the category hierarchy.
Placements. InsideArt provides more than 100,000 placements (Figure 1). *Simple* layouts use one item category; *middle* layouts combine multiple categories without stacking; and *hard* layouts increase category diversity or introduce stacking(Figure 3d). Multiple variants share container geometry and access annotations. Layouts undergo constraint-based checks and, where needed, gravity settling (Section 4); placement counts are distinct from item-asset and scene-record counts.

3.3 Data Visualization

Figure 4 shows one placement scene and its annotations. The container has four free spaces accessed through five articulated parts; the target is the upper drawer, so items are placed in the drawer's local frame. The instruction specifies each item's region, wall contacts, orientation, and support. A placement is correct if all items satisfy these constraints, avoid collisions, and remain stable under gravity. More visualizations are provided in Appendix B.

4 Data Generation Pipeline

Joint reasoning over articulation, interior space, and item placement requires supervision that explicitly links interior geometry, kinematic access, and internal item arrangements. For each container, this supervision includes usable support planes and their associated free spaces, the articulated parts to be actuated to access them, and valid item placements. We construct this supervision with a pipeline organized into three modules. The first annotates articulated containers (Stages 1–3); another branch prepares candidate items and their compatibility rules (Stages 4–5); the two streams merge in the third module, which generates placements and unifies all annotations (Stages 6–9).

4.1 Articulated Container Annotation

Stage 1: Multi-view, multi-state observation. Each container is observed under a set of articulation states. For every state we render RGB images, colored part renders, and depth maps, and we record the part states together with a part-identity map. These records allow annotations made in image space to be associated with the parts and surfaces in the configuration in which they were observed, while the closed-state image later serves as the conditioning input for generation. An illustration of the multi-view, multi-pose observation process is provided in Appendix G.
Stage 2: Structure description proposal and verification. A VLM analyzes the multi-pose images

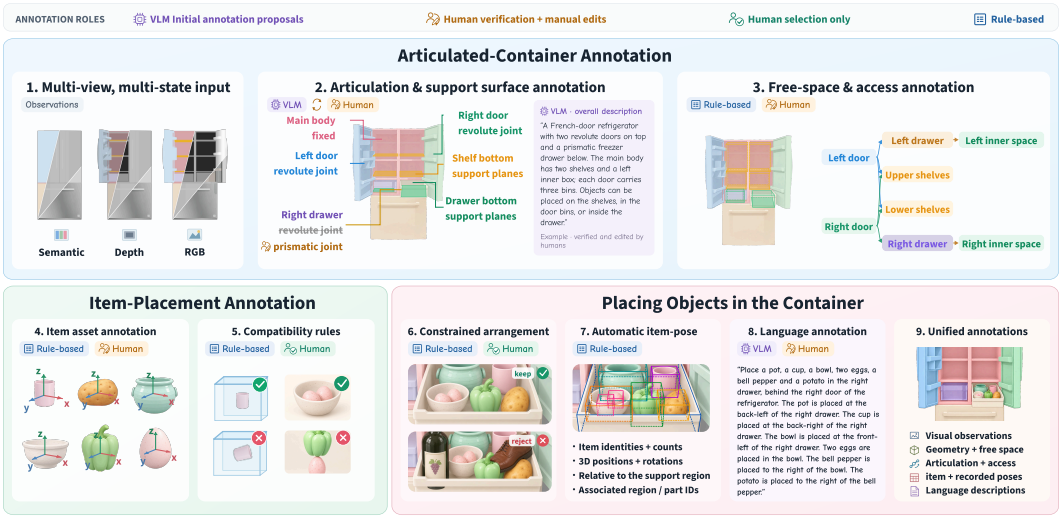


Figure 5: Overview of the annotation pipeline and the roles of vision–language models, human annotators, and rule-based processing. (1) Annotate container articulation, support surfaces, free space, and access; (2) define item properties and compatibility rules; (3) generate constrained placements with pose and language annotations.

and proposes a structured description of each container: its constituent parts, the joints that move them, a semantic description of the container and its interior, and candidate support points marking where internal items could rest. These are treated as proposals and verified by human annotators. Expert annotators review the proposals through a dedicated interface over several passes. They correct the positions of support points, check the identified parts and their descriptions, and confirm that each image-space annotation corresponds to the intended 3D geometry. The output is a verified inventory of parts, descriptions, and support points for every container.

Stage 3: Free-space and access-condition annotation. Verified support points are lifted to 3D in their observed configurations, and the usable free volume above each surface is estimated by expanding the region until geometric occlusion is reached. Annotators refine free-space boundaries, merge over-segmented cavities, and standardize surface names. Access conditions are then inferred by comparing closed and actuated configurations to identify moving parts and determine which parts occlude each surface. These relations are represented as access paths, capturing both alternative routes (e.g., opening either of two doors) and ordered dependencies (e.g., opening a door before extending a drawer). The stage outputs usable 3D volumes, their metric scale, a canonical coordinate frame, and the articulation conditions required to access each free space.

4.2 Item Placement Preparation

Another branch prepares the items to be placed. It runs independently of container annotation and joins the main pipeline at Stage 6.

Stage 4: Item asset annotation. Placeable rigid items are collected from several asset sources, deduplicated, and split into individual items when an asset contains several. Annotators then correct each item's geometry and pose, producing its category, principal axis and canonical upright axis, canonical resting pose, facing direction, and real metric scale and dimensions. These attributes allow every item to fit in the containers in a shared metric frame.

Stage 5: Compatibility rules. *Food is NEVER placed on a desk, and a milkshake MUST stand upright.* We specify category-level rules that govern how items may be combined with containers: the item categories allowed on each kind of container and surface, the orientations in which an item may rest, stacking relations that are semantically and practically valid, and nesting compatibility between items, such as which foods may be placed on which vessels. Together with the calibrated dimensions, these rules constrain both the selection of items and their arrangement.

Table 1: **Articulated reconstruction on InsideArt.** Surface accuracy, part poses, and sibling-part overlap, grouped by input setting.

Input setting	Method	CD ↓	F ↑	RS-dgIoU ↓	AS-dgIoU ↓	RS-dcDist ↓	AS-dcDist ↓	AOR ↓
PartNet-Mobility (Xiang et al., 2020)								
Image-only	PhysX-Anything (Cao et al., 2026)	116.39	59.05	1.009	1.014	0.250	0.452	**0.005**
	SPARK (He et al., 2026)	66.05	76.23	0.856	0.869	0.202	0.463	0.064
	URDF-Anything+ (Wu et al., 2026)	54.02	83.97	0.962	0.974	0.287	0.468	0.025
	PAct (Liu et al., 2026)	**25.01**	**96.14**	**0.578**	**0.595**	**0.077**	**0.202**	0.008
Image + text	GPT-6-Astra	34.88	91.74	0.655	0.670	0.092	0.160	0.000
Artiverse (Iliash et al., 2026)								
Image-only	PhysX-Anything (Cao et al., 2026)	142.40	47.99	0.910	0.914	0.233	0.365	**0.004**
	SPARK (He et al., 2026)	73.93	73.37	1.046	1.053	0.202	0.310	0.049
	URDF-Anything+ (Wu et al., 2026)	58.43	82.04	1.084	1.092	0.252	0.337	0.036
	PAct (Liu et al., 2026)	**36.99**	**93.00**	**0.750**	**0.760**	**0.111**	**0.158**	0.023
Image + text	GPT-6-Astra	31.72	94.75	0.565	0.583	0.061	0.078	0.000

4.3 Item Placement and Unified Annotation

Stage 6: Constrained arrangement. Given the verified free space, the annotated item pool with its compatibility rules, layouts are generated automatically. Candidate items are sampled and arranged within the placement volumes subject to size, containment, collision, support, orientation, and semantic constraints. For surfaces that admit a planar approximation, resting poses are computed analytically; otherwise, including all stacked arrangements, use rigid-body simulation with gravity and collision detection inside the opened container. Multiple variants of placements are produced per surface, ranging from repeated instances of one category to combinations of compatible categories and permitted stacks. Automatic checks flag potential geometric or semantic violations, and annotators then select appropriate arrangements also considering own judgment.
Stage 7: Automatic item-pose. For every accepted layout, the placement records are extracted: the identity and count of the placed items, their 3D positions and directions, their poses relative to the support region they rest on, and the identifiers of the associated region and articulated part. Because these records are derived from the same geometry used for generation, they remain consistent with the container annotations and can be regenerated when upstream annotations change.
Stage 8: Language annotation. The structured records are converted into natural-language descriptions of three kinds: spatial relations, which describe where items lie relative to the support surface and to one another; pose changes, which describe how items are oriented and stacked; and access operations, which describe the parts that must be opened or extended to reach the placed items.
Stage 9: Unified annotation. Finally, each sample packages the visual observations, the container geometry and free space, the articulation structure and access conditions, the placed items with their recorded poses, and the language descriptions into a single annotation. Container and item images are indexed in order of reference and linked to the corresponding text, so that every element of a scene can be traced back to the observation, geometry, and rule that produced it.

5 Experiments

Our full task is to generate an articulated container and its interior items from closed-state images, a structural description, and language instructions specifying the interior arrangement. Since current methods cannot jointly generate multiple 3D items with the required relative spatial relationships, we evaluate the task on InsideArt in two stages connected by the container's interior free space. *Container generation* evaluates whether the generated articulated asset matches the input image, has correct parts and joints, and provides usable free spaces with valid access paths (Sections 5.1 and 5.2). *Item placement* evaluates whether items are arranged in the correct compartments and positions, without collisions and with physical stability, using ground-truth containers to avoid confounding placement performance with generation errors (Section 5.3).

5.1 Articulated Reconstruction Quality

We compare two input tracks on both source subsets. The *Image-only* track includes PhysX-Anything (Cao et al., 2026), SPARK (He et al., 2026), URDF-Anything+ (Wu et al., 2026), and

Table 2: **Free-space and access-topology evaluation on InsideArt.** PAct and GPT-6-Astra results grouped by input setting.

Input setting	Method	Input image	FS-dgIoU ↓	UFVR ↓	Inst. F1 ↑	Topo. F1 ↑
PartNet-Mobility (Xiang et al., 2020)						
Image-only	PAct (Liu et al., 2026)	closed	0.457	0.300	0.662	0.652
	PAct (Liu et al., 2026)	open state	0.674	0.636	0.338	0.319
Image + text	GPT-6-Astra	closed	0.651	0.636	0.341	0.338
Artiverse (Iliash et al., 2026)						
Image-only	PAct (Liu et al., 2026)	closed	0.775	0.801	0.198	0.189
	PAct (Liu et al., 2026)	open state	0.745	0.726	0.253	0.240
Image + text	GPT-6-Astra	closed	0.498	0.328	0.647	0.632

PAct (Liu et al., 2026), using closed-state renderings without text. PAct additionally uses supplied part masks, so its input is not strictly RGB-only. The *Image + text* track contains GPT-6-Astra. Bold values indicate the best score within the image-only track for each source subset; we do not highlight cross-track winners. Table 1 reports surface accuracy (CD, F-score) together with part-level pose error in the resting and articulated states (RS/AS-dgIoU, RS/AS-dcDist) and sibling-part overlap (AOR). Metrics are averaged over items; definitions are given in Appendix D. Remaining part, joint and precision/recall numbers are in Appendix C. GPT-6-Astra is included as a reference: it produces procedural reconstructions from the closed-state image and an additional structured text prompt. Within the image-only input, PAct achieves the best surface reconstruction and part-state scores on both source subsets. GPT-6-Astra provides the text reference, with particularly strong scores on the Artiverse-derived subset. Despite plausible exterior geometry, the learning-based methods produce incomplete interiors, incorrect articulation, and inter-part collisions (Appendix F.5). We also build and test a rule-based method from multi-state 3D observations (Appendix F.6).

5.2 Free-Space and Access-Topology Evaluation

We evaluate whether generated containers recover the annotated interior free spaces and their access paths. We acquire the free space and the access path for the predictions with the same procedure. Predictions and references are normalized by their own closed-pose bounding boxes, and free spaces and movable parts are matched independently by centroid-based Hungarian assignment.

Metrics. **FS-dgIoU** reports the mean $1 - \mathrm{gIoU}$ over matched free-space boxes, capturing errors in both box location and extent. **UFVR** quantifies unmatched interior space as the fraction of total predicted and reference free-space volume that lacks a valid match at $\mathrm{IoU} \geq 0.5$, thereby penalizing both missing and spurious regions. **Instance F1** evaluates instance-level recovery using harmonic mean of free-space precision and recall at the same IoU threshold. Finally, **Topology F1** requires each geometric match to recover at least one annotated ordered access path, testing whether reconstructed regions have the correct opening sequence. Formal definitions are given in Appendix D.

Results. Table 2 reveals substantial source-dependent variation in interior recovery. With closed-state image input, PAct achieves an instance F1 of 0.662 on the PM-derived subset and 0.198 on the Artiverse-derived subset, while text-assisted GPT-6-Astra obtains 0.341 and 0.647, respectively. Topology F1 is consistently lower than instance F1, indicating additional errors in associating recovered regions with their access paths. Compared too closed-state inputs, open-state images provide a clearer interior, but PAct's open poses reduce free-space localization in closed-pose evaluation.

Articulation recovery pilot. As a complementary probe of the top-down route, we also recover movable parts and their joint motions from framewise reconstructions of rendered states, using color-aware registration (Park et al., 2017) and CUPID pose estimates (Huang et al., 2026) to transfer ground-truth part masks. The diagnostics, and three failure cases are reported in Appendices E–F.6; even with mask assistance, part recovery and motion selection remain unreliable.

5.3 Language-Guided Item Placement

We report results for VLMPose (Baik et al., 2026) with qualitative position instructions and GPT-6-Astra with metric instructions across easy, medium, and hard splits (Table 3). Target-region success measures the fraction of ground-truth objects whose predicted bounding-box centres lie inside

Table 3: **Language-guided item placement inside articulated containers.** (Q)/(I): qualitative/metric instructions. CFR = collision-free rate; Stab. = physics stability success. Values are percentages.

	Target-region success ↑			Position accuracy@5cm ↑			CFR ↑			Stab. ↑		
Method	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
VLMPose (Q)	81.3	79.4	80.1	—	—	—	67.6	21.7	10.0	46.6	23.6	11.5
GPT-6-Astra (I)	—	—	—	29.1	27.1	27.2	88.0	83.5	74.5	80.5	76.8	60.9



Figure 6: Downstream manipulation in simulation. A mobile Franka arm with a dexterous hand opens the lower door of a generated articulated cabinet and retrieves a basket from the bottom compartment. The container, its interior free space, and the placed items all come from our dataset.

the target free space, indicating coarse region correctness. Position accuracy@5cm measures the fraction whose predicted centres are within 5 cm of the ground-truth centres, indicating absolute positional accuracy. These are distinct criteria and their scores should not be compared directly. Collision-free rate measures the fraction of scenes without penetration into any open-pose container part or between objects, indicating geometric feasibility. Physics stability success measures the fraction of objects that remain within motion thresholds after a fresh gravity rollout, indicating stability under gravity. Both methods use the same collision and stability protocols (Appendix D).

VLMPose places about 80% of item centres in the target free space across all difficulty levels, but its collision-free rate drops from 67.6% on easy to 10.0% on hard scenes. Under metric instructions, GPT-6-Astra achieves position accuracy below 30% at the 5 cm threshold, while its collision-free rate and stability fall from 88.0% and 80.5% to 74.5% and 60.9% respectively. Within each instruction setting, position scores remain relatively stable as difficulty increases, while physical feasibility declines. Neither method achieves physically valid and reliable placement. Two other language-guided placement methods, Copy-Transform-Paste (Gatenyo & Fried, 2026) and PlaceIt3D (Abdelreheem et al., 2025), are not directly applicable to our setting. The former lacks fine-grained spatial grounding for instructions inside articulated compartments, while the latter targets room-scale ScanNet support surfaces and relies on asset-specific features, making it unsuitable for enclosed interiors and our object set.

5.4 Downstream Applications

InsideArt can instantiate open–retrieve–place manipulation episodes without manually authoring each scene, illustrating its potential for embodied policy training. In practice, an agent opens an articulated container in order to reach and grasp the items stored inside. Every container comes with verified joints, interior free space, and items placed under gravity, which lets a simulated robot open the door, reach into the compartment, and take an item out without any manual scene authoring (Figure 6). We expect such open-then-grasp episodes, generated at scale, to push mobile manipulation training one step further than scenes assembled from fixed asset libraries.

6 Conclusion

We introduced InsideArt, a dataset and benchmark connecting articulated containers, interior free space, access paths, and language-guided item placement. Experiments show that recovering accessible interiors and producing accurate, collision-free, stable placements remain challenging for current methods. Future work will use these structured scenes to train and evaluate robots on long-horizon tasks involving opening containers, placing items, and retrieving them.

AI USE STATEMENT

In this work, we used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, assist in writing proofs, implement methods, assist with translation, clean and reformat datasets, generative AI tools to generate synthetic datasets, and support qualitative and thematic data analysis. We have not used provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, or interpret results. No required-disclosure task categories are marked as not applicable to this work.

Additionally, we used generative AI tools to create or modify scientific figures or images, summarize or analyze existing literature, brainstorm, source and search for information, edit the paper to improve readability, identify relevant literature, and format references.

We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Ahmed Abdelreheem, Filippo Aleotti, Jamie Watson, Zawar Qureshi, Abdelrahman Eldesokey, Peter Wonka, Gabriel Brostow, Sara Vicente, and Guillermo Garcia-Hernando. PlaceIt3D: Language-guided object placement in real 3D scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Sangwon Baik, Gunhee Kim, Mingi Choi, and Hanbyul Joo. Text-guided 6D object pose rearrangement via closed-loop VLM agents. In *European Conference on Computer Vision (ECCV)*, 2026.

Ziang Cao, Fangzhou Hong, Zhaoxi Chen, Liang Pan, and Ziwei Liu. Physx-anything: Simulation-ready physical 3d assets from single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5839–5848, 2026.

Chuhao Chen, Isabella Liu, Xinyue Wei, Hao Su, and Minghua Liu. FreeArt3D: Training-free articulated object generation using 3D diffusion. In *SIGGRAPH Asia 2025 Conference Papers*, pp. 1–13, 2025.

Qiuyu Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Dieter Fox, and Abhishek Gupta. URDFormer: A pipeline for constructing articulated simulation environments from real-world images. In *Robotics: Science and Systems (RSS)*, 2024.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ProcTHOR: Large-scale embodied AI using procedural generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Weixi Feng, Wanrong Zhu, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. LayoutGPT: Compositional visual planning and generation with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Matthew Fisher, Daniel Ritchie, Manolis Savva, Thomas Funkhouser, and Pat Hanrahan. Example-based synthesis of 3D object arrangements. *ACM Transactions on Graphics*, 31(6), 2012.

Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, and Hao Zhang. 3D-FRONT: 3D furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Rotem Gatenyo and Ohad Fried. Copy-Transform-Paste: Zero-shot object-object alignment guided by vision-language and geometric constraints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14936–14945, 2026.

Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. GAPartNet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

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 (CVPR)*, 2026.

Binbin Huang, Haobin Duan, Yiqun Zhao, Zibo Zhao, Yi Ma, and Shenghua Gao. CUPID: Generative 3D reconstruction via joint object and pose modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Denys Iliash, Jiayi Liu, Egor Fokin, Qirui Wu, Ali Mahdavi-Amiri, Manolis Savva, and Angel X. Chang. Artiverse: A diverse and physically grounded dataset for articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An interactive 3D environment for visual AI. *arXiv preprint arXiv:1712.05474*, 2017.

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 (ICLR)*, 2025.

Chenguo Lin and Yadong Mu. InstructScene: Instruction-driven 3D indoor scene synthesis with semantic graph prior. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. PARIS: Part-level reconstruction and motion analysis for articulated objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Jiayi Liu, Hou In Ivan Tam, Ali Mahdavi-Amiri, and Manolis Savva. CAGE: Controllable articulation generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Jiayi Liu, Denys Iliash, Angel X Chang, Manolis Savva, and Ali Mahdavi-Amiri. SINGAPO: Single image controlled generation of articulated parts in objects. In *International Conference on Learning Representations (ICLR)*, 2025.

Qingming Liu, Xinyue Yao, Shuyuan Zhang, Yueci Deng, Guiliang Liu, Zhen Liu, and Kui Jia. PAct: Part-decomposed single-view articulated object generation. *arXiv preprint arXiv:2602.14965*, 2026.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.050.

Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 143–152, 2017. URL https://openaccess.thecvf.com/content_iccv_2017/html/Park_Colored_Point_Cloud_ICCV_2017_paper.html.

Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. ATISS: Autoregressive transformers for indoor scene synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. iGibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. DiffuScene: Denoising diffusion models for generative indoor scene synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. SceneFormer: Indoor scene generation with transformers. In *International Conference on 3D Vision (3DV)*, 2021.

Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. LEGO-Net: Learning regular rearrangements of objects in rooms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Zhuangzhe Wu, Yue Xin, Chengkai Hou, Minghao Chen, Yaoxu Lyu, Jieyu Zhang, and Shanghang Zhang. URDF-Anything+: Autoregressive articulated 3D models generation for physical simulation. *arXiv preprint arXiv:2603.14010*, 2026. URL https://arxiv.org/abs/2603.14010.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X Chang, Leonidas J Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. PhyScene: Physically interactable 3D scene synthesis for embodied AI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024a.

Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, Chris Callison-Burch, Mark Yatskar, Aniruddha Kembhavi, and Christopher Clark. Holodeck: Language guided generation of 3D embodied AI environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024b.

Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F Chan, and Stanley J Osher. Make it home: automatic optimization of furniture arrangement. *ACM Transactions on Graphics*, 30(4), 2011.

A Additional Dataset Statistics Details

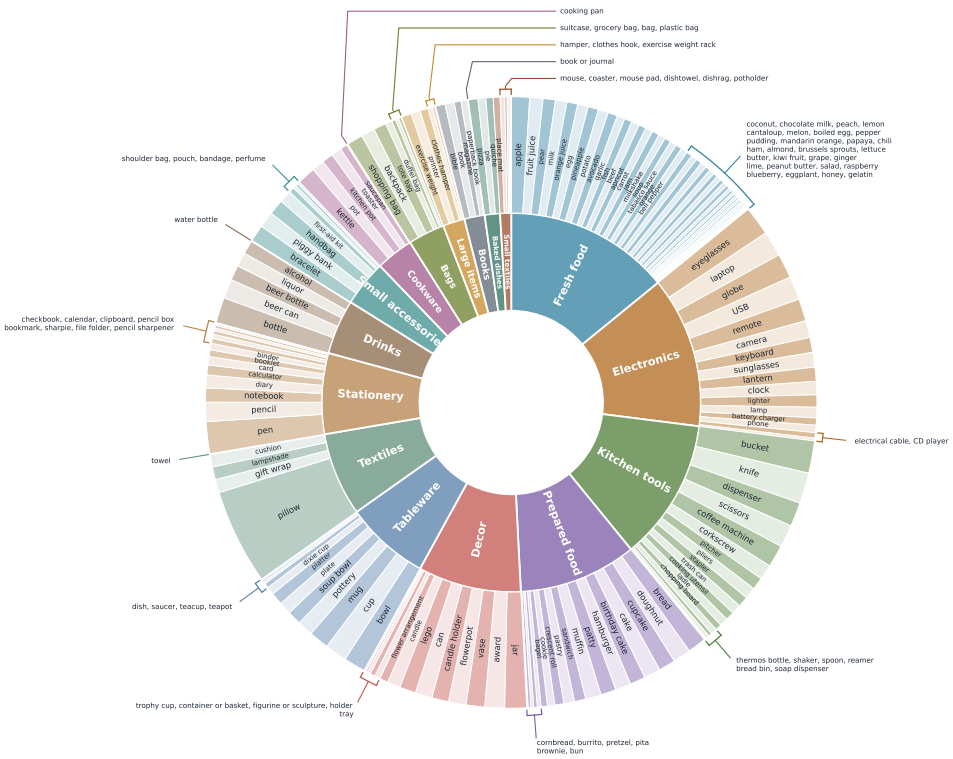


Figure 7: **Detailed item categories of InsideArt.** The 198 item categories as a two-level hierarchy grouped into the 16 semantic groups: the inner ring shows the semantic groups and the outer ring the categories, with angle proportional to the number of items.

This section expands the item statistics of Section 3.2. Figure 7 shows all 198 item categories as a two-level hierarchy under the 16 semantic groups. Because category names were unified across the three asset sources, a few categories keep merged names such as *container or basket* or *book or journal* where the sources labeled the same kind of asset differently.

Semantic groups. The 16 groups organize the categories by function and typical storage context, such as fresh and prepared food, tableware and cookware, or stationery and books. Figure 3c is colored by group, while the compatibility rules of Stage 5 (Section 4) are written at the category level. Group sizes are shown in Figure 3c. The four largest groups (fresh food, electronics, kitchen tools, and prepared food) hold 49% of the items and the four smallest 6%. The smallest group, small textiles, has 39 assets.

Category granularity. The number of categories per group varies more than the number of objects. Fresh food has 48 categories, kitchen tools and prepared food 19 each, and baked dishes three (pizza, pie, and quiche). The average category size ranges from about 12 items in fresh food to about 59 in textiles, where pillows dominate.

B Additional Data Visualization

Figure 8 shows further placement scenes from the easy, medium and hard splits, rendered and annotated as in Figure 4.

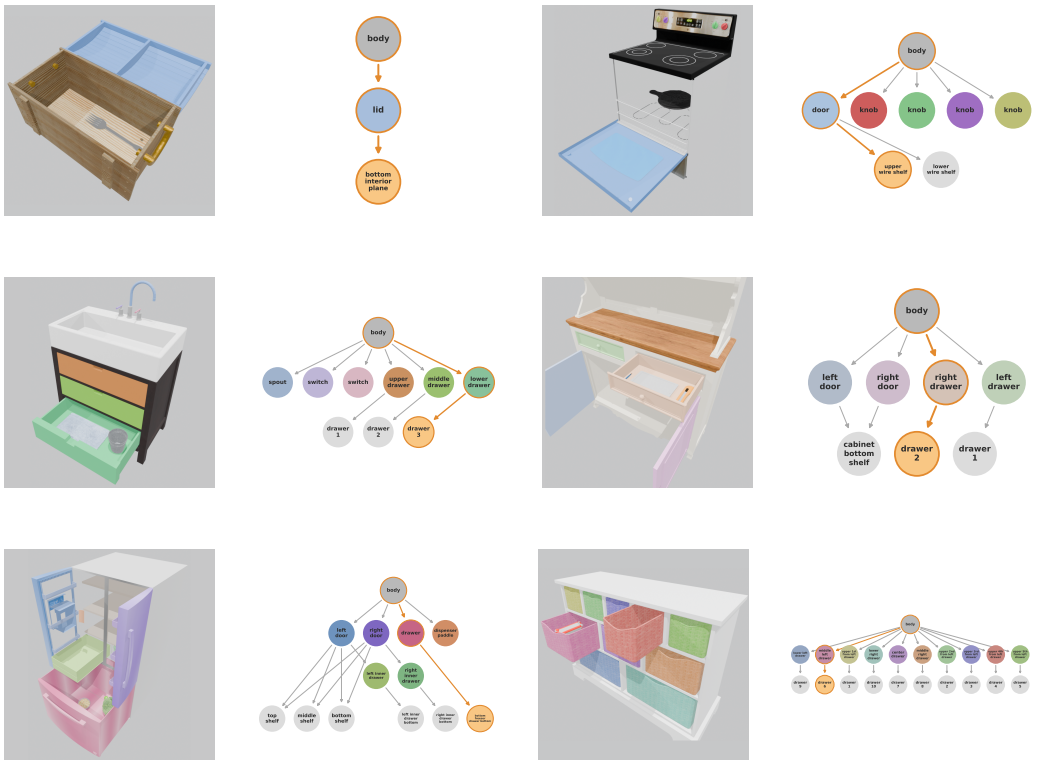


Figure 8: **Additional placement scenes.** Each pair shows the opened container with its placed objects (articulated parts tinted) and its access graph in the same colors, with the occupied free space and its access path highlighted. Placement examples at three difficulty levels, easy, medium, and hard, shown from top to bottom.

C Additional Quantitative Results

Table 4 lists the part, joint and precision/recall numbers omitted from Table 1: the surface precision and recall behind the F-score, the Chamfer-based part distances (RS/AS-dCD), the joint axis and pivot errors, joint-type accuracy, and part detection precision, recall and F1. Part detection counts a predicted part as correct when it is Hungarian-matched to a ground-truth part; joint accuracy is the fraction of matched parts whose joint type is also correct. The part and joint metrics of Tables 1 and 4 follow established definitions rather than new ones: part matching, the resting/articulated-state split and the d-gIoU family are those of SINGAPO (Liu et al., 2025), the average overlap ratio is that of CAGE (Liu et al., 2024), and the joint axis and pivot errors follow FreeArt3D (Chen et al., 2025); oriented-box intersections and AOR are estimated by Monte-Carlo sampling, so values may differ from the original implementations in the last reported digit.

Two non-articulated single-image generators are also available as references on shape alone. On the PM-derived subset, Hunyuan3D-2.1 reaches CD 32.72 / F 93.19 and TRELLIS CD 46.69 / F 88.94; on the Artiverse-derived subset, CD 32.81 / F 92.11 and CD 41.54 / F 91.02. They produce a single mesh with no parts or joints, so none of the part, joint, free-space or topology metrics apply to them.

Table 2 uses a matching threshold of τ=0.5. Relaxing it to τ=0.25 raises the closed-input numbers on the PM-derived subset to Inst. F1 0.828 and Topo. F1 0.807, and those on the Artiverse-derived subset to 0.531 and 0.486; the ordering between settings is unchanged. For GPT-6-Astra the same relaxation gives Inst. F1 0.688 and Topo. F1 0.679 on the PM-derived subset, and 0.864 and 0.839 on the Artiverse-derived subset.

Table 4: **Additional reconstruction metrics on InsideArt.** P_s, R_s = surface precision and recall at τ_F=0.05; dCD = part Chamfer distance; Axis, Pivot = joint axis angle and pivot position error; JAcc = joint-type accuracy; P_{part}, R_{part}, F_{part} = part detection scores. †See Table 1.

Asset source	Method	$P_s \uparrow$	$R_s \uparrow$	RS-dCD $\downarrow$	AS-dCD $\downarrow$	Axis(°) $\downarrow$	Pivot $\downarrow$	JAcc $\uparrow$	$P_{\text{part}} \uparrow$	$R_{\text{part}} \uparrow$	$F_{\text{part}} \uparrow$
PM	PhysX-Anything	0.594	0.606	0.131	0.511	17.86	0.357	0.664	0.707	0.599	0.618
	SPARK	0.726	0.819	0.100	0.375	18.28	0.566	0.754	0.715	0.701	0.698
	URDF-Anything+	0.861	0.834	0.146	0.394	42.16	0.294	0.444	0.685	0.512	0.554
	PAct	**0.967**	**0.958**	0.033	0.151	10.92	0.095	0.904	0.938	0.904	0.916
	GPT-6-Astra†	0.916	0.921	**0.025**	**0.079**	**0.09**	**0.069**	**0.956**	**0.957**	**0.925**	**0.932**
Artiverse	PhysX-Anything	0.503	0.487	0.157	0.372	21.85	0.293	0.283	0.740	0.365	0.439
	SPARK	0.688	0.815	0.098	0.208	14.41	0.475	0.704	0.660	0.591	0.614
	URDF-Anything+	0.845	0.816	0.133	0.237	62.38	0.261	0.200	0.604	0.368	0.415
	PAct	0.937	0.929	0.051	0.089	14.72	0.124	0.775	0.863	0.797	0.822
	GPT-6-Astra†	**0.946**	**0.951**	**0.018**	**0.020**	**0.62**	**0.055**	**0.961**	**0.970**	**0.945**	**0.955**

D METRIC DEFINITIONS

Table 5 summarizes the placement metrics used in Table 3. We write $\mathcal{F}$ for the annotated target free space inside the container cavity, and $\hat{c}_i$ and c_i for the predicted and ground-truth bounding-box centres of object i in a common scene frame, measured in metres. A placement is one complete multi-object scene on a target support region.

Table 5: **Placement metrics** reported in Table 3.

Metric	Definition	↑ / ↓
Target-region success	Fraction of ground-truth objects with a matched prediction satisfying $\hat{c}_i \in \mathcal{F}$; measures coarse region correctness	↑
Position accuracy@5cm	Fraction of ground-truth objects with a matched prediction satisfying $\|\hat{c}_i - c_i\|_2 \leq 0.05$; measures absolute positional accuracy	↑
Collision-free rate	Fraction of scenes with no penetration between any object and any container part, or between any two objects	↑
Physics stability success	Fraction of objects whose translation and rotation after a gravity rollout stay below thresholds	↑

Position-instruction diagnostics. Target-region success checks centre membership, not full-volume containment or compliance with every qualitative spatial relation. Position accuracy@5cm checks absolute centre error, not containment. Table 3 reports the former for qualitative instructions and the latter for metric instructions; their values are not directly comparable, and — means unreported rather than inapplicable. We additionally evaluate qualitative position instructions using normalized left/center/right and front/middle/back regions or explicit object–object order constraints. For metric instructions, distances refer to object surfaces and the corresponding interior walls. We report wall-clearance errors, target-center distance, and target-box IoU on the axes determined by the instruction; full 3D IoU is reported only when all three axes are determined. A single wall distance requires an independently annotated target size to reconstruct the target interval. Unspecified axes are not imputed from the prediction. These are diagnostics under position evaluation, not additional columns in Table 3.

Evaluation protocol. Predicted objects are matched to ground-truth instances of the same asset by Hungarian assignment on centre distance; ground-truth objects without a prediction count as failures in both position metrics and make the scene fail. Both position metrics use all ground-truth objects as the denominator. Collisions are first tested with exact mesh booleans (a $10^{-9}\,\text{m}^3$ intersection-volume tolerance distinguishes contact from penetration). Pairs whose bounding boxes overlap but whose meshes are not watertight cannot be resolved by booleans; for these we query penetration depth in PyBullet with the static container triangles and convex-decomposed objects, and count a collision when the depth exceeds 1 cm. Stability uses a fresh 3 s gravity rollout at 240 Hz with translation/rotation tolerances of 1 cm/5° and terminal-window speeds below 0.01 m/s and 0.1 rad/s; decomposition parameters are held fixed across methods.

Exact definition of metrics. All boxes are axis-aligned and live in the object's normalized closed-pose frame: each side is translated by its own closed-pose bounding-box centre c and divided by that box's longest edge s, so absolute scale and position cancel. Write $b = (x_1, x_2, y_1, y_2, z_1, z_2)$, $V(b) = \prod_{a\in\{x,y,z\}}(a_2 - a_1)$, and for two boxes

$$\begin{aligned} I(b, b') &= \prod_{a\in\{x,y,z\}} \max\big(0, \min(a_2, a_2') - \max(a_1, a_1')\big) \\ \mathrm{IoU} &= \frac{I}{V(b) + V(b') - I} \\ \mathrm{gIoU} &= \mathrm{IoU} - \frac{V(C) - U}{V(C)}, \end{aligned} \tag{1}$$

where $U = V(b) + V(b') - I$ and C is the smallest box enclosing both. Let $\mathcal{A}^g = \{a_i^g\}_{i=1}^{N_g}$ and $\mathcal{A}^p = \{a_j^p\}_{j=1}^{N_p}$ be the ground-truth and predicted articulated parts, and let $\mathcal{G}$ and $\mathcal{P}$ be their free-space sets.

Part correspondence. Let $\mu(a)$ denote the centre of part a's axis-aligned bounding box in the normalized closed-pose frame. We define the matching cost as

$$D_{ij} = \|\mu(a_i^g) - \mu(a_j^p)\|_2. \tag{2}$$

Hungarian assignment finds a one-to-one pair set $\mathcal{P}_A$ of size $\min(N_g, N_p)$ that minimizes $\sum_{(i,j)\in\mathcal{P}_A} D_{ij}$. The resulting ground-truth-to-prediction map is $M_A : \mathcal{A}^g \to \mathcal{A}^p \cup \{\perp\}$, defined by

$$M_A(a_i^g) = \begin{cases} a_j^p, & (i,j) \in \mathcal{P}_A, \\ \perp, & \text{if } a_i^g \text{ is unmatched.} \end{cases} \tag{3}$$

Here $\perp$ is a special unmatched symbol, not a predicted part. The default part assignment applies no distance or IoU rejection threshold: a distant pair may still be assigned, and ground-truth parts remain unmatched when there are fewer predicted parts. Access topology is not used for matching.

Free-space pairs $\mathcal{M}$ are obtained independently by Hungarian assignment on their bounding-box centre distances. Unlike the default part assignment, free-space matches are accepted only above an IoU threshold: $\mathcal{M}_\tau = \{(g,p) \in \mathcal{M} : \mathrm{IoU}(g,p) \geq \tau\}$, with $\tau = 0.5$.

Access paths. Each free space g carries a set of access paths Π_g; a path is the ordered tuple of parts that must be opened, nearest first, so a shelf behind a door inside a drawer is $(\text{door}, \text{drawer})$ and a shelf reachable through either door of a pair contributes two paths. Mapping through the part assignment,

$$S_g = \big\{\, (M_A(a_1), \ldots, M_A(a_n)) \;:\; (a_1, \ldots, a_n) \in \Pi_g \,\big\}, \qquad S_p = \Pi_p, \tag{4}$$

Any mapped ground-truth path containing $\perp$ cannot match a predicted path, since predicted paths contain only predicted-part identifiers.

Free-space and access-topology metrics.

$$\text{FS-dgIoU} = \frac{1}{|\mathcal{M}|} \sum_{(g,p)\in\mathcal{M}} \big(1 - \mathrm{gIoU}(g,p)\big) \tag{5}$$

$$\mathrm{UFVR} = \frac{\sum_{g\notin\mathcal{M}_\tau} V(g) + \sum_{p\notin\mathcal{M}_\tau} V(p)}{\sum_{g\in\mathcal{G}} V(g) + \sum_{p\in\mathcal{P}} V(p)} \tag{6}$$

$$\text{Inst. F1} = \frac{2\,\mathrm{TP}}{|\mathcal{P}| + |\mathcal{G}|}, \qquad \mathrm{TP} = |\mathcal{M}_\tau| \tag{7}$$

$$\text{Topo. F1} = \frac{2\,\mathrm{TP}_{\text{topo}}}{|\mathcal{P}| + |\mathcal{G}|}, \qquad \mathrm{TP}_{\text{topo}} = \big|\{(g,p) \in \mathcal{M}_\tau : S_g \cap S_p \neq \emptyset\}\big| \tag{8}$$

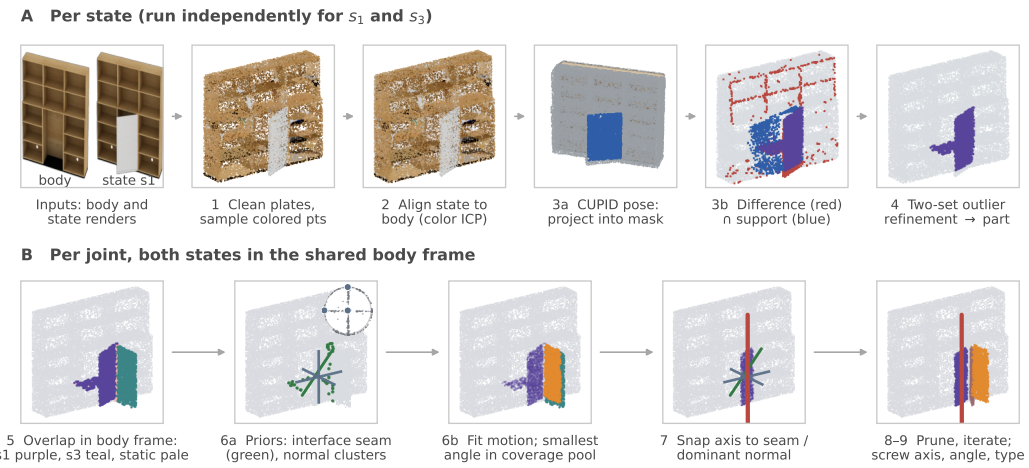


Figure 9: Articulation recovery pipeline on a bookcase door (joint a_0, states s_1 and s_3).

E Articulation Recovery Pipeline

We first locate the moving part, then estimate how it moves. Comparing each generated state with a reconstruction of the body alone identifies geometry that may belong to the part. An image mask narrows these candidates to the target region. We then compare the recovered parts in the body frame and alternate motion fitting with outlier removal. The outputs are colored part point sets and an estimated relative motion for each supported joint.

Setting and notation. For each joint, we use a Hunyuan body reconstruction with *all* movable parts removed, reconstructions of four rendered states $s_0, \ldots, s_3$, and each state's CUPID camera pose (Huang et al., 2026) and renderer-provided target-part mask. The body is shared across joints. Reconstruction models remain frozen, and the pilot is mask-assisted. Each mesh yields 30,000 colored surface points, denoted X_k for state k. We estimate a similarity G_k from each state to the body frame, then fit a rigid part motion H between s_1 and s_3 in that frame.

We use separate distance tolerances for comparison with the body (τ_b) and between states (τ_x). A motion's symmetric coverage measures how much of each part cloud it brings close to the other, averaging both directions.

Step-by-step summary. The stage runs once per joint; Figure 9 illustrates the steps on a bookcase door.

1. **Clean and sample.** Remove spurious thin plates from every generated mesh, re-normalize to a longest edge of two, and sample 30,000 colored surface points per mesh.
2. **Align each state to the body.** Estimate the similarity G_k with color-aware registration (Park et al., 2017). RGB provides an additional cue for disambiguating 180° flips of symmetric bodies.
3. **Difference and image support.** Collect state points farther than τ_b from the aligned body into $\mathcal{D}_k$. Separately, project points with the CUPID camera and retain visible points inside the part mask as $\mathcal{Q}_k$. Their intersection, $\mathcal{D}_k \cap \mathcal{Q}_k$, is the initial part.
4. **Two-set outlier refinement.** Compare each selected point's neighborhood inside the part with its neighborhood among the remaining state points. Remove it if the fifth-nearest neighbor outside is less than half as far away as the fifth-nearest neighbor inside. Add back unselected points satisfying the reverse condition.
5. **Cross-state overlap test.** Map the refined parts of s_1 and s_3 into the body frame with G_1 and G_3. Without further registration, remove part points within τ_x of the *full* cloud of the other state. This rejects shared occupied positions and can also discard overlapping surfaces of moving parts.

Table 6: Full-resolution mask-assisted refinement diagnostics over 244 states. Each state has 30,000 sampled points. Means weight states equally, including empty selections under Eq. 9. Recovered fragments are shown separately and then included in the refined part.

Selection	Mean points	Mean invisible	Precision	Recall
Initial intersection	1576.9	0.0	0.996	0.707
Cleaned intersection	1508.4	0.0	0.996	0.632
Recovered fragments	11.9	8.5	0.098	0.0001
Refined part	1520.3	8.5	0.993	0.632

6. **Fit the relative motion.** Fit a rigid H between the remaining parts using generic initializations and hypotheses based on seams and dominant normals. Score candidates by symmetric coverage. Among those within 0.30 of the best coverage, select the smallest rotation angle.
7. **Snap to priors.** Try the seam axis, the closest dominant normal, and the seam direction in that order. Keep the first constrained fit whose coverage drops by at most 0.05.
8. **Prune and iterate.** Remove points not explained by H within τ_x on either side, refit with the previous H as an additional initialization, and stop when the supports are unchanged or after six rounds.
9. **Report the joint.** Decompose the final H into axis direction, axis point, angle and axial displacement; label it revolute if the angle is at least 10° and prismatic otherwise.

F Articulation Diagnostics and Evaluation Boundaries

We analyze saved pilot outputs without rerunning reconstruction or fitting.

F.1 Coverage and No-Motion Screening

We start with 20 objects and 376 states. We exclude missing reconstructions and screen out joints whose masks remain nearly unchanged across states: the minimum IoU between s_0 and the other states must exceed 0.85. This screen can also remove small or silhouette-preserving motions. The resulting evaluation set contains 244 states across 61 joints. Insufficient part support prevents fitting for seven joints, leaving 54 fits and 52 ground-truth comparisons. Some fits reach the iteration cap, so a returned fit does not establish convergence.

F.2 Mask-Assisted Segmentation Metrics

For a selected point set $Z \subseteq X_k$, with V_k the camera-visible samples and $\mathcal{Q}_k$ the lifted ground-truth mask,

$$\mathrm{Prec}(Z) = \frac{|Z \cap \mathcal{Q}_k|}{\max(|Z \cap V_k|, 1)}, \qquad \mathrm{Rec}(Z) = \frac{|Z \cap \mathcal{Q}_k|}{\max(|\mathcal{Q}_k|, 1)}. \tag{9}$$

Intersecting with the ground-truth mask makes precision perfect for every nonempty initial selection. Empty selections score zero. Recall is therefore the useful measure of how much of the target remains. Table 6 shows that local refinement removes target points while recovering few others. Motion filtering further reduces mean recall to roughly one quarter in the fitted pairs. A surface present in one reconstruction may have no counterpart in the other and be discarded. The final points provide partial fitting support, with much of the original part geometry lost.

F.3 Joint-State Evaluation

The reference angle is the median magnitude of $R_3 R_1^\top$ over moved visuals. Revolute labels use a 1° reference threshold and a 10° estimate threshold. Table 7 reports rotation-magnitude error. Its small pooled median is dominated by prismatic cases, where near-zero rotation error does not establish that a drawer travels the correct distance.

Table 7: Current motion diagnostics on the 52 fitted pairs with comparable reference angles. Angle error is the absolute error in rotation magnitude. Prismatic rows do not measure translation accuracy. Type accuracy follows the different reference and estimate thresholds stated in the text.

Reference type	Pairs	Median error	Error $< 15^\circ$	Correct type
Revolute	22	7.85°	16/22	19/22
Prismatic	30	0.021°	29/30	29/30
All	52	0.022°	45/52	48/52

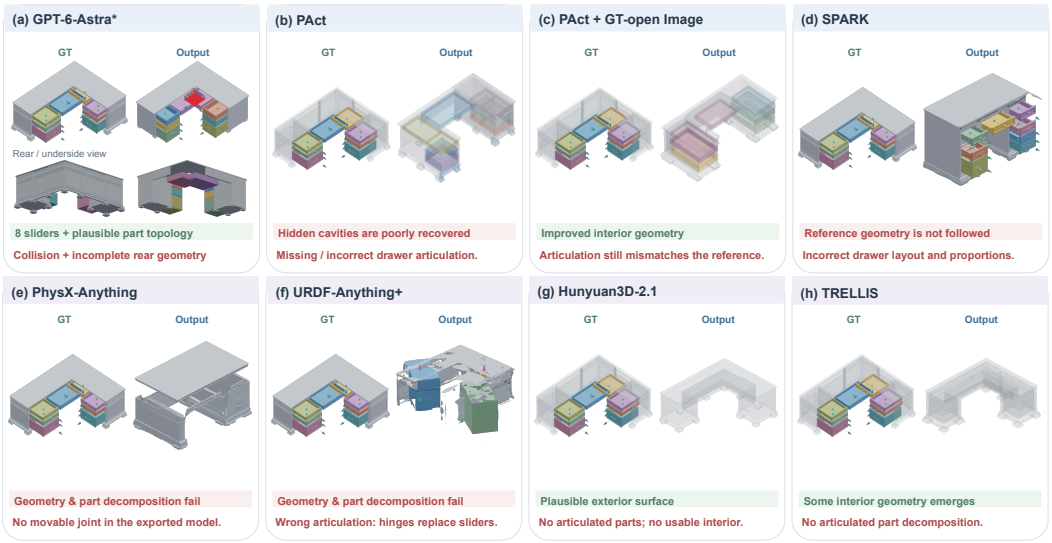


Figure 10: **Qualitative failure cases across eight method/input settings on a desk demo.** Each panel compares the reference (left) with a method output (right). Translucency exposes interiors, colors distinguish exported parts, and red highlights the verified drawer intersection.

The angle metric alone cannot determine whether the part follows the correct hinge or moves in the correct direction. The examples below expose ambiguities that remain despite apparently plausible angle errors.

F.4 Diagnostic Summary and Evaluation Scope

Over-selection is the most frequent diagnostic flag: unrelated surfaces enter the part candidate in 104 of 244 states. Low registration coverage is the next largest category, with 45 states. Other flags concern mask transfer and missing target points. These categories identify symptoms, while the cases below explain how they arise. The pilot uses ground-truth masks and only two states for motion fitting. Independent 3D segmentation accuracy remains untested. Without calibration between the generated and reference coordinate frames, we evaluate rotation magnitude only. Axis direction, axis position, translation magnitude, and rotation sign remain unvalidated. The results also do not isolate the benefit of the geometric priors.

F.5 Failure Cases of Learning-Based Method

Figure 10 compares eight method on the same articulated desk, revealing distinct gaps between exterior fidelity and usable articulated interiors. GPT-6-Astra follows the intended articulation and part topology, but produces colliding drawers and incomplete rear geometry (a). With only a closed-state image, PAct poorly reconstructs the hidden cavities and predicts seven rather than eight drawer joints (b). Providing the GT open-state image improves its interior geometry, but does not resolve articulation (c). SPARK's drawer layout and proportions deviate from the geometry specified by

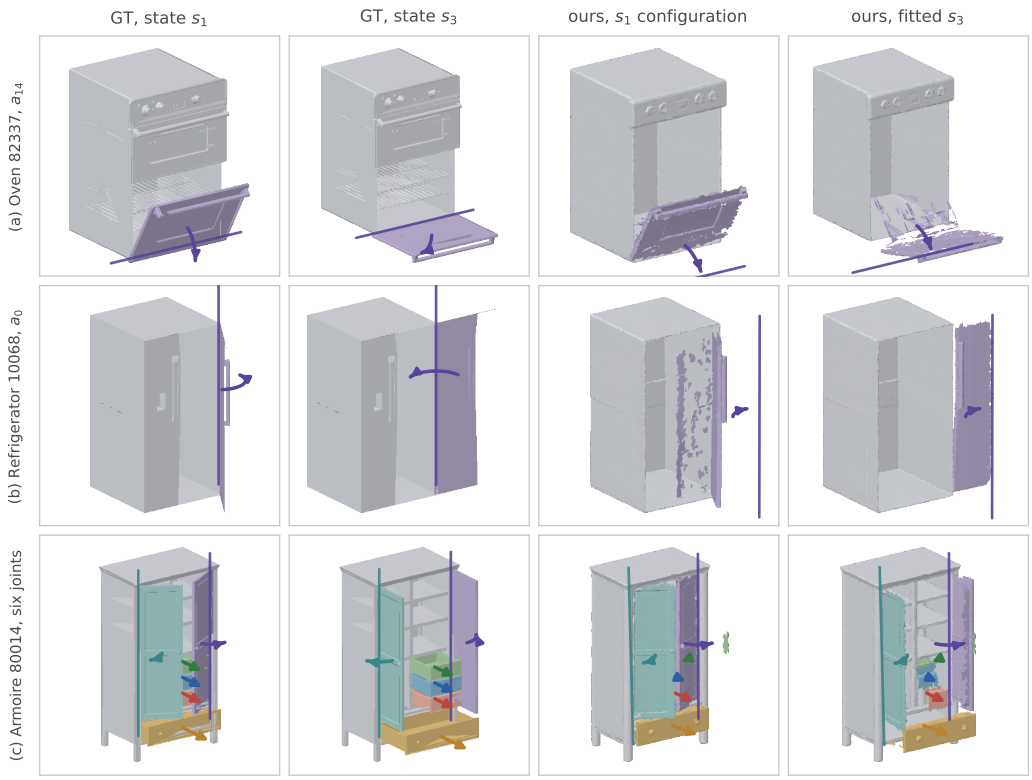


Figure 11: Recovered parts and motions compared with ground truth. The left pair shows the reference URDF at s_1 and s_3. The right pair shows parts cut from the generated s_1 mesh, placed on the generated body before and after the fitted motion. Bodies are grey and moving parts are colored. Arcs indicate rotation and straight arrows indicate translation. Compare both the recovered geometry and its motion: (a) the oven door opens too little, (b) the refrigerator door follows a much shorter sweep about a displaced hinge, and (c) refinement recovers drawer fronts within a nested assembly. The closed-state mask-transfer failure and raw difference regions discussed in the text are separate saved diagnostics and are not shown here.

the reference image and structural description (d). PhysX-Anything and URDF-Anything+ show failures in geometry, movable-part decomposition, and articulation (e,f). Hunyuan3D-2.1 captures the exterior surface but does not recover usable interior compartments or an articulated part decomposition(g). TRELLIS recovers some interior geometry, yet its whole-shape output likewise lacks articulated parts (h). These qualitative examples show that plausible surfaces or correct joint counts alone do not ensure collision-free, structurally faithful interiors, motivating joint evaluation of geometry, part structure, and articulation.

F.6 Failure Cases of Rule-Based Method

The saved examples reveal three distinct issues: image evidence attached to the wrong surface, similar endpoint geometry explained by the wrong motion, and small parts mixed with unrelated interior surfaces. Figure 11 shows the final recovered meshes and motions. The supporting point sets and figure scripts are archived with the outputs.

The door is found, but its image mask lands elsewhere (oven). In the saved closed-state output, the geometric difference correctly covers the door. Transferring the image mask places its support on the oven's top and back, so intersecting the two largely removes the door. The closed oven resembles a box, allowing a flipped alignment to score well geometrically. This example shows how an erroneous image constraint can discard a useful geometric candidate. Comparing the transfer

across open and closed states would provide a consistency check. Figure 11a instead shows the part recovered from open states, where this ambiguity is reduced, but the fitted door still opens less than the reference.

The plates overlap, but the opening motion is wrong (refrigerator). The reference door in Figure 11b swings 120° about its vertical front hinge. The fit rotates it 62° about a displaced vertical axis, yet the transformed plate closely overlaps the target surface. A saved near-reference candidate is rejected. Thin-plate symmetry admits competing correspondences and rotations. The minimum-angle rule favors the shorter alternative among candidates with similar coverage. It can even favor translation for a rotating door because translation has zero rotation angle. An intermediate state would test whether the same axis explains motion throughout the opening sequence.

A small drawer is surrounded by a large difference region (armoire). Opening the doors exposes shelves and inner walls around the target drawer. These surfaces also enter the geometric difference when the independent reconstructions disagree. The drawer occupies only a small portion of this candidate region, so the raw-difference IoU is low. Refinement nevertheless recovers its front, and the fitted outputs distinguish rotating doors from sliding drawers (Figure 11c). Raw candidate quality and refined part quality must therefore be assessed separately. A second issue appears when the doors are closed: some image support lands on the mirrored door, consistent with a left–right alignment ambiguity.

G Multi-view, Multi-pose Observation

Figure 12 illustrates the Stage 1 observations for two PartNet-Mobility containers. Each articulation state is rendered from fixed viewpoints, storing RGB, part-identity, and depth maps. Movable links retain consistent colors across views and states (gray denotes the static body), enabling pixels to be associated with their corresponding parts and surfaces; the examples show a partially open state and a fully opened state.

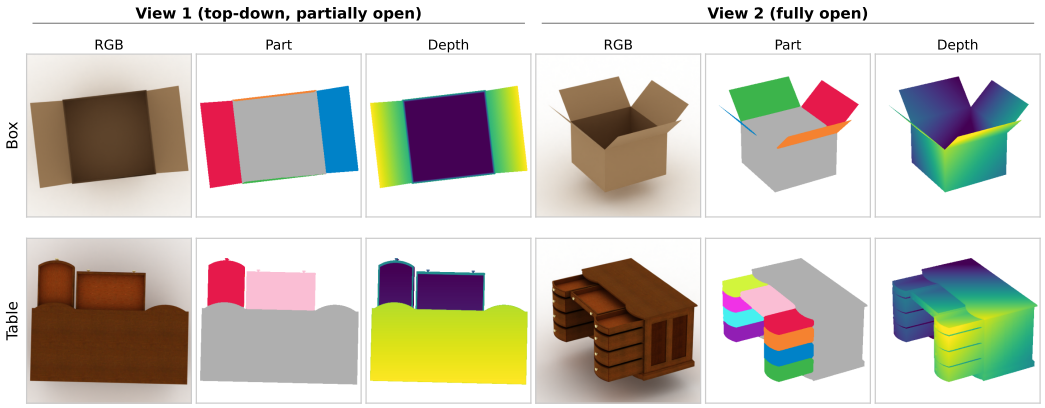


Figure 12: Multi-view, multi-pose observations of a box (top) and table (bottom). Each row shows RGB, part-identity (gray: static body), and depth renders from two fixed viewpoints. View 1 is a top-down view of a partially open state, while View 2 has all joints fully opened.

H Human Verification and Annotation Protocol

H.1 Articulation and Support Surface Annotation

Articulation annotation. Stage 2 alternates between the vision–language model and manual correction. The model proposes an annotation, we correct it, and the corrected version is returned to the model for the next round. The rounds stop when we find nothing left to correct. We check



Figure 13: Overview of the human verification and annotation protocol, including support surface annotation, free-space annotation, access annotation, and semantic and physical constraint checks.

whether the model has found every free-space region of the container, whether every part and region is described correctly, and whether the articulation type of each movable part is right. For the refrigerator in Figure 5 of the main paper, the verified annotation lists five placeable regions, five articulated parts, and five support points. The figure shows one such correction, where the right drawer proposed as a revolute joint is relabeled as prismatic. **Support surface annotation** (Figure 13). We check whether the support points proposed by the model are plausible, that is, whether an object could rest on that surface at that point. For each surface, we pick a viewpoint among the multi-pose images and mark the support point as an image coordinate (x, y) in that view.

H.2 Free-Space and Access Annotation

Free-space annotation (Figure 13). Free-space regions start from the verified support points of Stage 2. From each support point, a breadth-first search expands through the free volume and stops wherever it meets a container surface, both horizontally and vertically. The expanded region is converted into a bounding box on the support plane, which we then trim or extend. A box comes out too small when the mesh is uneven, because bumps on the support plane or on the surrounding walls stop the expansion early. A box comes out too large, reaching beyond the container, when the support plane is not enclosed by a closed mesh and the search leaks through the gaps. Dish racks and decorative cabinets with hollow centers are typical cases. **Access annotation.** Access annotation records the articulation graph that links movable parts to the regions they expose. For every region, we verify the ordered sequence of parts that has to be actuated before an object inside can be reached. For the refrigerator in Figure 5 of the main paper, an object in the left drawer is reached by opening the left door and then pulling out the left drawer, which exposes the left inner space.

H.3 Object Asset Annotation

Source assets arrive in arbitrary orientations. For each object, we determine the pose in which it rests stably on a horizontal surface, as it would in a real scene, and record two directions in that pose, the upward axis (z) and the front-facing direction. These two directions define the canonical resting pose used in all later stages.

H.4 Compatibility Rules

In our work, we judge the compatibility of items with a container in a 3D viewer and keep or reject each proposal without editing it. We first check the container type and the size of the target region, and whether the items listed for the scene match the objects placed in it. We then translate, rotate, and zoom the view to inspect the position and orientation of every object and to confirm that it

lies inside the container. When several objects are present, we also check the arrangement and the stacking and judge whether the objects are placed sensibly in the available space.

H.5 Constrained Arrangement

Semantic constraints. We first judge whether the objects belong in the container and next to each other, see Figure 13. A layout is rejected when an object is out of place for the container, such as shoes in a refrigerator, or when a stack pairs objects that do not belong together, such as a bottle on a pizza. **Physical constraints.** We then judge whether the layout obeys physical rules such as gravity. Every object has to rest on the support surface of the container or on another object, without floating, coincident or penetrating meshes, or collisions. In a stack, a larger object is not placed on a smaller one, and the volume of the object on top does not exceed that of the object below. Stacks also have to be stable under gravity. In the rigid-body simulation, a layout is accepted when each object slides by at most 5 cm and rotates by at most 20° between placement and rest, and comes to rest within the top surface of the object below.

I Annotation Examples

Both examples below are taken from the same container, an armoire from the artiverse split (id `80110`) with two doors and two drawers. The first shows the structured articulation and free-space record that Stage 3 of the pipeline produces (Section 4); the second shows the prompt for generating the 3D container based on the image input. Freespace boxes are given in the URDF frame of the closed container, in metres, and a box that belongs to a movable part (here the two drawers) records that part in `moves_with`. Access to each free-space region is specified by one or more alternative joint-actuation sequences (e.g. [[joint1],[joint2]]), with all joints in a chosen sequence actuated in the specified order (e.g. [[joint1, joint2]]). For example, the top shelf can be accessed by opening either the left or the right door.

I.1 An example of articulated & freespace template

Articulation and free-space record (JSON)


```
{
  "id": "80110",
  "category": "armoire",
  "scale": {
    "value": 1.0,
    "container_size_m": [0.451, 1.27, 1.668]
  },
  "articulation": {
    "status_actions": {
      "status1": {"joint": "link_2_door", "action": "The left door opens by rotating (
     revolute)", "closed": 0.0, "open": 1.571},
      "status2": {"joint": "link_3_door", "action": "The right door opens by rotating (
     revolute)", "closed": 0.0, "open": 1.571},
      "status3": {"joint": "link_4_drawer", "action": "The right drawer is pulled out (
     prismatic)", "closed": 0.0, "open": 0.342},
      "status4": {"joint": "link_5_drawer", "action": "The left drawer is pulled out (
     prismatic)", "closed": 0.0, "open": 0.379}
    },
    "joints": {
      "link_2_door": {"type": "revolute", "parent": "link_1_base", "axis": [0.0, -0.0,
     -1.0], "range": [0.0, 1.571]},
      "link_3_door": {"type": "revolute", "parent": "link_1_base", "axis": [0.0, 0.0,
     1.0], "range": [0.0, 1.571]},
      "link_4_drawer": {"type": "prismatic", "parent": "link_1_base", "axis": [0.0, -1.0,
     0.0], "range": [0.0, 0.342]},
      "link_5_drawer": {"type": "prismatic", "parent": "link_1_base", "axis": [0.0, -1.0,
     0.0], "range": [0.0, 0.379]}
    },
    "support_plane_access": {
      "shelf_2_left": [["status1"]],
      "shelf_3_left": [["status1"]],
      "shelf_4_left": [["status1"]],
      "shelf_1": [["status1"], ["status2"]],
      "shelf_2_right": [["status2"]],
```

```
      "shelf_3_right": [["status2"]],
      "shelf_4_right": [["status2"]],
      "drawer_1": [["status4"]],
      "drawer_2": [["status3"]]
    },
  },
  "support_planes": [
    {
      "id": "shelf_1",
      "name": "top shelf",
      "access_requirement": "Open the right door.",
      "dims_cm": [93, 27, 27],
      "max_weight": 10,
      "bbox3d_urdf": {"xmin": -0.467, "ymin": -0.109, "zmin": 0.275, "xmax": 0.467, "ymax"
     : 0.158, "zmax": 0.542},
      "moves_with": null,
      "access_joint_paths": [["link_2_door"], ["link_3_door"]]
    },
    {
      "id": "shelf_2_left",
      "name": "second left shelf",
      "access_requirement": "Open the left door.",
      "dims_cm": [37, 27, 17],
      "max_weight": 10,
      "bbox3d_urdf": {"xmin": -0.442, "ymin": -0.112, "zmin": 0.077, "xmax": -0.074, "ymax
     ": 0.155, "zmax": 0.244},
      "moves_with": null,
      "access_joint_paths": [["link_2_door"]]
    },
    {
      "id": "shelf_3_left",
      "name": "third left shelf",
      "access_requirement": "Open the left door.",
      "dims_cm": [40, 23, 17],
      "max_weight": 10,
      "bbox3d_urdf": {"xmin": -0.451, "ymin": -0.092, "zmin": -0.121, "xmax": -0.05, "ymax
     ": 0.141, "zmax": 0.046},
      "moves_with": null,
      "access_joint_paths": [["link_2_door"]]
    },
    {
      "id": "shelf_4_left",
      "name": "bottom left shelf",
      "access_requirement": "Open the left door.",
      "dims_cm": [40, 27, 17],
      "max_weight": 10,
      "bbox3d_urdf": {"xmin": -0.454, "ymin": -0.121, "zmin": -0.319, "xmax": -0.054, "
     ymax": 0.146, "zmax": -0.152},
      "moves_with": null,
      "access_joint_paths": [["link_2_door"]]
    },
    {
      "id": "shelf_2_right",
      "name": "second right shelf",
      "access_requirement": "Open the right door.",
      "dims_cm": [40, 27, 17],
      "max_weight": 10,
      "bbox3d_urdf": {"xmin": 0.065, "ymin": -0.123, "zmin": 0.077, "xmax": 0.465, "ymax":
     0.144, "zmax": 0.244},
      "moves_with": null,
      "access_joint_paths": [["link_3_door"]]
    },
    {
      "id": "shelf_3_right",
      "name": "third right shelf",
      "access_requirement": "Open the right door.",
      "dims_cm": [40, 27, 17],
      "max_weight": 10,
      "bbox3d_urdf": {"xmin": 0.061, "ymin": -0.109, "zmin": -0.121, "xmax": 0.462, "ymax"
     : 0.158, "zmax": 0.046},
      "moves_with": null,
      "access_joint_paths": [["link_3_door"]]
    },
    {
      "id": "shelf_4_right",
      "name": "bottom right shelf",
      "access_requirement": "Open the right door.",
      "dims_cm": [40, 27, 17],
      "max_weight": 10,
```

```
      "bbox3d_urdf": {"xmin": 0.06, "ymin": -0.101, "zmin": -0.319, "xmax": 0.46, "ymax":
     0.165, "zmax": -0.152},
      "moves_with": null,
      "access_joint_paths": [["link_3_door"]]
    },
    {
      "id": "drawer_1",
      "name": "drawer 1",
      "access_requirement": "Pull out the left drawer.",
      "dims_cm": [38, 35, 14],
      "max_weight": 5,
      "bbox3d_urdf": {"xmin": -0.431, "ymin": -0.16, "zmin": -0.715, "xmax": -0.054, "ymax
     ": 0.191, "zmax": -0.578},
      "moves_with": "link_5_drawer",
      "access_joint_paths": [["link_5_drawer"]]
    },
    {
      "id": "drawer_2",
      "name": "drawer 2",
      "access_requirement": "Pull out the right drawer.",
      "dims_cm": [38, 35, 14],
      "max_weight": 5,
      "bbox3d_urdf": {"xmin": 0.055, "ymin": -0.16, "zmin": -0.715, "xmax": 0.432, "ymax":
     0.191, "zmax": -0.578},
      "moves_with": "link_4_drawer",
      "access_joint_paths": [["link_4_drawer"]]
    }
  ]
}
```


I.2 An example of prompt of container

Container prompt (text input to the generator)

<image> is a large, white armoire with a pair of doors and a pair of drawers, with the drawers below the doors, about 45 × 127 × 167 cm (width × depth × height).

It has 4 movable parts:

1. The left door swings open forward and to the left (hinged along the left edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
2. The right door swings open forward and to the right (hinged along the right edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
3. The right drawer slides forward (slides on a linear rail);
4. The left drawer slides forward (slides on a linear rail).

I.3 An example of item placement (medium level)

A medium-level scene for the same container: two item categories with two instances each, placed without stacking on the third right shelf, which is reached by opening the right door. Positions are item bounding-box centres in the container frame (z-up, metres) after gravity settling; the scene record is abridged.

Placement record of a two-category scene (JSON)


```
{
  "scene_id": "80110/v04",
  "level": "middle",
  "container": {
    "id": "80110",
    "category": "armoire",
    "support_region": "shelf_3_right",
    "region_name": "third right shelf",
    "access_joint_paths": [["link_3_door"]],
    "free_space_box_m": {"xmin": 0.061, "xmax": 0.462, "ymin": -0.109, "ymax": 0.158, "
     zmin": 1.271, "zmax": 1.437},
    "free_space_dims_cm": [40, 27, 17]
  },
```

```
  "content_class": "non-food",
  "layout": "scatter",
  "type_counts": {
    "scissors": 2,
    "pitcher": 2
  },
  "items": [
    {
      "index": 0,
      "asset_id": "10960",
      "source": "partnet",
      "category": "scissors",
      "scale": 1.0,
      "up_axis": "y_up",
      "dims_cm": [21.1, 7.4, 0.9],
      "position_m": [0.198, 0.018, 1.276],
      "rotation_xyzw": [0.0003, 0.0007, 0.0, 1.0],
      "support": "floor",
      "layer": 0
    },
    {
      "index": 1,
      "asset_id": "10960",
      "source": "partnet",
      "category": "scissors",
      "scale": 1.0,
      "up_axis": "y_up",
      "dims_cm": [21.1, 7.4, 0.9],
      "position_m": [0.223, 0.117, 1.276],
      "rotation_xyzw": [0.0003, 0.0007, 0.0, 1.0],
      "support": "floor",
      "layer": 0
    },
    {
      "index": 2,
      "asset_id": "4e8dec7f7e84...",
      "source": "objaverse",
      "category": "pitcher",
      "scale": 1.0,
      "up_axis": "y_up",
      "dims_cm": [9.8, 9.8, 15.7],
      "position_m": [0.377, -0.022, 1.35],
      "rotation_xyzw": [0.0, 0.0, 0.0, 1.0],
      "support": "floor",
      "layer": 0
    },
    {
      "index": 3,
      "asset_id": "4e8dec7f7e84...",
      "source": "objaverse",
      "category": "pitcher",
      "scale": 1.0,
      "up_axis": "y_up",
      "dims_cm": [9.8, 9.8, 15.7],
      "position_m": [0.41, 0.101, 1.35],
      "rotation_xyzw": [0.0, 0.0, 0.0, 1.0],
      "support": "floor",
      "layer": 0
    }
  ],
  "physics": {
    "backend": "pybullet",
    "passed": true,
    "max_displacement_cm": 0.91,
    "n_toppled": 0,
    "all_inside_after_settle": true
  },
  "language": {
    "per_item": [
      "the scissors is in the center of the interior, mid-depth and horizontally centered,
      resting on the floor",
      "the scissors is in the back-center of the interior, near the back and horizontally
     centered, resting on the floor",
      "the pitcher is in the front-right corner of the interior, near the front and
     against the right wall, resting on the floor",
      "the pitcher is in the back-right corner of the interior, near the back and against
     the right wall, resting on the floor"
    ],
    "relations": ["the scissors are to the left of the pitchers"],
```

```
    "summary": "4 items (2 scissors, 2 pitchers) arranged in a scatter layout on the
     interior floor"
  }
}
```

I.4 An example of the integrated prompt with 3D assets

When 3D meshes are available for both the container and the items, the integrated prompt combines the container description, the target support region and its access condition, and the items to be placed, each referenced by its mesh. Every scene comes with two versions of this prompt that differ only in how positions are stated. The *precise* version gives metric distances: the first item is located by its distances to two walls of the support region, and each later item by its offset from the previous one. The *vague* version uses only qualitative directions such as "the back-right corner". The two boxes below show both versions for a medium-level scene of the armoire in Appendix I.2; the position phrases, the only part that differs, are highlighted. Paths in the `[objects]` block are relative to the prompt file.

Integrated placement prompt with 3D assets: precise version

<container> is a large, white armoire with a pair of doors and a pair of drawers, with the drawers below the doors, about 45 × 127 × 167 cm (width × depth × height).

It has 4 movable parts:

1. The left door swings open forward and to the left (hinged along the left edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
2. The right door swings open forward and to the right (hinged along the right edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
3. The right drawer slides forward (slides on a linear rail);
4. The left drawer slides forward (slides on a linear rail).

<item1> is a mouse, about 7 × 13 × 5 cm. It is placed on the bottom right shelf, behind the right door: **its right edge 2.3 cm from the right wall and its back edge 2.1 cm from the back wall**, resting on the floor, standing upright, in the same orientation as its model file, its front (model +Z) facing the front (the opening).

<item2 > is an award, about 3 × 2 × 9 cm. It is placed on the bottom right shelf, behind the right door: **its center 29.1 cm to the left of and 8.1 cm in front of the center of the mouse**, resting on the floor, standing upright, turned 180° from its model file, so its front (model +Z) faces the back wall.

Integrated placement prompt with 3D assets: vague version

<container> is a large, white armoire with a pair of doors and a pair of drawers, with the drawers below the doors, about 45 × 127 × 167 cm (width × depth × height).

It has 4 movable parts:

1. The left door swings open forward and to the left (hinged along the left edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
2. The right door swings open forward and to the right (hinged along the right edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
3. The right drawer slides forward (slides on a linear rail);
4. The left drawer slides forward (slides on a linear rail).

<item1> is a mouse, about 7 × 13 × 5 cm. It is placed on the bottom right shelf, behind the right door: **in the back-right corner of the interior, near the back and against the right wall**, resting on the floor, standing upright, in the same orientation as its model file, its front (model +Z) facing the front (the opening).

<item2> is an award, about 3 × 2 × 9 cm. It is placed on the bottom right shelf, behind the right door: **on the left side of the interior, mid-depth and against the left wall**, resting on the floor, standing upright, turned 180° from its model file, so its front (model +Z) faces the back wall.

I.5 An example of the integrated prompt with 2D images

When only 2D images are available for the container and the items, the integrated prompt refers to each of them through its image and conveys the same information in text. The boxes below show the same medium-level scene as Appendix I.4, again in a precise and a vague version. Compared with the 3D-asset prompt, the container is referred to by its closed-state rendering and each item by a reference image. The orientation clauses relative to the model file (e.g. "turned 180° from its model file") are omitted, since an image has no model frame. The position phrases, highlighted, are identical to those of the 3D-asset version.

Integrated placement prompt with 2D images: precise version

<image1> is a large, white armoire with a pair of doors and a pair of drawers, with the drawers below the doors, about 45 × 127 × 167 cm (width × depth × height).

It has 4 movable parts:

1. The left door swings open forward and to the left (hinged along the left edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
2. The right door swings open forward and to the right (hinged along the right edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
3. The right drawer slides forward (slides on a linear rail);
4. The left drawer slides forward (slides on a linear rail).

<image2> is a mouse, about 7 × 13 × 5 cm. It is placed on the bottom right shelf, behind the right door: **its right edge 2.3 cm from the right wall and its back edge 2.1 cm from the back wall**, resting on the floor.

<image3> is an award, about 3 × 2 × 9 cm. It is placed on the bottom right shelf, behind the right door: **its center 29.1 cm to the left of and 8.1 cm in front of the center of the mouse**, resting on the floor.

Integrated placement prompt with 2D images: vague version

<image1> is a large, white armoire with a pair of doors and a pair of drawers, with the drawers below the doors, about 45 × 127 × 167 cm (width × depth × height).

It has 4 movable parts:

1. The left door swings open forward and to the left (hinged along the left edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
2. The right door swings open forward and to the right (hinged along the right edge); inside it, left door and right door share the same interior space, which is evenly divided into four levels;
3. The right drawer slides forward (slides on a linear rail);
4. The left drawer slides forward (slides on a linear rail).

<image2> is a mouse, about 7 × 13 × 5 cm. It is placed on the bottom right shelf, behind the right door: **in the back-right corner of the interior, near the back and against the right wall**, resting on the floor.

<image3> is an award, about 3 × 2 × 9 cm. It is placed on the bottom right shelf, behind the right door: **on the left side of the interior, mid-depth and against the left wall**, resting on the floor.

J Robot Manipulation in Closed Containers

Figure 14 shows four simulated retrieval episodes built from InsideArt scenes. Each container is loaded from its URDF in the closed state and placed in the scene frame; the objects keep their annotated placements. A mobile manipulator opens the parts on the annotated access path of the target free space, grasps the target object, lifts it out, and places it on a side table. Simulation uses MuJoCo; grasps attach only after both fingers are in contact. From top to bottom, the four tasks are: (1) a UFactory xArm7 retrieving a knife from a sideboard drawer; (2) a UR5e with a Robotiq 2F-85 retrieving a fork from a lidded box; (3) a Franka Panda with a Shadow DEX-EE hand retrieving a

wallet from a TV-stand drawer; and (4) a Kinova Gen3 with a Robotiq 2F-85 retrieving an egg from a refrigerator bin.

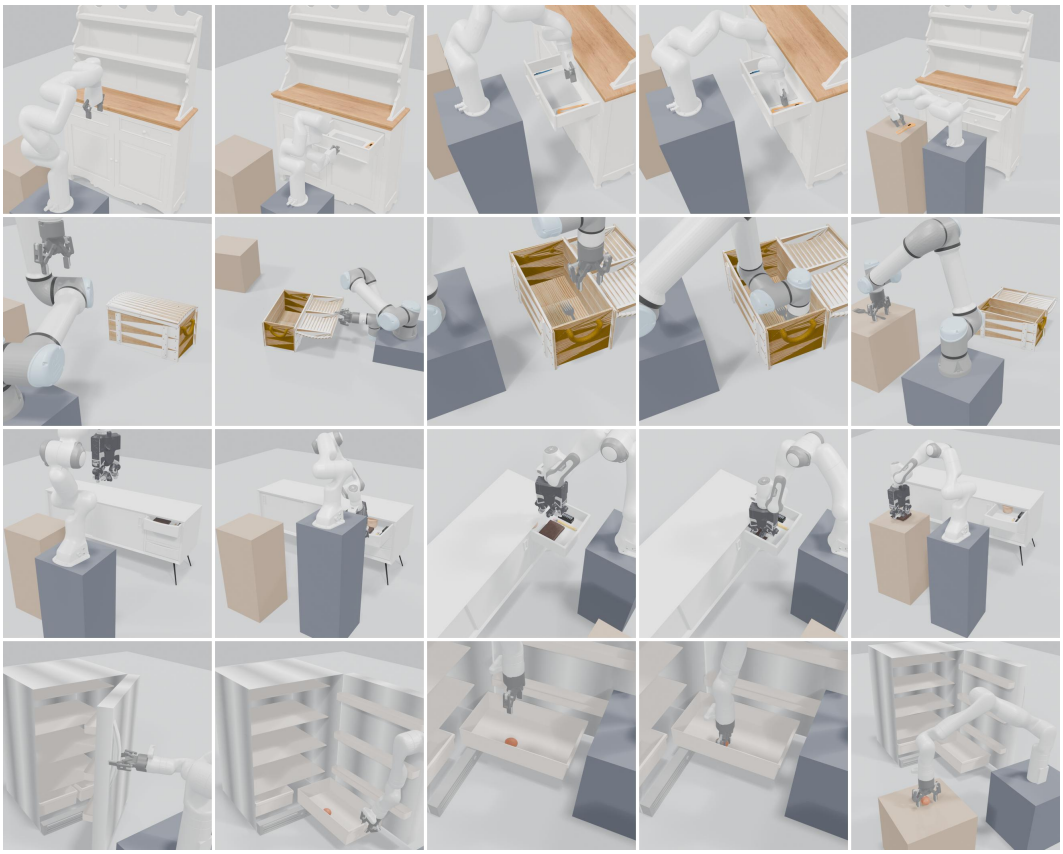


Figure 14: **Retrieving objects from closed articulated containers.**