TOWARDS PHYSICALLY GROUNDED ARTICULATED 3D ASSETS: A MULTI-AGENT PIPELINE AND BENCHMARK

Anonymous authors
Paper under double-blind review

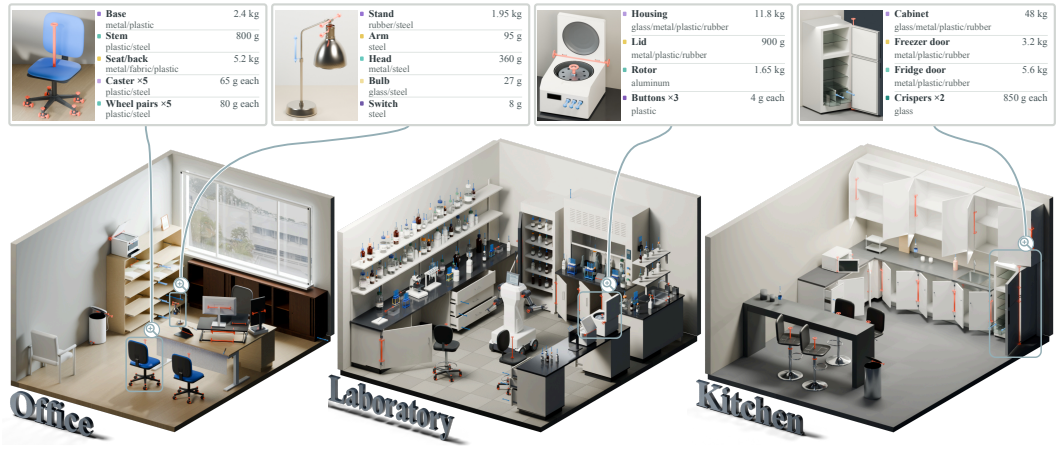


Figure 1: **Physically grounded articulated 3D objects generated by Artisan**. Here, the generated assets across diverse categories are assembled into office, laboratory, and kitchen scenes. Orange and blue annotations indicate revolute and prismatic axes, respectively. Zoom in for more details.

ABSTRACT

Recent advances in 3D generation have significantly improved aesthetic quality, but they do not ensure that objects behave correctly under physical interaction. In this paper, we study *physically grounded articulated 3D asset generation*—moving the goal from reproducing how objects look to capturing how they move and function. Inspired by the literature on procedural and agentic generation, we present ARTISAN, a system that extends the latest method Articraft by introducing a *multi-agent* pipeline with *physical simulation feedback*, enabling comprehensive attributes beyond geometry, appearance, and physically plausible articulation, such as collision-free motion, stability under gravity, and accurate mass and material properties. Furthermore, we construct a unified benchmark spanning four key aspects: spatial fidelity, physical plausibility, downstream task performance, and overall quality. The benchmark encompasses 100 reference assets from ArtVIP and LightWheel, 50 stability prompts, and 50 ABO objects with reference masses, with protocols evaluating various metrics. Through extensive evaluation of more than 10 representative models, ARTISAN achieves competitive spatial fidelity while attaining the lowest collision rates and highest stability rates among the evaluated methods. Together, our system and benchmark establish insights for advancing 3D generation towards physically plausible, functionally usable assets.

1 INTRODUCTION

We consider the problem of generating physically grounded articulated 3D assets from natural language and, optionally, reference images. Specifically, we aim to reconstruct the objects' appearance and geometry, as well as structural, functional, and physical properties such as part hierarchies, joints, and mass. The intent is to capture the configuration space of the object and to support simulation of

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.

contact and gravity. By solving this problem, we can reduce the cost of creating assets for physics simulation (Genesis Authors, 2024; Genesis AI Team, 2026), product design (Willis et al., 2021), and robotics (Li et al., 2023; Nasiriany et al., 2026).

Appearance, shape, articulation, and physical attributes are interdependent and should be modeled *holistically*. However, most existing methods address these properties separately, focusing on shape and appearance (Xiang et al., 2025; Hunyuan3D et al., 2025; Wu et al., 2025c), articulation (Wu et al., 2026; Li et al., 2026a;b), or specific physical properties, such as materials (Wu et al., 2025b; Youwang et al., 2026), mass (Zhai et al., 2024; Cao et al., 2026a;b; Li et al., 2026c), stability (Chen et al., 2024b; Li et al., 2025), and collision-free motion (Luo et al., 2025; Kreber & Stueckler, 2025). Addressing each aspect in isolation makes it difficult to satisfy all of them, resulting in artifacts and defects in the reconstructions and ultimately limiting the usability of the generated assets.

As alternatives to these learned models, procedural generators (Lian et al., 2025; Joshi et al., 2025) assemble assets programmatically, based on predefined rules that enforce desirable constraints and make at least some physical properties explicit and controllable. However, such rule-based systems limit the types of objects that can be generated and are difficult to use in tasks like image-based reconstruction. This limitation is addressed by several recently proposed agentic methods. For instance, Articraft (Zhou et al., 2026) uses a Large Language Model (LLM) to write, validate, and revise programs that construct 3D objects, which is much more flexible than using canned rules of traditional procedural generators. However, Articraft and other similar approaches use a single agent to both construct and verify the '3D program', which can lead to *self-evaluation bias*, where the agent may satisfy its own checks without genuinely ensuring the quality of the generated assets. Moreover, while their checks are meant to increase the likelihood that the generated assets would work in a physics-simulated environment, the latter is not directly tested and may still fail.

In this paper, we address these gaps by introducing ARTISAN, a new pipeline that improves on the blueprint of Articraft (Zhou et al., 2026) by switching to a *multi-agent* design and introducing *physical simulator feedback*. The new pipeline (Figure 2) consists of: **(i)** the *Design Agent*, which produces a blueprint of the object specifying its parts, joints, and articulations; **(ii)** the *Coding Agent*, which translates the blueprint into a program that outputs the geometry, articulations, and physical parameters of the object; and **(iii)** the *Review Agent*, which validates the built asset and provides feedback to the Coding Agent to revise the program. The agents work *independently*, exchanging only the blueprint, 3D asset, and feedback as explained. Their independence reduces the self-evaluation bias mentioned above. Furthermore, the validation step is augmented with feedback obtained from a physical simulator, which verifies physical properties like collisions and stability within the target simulator environment.

Existing benchmarks offer limited assessment of whether visual fidelity translates into physical plausibility and even functional usability to evaluate the assets produced by such pipelines. We therefore introduce a benchmark that jointly assesses all of these properties. It evaluates: *spatial fidelity*, comparing the object's appearance, geometry, and kinematics to the reference inputs; *physical attributes*, measuring mass, material, stability under gravity, and collision under motion; *downstream tasks*, measuring the usability of the generated assets in tasks like robotic manipulation; and *overall quality*, based on human judgments after manipulating objects in the simulator. Each aspect is evaluated with multiple metrics.

Using the new benchmark, we conducted extensive experiments to compare our method with state-of-the-art baselines. Empirically, we find that the assets constructed by Artisan: **(i)** are faithful to the input references; **(ii)** are physically valid and articulate correctly; and **(iii)** work well in downstream tasks such as simulated robotic manipulation. We also show that Artisan outperforms the recent and strong baselines in most metrics and in human evaluation, especially improving physical properties like collision and stability. In most cases, our general-purpose pipeline is more accurate than task-specific feed-forward 3D reconstruction models. We also show that the constructed objects are functional and support downstream robotic manipulation tasks.

To summarize, our key contributions are: **(i)** a new multi-agent 3D coding system, ARTISAN, that combines asset construction with physical simulator feedback and an independent design-coding-review workflow; **(ii)** a benchmark along with diverse datasets covering comprehensive evaluation; and **(iii)** through comprehensive evaluation, we show gaps and reveal key insights and challenges in physical asset generation, providing a foundation for future research in this area.

2 Related Work

Creating Functional 3D Assets. Early methods build articulated objects through per-instance optimization (Mu et al., 2021; Wu et al., 2022; Tseng et al., 2022; Liu et al., 2023; Weng et al., 2024; Wu et al., 2025a), but are limited to specific instances. To improve generalizability, learning-based methods create articulated objects from pure noise (Lei et al., 2023; Gao et al., 2025), text (Liu et al., 2024; Su et al., 2025), images (Liu et al., 2025; Jiang et al., 2022; Chen et al., 2024a; He et al., 2026; Chen et al., 2025; Wu et al., 2026), or 3D models (Li et al., 2026a; Zhang et al., 2026; Li et al., 2026b), learning from large-scale datasets (Xiang et al., 2020; Li et al., 2026b). Recent methods focus on inferring independent physical properties from visual representations (Zhai et al., 2024; Shuai et al., 2025), jointly predicting these alongside joint motion characteristics (Cao et al., 2026a;b; Li et al., 2026c; Huang et al., 2026), and leveraging simulation feedback to refine aspects such as support, clearances, or mobility (Chen et al., 2024b; Li et al., 2025; Luo et al., 2025; Kreber & Stueckler, 2025; Xu et al., 2026). In contrast, ARTISAN generates articulated objects that take diverse physical properties into account.

Procedural and Agentic 3D Asset Creation. Procedural systems construct editable and simulation-ready variations within the authored rules (Lian et al., 2025; Joshi et al., 2025). The latest agent-based pipelines extend this coverage through executable articulation, retrieval with visual critique, or semantic search over kinematic structures (Mandi et al., 2025; Le et al., 2025; Qiu et al., 2025; Wang et al., 2025). Closer to our approach, Articraft (Zhou et al., 2026) builds articulated objects through an agent with Software Development Kit (SDK)s. Recent systems improve reliability through geometric and visual checks, and revisions to earlier construction stages when checks fail (Wang et al., 2026; Shui et al., 2026; Lin et al., 2026). In contrast, ARTISAN introduces a multi-agent pipeline with physical simulator feedback, enabling physically grounded articulated 3D asset generation.

Datasets and Benchmarks for Physically Grounded 3D Assets. Articulated datasets provide part geometry and motion annotations (Xiang et al., 2020; Geng et al., 2023; Liu et al., 2022; Jin et al., 2026). Program-generated collections further expand the available assets (Zhou et al., 2026; Cao et al., 2026b; Li et al., 2026c). Existing benchmarks separately assess geometric and semantic fidelity (Li et al., 2026a; Duggal et al., 2025), programmatic modelling quality (Yang et al., 2026; Gao et al., 2026), material properties (Cao et al., 2026b; Li et al., 2026c), and motion behavior (Luo et al., 2025; Pham, 2026). Our benchmark comprehensively evaluates the spatial fidelity, physical plausibility, and downstream task utility of generated articulated assets using a set of distinct metrics.

3 Method: A Multi-Agentic Pipeline with Physical Validation

Our goal is to construct a system $\mathcal{P}$ that, given as input a natural language prompt $\boldsymbol{p}$ and, optionally, one or more reference images $\mathcal{I}$, generates as output an articulated 3D asset $\mathcal{O}$:

$$\mathcal{P} : (\boldsymbol{p}, \mathcal{I}) \mapsto \mathcal{O} = \{\{\mathcal{G}_i, \rho_i, \mathcal{M}_i\}_{i=1}^{P}, (\mathcal{K}, \mathcal{J})\}. \tag{1}$$

The object $\mathcal{O}$ consists of P rigid parts defined by their geometry $\mathcal{G}_i$, mass ρ_i, and material $\mathcal{M}_i = \{\boldsymbol{c}_i, m_i, r_i, \alpha_i\}$ specified by its base color $\boldsymbol{c}_i$, metalness m_i, roughness r_i and opacity α_i (i.e., standard Physically Based Rendering (PBR) parameters).[1] $\mathcal{K}$ specifies the *kinematic structure*, defining the parent-child relationships among the P parts, while $\mathcal{J}$ specifies the joint types, axes, motion limits, and any coupling constraints between dependent joints. The pipeline aims to output an object $\mathcal{O}$ that faithfully reproduces the inputs $\boldsymbol{p}$ and $\mathcal{I}$ and is physically sound: parts should not collide, freestanding objects should remain stable under gravity, and it must be possible to grasp and manipulate it. The pipeline consists of three independent LLM agents equipped with tools and execution harnesses, $\mathcal{P} = \{\mathcal{A}_i\}_{i=1}^{3}$, described below, with implementation details given in appendix A.

3.1 Design Agent

The *Design Agent* $\mathcal{A}_{\text{design}}$ takes the input prompt and image, and outputs a structured textual blueprint $\mathcal{B}$ and a design board $\mathcal{D}$:

$$\mathcal{A}_{\text{design}}(\boldsymbol{p}, \mathcal{I}) \to (\mathcal{B}, \mathcal{D}), \text{where } \mathcal{B} = \left\{\{\tilde{\mathcal{G}}_i^{\text{text}}, \tilde{\mathcal{M}}_i^{\text{text}}\}_{i=1}^{P},\ (\tilde{\mathcal{K}}^{\text{text}}, \tilde{\mathcal{J}}^{\text{text}}),\ \mathcal{Q}^{\text{text}}\right\}. \tag{2}$$

[1] The SDK supports distinct PBR materials with optional textures for components within each part.

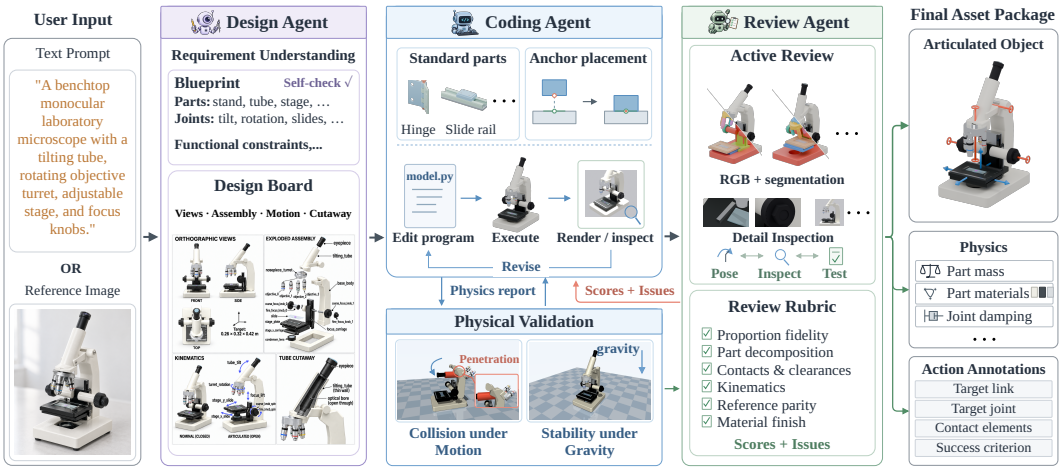


Figure 2: The staged pipeline of the three agents: Design Agent, Coding Agent, and Review Agent.

The superscript $*^{\text{text}}$ indicates that *these are all textual descriptions* rather than the final actual values. Text $\tilde{\mathcal{G}}_i^{\text{text}}$ and $\tilde{\mathcal{M}}_i^{\text{text}}$ describe the geometry and material of part i and $\tilde{\mathcal{K}}^{\text{text}}$ and $\tilde{\mathcal{J}}^{\text{text}}$ describe the kinematic structure and joints of the asset.

Beyond geometry, materials, and articulation considered in prior work (Zhou et al., 2026), $\mathcal{Q}^{\text{text}}$ describes novel *functional* properties. First, it specifies coupled motions, such as a bin lid opening when its pedal is pressed. It also defines interaction tasks in plain language, such as "*raise the mixing head and lift out the mixing bowl*". These functional constraints guide asset verification in Section 3.3 and the generation of downstream task annotations in Section 5.2, helping ensure that the asset is both physically plausible and functionally usable in downstream applications.

Blueprint Verification and Visual Reference. $\mathcal{A}_{\text{design}}$ checks the blueprint against the input requirements and corrects inconsistencies in parts, motions, and functions. An image-generation model (OpenAI, 2026b) converts the blueprint into a *design board* $\mathcal{D}$ (Figure 2). It lays out orthographic views of the articulated object, an exploded view of the parts, a kinematic diagram of the moving parts in their closed and open states, and cutaway sections of the parts whose interiors matter.

3.2 Coding Agent

The *Coding Agent* $\mathcal{A}_{\text{coding}}$ takes as input the user prompt and the optional reference images, the blueprint $\mathcal{B}$ and design board $\mathcal{D}$ from the Design Agent, the previous program Π_{t-1}, the corresponding 3D asset $\mathcal{O}_{t-1}$, the physics report $\mathcal{E}_{t-1}$, and the review scores $\mathcal{S}_{t-1}$ and list of issues $\mathcal{R}_{t-1}$ from the Review Agent, and produces the next program Π_t and asset $\mathcal{O}_t$:

$$\mathcal{A}_{\text{coding}}\Big(\underbrace{(\boldsymbol{p},\mathcal{I})}_{\text{Input}},\underbrace{(\mathcal{B},\mathcal{D})}_{\mathcal{A}_{\text{design}}},\underbrace{(\Pi_{t-1},\mathcal{O}_{t-1})}_{\mathcal{A}_{\text{coding}}},\mathcal{E}_{t-1},\underbrace{(\mathcal{S}_{t-1},\mathcal{R}_{t-1})}_{\mathcal{A}_{\text{review}}}\Big)\rightarrow(\Pi_t,\mathcal{O}_t), \tag{3}$$

Here, t indexes a coding–review iteration. The previous program Π_0, asset $\mathcal{O}_0$, physics report $\mathcal{E}_0$ (from the Physics Simulator in Section 3.4), and review feedback $\mathcal{S}_0$ and $\mathcal{R}_0$ (returned by the Review Agent $\mathcal{A}_{\text{review}}$, described in Section 3.3) are absent on the first iteration.

Building on Articraft's SDK and execution harness (Zhou et al., 2026), the agent iteratively *edits* the program Π_t (written in PYTHON), *executes* it to obtain a corresponding asset $\mathcal{O}_t$, and then *estimates* its quality for iterative improvement. We refer readers to Zhou et al. (2026) for more details. We extend the workflow for physical plausibility with three components: **(i)** *tools* for geometric and visual inspection (detailed in appendix A.2), **(ii)** reusable *standard connectors* for joint connections, and **(iii)** *physical simulation feedback* (Section 3.4) that guides repairs to physical attributes.

Standard Connectors for Joint Connections. Inspired by standard connectors in mechanical design (Poli, 2001), we extend the Articraft SDK with reusable, parameterized connectors, such as hinges, slide rails, and rotary connectors (Figure 5). $\mathcal{A}_{\text{coding}}$ only needs to select a connector

and adjust its dimensions and mounting positions, instead of designing a geometrically accurate connector from scratch. This reduces floating connections and unintended interpenetration, making the generated asset more collision-free as measured in Section 3.4.

Action Annotation. For each task τ described in natural language in $\mathcal{Q}^{\text{text}}$, the Coding Agent records the target rigid part ℓ_τ, the geometric components forming its grasp or press region V_τ, the joint to actuate j_τ, and the fraction of joint travel required for success σ_τ. These are written to a manifest beside the exported meshes so that a downstream pipeline needs no manual labelling.

3.3 Review Agent

The *Review Agent* $\mathcal{A}_{\text{review}}$ evaluates the asset against the intended design and physical constraints. It takes as input the user prompt $\boldsymbol{p}$ and optional images $\mathcal{I}$, the blueprint $\mathcal{B}$ and design board $\mathcal{D}$, the current program Π_t and asset $\mathcal{O}_t$, and the corresponding physics report $\mathcal{E}_t$ from Section 3.4 and outputs a set of scores $\mathcal{S}_t$, one for each evaluation criterion, as well as a structured list of issues $\mathcal{R}_t$:

$$\mathcal{A}_{\text{review}}\Big(\underbrace{(\boldsymbol{p},\mathcal{I})}_{\text{Input}},\underbrace{(\mathcal{B},\mathcal{D})}_{\mathcal{A}_{\text{design}}},\underbrace{(\Pi_t,\mathcal{O}_t)}_{\mathcal{A}_{\text{coding}}},\mathcal{E}_t\Big)\rightarrow(\mathcal{S}_t,\mathcal{R}_t). \tag{4}$$

The Review Agent critically reviews repairs against the blueprint and visual references to check that intended functions are preserved. For example, $\mathcal{A}_{\text{coding}}$ may shorten a door's travel to pass collision checks, while $\mathcal{A}_{\text{review}}$ flags the reduced opening range as inconsistent with the blueprint.

Each issue identifies the affected component, supporting evidence, severity, and what the Coding Agent needs to fix. The pipeline uses the scores and issue severity to request repairs when necessary; critical structural or functional failures cannot be offset by a high overall score. Note that the feedback $(\mathcal{S}_t,\mathcal{R}_t)$ guides the next coding iteration, but the Review Agent itself does not alter the asset program.

3.4 Physical Validation

We validate each candidate asset $\mathcal{O}_t$ through collision and stability checks in MuJoCo (Todorov et al., 2012). The resulting physics report $\mathcal{E}_t$ records colliding part pairs and penetration depths at the tested joint configurations, together with tilt measurements under gravity, to guide coding and review.

Collision under Motion. The collisions are determined based on whether different parts interpenetrate during the simulated motion sequences, across sampled configurations of single-joint, double-joint, and multi-joint motions (details defined in Section 4.1). Specifically, we first perform convex decomposition (Wei et al., 2022) to construct mesh collision geometry for simulation. Then, we simulate the motion of each joint in isolation and record any collisions between parts. Collisions that persist during individual joint operation require remediation. Collision findings across motions include the affected part pairs, poses, and penetration depths, enabling the Review Agent to assess whether these collisions occur during normal operation (Section 3.3).

Stability under Gravity. We simulate the assembly under gravity and measure its tilt angle relative to the initial pose to assess whether it remains upright. We run the MuJoCo simulation for up to 5 seconds under gravity. The 3D asset is considered unstable if it tilts beyond a certain threshold during the simulation. The tilt measurements are recorded in $\mathcal{E}_t$ for review against the intended use.

4 Benchmark

A key contribution of our paper is to build a comprehensive benchmark for physically grounded articulated 3D assets. To this end, we design a set of evaluation protocols and metrics, and curate benchmark data with abundant annotations for these metrics.

4.1 Evaluation Protocol and Metrics

Spatial Fidelity. The spatial fidelity metrics evaluate how well the 3D shape of the generated asset matches the reference asset in both the rest-pose and the articulated states. We first perform

point sampling and align the meshes using ICP registration (Besl & McKay, 1992), then establish correspondences between parts based on their centers using the Hungarian matching algorithm (Kuhn, 1955). Due to the difficulty of clearly distinguishing between states, outputs from different states may sometimes be confused. To avoid erroneous penalties, we test both state pairings and retain the one with the lower alignment error across the two states as the final aligned result.

Appearance: After aligning the generated 3D asset to the reference asset, we render both using Blender at 1024^2 resolution from three viewpoints per state. We report the Intersection over Union (IoU) of the rendered masks, pixel-level PSNR, patch-level SSIM (Wang et al., 2004), feature-level LPIPS (Zhang et al., 2018), and CLIP score (Radford et al., 2021) over all six images.

Rest-Pose and Articulated Geometry: We calculate 3D metrics in both states. We report the object Chamfer Distance (oCD) for the object as a whole. We also report the part-aware CD (pCD) and mean part IoU (pIoU) by averaging the metrics of all matched parts, following (Li et al., 2026a).

Physical Properties. We assess the fidelity of the reconstructed *materials* and *mass*, the absence of *collisions* during articulated motion, and *stability* under gravity.

Materials: Following PBR-NeRF (Wu et al., 2025b) and LMRM (Youwang et al., 2026), we evaluate the materials of the generated assets by rendering the colour, metallic, and roughness buffers from the same viewpoints used for appearance evaluation, and then calculating PSNR against corresponding renders of the reference asset. To mitigate the impact of geometric inconsistencies, we also develop a protocol to transfer predicted materials to the ground-truth geometry so that the material attributes can be rendered and compared in the same geometry (results in Appendix Table 11).

Mass: We compare the mass $\hat{m}$ of the object against the corresponding ground-truth mass m. Here, we report three scores following (Standley et al., 2017; Zhai et al., 2024): $\mathrm{ADE} = |\hat{m} - m|$ in kilograms, $\mathrm{APE} = |\hat{m} - m|/m$ and $\mathrm{MnRE} = \min(r, 1/r)$, where $r = \hat{m}/m$.

Collision: We evaluate the collision of the generated asset at rest and during articulated motion. We sample the motion in three levels: single-joint, double-joint, and multi-joint. Specifically, for single-joint motion, we uniformly sample 9 states ranging from the rest state to its (predicted) upper-limit state. For double-joint motion, we then combine two joints to form a 9×9 grid of motion states. To efficiently evaluate multi-joint motion, we randomly sample a state for each joint and perform a total of 200 trials. To avoid penalizing collisions from joint combinations that violate motion dependencies, we establish joint correspondences using Hungarian matching (Kuhn, 1955), reproduce each sampled configuration on the ground-truth asset, and exclude colliding part pairs that also contact in that asset. At each sampling level, we report the percentage of collisions over all sampled states.

Stability: Prior work (Guo et al., 2024; Li et al., 2025) evaluates stability under gravity primarily considering the whole object as a single rigid body, with uniform mass distribution. We extend this evaluation to a challenging benchmark of 50 articulated objects with high centers of mass. We simulate each object in MuJoCo (Todorov et al., 2012) for 10 seconds in both its rest and fully articulated states, and report the percentage of stable assets separately for each state. An object is considered stable if its maximum tilt angle relative to its initial pose remains below 30°, its positional change is below 1 mm, and its rotational change does not exceed 0.01 rad.

Robotic Task Performance. Beyond spatial fidelity and physical plausibility, we evaluate whether the generated assets are functionally usable and improve downstream task performance.

Success Rate: We evaluate whether training with generated articulated assets improves the performance of the vision-language-action policy $\pi_{0.5}$ (Black et al., 2025) on unseen instances. For each held-out asset, we perform policy rollouts to evaluate the performance of $\pi_{0.5}$, which is fine-tuned on robotic demonstrations generated with predefined interactions on articulated assets. We report the mean success rate for each task and the aggregate success rate across all tasks.

Asset Qualification: To further assess the simulator-readiness of the assets, we additionally report how many generated assets can be utilized for robotic tasks. The qualification consists of two stages: **(i)** *physical stability and task feasibility (Round 1)*, which assesses whether an asset remains in its intended initial configuration after being placed in the simulator and whether it can physically reach the benchmark-defined success state under valid interactions; **(ii)** *demonstration generation success (Round 2)*, which assesses whether the asset supports the generation of successful demonstrations using the benchmark's synthetic data generation pipeline within a predefined trial budget.

Table 1: **Spatial fidelity scores** on ArtVIP (Jin et al., 2026), reported in the rest and the open state. For simplicity, we only report the up-to-date results of Articraft and Codex, while the full results are available in Table 8. **Bold** marks the best value and underline marks the second-best value per metric.

Method	Appearance					Rest-Pose Geometry			Articulated Geometry		
	IoU↑	SSIM↑	PSNR↑	LPIPS↓	CLIP↑	oCD↓	pCD↓	pIoU↑	oCD↓	pCD↓	pIoU↑
Instruct-Particulate	0.6299	0.8585	17.18	0.2708	0.9166	0.0027	0.1137	0.2440	**0.0010**	0.1126	0.2730
PhysX-Anything	0.4547	0.8450	15.26	0.3889	0.8480	0.0097	0.3721	0.0822	0.0123	0.3735	0.0772
URDF-Anything+	0.5568	0.8604	16.39	0.3802	0.8323	0.0062	0.2913	0.1262	0.0068	0.3034	0.1236
Articulate-Anything	0.5785	0.8202	14.46	0.3902	0.8440	0.0087	0.1923	0.1223	0.0097	0.2076	0.1254
Articraft (GPT-5.5)	0.6744	0.8584	17.53	0.2784	0.9029	0.0026	0.0212	0.2791	0.0040	0.0236	0.2727
Articraft (GPT-6 Astra)	0.7085	0.8758	18.74	0.2343	0.9195	0.0022	0.0136	0.3233	0.0034	0.0229	0.3215
Codex (GPT-6 Astra)	0.6804	0.8668	17.99	0.2523	0.9168	0.0033	0.0199	0.3354	0.0039	0.0276	0.3126
Claude (Fable 5.1)	0.6751	0.8772	18.39	0.2608	0.9047	0.0015	0.0290	0.3233	0.0050	0.0432	0.2922
Agy (Gemini-Flash-3.8)	0.7031	0.8777	18.26	0.2548	0.9130	**0.0010**	0.0131	0.3555	0.0050	0.0236	0.3110
Ours (GPT-5.5 w/o phys.)	0.6783	0.8652	17.70	0.2745	0.9050	0.0021	0.0163	0.2975	0.0052	0.0255	0.2828
Ours (GPT-5.5)	0.6841	0.8597	17.61	0.2744	0.9059	0.0016	0.0167	0.2905	0.0030	0.0210	0.2870
Ours (GPT-5.6 Sol)	0.6781	0.8725	17.83	0.2679	0.9146	0.0018	0.0152	0.3006	0.0023	0.0212	0.3064
Ours (GPT-6 Astra)	**0.7243**	**0.8785**	**18.94**	**0.2264**	**0.9268**	0.0011	**0.0089**	**0.3634**	0.0018	**0.0138**	**0.3516**



Figure 3: **Qualitative comparison** of articulated assets generated by different models. Orange rods indicate revolute axes; blue arrows indicate prismatic directions.

Overall Quality. We also add a *human* evaluation to assess the overall quality. Participants are presented with the input text prompt and reference images, rendered images of the generated assets in rest and open states, and interactive *MuJoCo* scenes in which each asset can be operated to inspect how its parts move, with six methods' outputs anonymized and randomly ordered. Each participant selects and ranks the top three of six results according to overall quality, considering visual fidelity, articulation, and physical interaction (detailed in appendix B.4).

4.2 Datasets

We evaluate spatial fidelity on 50 objects, each derived from ArtVIP (Jin et al., 2026) and LightWheel (Lightwheel, 2025), using ArtVIP as the main benchmark. We use our system to reconstruct each of the assets from a text description and a reference image. For physical evaluation, we use ArtVIP for material and collision, 50 articulated objects from ABO-500 (Collins et al., 2022) for mass, and a curated set of 50 objects for stability. The downstream evaluation covers three tasks of RoboTwin 2.0 (Chen et al., 2026) and eight from RoboCasa365 (Nasiriany et al., 2026), with disjoint native training and test assets, generated assets that correspond to the training distribution, and 50 demonstrations per asset. More details of the dataset are provided in appendices B.3 and E.

4.3 Baselines

We compare against three groups of baselines: **(i)** *task-specific learned articulation models*, including the latest feed-forward method Instruct-Particulate (Li et al., 2026b), PhysX-Anything (Cao et al., 2026a) and URDF-Anything+ (Wu et al., 2026); **(ii)** *specialized agentic methods* for articulated asset construction, Articraft (Zhou et al., 2026) and Articulate-Anything (Le et al., 2025); and **(iii)** *general-purpose coding agents*, Codex (OpenAI, 2026a), Claude (Fable 5.1 high), and Antigravity (Agy) with Gemini-Flash-3.8. For Articraft, Codex, and ARTISAN, we report `GPT-5.5`, `GPT-5.6 Sol`, and `GPT-6 Astra` variants where available, as indicated in the tables.

Table 2: **Physical attribute scores** on ArtVIP (Jin et al., 2026) (Collision, Material), our own dataset (Stability), and ABO-500 (Collins et al., 2022) (Mass). The full results are available in Table 10. **Bold** marks the best value and underline marks the second-best value per metric. – indicates an unavailable result due to the problem scope of each method.

Method	Collision			Stability		Material			Mass		
	Single ↓	Double ↓	Multiple ↓	Rest ↑	Open ↑	Base color ↑	Metallic ↑	Roughness ↑	MnRE ↑	ADE (kg) ↓	APE ↓
NeRF2Physics	–	–	–	–	–	–	–	–	0.601	10.78	0.509
PUGS	–	–	–	–	–	–	–	–	0.412	44.50	3.016
Instruct-Particulate	70.4	90.1	98.7	68%	42%	11.48	**17.86**	10.94	–	–	–
PhysX-Anything	55.0	76.0	81.3	74%	22%	9.22	–	–	0.302	198.07	19.292
URDF-Anything+	84.6	99.2	100.0	64%	48%	–	–	–	–	–	–
Articulate-Anything	31.2	40.5	32.8	64%	44%	7.75	–	–	–	–	–
Articraft (GPT-5.5)	42.5	57.5	79.4	82%	78%	10.93	–	–	–	–	–
Articraft (GPT-6 Astra)	45.5	62.0	78.8	80%	80%	11.99	–	–	–	–	–
Codex (GPT-6 Astra)	32.4	48.1	63.7	86%	82%	10.29	–	–	0.763	6.63	0.303
Claude (Fable 5.1)	39.9	50.6	64.7	84%	86%	10.98	–	–	0.743	6.51	0.388
Agy (Gemini-Flash-3.8)	47.8	60.1	78.2	80%	78%	**12.23**	–	–	**0.774**	**5.49**	0.296
Ours (GPT-5.5 w/o phys.)	37.0	47.4	77.0	84%	82%	11.02	16.75	12.09	0.742	6.07	0.340
Ours (GPT-5.5)	3.5	6.6	12.3	96%	94%	12.11	16.85	12.27	0.720	7.22	0.385
Ours (GPT-5.6 Sol)	1.9	4.8	14.5	**98%**	**96%**	10.94	16.58	**14.10**	0.711	8.16	0.484
Ours (GPT-6 Astra)	**1.6**	**3.5**	**12.0**	**98%**	**96%**	12.05	16.99	12.16	0.769	6.07	**0.264**

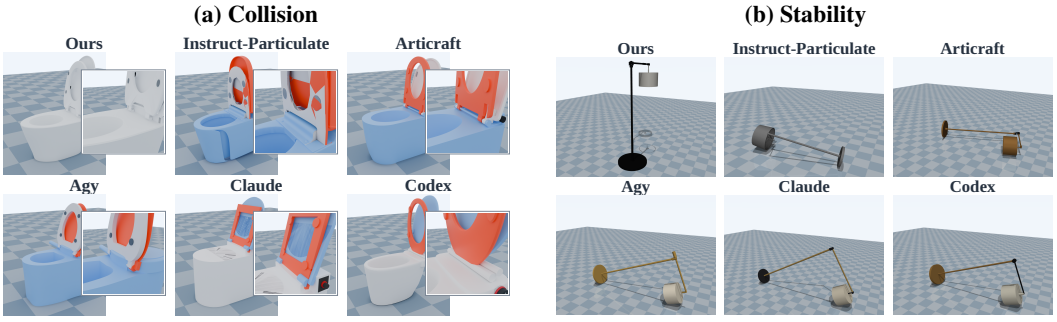


Figure 4: **Qualitative collision and stability comparisons.** (a) Visualization of Collision. Red and blue denote the major intersected part pair. ARTISAN is clean, whereas the other baselines involve collisions between components. (b) Visualization of stability under gravity. ARTISAN remains upright, whereas all five baselines topple after gravity is applied.

5 RESULTS

5.1 ANALYSIS OF BENCHMARK RESULTS

Main Results. We compare ARTISAN to the state-of-the-art baselines described in Section 4.3. We show spatial fidelity results in Table 1 and Figure 3 and physical attribute results in Table 2 and Figure 4. Without bells and whistles, ours (GPT-6 Astra) outperforms the other methods in 9 of 11 metrics for spatial fidelity in Table 1. More importantly, it dramatically improves physical plausibility, reducing collision occurrences to 1.6% on single-joint and 12.0% on multi-joint evaluations, and achieving 98% and 96% stability in the rest and articulated states, respectively. This highlights the significant advantage of our method in generating physically grounded articulated assets. In the following, we summarize several key findings. More details are provided in Appendix C.

Agentic generation methods can rival or even surpass state-of-the-art task-specific methods. Compared to Instruct-Particulate, whose input mesh is generated by Hunyuan-3.1 (Tencent Hunyuan3D Team, 2026) from the reference image, our method consistently achieves superior results in most metrics. Figure 3 illustrates the visual results in two additional reference-guided examples: a motorcycle and a miniature steam traction engine. Ours provides better appearance and articulated structure. This reveals the potential impact and versatility of agentic generation methods.

Physical validation is the key to generating a ready-to-use asset. As shown in Tables 1 and 2, we ablate ours (GPT-5.5) by disabling the physical validation (GPT-5.5 w/o phys.). Experiments demonstrate that, without physical validation, models can still achieve comparable performance in terms of appearance, geometry, and even kinematics. However, their physical behavior degrades drastically with a high rate of collision detection, indicating that such assets cannot execute their

Table 3: **Downstream robotic evaluation of articulated assets.** **(a)** Task success rate (%) of $\pi_{0.5}$ (Black et al., 2025) fine-tuned with different training-asset compositions, averaged over 3 RoboTwin 2.0 (Chen et al., 2026) tasks and 8 RoboCasa365 (Nasiriany et al., 2026) tasks. All policies are evaluated on held-out native assets that are not seen during fine-tuning. **(b)** Percentage of assets that pass the qualification test (Round 1) and support successful demonstration generation (Round 2).

(a) Task Success Rate (%)

Assets	RoboTwin	RoboCasa
Native only	80%	39%
Articraft only	60%	5%
ARTISAN only	63%	27%
Native + ARTISAN	**84%**	**43%**

(b) Asset Qualification Rate (%)

Assets	RoboTwin		RoboCasa	
	Round 1	Round 2	Round 1	Round 2
Native	100%	100%	98%	76%
Articraft	62%	62%	39%	8%
ARTISAN	100%	100%	95%	44%

intended articulated motion within the simulator. As shown in Figure 4, our model with physical validation is the only method that avoids component intersections and maintains stability within the simulator. Therefore, compared with existing agentic or feed-forward methods, our method brings the generated assets much closer to being truly ready to use.

Multi-agent design helps eliminate the self-evaluation bias. In Tables 1 and 2, a comparison between "Ours (GPT-5.5 w/o phys.)" and "Articraft (GPT-5.5)" shows average improvements of 6.7% in pCD, 5.2% in pIoU, and 11.2% in collision rates. This demonstrates the effectiveness of the multi-agent design. This further suggests that designing effective and efficient harnessing of agentic systems for physical 3D asset generation is useful, but it is a new and emerging area of physical 3D asset generation research.

5.2 ASSESSING ROBOTIC TASKS AND HUMAN JUDGEMENTS

Robotic Tasks. We report task success and asset qualification rates in Table 3. Based on the experimental results, we have the following key findings: (i) *Our assets are more suitable for robotic tasks:* When trained on robotic demonstrations generated with ARTISAN, the $\pi_{0.5}$ model achieves a higher success rate than that trained on Articraft demonstrations on both benchmarks (rows 2 & 3 in Table 3(a)); (ii) *Our assets can be a good extension to the current robotic training sets:* After incorporating our assets into the original training set, the newly trained model achieves a higher success rate on the same test set (rows 1 & 4 in Table 3(a)); (iii) *Our assets have better availability:* compared to Articraft assets, our assets achieve higher pass rates (37% higher on average in final pass rates) across two rounds of qualification reviews on two benchmarks (Table 3(b)).

Human Judgements. Details of the human evaluation are provided in Appendix B.4. The results show that ARTISAN achieves 43.3% of the Top-1 votes and 85.1% of the Top-3 votes, far surpassing the runner-up Codex (21.1% in Top-1 and 68.7% in Top-3). These results are based on user feedback following object manipulation in the MuJoCo simulation environment, further demonstrating that the assets produced by our method are closer to being simulation-ready.

6 CONCLUSION

We presented ARTISAN, a pipeline that achieves physically grounded, articulated 3D asset generation by combining multiple agents and physical simulation feedback. We also introduced a benchmark to evaluate various properties of the generated assets, including appearance, geometry, kinematics, materials, collision, mass, stability, and even robotic task performance. Empirically, we found that ARTISAN achieves the best overall performance across a wide range of evaluation metrics, and even outperforms existing task-specific models in most areas. We also found that multi-agent harnessing with physical simulation feedback significantly improves the spatial and physical fidelity of the generated assets.

Our findings motivate further research for our community: revisiting the role of feed-forward models, developing efficient agent harnesses across tasks, providing richer and more actionable simulator feedback, and extending simulators to capture a broader range of physical phenomena. We hope this work will help the community measure and advance progress toward physically grounded 3D worlds.

AI USE STATEMENT

We used generative artificial intelligence tools to assist in manuscript editing, including correcting grammar, refining sentence structure, and improving clarity and conciseness. The authors reviewed and revised the AI-assisted text and verified its consistency with the methods and experimental results. The authors assume full responsibility for the final manuscript, including its scientific claims and reported results.

ETHICS STATEMENT

Our human evaluation involves inspecting rendered assets and interacting with virtual objects in a physics simulator, as described in Appendix B.4. Participants assess visual quality, articulation, and physical behavior through simulated interactions. We report aggregate preference statistics, and the supplementary offline demo contains no real participant data. The generated assets are intended for simulation and evaluation; their use in real-world robotic applications should undergo additional validation appropriate to the deployment setting.

REPRODUCIBILITY STATEMENT

We describe the generation pipeline in Section 3 and the datasets and evaluation metrics in Section 4. Appendices A and B provide implementation details, dataset construction procedures, and evaluation protocols, including the human-study setup. Appendices C–F document complete results, component ablations, generation budgets and computational costs, robotic experiment settings, and representative agent traces. The supplementary material also includes an offline demonstration of the human-evaluation interface, allowing readers to inspect the rendered candidates and interact with the simulated assets.

REFERENCES

Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992. doi: 10.1109/34.121791. URL https://doi.org/10.1109/34.121791.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025. URL https://proceedings.mlr.press/v305/black25a.html.

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 (CVPR)*, pp. 5839–5848, 2026a.

Ziang Cao, Yinghao Liu, Haitian Li, Runmao Yao, Fangzhou Hong, Zhaoxi Chen, Liang Pan, and Ziwei Liu. PhysX-Omni: Unified simulation-ready physical 3D generation for rigid, deformable, and articulated objects. *arXiv preprint arXiv:2605.21572*, 2026b.

Honghua Chen, Yushi Lan, Yongwei Chen, and Xingang Pan. ArtiLatent: Realistic articulated 3D object generation via structured latents. In *SIGGRAPH Asia*, 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 *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024a. doi: 10.15607/RSS.2024.XX.124.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=itonej9GIV.

Yunuo Chen, Tianyi Xie, Zeshun Zong, Xuan Li, Feng Gao, Yin Yang, Ying Nian Wu, and Chenfanfu Jiang. Atlas3D: Physically constrained self-supporting text-to-3D for simulation and fabrication. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, pp. 102501–102530, 2024b. doi: 10.52202/079017-3255.

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

Shivam Duggal, Yushi Hu, Oscar Michel, Aniruddha Kembhavi, William T. Freeman, Noah A. Smith, Ranjay Krishna, Antonio Torralba, Ali Farhadi, and Wei-Chiu Ma. Eval3D: Interpretable and fine-grained evaluation for 3D generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025*, pp. 13326–13336. IEEE, 2025. doi: 10.1109/CVPR52734.2025.01244.

Daoyi Gao, Yawar Siddiqui, Lei Li, and Angela Dai. MeshArt: Generating articulated meshes with structure-guided transformers. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 618–627, 2025.

Yipeng Gao, Lei Shu, Genzhi Ye, Xi Xiong, Ameesh Makadia, Meiqi Guo, Laurent Itti, and Jindong Chen. 3DCodeBench: Benchmarking agentic procedural 3D modeling via code. *arXiv preprint arXiv:2606.01057*, 2026.

Genesis AI Team. The role of simulation in scalable robotics, Genesis World 1.0, and the path forward. Genesis AI Blog, May 2026. URL https://www.genesis.ai/blog.

Genesis Authors. Genesis: A generative and universal physics engine for robotics and beyond, December 2024. URL https://github.com/Genesis-Embodied-AI/genesis-world.

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 *CVPR*, pp. 7081–7091, June 2023.

Minghao Guo, Bohan Wang, Pingchuan Ma, Tianyuan Zhang, Crystal E. Owens, Chuang Gan, Joshua B. Tenenbaum, Kaiming He, and Wojciech Matusik. Physically compatible 3D object modeling from a single image. *NeurIPS*, 37:119260–119282, 2024.

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)*, pp. 7233–7243, 2026.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jie Huang, Xiaohe Li, Jiahao Li, Fangli Mou, Chen Qian, Yuqiang Fang, Junhao Fan, Kaixin Zhang, and Zide Fan. PhysX-CoT: Structured physical reasoning from a single image to simulation-ready 3D assets. *arXiv preprint arXiv:2608.08053*, 2026.

Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, Qingxiang Lin, Zeqiang Lai, Xianghui Yang, Huiwen Shi, Zibo Zhao, Bowen Zhang, Hongyu Yan, Lifu Wang, Sicong Liu, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Dongyuan Guo, Junlin Yu, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Shida Wei, Chao Zhang, Yonghao Tan, Yifu Sun, Lin Niu, Shirui

Huang, Bojian Zheng, Shu Liu, Shilin Chen, Xiang Yuan, Xiaofeng Yang, Kai Liu, Jianchen Zhu, Peng Chen, Tian Liu, Di Wang, Yuhong Liu, Linus, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3D 2.1: From images to high-fidelity 3D assets with production-ready PBR material. *arXiv*, 2506.15442, 2025.

Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *CVPR*, pp. 5616–5626, 2022.

Zhao Jin, Zhengping Che, Tao Li, Zhen Zhao, Kun Wu, Yuheng Zhang, Yinuo Zhao, Zehui Liu, Qiang Zhang, Xiaozhu Ju, Jing Tian, Yousong Xue, and Jian Tang. ArtVIP: Articulated digital assets of visual realism, modular interaction, and physical fidelity for robot learning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=SqPLEZ66BO.

Abhishek Joshi, Beining Han, Jack Nugent, Max Gonzalez Saez-Diez, Yiming Zuo, Jonathan Liu, Hongyu Wen, Stamatis Alexandropoulos, Karhan Kayan, Anna Calveri, Tao Sun, Gaowen Liu, Yi Shao, Alexander Raistrick, and Jia Deng. Procedural generation of articulated simulation-ready assets, 2025. URL https://arxiv.org/abs/2505.10755.

Jens U. Kreber and Joerg Stueckler. Guiding diffusion-based articulated object generation by partial point cloud alignment and physical plausibility constraints. In *IEEE/CVF International Conference on Computer Vision, ICCV 2025*, pp. 3206–3214. IEEE, 2025. doi: 10.1109/ICCV51701.2025.00307.

Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

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 *ICLR*, 2025.

Jiahui Lei, Congyue Deng, William B. Shen, Leonidas J. Guibas, and Kostas Daniilidis. NAP: Neural 3D articulated object prior. *NeurIPS*, 36:31878–31894, 2023.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. BEHAVIOR-1K: A benchmark for embodied AI with 1,000 everyday activities and realistic simulation. In *Conf. Robot Learn.*, pp. 80–93. PMLR, 2023.

Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. DSO: Aligning 3D generators with simulation feedback for physical soundness. In *ICCV*, 2025.

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

Ruining Li, Yuxin Yao, Matt Zhou, Chuanxia Zheng, Christian Rupprecht, Joan Lasenby, Shangzhe Wu, and Andrea Vedaldi. Instruct-Particulate: Scaling feed-forward 3D object articulation with kinematic control. *arXiv preprint arXiv:2606.14699*, 2026b.

Xian Li, Rong Wei, Lujie Yang, Haolin Huang, Junyuan Fang, Siliang Tang, Jun Xiao, Rui Tang, and Juncheng Li. UniPhysGen: Unified physical grounding for simulation-ready 3D assets. *arXiv preprint arXiv:2607.13586*, 2026c.

Xinyu Lian, Zichao Yu, Ruiming Liang, Yitong Wang, Li Ray Luo, Kaixu Chen, Yuanzhen Zhou, Qihong Tang, Xudong Xu, Zhaoyang Lyu, et al. Infinite Mobility: Scalable high-fidelity synthesis of articulated objects via procedural generation. *arXiv preprint arXiv:2503.13424*, 2025.

Lightwheel. SimReady: Simulation-ready 3D assets. https://simready.com/, 2025. Accessed: 2026.

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

Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. PARIS: Part-level reconstruction and motion analysis for articulated objects. In *ICCV*, pp. 352–363, 2023.

Jiayi Liu, Hou In Ivan Tam, Ali Mahdavi-Amiri, and Manolis Savva. CAGE: Controllable articulation generation. In *CVPR*, pp. 17880–17889, 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 *ICLR*, 2025.

Liu Liu, Wenqiang Xu, Haoyuan Fu, Sucheng Qian, Qiaojun Yu, Yang Han, and Cewu Lu. AKB-48: A real-world articulated object knowledge base. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 14789–14798. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01439.

Rundong Luo, Haoran Geng, Congyue Deng, Puhao Li, Zan Wang, Baoxiong Jia, Leonidas J. Guibas, and Siyuan Huang. PhysPart: Physically plausible part completion for interactable objects. In *IEEE International Conference on Robotics and Automation, ICRA 2025*, pp. 12386–12393. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127496.

Zhao Mandi, Yijia Weng, Dominik Bauer, and Shuran Song. Real2Code: Reconstruct articulated objects via code generation. In *ICLR*, 2025.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1820–1864. PMLR, 2023. URL https://proceedings.mlr.press/v229/mandlekar23a.html.

Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-SDF: Learning disentangled signed distance functions for articulated shape representation. In *ICCV*, pp. 12981–12991. IEEE, 2021.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *Proc. ICLR*, 2026.

OpenAI. Codex CLI. https://developers.openai.com/codex/cli, 2026a. [Online; accessed 06-May-2026].

OpenAI. GPT-Image-2. https://developers.openai.com/api/docs/models/gpt-image-2, 2026b. [Online; accessed 03-Sep-2026].

Anh-Quan Pham. Automatically improving simulation physics for articulated objects. *arXiv preprint arXiv:2605.19136*, 2026.

Corrado Poli. *Design for Manufacturing: A Structured Approach*. Butterworth-Heinemann, 2001. ISBN 9780750673419.

Xiaowen Qiu, Jincheng Yang, Yian Wang, Zhehuan Chen, Yufei Wang, Tsun-Hsuan Wang, Zhou Xian, and Chuang Gan. Articulate AnyMesh: Open-vocabulary 3D articulated objects modeling. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 4585–4606. PMLR, 2025. URL https://proceedings.mlr.press/v305/qiu25b.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL http://proceedings.mlr.press/v139/radford21a.html.

Yinghao Shuai, Ran Yu, Yuantao Chen, Zijian Jiang, Xiaowei Song, Nan Wang, Jv Zheng, Jianzhu Ma, Meng Yang, Zhicheng Wang, Wenbo Ding, and Hao Zhao. PUGS: zero-shot physical understanding with Gaussian splatting. In *IEEE International Conference on Robotics and Automation, ICRA 2025, Atlanta, GA, USA, May 19-23, 2025*, pp. 4478–4485. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11128683. URL https://doi.org/10.1109/ICRA55743.2025.11128683.

Yuan Shui, Yandong Guan, Zhanwei Zhang, Juncheng Hu, Jing Zhang, Dong Xu, and Qian Yu. ArtiCAD: Articulated CAD assembly design via multi-agent code generation. *arXiv preprint arXiv:2604.10992*, 2026.

Trevor Standley, Ozan Sener, Dawn Chen, and Silvio Savarese. image2mass: Estimating the mass of an object from its image. In *Conf. Robot Learn.*, pp. 324–333. PMLR, 2017.

Jiayi Su, Youhe Feng, Zheng Li, Jinhua Song, Yangfan He, Botao Ren, and Botian Xu. ArtFormer: Controllable generation of diverse 3D articulated objects. In *CVPR*, pp. 1894–1904, June 2025.

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Reed Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fábio Ramos, Nathan Ratliff, and Dieter Fox. CuRobo: Parallelized collision-free robot motion generation. In *IEEE International Conference on Robotics and Automation, ICRA 2023*, pp. 8112–8119. IEEE, 2023. doi: 10.1109/ICRA48891.2023.10160765.

Tencent Hunyuan3D Team. Hunyuan3D 3.1. https://3d.hunyuan.tencent.com/, 2026.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109. URL https://doi.org/10.1109/IROS.2012.6386109.

Wei-Cheng Tseng, Hung-Ju Liao, Lin Yen-Chen, and Min Sun. CLA-NeRF: Category-level articulated neural radiance field. In *IEEE Int. Conf. Robot. Autom.*, pp. 8454–8460. IEEE, 2022.

Jiawei Wang, Dingyou Wang, Jiaming Hu, Qixuan Zhang, Jingyi Yu, and Lan Xu. Kinematify: Open-vocabulary synthesis of high-dof articulated objects. *arXiv preprint arXiv:2511.01294*, 2025.

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

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3D meshes with collision-aware concavity and tree search. *ACM Trans. Graph.*, 41(4):42:1–42:18, 2022. doi: 10.1145/3528223.3530103. URL https://doi.org/10.1145/3528223.3530103.

Yijia Weng, Bowen Wen, Jonathan Tremblay, Valts Blukis, Dieter Fox, Leonidas Guibas, and Stan Birchfield. Neural implicit representation for building digital twins of unknown articulated objects. In *CVPR*, pp. 3141–3150, 2024.

Karl D. D. Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G. Lambourne, Armando Solar-Lezama, and Wojciech Matusik. Fusion 360 Gallery: A dataset and environment for programmatic CAD construction from human design sequences. *ACM Transactions on Graphics (TOG)*, 40(4): 1–24, 2021.

Di Wu, Liu Liu, Zhou Linli, Anran Huang, Liangtu Song, Qiaojun Yu, Qi Wu, and Cewu Lu. REArtGS: Reconstructing and generating articulated objects via 3D Gaussian splatting with geometric and motion constraints. In *NeurIPS*, 2025a.

Sean Wu, Shamik Basu, Tim Brödermann, Luc Van Gool, and Christos Sakaridis. PBR-NeRF: Inverse rendering with physics-based neural fields. In *CVPR*, pp. 10974–10984. IEEE, 2025b.

Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. D^2NeRF: Self-supervised decoupling of dynamic and static objects from a monocular video. *NeurIPS*, 35: 32653–32666, 2022.

Tianhao Wu, Chuanxia Zheng, Frank Guan, Andrea Vedaldi, and Tat-Jen Cham. Amodal3R: Amodal 3D reconstruction from occluded 2D images. In *ICCV*, 2025c.

Zhuangzhe Wu, Yue Xin, Chengkai Hou, Minghao Chen, Yaoxu Lyu, Jieyu Zhang, and Shanghang Zhang. URDF-Anything+: End-to-end generation for simulation-ready articulated assets. *CoRR*, abs/2603.14010, 2026. doi: 10.48550/ARXIV.2603.14010. URL https://doi.org/10.48550/arXiv.2603.14010.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. SAPIEN: A simulated part-based interactive environment. In *CVPR*, pp. 11097–11107, 2020.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D latents for scalable and versatile 3D generation. In *Proc. CVPR*, 2025.

Wenbo Xu, Liu Liu, Li Zhang, Dan Guo, and Ruonan Liu. MotionAnymesh: Physics-grounded articulation for simulation-ready digital twins. *arXiv preprint arXiv:2603.12936*, 2026.

Yikang Yang, Zhanpeng Hu, Youtian Lin, Mengqi Zhou, Jingxi Xu, Feihu Zhang, Jiaheng Liu, and Yao Yao. P3D-Bench: Benchmarking MLLMs for parametric 3D generation and structural reasoning. *arXiv preprint arXiv:2606.11152*, 2026.

Kim Youwang, Jon Hasselgren, Peter Kocsis, Andrea Weidlich, Tae-Hyun Oh, and Jacob Munkberg. Extracting neural materials from multi-view images. *arXiv preprint arXiv:2606.26715*, 2026.

Albert J. Zhai, Yuan Shen, Emily Y. Chen, Gloria X. Wang, Xinlei Wang, Sheng Wang, Kaiyu Guan, and Shenlong Wang. Physical property understanding from language-embedded feature fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 28296–28305. IEEE, 2024. doi: 10.1109/CVPR52733.2024.02673. URL https://doi.org/10.1109/CVPR52733.2024.02673.

Chuanrui Zhang, Minghan Qin, Yuang Wang, Baifeng Xie, Hang Li, and Ziwei Wang. SimArt: Decomposing monolithic meshes into sim-ready articulated assets via MLLM. In *ACM SIGGRAPH 2026 Conference Papers*, pp. 1–12. ACM, 2026. doi: 10.1145/3799902.3811173.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 586–595. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00068. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

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

APPENDIX CONTENTS

A Implementation Details **17**

A.1 SDK Extensions 17

A.2 Agent Tools 17

A.3 Physical Simulation 18

B Benchmark Details **19**

B.1 Spatial Fidelity Protocol 19

B.2 Material Transfer Protocol 19

B.3 Evaluation Datasets 19

B.4 Human Evaluation 22

C Additional Results **24**

C.1 Full Metric Tables 24

C.2 Ablation 26

C.3 Additional Qualitative Examples 27

D API and Time Cost **31**

E Robotic Experiments **31**

E.1 RoboTwin 2.0 32

E.2 RoboCasa365 33

F Agent Trace Examples **37**

F.1 Stand Mixer: Control Accessibility 37

F.2 Shoe Cabinet: Drawer Connections 38

A Implementation Details

A.1 SDK Extensions

Physical Property Abstraction. The SDK exposes three groups of physical attributes: part materials, part masses, and joint damping and friction. For materials, the Coding Agent selects a named material preset and can directly adjust its PBR values, including albedo, metallicity, roughness, and opacity. For mass, the agent reasons from the blueprint to estimate each part's mass, considering its dimensions, materials, and construction. We deliberately avoid computing mass as material density times geometric volume, since real objects often combine multiple materials and contain hollow regions, making this approximation prone to large errors. The SDK computes each part's default inertia from its geometry and estimated mass. The agent can also adjust each part's inertial properties to reflect its mass distribution. Joint damping and friction are configured according to the motions and mechanisms specified in the blueprint.

Standard Parts for Joint Connections. We provide six commonly used families of parameterized *standard parts*: hinges, slide rails, pinned joints, rotary connectors, telescoping slides, and threaded sleeves (Figure 5). Each connector is built from matching components with shared dimensions, alignment, and clearance parameters. For example, hinge halves share a pin axis, while a slide rail pairs a guide with a matching runner. The Coding Agent adapts their dimensions and mounting positions to the asset, reducing repeated geometric design and mismatches between mating components.

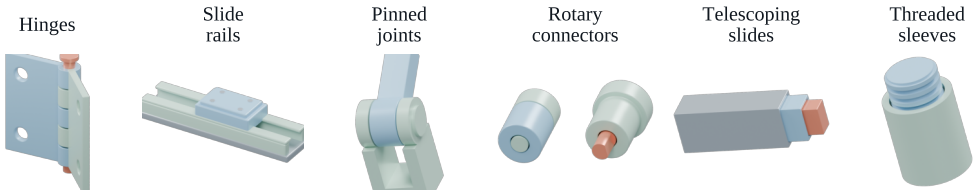


Figure 5: Illustrations of the six families of parameterized standard parts.

Anchor-based Placement. Authoring every pose in explicit coordinates requires the agent to calculate distances, orientations, and clearances, and to update them whenever geometry changes. We instead let the agent specify the intended spatial relationship and let the SDK solve the corresponding pose from the built geometry. An *anchor* names a geometric feature on an existing element: an oriented plane from a face, an axis from an edge or a round feature's center line, or a point from a corner. Given a subject anchor a_{sub}, reference anchors $\mathcal{A}_{\text{ref}}$, and placement parameters θ, the interface computes $T = \omega(a_{\text{sub}}, \mathcal{A}_{\text{ref}}; \theta)$, where T is the subject's rigid pose and θ specifies offsets, clearances, or a translation direction. Common operations are $\omega \in \{\text{ATTACH}, \text{CENTRE}, \text{SNAP}\}$: ATTACH aligns mating anchors with the requested orientation and clearance; CENTER centers the subject between two references along a chosen axis; and SNAP translates it along a specified direction until it reaches a support at the prescribed gap. The SDK resolves the anchors in a shared coordinate frame and computes the rotation and translation required by the selected relation, preserving pose components that the operation does not constrain. Operations can be composed, for example, to center a shelf between side panels and then seat it against the back panel. Changes to the underlying geometry automatically propagate through these relationships, keeping dependent placements consistent without manually recomputing coordinates.

A.2 Agent Tools

Following Articraft's focused tool interface (Zhou et al., 2026), we expose actions for editing, execution, inspection, and reference retrieval (Table 4). The Coding Agent can modify the asset program, while the Review Agent can inspect the asset and edit its own tests without changing the asset itself.

Editing and execution. The Coding Agent writes or patches the construction program and can execute Blender scripts to create additional geometry. Compilation builds the asset and returns

Family	Tool(s)	C	R	Purpose
Edit	`write_file`, `replace`	✓		Create or revise the asset program and geometry scripts.
	`write_test`, `edit_test`		✓	Create or revise independent tests against the blueprint.
Execute	`run_blender`	✓		Execute a geometry script and return the mesh and its diagnostics.
	`compile_model`	✓		Build the asset and return structural, geometric, and test feedback.
	`run_tests`		✓	Run the Review Agent's tests and return failures or errors.
Inspect	`probe_model`	✓	✓	Query part and joint structure or geometric measurements.
	`read_gate_evidence`	✓	✓	Retrieve motion and physics evidence for the current asset.
	`request_render`	✓	✓	Return RGB views or semantic maps with a distinct colour per part.
	`crop_region`		✓	Enlarge a selected region of an existing render.
Reference	`read_file`	✓	✓	Read source code, task specifications, or SDK documentation.
	`find_examples`	✓		Retrieve curated examples of geometry and mechanisms.
	`lookup_cadquery_api`, `lookup_blender_api`	✓		Look up signatures and documentation in the installed APIs.

Table 4: Tools available to the Coding (C) and Review (R) Agents. The Coding Agent edits the asset program; the Review Agent edits only its own tests.

diagnostics for invalid structure, disconnected geometry, unintended overlaps, and failed tests. The Review Agent separately writes and revises tests against the blueprint, then executes them on the asset to obtain failures and supporting details.

Inspection. Both agents can query the part hierarchy and joint axes and limits, or run read-only probes to measure dimensions, clearances, overlaps, and connectivity. These measurements identify the parts and spatial relationships involved in a suspected defect. They can also retrieve the motion and physics evidence associated with the current asset to examine failures under articulation or gravity.

Visual Inspection. `request_render` uses Blender to produce RGB images or semantic maps that assign a distinct flat colour to each part, making part boundaries easier to distinguish. Both agents can request front, side, top, or perspective views in open or closed configurations and focus on named parts or joints. The Review Agent can additionally specify joint positions and camera angles, or crop an existing image to inspect a local detail. RGB views reveal appearance and proportions, while semantic maps help locate unexpected part visibility or apparent interpenetration for further geometric inspection.

A.3 Physical Simulation

Collision under Motion. We use MuJoCo to check collisions at sampled joint configurations, with mesh collision geometry constructed by CoACD using a decomposition threshold of 0.05, at most 32 convex hulls per mesh, and random seed 0. The initial collision screen includes the rest pose, seven evenly spaced values along each joint's travel, pairwise 7×7 grids, and 200 random multi-joint samples drawn from the same joint-value grids. We cap this initial set at 1,500 configurations and use sampling seed 0. The functional constraints in $\mathcal{Q}^{\text{text}}$ guide checks of dependent motions: for example, an internal drawer is tested with its access door opened, so an invalid operation sequence is not mistaken for a geometric defect. Collisions that persist during individual joint operation are sent

to the Coding Agent for repair. For collisions found only when multiple joints move together, the Review Agent determines whether they occur during normal use and need repair. For each affected part pair, the report records a representative configuration and penetration depth, allowing the agents to reproduce and inspect the finding.

Stability under Gravity. For free-standing assets, we test stability under gravity in MuJoCo at rest and with each joint opened in turn. In each test, the assembly is placed on a ground plane with a free root, allowing it to settle or topple. We simulate for up to 5 s, stopping early when the root's linear and angular speeds remain at most 0.02 m/s and 0.05 rad/s, respectively, for 0.4 s. Tilt is measured as the angle between the root's initial and current up directions; a final tilt above 3° is flagged for review. The physics report $\mathcal{E}_t$ records, for each pose, the final and maximum tilt, whether the assembly settled, and the simulated duration, so the Review Agent can assess the result against the asset's intended use.

B Benchmark Details

B.1 Spatial Fidelity Protocol

Alignment and Correspondence. We sample 100,000 points uniformly over each asset's surface and record their positions in both endpoint poses. To resolve ambiguity in joint-axis motion direction, we test both ways of pairing the predicted poses with the ground-truth closed and open states. For each pairing, we align both states with ICP from several initial orientations. We select the pairing with the lower alignment error across both states, retaining the original ordering when the difference is negligible. After alignment, we normalize both assets so that the longest side of the ground-truth asset's closed-state bounding box has unit length. The Hungarian algorithm matches parts using distances between their centroids in the aligned closed state, and these matches are retained in the open state.

B.2 Material Transfer Protocol

To separate material errors from geometric discrepancies, we transfer predicted materials onto the ground-truth geometry. We use Hungarian assignment to match parts by their relative positions, shapes, and motion between the rest and open states. Within each matched part pair, component matching minimizes a weighted cost combining differences in normalized centers, bounding-box dimensions and aspect ratios, and relative surface area and bounding-box volume. We transfer materials between the resulting component matches. Numeric material parameters are copied directly, while texture inputs are reduced to surface-area-weighted means because the meshes do not share UV layouts.

Keeping the ground-truth geometry and joint configurations fixed, we render base-color, metallic, and roughness buffers in Blender at 512^2 resolution from three viewpoints in each state. We compute PSNR over visible object interiors, excluding boundary pixels. Scores are averaged over the six images and then over objects. Table 11 reports the transferred-material scores.

B.3 Evaluation Datasets

Spatial Fidelity Evaluation Datasets. LightWheel (Lightwheel, 2025) is a library of simulation-ready OpenUSD assets modelled by artists and released under a CC-BY-NC license. Recent articulation work adopts 243 of these assets, spanning 14 categories, as an evaluation set (Li et al., 2026a;b). It contains fourteen kitchen-appliance categories, including blenders, electric kettles, range hoods, and stovetops. We sample 50 objects evenly across the fourteen categories (Figure 6a).

ArtVIP (Jin et al., 2026) comprises 992 digital-twin articulated objects over 9 categories and 37 subcategories. Each asset is built by professional 3D modellers to a unified standard and ships with tuned dynamic parameters, embedded modular interaction behaviours, and pixel-level affordance annotations. Its categories extend beyond kitchen appliances to furniture, fixtures, cleaning tools, stationery, storage, and mechanical equipment. This diversity challenges methods to construct a wider range of geometries and mechanisms, motivating our use of ArtVIP as the main benchmark. We select 50 objects through category-stratified sampling to span diverse object types and joint counts, and

refine the selection by inspecting the models and renders. Figure 6b–c shows the subset's category and joint-count distributions. Table 5 lists all 50 objects using the same category grouping.

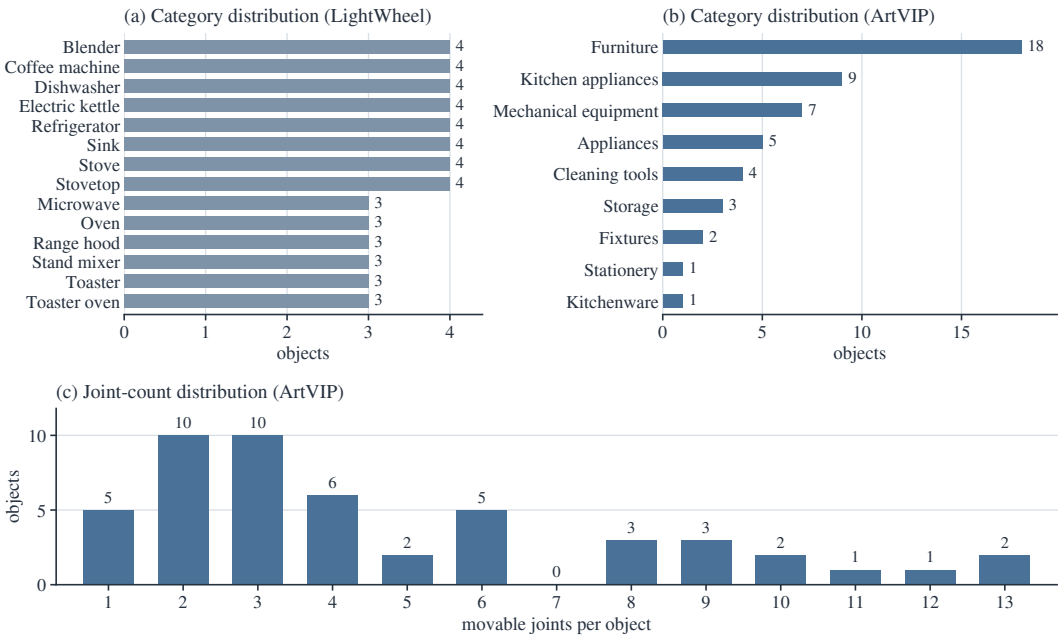


Figure 6: Composition of the spatial-fidelity evaluation subsets. **(a)** The LightWheel subset contains 50 objects across 14 categories. **(b)** The ArtVIP subset contains 50 objects across nine categories. **(c)** The ArtVIP objects have 1–13 movable joints each, with fixed joints excluded.

Table 5: The 50 ArtVIP objects used for evaluation. Names follow the source asset identifiers.

Category	Count	Object names
Furniture	18	GULLABERG_Shoe_cabinet_with_2_compartments_grey_ 88x30x122cm, KILBERGET_chest_of_6_drawers_white_anchor_ unlock_function_144x78cm, MUSKEN_bedside_table_ brown_45x58cm, NORDKISA_Dressing_table_bamboo_76x47cm(2), RIDSPO_Desk_ oak_140X70cm, STOCKHOLM_2025_Sideboard_oak_veneer_ 161x42x83cm, bookcase006, cabinet_1, chest_of_drawers_1, clock007, double_door_4, lab_bench001, office_chair_3, poison_cabinet001, table_3, wardrobe_1, wardrobe_21, wardrobe_22
Kitchen appliances	9	capsule_coffee_maker_3, dishwasher_2, fridge_1, l_dishwasher_1, microwave_2, oven_2, refrigerator_6, rice_cooker_7, steamer
Mechanical equipment	7	dolly, elevator, incubator, medical_cart001, medical_monitor, microscope001, oscillator
Appliances	5	big_electric_fan_9, desk_lamp_008, electric_fan_10, television003, washing_machine_2
Cleaning tools	4	mop_10, mop_9, steping_dustbin_20, stepping_dustbin_6
Storage	3	basket_21, carton_4, clamshell _tool_box_2
Fixtures	2	closestool_3, faucet_7
Stationery	1	folder_25
Kitchenware	1	tongs_10

Stability Evaluation Dataset. We construct 50 object prompts around common destabilization mechanisms in everyday articulated objects, such as cantilevered loads, telescoping extensions, and

tilting panels, as shown in Figure 7. By varying both object categories and mechanisms, the set tests whether generated assets remain stable across diverse support geometries and joint configurations. Each prompt is paired with a reference image generated using `gpt-image-2` (OpenAI, 2026b) from the object's appearance and scale description.

For example, one prompt used in the benchmark reads:

> *A narrow hallway shoe cabinet with a slim upright carcass and three stacked tilt-out shoe bins. Each bin rotates outward and downward on its own horizontal revolute hinge at the bottom front edge of its opening, carrying the whole bin body clear of the carcass front. Hall-furniture scale, tall and shallow with a footprint barely deeper than a shoe.*

Figure 7 shows the distributions by the destabilization mechanism (left) and the category of objects (right).

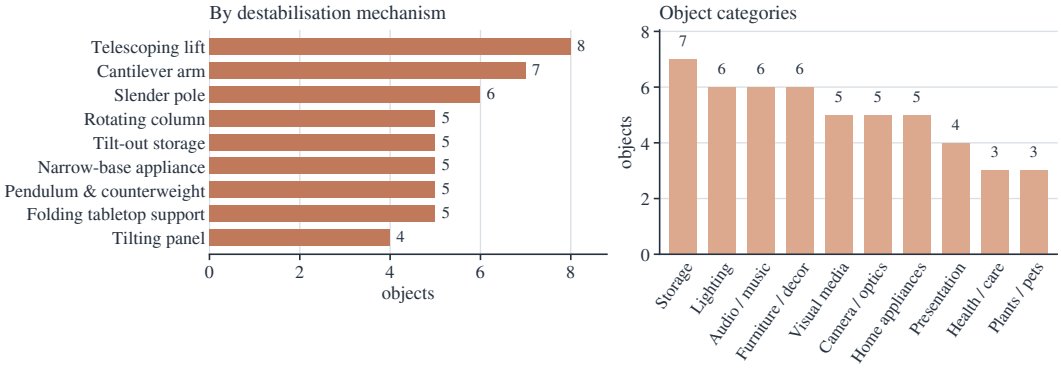


Figure 7: Composition of the stability dataset. **Left:** The 50 objects cover nine destabilization mechanisms. **Right:** The objects span ten categories defined by their primary use.

Mass Evaluation Dataset. Following NeRF2Physics (Zhai et al., 2024), we use ABO-500, a collection of 500 objects from Amazon Berkeley Objects (ABO) (Collins et al., 2022) with calibrated multi-view images and ground-truth masses from product listings. We select 50 articulated objects from distinct product families to cover diverse articulation mechanisms and a ground-truth mass range of 0.086–205 kg. For each object, we provide a brief prompt specifying its category and overall dimensions, together with a reference image. Figure 8 shows the ground-truth mass distribution (left) and object categories (right), and Table 6 lists the selected objects and their ground-truth masses.

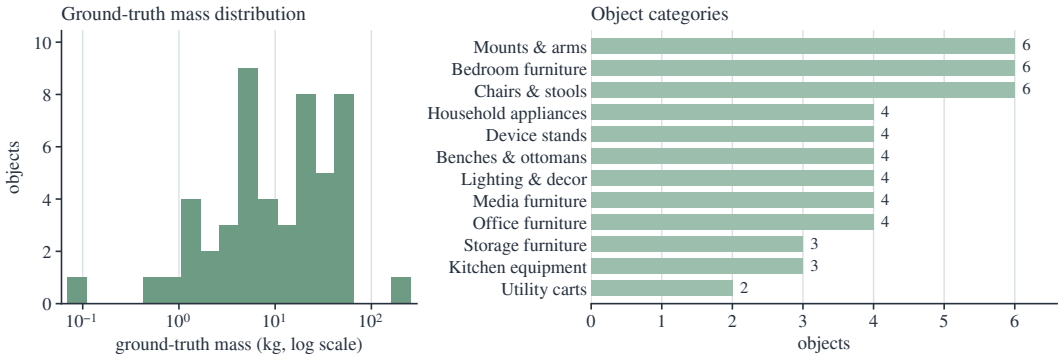


Figure 8: Composition of the mass-evaluation dataset. **Left:** The ground-truth masses range from 0.086 to 205 kg. **Right:** The 50 objects span 12 categories grouped by primary use.

Downstream Tasks Datasets. We evaluate three articulated operation tasks from RoboTwin 2.0 (Chen et al., 2026) and eight from RoboCasa365 (Nasiriany et al., 2026), using disjoint native training and test assets and generating additional assets matched to the training distribution. For RoboTwin

ASIN	Category	Mass (kg)	ASIN	Category	Mass (kg)
B006ZT4VA0	tablet stand	0.086	B07TFQM3JZ	utility cart	10.478
B082L6NQZ7	speaker wall mount	0.617	B084W2S628	barstool	10.977
B075HXHKVF	jewelry box	0.717	B075X463J2	ottoman	12.247
B00WRDS8H0	laptop stand	1.102	B073GGGF7H	storage bench	17.236
B0742D9X4G	table lamp	1.361	B082L28P18	file cabinet	18.144
B07MSYK43S	projector ceiling bracket	1.383	B007IHF36E	sideboard	19.999
B07MF1S1WL	wall sconce lamp	1.397	B07DMGM9T6	kitchen cart	21.319
B07BL7H8N5	frying pan with a lid	1.896	B07L48L3BH	kitchen cart	23.501
B07B8PXMRT	wall clock	2.336	B07H8JNCF3	bedside table	24
B07WFNW48N	step stool	3.452	B07N6RT9SM	portable air conditioner	25.301
B00MIBN16O	monitor arm	3.538	B07TLDMQS2	desk	26
B07XMV34PX	floor fan	3.992	B075X4WBNH	storage ottoman	29.483
B07DHNHZ2Y	dual monitor arm	4.286	B07H8P3K2R	TV stand	32
B07QVC3SLK	shop stool	4.5	B07TR659GF	TV stand	32
B07KR4BT31	ceiling TV mount	4.64	B07GFFPKBG	TV stand	33
B07Q16JDH5	keyboard stand	4.74	B07GFRP9RY	chest of drawers	40
B07T9NM5QR	DJ laptop stand	4.931	B07K6S2G5Z	corner desk	47
B07QWC7NYS	garage creeper seat	5.198	B07QBMQ737	recliner	48.081
B07TX2D4WW	storage cube	5.418	B07H8T5XPT	TV stand	50
B07M9KPR1P	TV wall mount	6.21	B07K6RH9HS	desk	51
B07RKJCGXR	piano bench	6.532	B07Y2B738L	window air conditioner	51.891
B000S6N026	paper shredder	6.68	B07H8SSY4F	sideboard	56
B07WBYQQ8F	garden utility wagon	8.8	B075YZPG5Y	dresser	60.328
B079VKDKC1	bedside cabinet	9.7	B07GFRWJS7	double bed frame with storage	60.5
B07TF7YLJC	desk chair	10.4	B07H8VCM68	wardrobe	205

Table 6: The 50 ABO objects used for mass evaluation, ordered by ground-truth mass.

2.0, each task contains 2–8 native assets; we hold out 1–2 for testing and generate one asset per remaining training asset. For RoboCasa365, we generate demonstrations on 10 native training assets, 10 generated assets, and the rest of the native training assets as held-out test assets for each task. We collect 50 demonstrations per asset using RoboTwin 2.0's task scripts with cuRobo (Sundaralingam et al., 2023) or RoboCasa365's MimicGen pipeline (Mandlekar et al., 2023). Further details on the train-test split and demonstration collection can be found in appendix E.

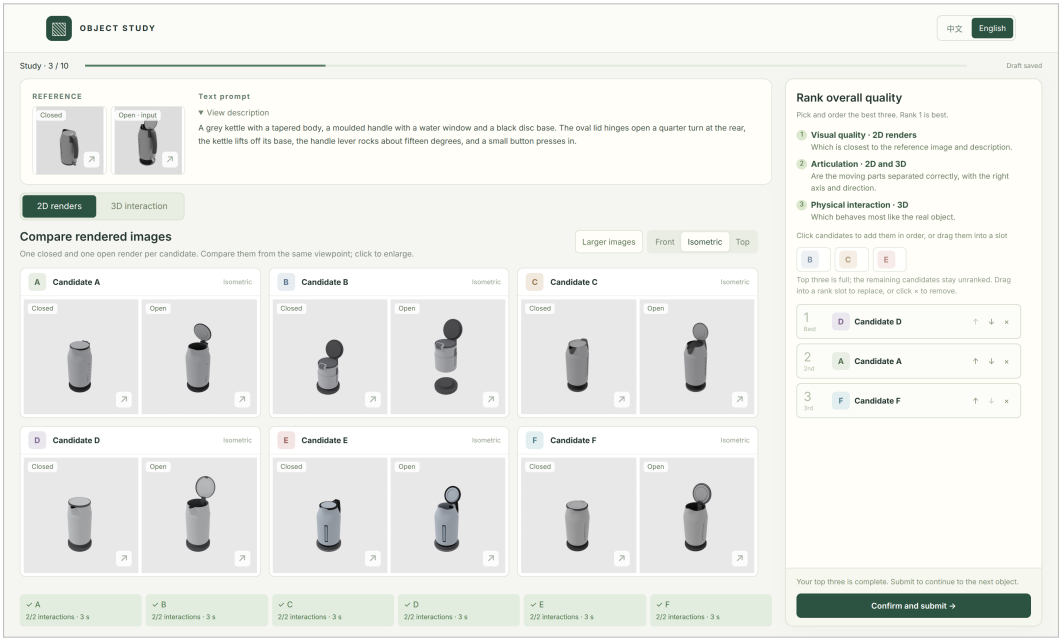
B.4 Human Evaluation

We introduce an interactive user-study protocol in which participants assess generated assets through direct manipulation in MuJoCo alongside rendered views. Making physical interaction part of human evaluation is essential for physically grounded assets: an object may look convincing yet fail to operate as expected. Participants can move articulated parts and observe their responses to applied forces and contact, allowing functional failures to inform their judgments.

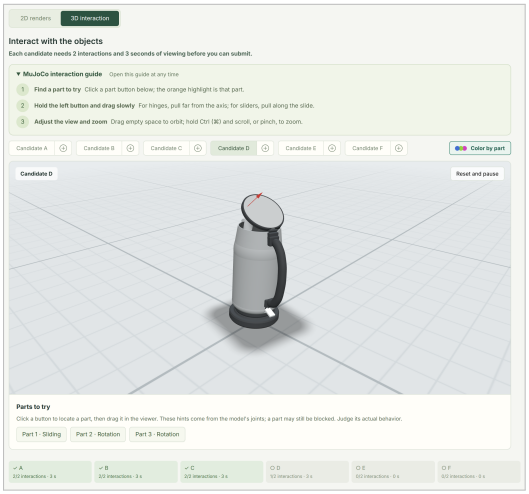
Assets and Comparisons. The study covers 100 ground-truth assets, comprising the 50-object evaluation subsets from ArtVIP and LightWheel. For each object, we compare ARTISAN (GPT-6-Astra) with five representative baselines: Instruct-Particulate, Codex (GPT-6-Astra), Articraft (GPT-6-Astra), Claude (Fable-5.1), and agy (Gemini-3.8-Flash). Participants receive the text prompt and reference images, inspect closed- and open-state renders from matched viewpoints, and interact with all six candidates in the simulator. Method identities are hidden and candidate order is randomized.

Evaluation Criteria. Participants select and rank the three best candidates by prompt alignment and overall quality, considering three complementary aspects: *visual quality*, whether appearance matches the reference image and description; *articulation*, whether movable parts are correctly separated and move about the intended axes and in the intended directions; and *physical interaction*, whether the asset behaves as its real-world counterpart would during manipulation. These criteria guide a single overall ranking for each object. For each method, we report first-place and top-three selection rates, averaged equally across objects.

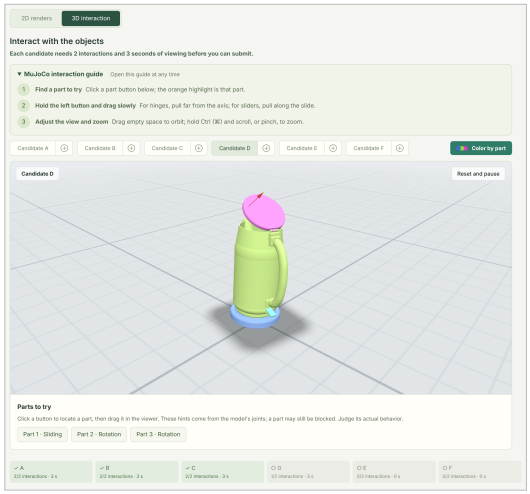
Study Interface. Figure 9 shows the web interface, which presents each object on one page with a 2D-render tab and a 3D-interaction tab that runs all candidates in MuJoCo in the browser. Before



(a) *2D renders* tab



(b) *3D interaction* tab



(c) *3D interaction* tab with *Color by part*

Figure 9: User study interface. **(a)** Closed- and open-state renders of the six anonymized candidates from a shared viewpoint, next to the reference images and text prompt; the right panel lists the three criteria and collects the top-three ranking. **(b)** The same page on the *3D interaction* tab, cropped to the interaction area. Each candidate is loaded into an in-browser MuJoCo viewer, where the participant drags a part (red arrow: applied force); buttons below the viewer locate the movable parts, and the bottom bar tracks each candidate's interactions. **(c)** The same moment with *Color by part* enabled, which draws each rigid body in its own color to reveal the part decomposition.

submitting a ranking, participants must drag each candidate at least twice and spend at least three seconds in its 3D viewer.

Participants and Results. We recruit 30 non-expert college students from diverse academic backgrounds, each evaluating 10 randomly assigned objects. The study yields 275 valid responses. Table 7 reports the selection rates for all six methods.

Table 7: Human evaluation results. Top-1 and Top-3 denote first-place and top-three selection rates, respectively.

Method	Top-1 (%)↑	Top-3 (%)↑
ARTISAN	**43.3**	**85.1**
Codex	21.1	68.7
Articraft	9.8	52.4
Claude	14.2	43.3
Agy	10.5	37.5
Instruct-Particulate	1.1	13.1

C ADDITIONAL RESULTS

C.1 FULL METRIC TABLES

Tables 8 and 9 report complete spatial fidelity results on ArtVIP and LightWheel, including all evaluated model variants. Table 10 reports collision rates on both datasets, together with stability, direct-material, and mass results. Table 11 separately reports material transfer onto the ground-truth geometry on ArtVIP. All results follow the protocols in Section 4.1 and appendices B.1 and B.2.

Spatial Fidelity. With GPT-6-Astra, ARTISAN leads on nine of the eleven ArtVIP metrics, including all five appearance metrics. On LightWheel, it achieves competitive appearance scores and ranks first on four of the six geometry metrics, with the lowest object Chamfer distance in both states. Instruct-Particulate, whose training mixture includes household assets from PartNet-Mobility and GRScenes (Li et al., 2026b), leads on four of the five LightWheel appearance metrics and open-state part IoU. These results show that agentic construction can match or surpass learned articulation pipelines in spatial fidelity; its stronger performance on the more diverse ArtVIP benchmark further suggests broader generalization across object categories.

Physical Attributes. ARTISAN (GPT-6-Astra) achieves the lowest collision rates, with ground-truth contacts excluded, at all three sweep levels on both datasets, together with stability rates of 98% at rest and 96% when open. It also achieves the lowest mass APE, although Agy has lower ADE and higher MnRE. We also estimate mass by multiplying the densities of the agent-assigned materials by their geometric volumes. This density-times-volume estimate yields an MnRE of 0.525, an ADE of 23.70 kg, and an APE of 2.379, substantially underperforming our direct mass estimates. This supports our design of reasoning directly about part mass to account for hollow and composite construction. Its material fidelity is comparable to Instruct-Particulate, with higher base-color and roughness PSNR but lower metallic PSNR.

Model Capacity. For Articraft, Codex, and ARTISAN, moving from GPT-5.5 to GPT-6-Astra improves every reported spatial fidelity metric on both datasets. However, the GPT-6-Astra baselines still exhibit substantially higher collision rates than ARTISAN. Stronger models improve reconstruction quality, but do not remove the need for explicit physical validation.

Table 8: **Complete spatial fidelity results on ArtVIP**. **Bold** and underline indicate the best and second-best values per metric, including ties at the reported precision.

Method	Appearance					Rest-Pose Geometry			Articulated Geometry		
	IoU↑	SSIM↑	PSNR↑	LPIPS↓	CLIP↑	oCD↓	pCD↓	pIoU↑	oCD↓	pCD↓	pIoU↑
Instruct-Particulate	0.6299	0.8585	17.18	0.2708	0.9166	0.0027	0.1137	0.2440	**0.0010**	0.1126	0.2730
PhysX-Anything	0.4547	0.8450	15.26	0.3889	0.8480	0.0097	0.3721	0.0822	0.0123	0.3735	0.0772
URDF-Anything+	0.5568	0.8604	16.39	0.3802	0.8323	0.0062	0.2913	0.1262	0.0068	0.3034	0.1236
Articulate-Anything	0.5785	0.8202	14.46	0.3902	0.8440	0.0087	0.1923	0.1223	0.0097	0.2076	0.1254
Articraft (GPT-5.5)	0.6744	0.8584	17.53	0.2784	0.9029	0.0026	0.0212	0.2791	0.0040	0.0236	0.2727
Articraft (GPT-5.6-Sol)	0.7015	0.8682	17.80	0.2647	0.9132	0.0023	0.0210	0.2787	0.0042	0.0278	0.2756
Articraft (GPT-6-Astra)	0.7085	0.8758	18.74	0.2343	0.9195	0.0022	0.0136	0.3233	0.0034	0.0229	0.3215
Codex (GPT-5.5)	0.6341	0.8509	16.81	0.3041	0.8997	0.0037	0.0258	0.2352	0.0051	0.0376	0.2397
Codex (GPT-5.6-Sol)	0.6772	0.8591	17.30	0.2863	0.9052	0.0032	0.0243	0.2675	0.0045	0.0312	0.2484
Codex (GPT-6-Astra)	0.6804	0.8668	17.99	0.2523	0.9168	0.0033	0.0199	0.3354	0.0039	0.0276	0.3126
Claude (Fable 5.1)	0.6751	0.8772	18.39	0.2608	0.9047	0.0015	0.0290	0.3233	0.0050	0.0432	0.2922
Agy (Gemini-Flash-3.8)	0.7031	0.8777	18.26	0.2548	0.9130	**0.0010**	0.0131	0.3555	0.0050	0.0236	0.3110
Ours (GPT-5.5 w/o phys.)	0.6783	0.8652	17.70	0.2745	0.9050	0.0021	0.0163	0.2975	0.0052	0.0255	0.2828
Ours (GPT-5.5)	0.6841	0.8597	17.61	0.2744	0.9059	0.0016	0.0167	0.2905	0.0030	0.0210	0.2870
Ours (GPT-5.6-Sol)	0.6781	0.8725	17.83	0.2679	0.9146	0.0018	0.0152	0.3006	0.0023	0.0212	0.3064
Ours (GPT-6-Astra)	**0.7243**	**0.8785**	**18.94**	**0.2264**	**0.9268**	0.0011	**0.0089**	**0.3634**	0.0018	**0.0138**	**0.3516**

Table 9: **Complete spatial fidelity results on LightWheel**. **Bold** and underline indicate the best and second-best values per metric.

Method	Appearance					Rest-Pose Geometry			Articulated Geometry		
	IoU↑	SSIM↑	PSNR↑	LPIPS↓	CLIP↑	oCD↓	pCD↓	pIoU↑	oCD↓	pCD↓	pIoU↑
Instruct-Particulate	**0.8847**	**0.9315**	**21.65**	**0.0890**	0.9410	0.0026	0.0781	0.2822	0.0019	0.0844	**0.3295**
PhysX-Anything	0.6481	0.9062	17.10	0.2020	0.8958	0.0071	0.1646	0.1076	0.0138	0.1867	0.0922
URDF-Anything+	0.7409	0.9151	16.22	0.2380	0.8458	0.0048	0.3502	0.1220	0.0062	0.3578	0.1264
Articulate-Anything	0.6376	0.9087	16.62	0.2205	0.8600	0.0093	0.1677	0.1080	0.0134	0.1815	0.1000
Articraft (GPT-5.5)	0.7940	0.9126	18.58	0.1520	0.9120	0.0019	0.0504	0.2092	0.0044	0.0615	0.1781
Articraft (GPT-6-Astra)	0.8496	0.9273	20.86	0.1042	**0.9426**	0.0016	**0.0261**	0.2677	0.0025	0.0328	0.2481
Codex (GPT-5.5)	0.7665	0.9065	18.15	0.1615	0.9053	0.0024	0.0615	0.1913	0.0037	0.0729	0.1687
Codex (GPT-6-Astra)	0.8403	0.9222	20.45	0.1085	0.9370	0.0013	0.0338	0.2589	0.0022	0.0421	0.2545
Claude (Fable 5.1)	0.8067	0.9236	19.46	0.1408	0.9218	0.0019	0.0660	0.2234	0.0034	0.0744	0.1842
Agy (Gemini-Flash-3.8)	0.8028	0.9092	18.62	0.1427	0.9155	0.0019	0.0683	0.2420	0.0038	0.0748	0.2050
Ours (GPT-5.5 w/o phys.)	0.7986	0.9172	18.66	0.1542	0.9116	0.0017	0.0459	0.2301	0.0036	0.0622	0.2046
Ours (GPT-5.5)	0.8033	0.9184	18.63	0.1504	0.9087	0.0018	0.0435	0.2216	0.0032	0.0523	0.1909
Ours (GPT-6-Astra)	0.8688	0.9297	21.01	0.1035	0.9387	**0.0012**	0.0267	**0.2916**	**0.0018**	**0.0298**	0.2817

Table 10: **Complete physical attribute results**. Collision rates are reported on ArtVIP and LightWheel; direct material PSNR uses ArtVIP, stability our stability dataset, and mass uses the ABO subset. Bold and underline mark the best and second-best values per metric; – denotes unavailable results.

Method	Collision (%)						Stability		Material (dB)			Mass		
	ArtVIP			LightWheel										
	Single↓	Double↓	Multi.↓	Single↓	Double↓	Multi.↓	Rest↑	Open↑	Base color↑	Metallic↑	Roughness↑	MnRE↑	ADE (kg)↓	APE↓
NeRF2Physics	–	–	–	–	–	–	–	–	–	–	–	0.601	10.78	0.509
PUGS	–	–	–	–	–	–	–	–	–	–	–	0.412	44.50	3.016
Instruct-Particulate	70.4	90.1	98.7	45.1	66.2	80.6	68%	42%	11.48	**17.86**	10.94	–	–	–
PhysX-Anything	55.0	76.0	81.3	58.6	86.5	96.3	74%	22%	9.22	–	–	0.302	198.07	19.292
URDF-Anything+	84.6	99.2	100.0	92.0	100.0	100.0	64%	48%	–	–	–	–	–	–
Articulate-Anything	31.2	40.5	32.8	24.2	39.0	61.2	64%	44%	7.75	–	–	–	–	–
Articraft (GPT-5.5)	42.5	57.5	79.4	43.8	66.1	82.7	82%	78%	10.93	–	–	–	–	–
Articraft (GPT-5.6-Sol)	46.8	66.1	81.7	–	–	–	82%	76%	10.98	–	–	–	–	–
Articraft (GPT-6-Astra)	45.5	62.0	78.8	42.4	58.0	75.6	80%	80%	11.99	–	–	–	–	–
Codex (GPT-5.5)	61.6	72.8	80.0	49.7	68.6	82.5	–	–	–	–	–	–	–	–
Codex (GPT-5.6-Sol)	38.4	56.9	65.9	–	–	–	80%	86%	11.11	–	–	–	–	–
Codex (GPT-6-Astra)	32.4	48.1	63.7	25.2	38.0	55.6	86%	82%	10.29	–	–	0.763	6.63	0.303
Claude (Fable 5.1)	39.9	50.6	64.7	27.1	39.0	62.6	84%	86%	10.98	–	–	0.743	6.51	0.388
Agy (Gemini-Flash-3.8)	47.8	60.1	78.2	33.6	51.6	63.1	80%	78%	**12.23**	–	–	**0.774**	**5.49**	0.296
Ours (GPT-5.5 w/o phys.)	37.0	47.4	77.0	40.0	61.3	74.8	84%	82%	11.02	16.75	12.09	0.742	6.07	0.340
Ours (GPT-5.5)	3.5	6.6	12.3	3.3	5.5	9.8	96%	94%	12.11	16.85	12.27	0.720	7.22	0.385
Ours (GPT-5.6-Sol)	1.9	4.8	14.5	–	–	–	**98%**	**96%**	10.94	16.58	**14.10**	0.711	8.16	0.484
Ours (GPT-6-Astra)	**1.6**	**3.5**	**12.0**	**2.1**	**3.9**	**8.3**	**98%**	**96%**	12.05	16.99	12.16	0.769	6.07	**0.264**

Table 11: **Material transfer evaluation on ArtVIP**. PSNR (dB) uses predicted materials transferred to ground-truth geometry. **Bold** and underline mark the best and second-best scores; – denotes unavailable results due to the problem scope of methods.

Method	Base color↑	Metallic↑	Roughness↑
Instruct-Particulate	11.11	**18.37**	10.72
PhysX-Anything	9.51	–	–
Articulate-Anything	7.81	–	–
Articraft (GPT-5.5)	11.64	–	–
Articraft (GPT-5.6-Sol)	12.21	–	–
Articraft (GPT-6-Astra)	12.30	–	–
Codex (GPT-5.5)	11.19	–	–
Codex (GPT-5.6-Sol)	**12.65**	–	–
Codex (GPT-6-Astra)	11.62	–	–
Claude (Fable 5.1)	11.49	–	–
Agy (Gemini-Flash-3.8)	12.31	–	–
Ours (GPT-5.5 w/o phys.)	12.26	17.54	11.38
Ours (GPT-5.5)	12.25	17.47	11.50
Ours (GPT-5.6-Sol)	11.63	17.05	**14.58**
Ours (GPT-6-Astra)	12.48	17.19	11.84

C.2 Ablation

We examine three components by removing the Design Agent, the Review Agent, or physical validation from ARTISAN with GPT-5.6-Sol (Table 12). The geometry scores here are the average of the rest and open states.

Table 12: **Component ablations with GPT-5.6-Sol**. Each variant removes the Design Agent, Review Agent, or physical validation. Appearance, geometry, collision, and mean part count are measured on ArtVIP; stability and mass use their respective datasets. Geometry scores average the rest and open states, and mass ADE is in kilograms. **Bold** and underline mark the best and second-best scores per metric; part count is descriptive.

Variant	Appearance		Geometry			Parts	Collision (%)			Stability		Mass		
	PSNR↑	CLIP↑	oCD↓	pCD↓	pIoU↑	N_p	Single↓	Double↓	Multi.↓	Rest↑	Open↑	MnRE↑	ADE↓	APE↓
w/o Design	17.56	0.9103	0.0032	0.0217	0.2691	5.86	**0.6**	**2.5**	13.4	96%	**96%**	0.630	9.57	0.710
w/o Review	17.03	0.9068	0.0034	0.0224	0.2641	6.55	2.3	4.8	**13.3**	96%	94%	0.646	8.54	0.723
w/o Physics	17.56	0.9108	0.0033	0.0215	0.2741	6.66	34.3	52.9	69.6	90%	90%	0.648	**8.01**	0.625
Full	**17.83**	**0.9146**	**0.0021**	**0.0182**	**0.3035**	6.71	1.9	4.8	14.5	**98%**	**96%**	**0.711**	8.16	**0.484**

Design Agent. Without the Design Agent, part IoU falls from 0.3035 to 0.2691 and mass APE rises from 0.484 to 0.710. The mean part count also falls from 6.71 to 5.86, alongside lower single- and double-joint collision rates. This combination shows why collision rates must be interpreted together with structural fidelity: simpler assemblies can exhibit fewer collisions while reproducing the target less accurately. These results suggest that explicit design encourages richer part structures while preserving the intended functionality.

Review Agent. Removing the Review Agent reduces PSNR from 17.83 to 17.03 and part IoU from 0.3035 to 0.2641, while increasing mass APE from 0.484 to 0.723. Its collision rates remain close to those of the full system, with a lower multi-joint rate. The independent review thus helps assess whether the generated asset preserves the intended appearance, structure, and physical properties.

Physical Validation. Without physical validation, single-joint collisions rise from 1.9% to 34.3% and multi-joint collisions from 14.5% to 69.6%, while open-state stability falls from 96% to 90%. These findings reinforce the need for simulation feedback when constructing assets intended for physical interaction.

C.3 Additional Qualitative Examples

In this section, we provide more qualitative comparisons for the ArtVIP and LightWheel benchmarks, as well as for stability and collision. For Articraft, Codex, and our method, we use GPT-6-Astra to generate the assets. For Agy, we use Gemini-3.8-Flash and for Claude we use Claude-Fable-5.1. For all LLMs, the thinking level is set to high.

Figure 10 provides more qualitative comparisons on the ArtVIP and LightWheel benchmarks, showing the reference image, the ground-truth asset, and the assets generated by the baselines and our method. The results show that our method reproduces the reference appearance more faithfully, recovers part decompositions that follow the ground-truth structure, and matches the ground-truth part motion better, whereas the baselines lose the reference colour, merge or fragment parts, and deviate from the ground-truth asset in the open state.

Figure 11 visualizes more stability results under gravity in MuJoCo. The results show that our method can generate stable assets under gravity, while the baselines often produce unstable assets that fail the stability check.

Figure 12 visualizes more collision results. Our assets are collision-free while most baselines exhibit interpenetration, such as drawers, doors, and racks passing into the surrounding body.



Figure 10: Additional qualitative comparisons on the ArtVIP and LightWheel benchmarks.

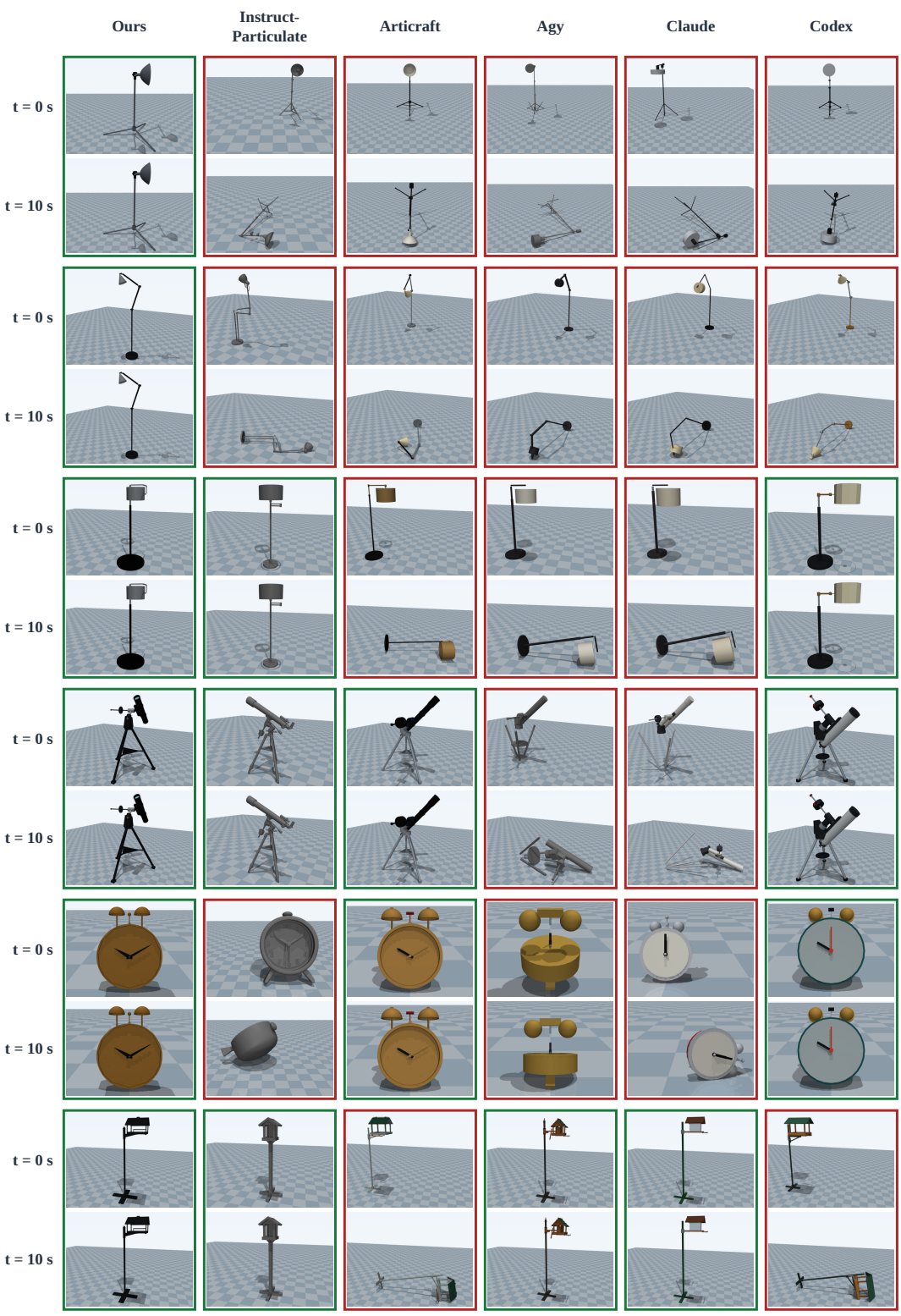


Figure 11: Additional visualizations of stability under gravity, each shown at rest and after ten seconds in MuJoCo. A green frame marks an asset that passes the stability check and a red frame one that does not.

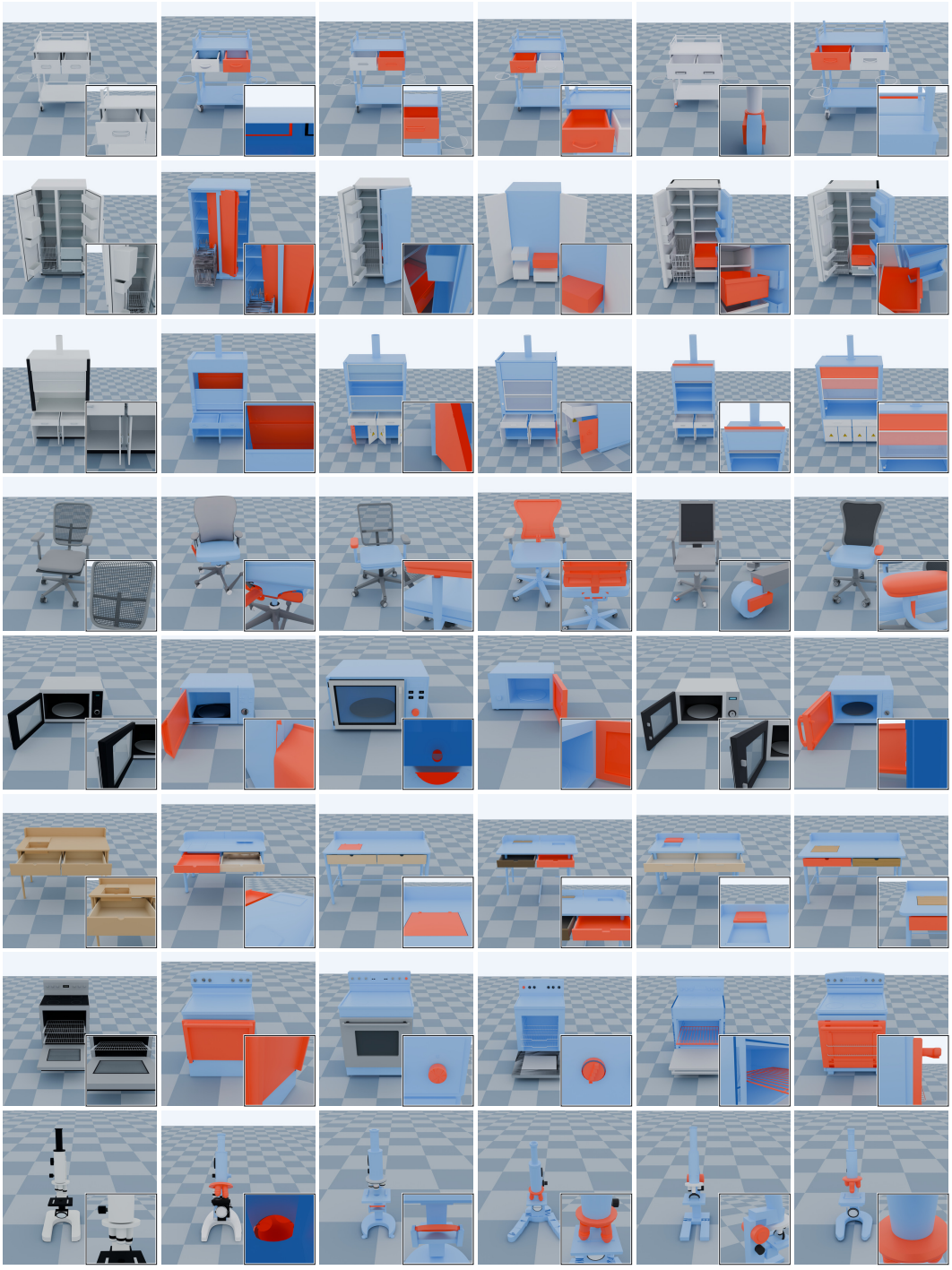


Figure 12: Additional collision comparisons. Red and blue highlight the most severely interpenetrating pair of parts, enlarged in the lower-right inset.

D API AND TIME COST

The Design Agent is allocated up to 12 turns for blueprint generation, and the Coding and Review Agents up to 150 turns each. We set an API cost budget for each complete asset-generation run of $10 for GPT-5.5 and GPT-5.6-Sol, and $15 for GPT-6-Astra. The repair loop between the Coding Agent and the Review Agent is bounded at five rounds and stops early after two consecutive rounds in which no rubric score rises. To curb the rapid increase in API costs as conversation contexts grow, we trigger context compression at 110,000 tokens by capping and deduplicating tool outputs, dropping superseded renders, and summarizing earlier turns. For each method–model configuration, we report the mean API cost and generation time per asset over all evaluated samples pooled from the available benchmarks.

Table 13: **Average API cost and generation time per asset.** Costs are reported in USD and times in minutes.

Method	Model	Avg. cost (USD)	Avg. time (min)
Ours (ARTISAN)	GPT-5.6-Sol	6.03	34.7
Ours (ARTISAN)	GPT-6-Astra	5.79	34.3
Ours (ARTISAN)	GPT-5.5	7.35	33.9
Vanilla Codex	GPT-5.6-Sol	0.33	9.9
Vanilla Codex	GPT-6-Astra	1.07	6.3
Vanilla Codex	GPT-5.5	0.51	1.9
Articraft	GPT-5.6-Sol	0.94	8.2
Articraft	GPT-6-Astra	0.95	5.4
Articraft	GPT-5.5	1.32	6.0
Articulate-Anything	GPT-5.6-Sol	0.46	6.8
Claude Code CLI	claude-fable-5-1	0.54	1.5
agy CLI	gemini-3.8-flash-high	0.92	7.3

Although our method incurs higher API costs, it produces assets with fewer interpenetrations and greater stability, making them more usable in physical simulation (Table 2). Figure 13 shows the API cost and generation time breakdown of ARTISAN, with simulation time reported separately.

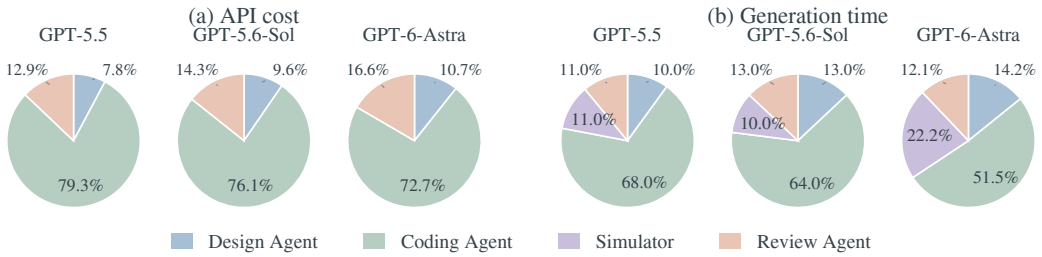


Figure 13: **API cost and generation time breakdown of ARTISAN.** The left three pies show API cost shares of the Design, Coding, and Review Agents; the right three show generation time shares, including the simulator.

E ROBOTIC EXPERIMENTS

To evaluate whether our generated articulated assets are suitable for downstream robot learning, we integrate them into two established robotic simulation benchmarks and use them for demonstration collection and VLA policy training. Our evaluation addresses two complementary questions: **(i)** whether generated assets can be directly instantiated in physics-based robotic simulators and support successful demonstration collection; and **(ii)** whether augmenting benchmark-native training assets

Table 14: Number of assets used in each RoboTwin 2.0 task. Native assets are split into disjoint training and held-out test sets. Artisan and Articraft denote generated assets physically able to be used for downstream evaluation and policy training.

Task	Train	Test	ARTISAN	Train + ARTISAN	Articraft
OpenMicrowave	1	1	6	7	6
OpenLaptop	9	2	9	18	9
TurnSwitch	6	2	3	9	1
Total	16	5	18	34	16

with our generated assets improves policy generalization to held-out native objects. We therefore compare policies trained on demonstrations collected from **Train**, **ARTISAN**, and **Train + ARTISAN** asset sets. Where feasible, we additionally evaluate assets generated by Articraft as a baseline for downstream asset usability.

Shared Setup. For each task, we partition the benchmark-native assets into disjoint training and held-out test sets. Our generated assets are constructed from the corresponding training distribution and are used only for demonstration collection and policy training; all policy evaluations are performed on held-out native assets that are not observed during fine-tuning. This design allows us to measure whether generated assets provide useful object-level variation that transfers back to real benchmark assets rather than merely improving performance on generated instances.

With ARTISAN and Articraft, we generate the same number of assets as the training set of each benchmark and run qualification tests on them to test that they **(i)** can stably maintain the initial poses and states when placed into the simulator and **(ii)** can achieve the success condition upon the predefined interaction. Then we run demonstration collection for 50 trials, and for those that yielded at least 1 successful demonstration, we run a maximum of 3,000 demonstration trials to collect 50 demonstrations per asset. Failed assets are discarded. To rule out other influencing factors and study the contribution of having more assets, we choose the simplest possible setup: we choose the clean tabletop configuration on RoboTwin and a fixed kitchen setup for RoboCasa.

For each asset set, we fine-tune a pretrained $\pi_{0.5}$ policy (Black et al., 2025) using LoRA (Hu et al., 2022). We use rank-16 LoRA adapters for the PaliGemma 2B backbone and rank-32 adapters for the 300M action expert. The batch size is 64, and each policy is trained for 8,000 optimization steps. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, and gradient clipping at 1.0. The learning rate follows a cosine schedule with 1,000 warmup steps, a peak learning rate of 5×10^{-5}, and a configured 30,000-step decay horizon to 5×10^{-6}. EMA is disabled, and the vision encoder remains trainable during fine-tuning.

E.1 ROBOTWIN 2.0

E.1.1 EXPERIMENT DETAILS

We select the three RoboTwin 2.0 tasks that involve articulated-object manipulation: *Open Microwave*, *Open Laptop*, and *Turn Switch*. Table 14 summarizes the native training and held-out test assets, together with the generated assets used for each task.

For RoboTwin 2.0, demonstrations are collected using the benchmark-provided task scripts together with the cuRobo motion planner (Sundaralingam et al., 2023). To ensure compatibility, contact points are annotated on the assets generated by ARTISAN and Articraft. Because the tasks are executed in relatively simple manipulation scenes with scripted trajectory generation, this setting primarily evaluates whether the generated assets can be integrated into the simulator and manipulated successfully without task-specific repair or manual trajectory annotation.

E.1.2 TASK EXECUTION SUCCESS RATES

Table 15 reports the task execution success rates of $\pi_{0.5}$ under the different training-asset compositions. All policies are evaluated on held-out native assets that are disjoint from the assets used for

Table 15: Task execution success rate (%) of $\pi_{0.5}$ on RoboTwin 2.0. All policies are evaluated on held-out native test assets unseen during training. Native, **Ours**, Native + **Ours**, and Articraft denote policies fine-tuned on demonstrations collected from the corresponding asset sets. Higher is better.

Task	Native	Native + ARTISAN	Articraft	ARTISAN
Open Microwave	64%	78%	46%	40%
Open Laptop	98%	97%	86%	97%
Turn Switch	78%	76%	49%	53%
Average	**80%**	**84%**	**60%**	**63%**

Table 16: Number of assets used in each RoboCasa365 task. Native assets are split into disjoint training and held-out test sets. Artisan and Articraft denote generated assets physically able to be used for downstream evaluation and policy training. Each training asset contributes 50 demonstration episodes.

Task	Train	Test	ARTISAN	Train + ARTISAN	Articraft
OpenElectricKettleLid	8	9	10	18	0
OpenFridgeDrawer	5	11	1	6	0
OpenStandMixerHead	9	12	8	17	0
OpenToasterOvenDoor	6	5	3	9	1
SlideDishwasherRack	10	4	1	11	0
SlideOvenRack	3	9	2	5	0
TurnOnStove	10	1	5	15	0
TurnSinkSpout	10	12	5	15	5
Total	**61**	**63**	**35**	**96**	**6**

Entries denote the number of assets.

demonstration collection. This experiment therefore measures whether demonstrations generated from our assets provide useful variation for object-level generalization beyond the benchmark's native training set.

E.1.3 FAILURE MODE ANALYSIS FOR DATA GENERATION

Although most assets are physically stable and can be used to generate successful demonstrations, 5 out of 6 laptop assets and 3 out of 6 microwave assets generated with Articraft have incorrect collision meshes that caused grasp failures and are excluded.

E.2 ROBOCASA365

E.2.1 EXPERIMENT DETAILS

We further evaluate our assets in RoboCasa365, where articulated objects are embedded in substantially more complex kitchen scenes. In our evaluation, we select eight tasks that span different types of articulated objects. Table 16 summarizes the corresponding native training and held-out test assets together with our generated assets and, where applicable, Articraft assets.

We collect demonstrations using RoboCasa365's synthetic demonstration-generation workflow based on MimicGen (Mandlekar et al., 2023). Compared to the scripted RoboTwin 2.0 setting, successful data collection in RoboCasa365 depends more strongly on the interaction between object geometry, articulation, collision geometry, surrounding fixtures, and the transferred manipulation trajectory. We therefore evaluate both the success rate of demonstration generation and the downstream performance of policies trained on the resulting data.

Table 17: Number of assets that pass the quality control test and are successful in generating demonstrations on the RoboCasa365 tasks, for the native assets, the assets generated by ARTISAN, and the assets generated by Articraft.

Task	Native	ARTISAN	Articraft
OpenElectricKettleLid	8/10	10/10	0/10
OpenFridgeDrawer	5/10	1/10	0/10
OpenStandMixerHead	9/10	8/10	0/10
OpenToasterOvenDoor	6/10	3/10	1/10
SlideDishwasherRack	10/10	1/10	0/10
SlideOvenRack	3/10	2/10	0/10
TurnOnStove	10/10	5/10	0/10
TurnSinkSpout	10/10	5/10	5/10

Entries denote # successful assets / # total assets.

Table 18: Task execution success rate (%) of $\pi_{0.5}$ on RoboCasa365. All policies are evaluated on the held-out native test assets listed in Table 16, with 50 evaluation rollouts per test asset. Native, Native + ARTISAN, Articraft, and ARTISAN denote policies fine-tuned on demonstrations collected from the corresponding asset sets. Higher is better.

Task	**Native**	**Native + ARTISAN**	**Articraft**	**ARTISAN**
OpenElectricKettleLid	11%	12%	0%	14%
OpenFridgeDrawer	18%	23%	0%	4%
OpenStandMixerHead	17%	29%	0%	12%
OpenToasterOvenDoor	30%	35%	23%	21%
SlideDishwasherRack	6%	13%	0%	72%
SlideOvenRack	87%	84%	0%	59%
TurnOnStove	92%	98%	0%	8%
TurnSinkSpout	51%	49%	19%	28%
Average	**39%**	**43%**	**5%**	**27%**

E.2.2 DEMONSTRATION GENERATION SUCCESS RATES

Before training the VLA policies, we measure whether each asset can generate successful demonstrations. Unlike RoboTwin 2.0, where trajectories are generated in comparatively simple scenes using task-specific scripts, RoboCasa365 requires manipulation trajectories to remain valid under more constrained object geometries, richer scene context, and the MimicGen trajectory-transfer procedure.

Table 17 reports the demonstration generation success rate for native assets, our generated assets, and Articraft assets. This metric provides an end-to-end measure of whether an asset can be instantiated, simulated, and manipulated successfully by the benchmark's data-generation pipeline, MimicGen. Notably, all three asset groups, native, ARTISAN, and Articraft, contain failure cases. But Articraft shows zero generation success rates in six out of eight task groups. There are two primary reasons for the failures: 1) physical instability; 2) collision and mesh interpenetration.

E.2.3 TASK EXECUTION SUCCESS RATES

Table 18 reports task execution success rates for policies trained using the different asset compositions. As in RoboTwin 2.0, all policies are evaluated exclusively on held-out native assets. The comparison between **Native** and **Native + Ours** therefore isolates whether the additional object variation introduced by our generated assets improves generalization to unseen benchmark objects.

E.2.4 FAILURE MODE ANALYSIS FOR DATA GENERATION

Successful demonstration generation is not guaranteed in RoboCasa365, even when an asset can be loaded and simulated correctly. Failures may arise from the interaction between scene constraints,

articulated mechanism complexity, and limitations of trajectory transfer. We identify three recurring factors.

Scene Complexity. RoboTwin 2.0 primarily evaluates the selected tasks in relatively simple tabletop environments, whereas RoboCasa365 embeds articulated objects within structured kitchen scenes containing surrounding fixtures, support surfaces, and other collision geometry. Consequently, the feasible approach directions and manipulation trajectories are more constrained. Small changes in an object's geometry, scale, pose, or articulation can alter clearances between the robot, the manipulated part, and nearby scene elements. An asset can therefore be physically valid in isolation while still causing a previously valid manipulation trajectory to collide with the environment or become kinematically infeasible.

Task Difficulty. Several RoboCasa365 tasks require interaction with multi-part articulated mechanisms rather than direct actuation of the target joint. For example, opening an electric kettle lid may require the robot to first localize and press a small button, whose motion subsequently actuates the lid mechanism. Such tasks impose stronger requirements on local geometry, contact placement, collision geometry, joint configuration, and mechanical coupling than direct opening or closing motions. Small deviations in the location or dimensions of the interaction region can therefore produce substantial changes in task execution, even when the overall appearance and articulation of the generated object remain plausible. Figure 14 shows the details of such an asset.

MimicGen Limitations. A failure to generate a demonstration does not necessarily imply that the corresponding asset is physically invalid. MimicGen transfers manipulation behavior from source demonstrations to novel object and scene configurations. The transferred trajectories therefore retain assumptions about the local geometry and interaction regions of the source asset. Changes in handle thickness, button position, contact-surface orientation, lid geometry, or surrounding clearance can make a transferred end-effector trajectory invalid even when the modified object remains physically plausible and readily manipulable by an alternative trajectory.

This distinction is particularly important when evaluating generated assets. Asset validity and demonstration-transfer validity measure related but different properties: the former concerns whether the articulated object behaves correctly under simulation and contact, whereas the latter additionally depends on whether a specific demonstration-generation procedure can discover or transfer a valid manipulation trajectory. We therefore report demonstration generation success separately from downstream policy success and qualitatively analyse representative failures. Figure 14 shows an example where MimicGen fails to generate valid demonstrations for an electric kettle that has a hinged button instead of the sliding button as in the human demonstrations.

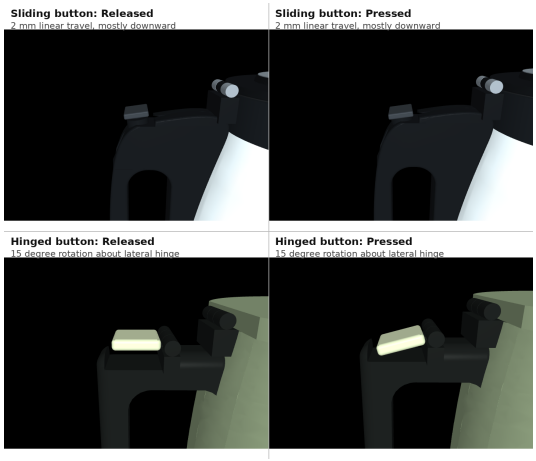


Figure 14: Close-up comparison of electric kettle buttons. **Top:** The native asset uses a sliding button in the human demonstration. **Bottom:** The generated asset uses a hinged button, which is plausible but prevents successful demonstration transfer with MimicGen.

F AGENT TRACE EXAMPLES

We show outputs, tool calls, and images from two real agent traces; `[...]` marks omissions. Turns count recorded responses within each agent across rounds, with separate intake and blueprint counters for Design.

F.1 STAND MIXER: CONTROL ACCESSIBILITY

Prompt. *"A slate blue stand mixer with a heavy rounded head, a matching pedestal and a steel mixing bowl. The head tips back a quarter turn, the bowl lifts off its seat, the whisk spins on the head, the side speed dial turns, and a release button presses in."*

Design Agent Blueprint and design board — **design specified**

Intake Turn 1. Identifies the appliance, visible motions, and user interactions from the input image.
Blueprint Turns 1–4. Authors the blueprint (1), checks its consistency (2), corrects the accessory-cap location (3), and verifies the revision (4). A design board is generated from the resulting blueprint.

Selected outputs (abbreviated).
Parts: base, mixing head, whisk, speed selector, mixing bowl.
Geometry/materials: a rounded $0.29 \times 0.46 \times 0.31$ m slate-blue base; a hollow stainless-steel bowl with an integral handle; an open wire whisk; [...].
Structure/joints: base $\rightarrow$ head, revolute about $+X$, travel 1.48 rad; head $\rightarrow$ whisk, continuous rotation; selector rotation 2.36 rad and bowl lift 0.22 m.
Dependencies/tasks: raise the head before lifting out the bowl; turn the selector to choose a setting. Electrical speed selection imposes no mechanical angle coupling to the whisk.
Functional constraint: the selector tab must be reachable with the bowl seated.
Design board: orthographic, articulated, exploded, and mechanism views.

Coding Agent First build — **construction and checks**

Turns 1–13 **Construction.** uses `read_file` and `lookup_cadquery_api`, then `write_file`, `replace`, and `compile_model` to build the five parts and their joints. `probe_model` and `request_render` check dimensions and the head connection.
Turns 14–33 **Motion repair.** examines motion feedback, probes intermediate poses, and revises the head housing and bearing geometry. Further compilation and rendering precede handoff.


Early build: head raised, bowl removed


After motion-clearance revisions


Candidate sent for review

Physical validation First candidate — **checks passed**

MuJoCo reports passing collision and gravity-stability checks for the submitted candidate. The report supplies the tested poses and measurements to the Review Agent; it does not assess finger access to the selector.

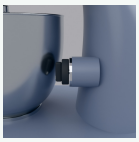
Review Agent First review — **functional repair requested**

Turns 1–3 **Evidence and tests.** reads the program, task bindings, and physics report; uses `write_test` and `run_tests` to check articulation, clearances, and bowl removal. All 24 checks pass.
Turns 4–9 **Focused inspection.** uses `crop_region`, `request_render`, and `probe_model` to inspect the neck and selector. The tab is trapped behind the seated bowl, with only 4.499 mm clearance in the rest pose.
Turns 10 **Feedback.** requests a selector location that satisfies the access requirement, together with a more enclosed rear hinge housing.

"Revise the stationary socket/bezel location and matching selector joint together so the control is beside the bowl rather than trapped behind its wall. [...] do not solve access by requiring bowl removal."


Original review crop: exposed hinge


Original review crop: selector


Matched detail: 4.499 mm gap

Coding Agent Second build — **control and housing revised**

Turns 34–36 **Revision.** reads the review, probes the affected parts, renders the hinge, and uses `replace` to move the selector/socket assembly to an outer pedestal shoulder and add a hinge cover.
Turns 37–41 **Verification.** probes the motion and selector clearance, recompiles, and inspects full and focused renders before resubmission.
Preserved: bowl size, all part masses, and all joint ranges. The selector's placement is changed to meet the blueprint's higher-level access requirement.

Coding Agent Second build (continued)



Revised raised-head configuration



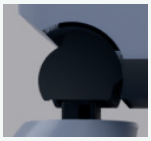
Selector relocated to the side

Physical validation Revised candidate — **checks passed**

The revised candidate continues to pass collision and stability checks. The updated physics report is supplied for final review.

Review Agent Final review — **repair verified**

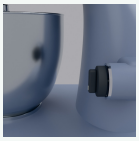
Turns 11–12 **Independent verification.** reads the revised program and physics evidence, crops the selector and hinge, and probes 17 selector positions. The existing test suite remains $24/24$.
Turns 13 **Decision.** approves the revision with an overall score of 8 and no listed issues, compared with 6 in the first review.
Minimum tab-to-bowl distance over 17 poses: 72.378 mm. This geometric measurement and the renders support the access assessment; no hand is simulated.



Original review crop: covered hinge



Original review crop: side approach



Matched detail after repair

Figure 15: **Stand-mixer construction and review trace.** Inspection identifies restricted control access despite passing physical checks; the Coding Agent relocates the selector while preserving the bowl and joint ranges. Small panels include original intermediate renders/crops and matched detail views of the archived candidates.

F.2 Shoe Cabinet: Drawer Connections

Prompt. *"Reconstruct this articulated object. A slim grey shoe cabinet with two small drawers across the top and two tall shoe compartments below, all fitted with round knobs. The drawers pull straight out, while each shoe compartment tilts forward from its lower edge like a hopper. [...]"*

Design Agent Blueprint and design board — **design specified**

Intake Turn 1. Describes the cabinet and its four independent storage motions.
Blueprint Turns 1–2. Writes the specification (1), then confirms completion (2). A design board is generated from its geometry and mechanism descriptions.

Selected outputs (abbreviated).
Parts: one structural carcass, two drawers, two tilt-out shoe compartments.
Geometry/materials: an open-front $0.82 \times 0.31 \times 1.40$ m frame in light warm-grey painted wood; hollow drawer trays with 10–12 mm walls; shoe compartments with curved black ABS liners; [...].
Structure/joints: each drawer connects to the carcass by a $+Y$ prismatic joint with 0.20 m travel; each compartment has a lower-edge revolute joint about $-X$ with 0.96 rad travel. No mechanical couplings are specified.
Tasks: pull either drawer knob or tilt either compartment using its front knob.
Functional constraints: runners remain engaged at full drawer extension; all four storage members can be open together without interpenetration.
Design board: orthographic, articulated, exploded, and connection views.

Coding Agent First build — **construction and checks**

Turns 1–10 **Construction.** uses `read_file`, `find_examples`, and `lookup_cadquery_api` to inspect slide/hinge interfaces; writes the asset and probes its dimensions before compilation.
Turns 11–23 **Geometry and appearance.** renders open and closed states, probes connectivity and overlaps, revises the hinges and facade, and writes the task bindings.
Turns 24–31 **Repair and handoff.** responds to disconnected drawer geometry using `replace`, `probe_model`, and `compile_model`; checks a new front render and submits the candidate.



Intermediate front render



Intermediate open-state render

Physical validation First candidate — **checks passed**

The submitted candidate passes MuJoCo collision and gravity-stability checks. This does not establish that the drawer runners remain mechanically engaged at the ends of their travel.

Review Agent First review — **connection repair requested**

Turns 1–4 Independent tests. reads the blueprint, asset program, and task bindings; uses `probe_model`, `write_test`, and `run_tests` to check scale, motion, retention, and clearances. Only $7/9$ tests pass.
Turns 5–7 Focused inspection. crops the facade, drawers, and compartment interiors, then probes their gaps and connections. Both drawer runners lie 15 mm beyond their guide channels at full extension; facade gaps are 21–65 mm.
Turns 8 Feedback. requests retained runner engagement, tighter facade gaps, smoother liners, and the specified grey finish.


Original review crop: loose facade


Under-drawer detail: detached runner

Coding Agent Second build — **runners and facade revised**

Turns 32–34 Revision. queries curve-construction APIs, probes runner/channel bounds, and replaces the runner and liner geometry. Runner length increases from 80 to 140 mm; the exterior finish changes to light grey.
Turns 35–40 Checks. probes the revised structure, adjusts facade rails, recompiles, and renders the result. All part masses and joint ranges are preserved.
Selected change: `runner_length=0.080` → `runner_length=0.140`.


Revised runner length and finish


Tighter facade and grey finish

Physical validation Repair feedback — **new interference detected**

The motion check reports interference between the revised drawer assembly and the carcass, including approximately 26 mm penetration along the slide direction. The finding is returned to the Coding Agent for another correction before final review.

Coding Agent Follow-up correction — **runner paths cleared**

Turns 41–46 Repair and revalidation. uses `replace` to split the fixed front rail around the two runner paths, verifies the gaps with `probe_model`, recompiles, and checks the front render. The final candidate passes collision and stability checks.
The drawer travel remains 0.20 m; the repair changes surrounding geometry rather than shortening the motion.

Review Agent Final review — **engagement verified**

Turns 9–11 Verification. reads the revised program and tests, independently probes both runners at full extension, and crops the updated facade and interiors. The test suite now passes $9/9$.
Turns 12 Decision. approves the revision with an overall score of 8, up from 5, while noting a minor drawer-interior finish discrepancy.
Full-extension engagement: 15 mm longitudinally, with 0.7 mm surface clearance and no collision. All joint ranges remain unchanged.


Original review crop: revised facade


Under-drawer detail: retained engagement

Figure 16: **Shoe-cabinet construction and review trace.** Longer runners and revised front rails restore engagement without shortening travel. Under-drawer views use matched cameras and supplementary lighting on the unmodified archived models.