SMART: ZERO-SHOT SIM-TO-REAL ARTICULATED OBJECT MANIPULATION VIA LARGE-SCALE SYNTHETIC PRETRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

The ability to interact with articulated objects is essential for intelligent embodiments, but real-world large-scale demonstration collection for this remains challenging due to the precise contact and constraint-following motions involved. Although simulation provides a promising alternative, existing synthetic data efforts cover limited articulated-object categories, while general-purpose synthesis pipelines lack explicit designs for part-level semantics and articulation constraints, hindering agentic task generation and scalable synthesis of high-quality articulated-manipulation demonstrations. To bridge this gap, we introduce **SMART**, a scalable system leveraging large-scale **S**ynthesized **M**anipulation demonstrations for **ART**iculated-object manipulation. At its core, we develop **SMART-Sim**, a simulation platform with articulation-aware design that enables effective task generation and efficient demonstration collection. Building on *SMART-Sim*, we apply agentic task generation and design a scalable distributed synthesis system, using them to synthesize **SMART-Data**, comprising over 1M demonstrations across 44 atomic task types, 5 robot setups, and 2,507 articulated objects. The VLA model pretrained on *SMART-Data* shows competitive performance on simulation benchmarks and achieves zero-shot sim-to-real transfer and scalable performance in real-world articulated-object manipulation tasks. This highlights the potential of synthetic demonstrations in providing effective and scalable supervision for improving VLA model performance in contact-rich articulated-object manipulation.

1 INTRODUCTION

Articulated objects are ubiquitous in human environments and central to many everyday activities. Their movable parts enable rich functionality while imposing structured geometric and physical constraints on interactions. Compared with rigid-object manipulation, articulated-object manipulation requires reasoning about functional parts and joint-constrained motions while maintaining precise and sustained contact throughout the interaction (Ma et al., 2023; Wu et al., 2025c; Wang et al., 2025). Enabling robots to engage in such interactions can facilitate the emergence of intelligent behaviors, making it an important step toward embodied intelligence.

Traditional approaches to articulated-object manipulation often rely on explicit object models, online kinematic estimation, and affordance prediction (Ma et al., 2023; Jiang et al., 2025; Wang et al., 2024b; Wu et al., 2023; Ling et al., 2024; Chen et al., 2026), which require task-specific assumptions and limit scalability across diverse objects and long-horizon interactions. Recently, vision-language-action (VLA) models have emerged as a promising paradigm for general-purpose manipulation (Brohan et al., 2023; Zitkovich et al., 2023; Kim et al., 2025; Black et al., 2025), with their capability largely shaped by the scale, diversity, and quality of training demonstrations (Black et al., 2025; NVIDIA et al., 2025a; Wu et al., 2026). Yet such data is still predominantly collected through human teleoperation (Wu et al., 2025b; AgiBot-World-Contributors et al., 2025), which is costly and hard to scale. To alleviate the burden of robot-based data collection, Universal Manipulation Interface (Chi et al., 2024; Zhaxizhuoma et al., 2025) and egocentric approaches (Hoque et al., 2026; Kareer et al., 2025) have been proposed to collect demonstrations through handheld devices or human-centric embodiments. However, transferring these demonstrations to robot platforms introduces embodiment mismatches and physical infeasibilities (Zhang et al., 2026a; Yang

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



Figure 1: **SMART** synthesizes large-scale simulation data for articulated-object manipulation by combining diverse embodiments and articulated objects, rich manipulation skills, and comprehensive domain randomization.

et al., 2025; Zheng et al., 2026), which can degrade demonstration quality. These challenges are particularly pronounced in articulated-object manipulation, where effective demonstrations require precise and sustained contact with functional parts while satisfying articulation constraints. As a result, high-quality real-world demonstrations for articulated-object manipulation remain difficult to collect at scale.

Simulation offers a natural alternative to these challenges. Prior works (Mandlekar et al., 2023; Nasiriany et al., 2026; Li et al., 2023) show that simulation can scale demonstration collection at low cost. Recent systems such as InternData-A1 (Tian et al., 2025) and MolmoBot (Deshpande et al., 2026) further highlight this potential by generating large-scale synthetic datasets for general manipulation and evaluating sim-to-real transfer using only simulated data. Despite this progress, existing simulation-based datasets for articulated-object manipulation often cover only a narrow range of object categories or remain limited in scale (Cui et al., 2025; Wang et al., 2025). Although works such as InternData-A1 (Tian et al., 2025) and MolmoBot (Deshpande et al., 2026) demonstrate sim-to-real transfer across several manipulation tasks, their synthetic data covers only a limited range of articulated-object categories. Besides, their general-purpose demonstration synthesis pipelines do not explicitly account for part-level semantics and articulation constraints, leading to limited adoption of agentic task generation and challenges in synthesizing high-quality demonstrations for articulated-object manipulation. Consequently, large-scale synthetic data for VLA learning and sim-to-real transfer across diverse articulated-object manipulation remains underexplored.

To fill this gap, we propose **SMART**, a scalable system leveraging **S**ynthetic **M**anipulation data for **ART**iculated object manipulation. We first introduce **SMART-Sim**, a simulation platform with specialized articulation-aware designs for rapid task specification and effective motion generation. We further employ agentic task generation to create articulated object manipulation tasks with varying complexity and temporal horizons. Combined with our distributed data synthesis system, this enables efficient and scalable generation of articulated manipulation tasks and trajectories, resulting in **SMART-Data**, a large-scale synthetic dataset covering diverse articulated objects and manipulation skills. We conduct behavior cloning pretraining exclusively on **SMART-Data**, with the pretraining effect first evaluated on simulation benchmarks. To further explore **SMART-Sim**'s potential in data synthesis, we conduct zero-shot sim-to-real experiments and performance scaling experiments on representative articulated-object manipulation tasks. Our contributions are summarized as follows:

- We propose **SMART-Sim**, a scalable simulation platform with specialized articulation-aware designs for asset annotation, motion generation, skill design, and domain randomization, enabling efficient and effective task and motion generation for articulated-object manipulation.
- We develop an **agentic task generation approach** and a **distributed synthesis system**, and use them to construct **SMART-Data**, a large-scale synthetic articulated-object manipulation dataset with over 1M trajectories across 5 robot setups and 23 object categories, providing diverse supervision for articulated-object manipulation learning.

- Extensive simulation and real-world experiments show that pretraining on **SMART-Data** can improve VLA performance on downstream tasks. Furthermore, data synthesized by **SMART-Sim** enables zero-shot sim-to-real policy transfer with positive scaling trends, highlighting the potential of synthetic data as a scalable supervision source for contact-rich articulated-object manipulation.

2 Related Work

Articulated-object manipulation. Articulated-object manipulation is challenging because it requires reasoning about functional parts and joint constraints while maintaining sustained contact during interaction. Early works address these challenges by leveraging pretrained encoders to extract auxiliary information such as the articulation axis (Wang et al., 2024b) and affordance (Wu et al., 2023; Ling et al., 2024) for manipulation. Recognizing the value of simulation for articulation modeling and supervision, GAPartManip (Cui et al., 2025) and Infinigen-Articulated (Joshi et al., 2025) develop automated pipelines for generating simulation-ready articulated assets with annotations. Subsequent works either construct explicit models for manipulation (Ma et al., 2023; Jiang et al., 2025) or train more capable affordance prediction models (Chen et al., 2026). Despite their effectiveness, these approaches often rely on simplified object models and accurate perception of object geometry and articulation, limiting their scalability to diverse objects and real-world settings.

Simulation and synthesized data for robot manipulation. Simulation offers a scalable and repeatable setting for generating executable robot trajectories. Frameworks such as IsaacLab (NVIDIA et al., 2025b) and robosuite (Zhu et al., 2020) provide reusable assets and infrastructure, while RoboCasa (Nasiriany et al., 2024), RoboCasa365 (Nasiriany et al., 2026), and BEHAVIOR-1K (Li et al., 2023) offer diverse household environments and task suites. Building on this infrastructure, MimicGen (Mandlekar et al., 2023), RoboTwin 2.0 (Chen et al., 2025), and HumanoidGen (Jing et al., 2025) further scale trajectory generation through demonstration recombination, automatic task programming, and LLM-guided trajectory optimization, respectively. At a larger scale, InternData-A1 (Tian et al., 2025) and MolmoBot (Deshpande et al., 2026) demonstrate the potential of synthetic data for policy pretraining and real-world transfer. However, large-scale synthetic data for VLA learning remains underexplored for diverse articulated-object manipulation, especially for contact-intensive tasks that demand precise contact and constrained motion.

3 Simulation Platform

We propose **SMART-Sim**, a simulation platform built on Isaac Lab (NVIDIA et al., 2025b) that streamlines demonstration synthesis with rich embodiment and asset support, automatic asset annotation, skills for motion generation, efficient demonstration collection, and comprehensive domain randomization (Fig. 2). A framework comparison is shown in Tab. 2. Notably, we introduce articulation-aware designs throughout asset annotation, motion generation, skill definition, and domain randomization to accommodate the multi-part structure and physical constraints of articulated objects, enabling efficient collection of high-quality demonstrations.

3.1 Simulation Assets

1) Embodiments. *SMART-Sim* supports 7 robot embodiments via a setup assembly module that pairs them with configurable camera layouts to quickly create new hardware configurations (Tab. 3). **2) Manipulable assets.** We prepare **9,116** rigid objects across **1,071** categories from open-source datasets (Chen et al., 2025; Gao et al., 2025) and 3D generation (Yang et al., 2023), and **2,507** articulated objects across **88** functional categories processed from PartNet-Mobility (Xiang et al., 2020), GRU-Scene (Wang et al., 2024a), and a commercially acquired dataset. **3) Visual scenes and backgrounds.** Visual scenes include mesh scenes from SDG-Scene (Gao et al., 2026) with 3DGS scenes from InteriorGS (SpatialVerse Research Team, 2025) and GN0 (Li et al., 2026) (1,452 in total), augmented by over 10K high-resolution interior images generated by Qwen-Image (Wu et al., 2025a) as pseudo-backgrounds and illumination sources.

3.2 Manipulation Motion Generation

Asset annotation. We adopt the action frame definition in HumanoidGen (Jing et al., 2025), defining a consistent end-effector action frame to provide pose targets for motion generation. For



Figure 2: Overview of **SMART-Sim**. **SMART-Sim** provides rich asset support, asset annotation, skills for motion generation, efficient demonstration collection, and comprehensive domain randomization.

rigid body annotation, we design an AnyGrasp (Fang et al., 2023)-based pipeline. We refer readers to App. B.4 for more details about annotation definitions and rigid body annotation.

To account for part-level interactions and articulation constraints in articulated objects, we design a dedicated annotation pipeline. For each object, a VLM first screens an orthographic view to coarsely identify manipulable parts, such as handles and knobs. We then annotate and calibrate their joint types, valid motion ranges, and manipulation semantics, enabling agentic task generation (Sec. 4.1) to select semantically consistent actions. Next, DINO-X (Ren et al., 2024) precisely localizes each part in the 2D view, from which an anchor action frame is extracted through iterative erosion and 2D-to-3D back-projection (App. 9b). Based on the joint type, we further apply rule-based sampling around each anchor to obtain a set of feasible action frames, introducing interaction diversity for subsequent motion generation. Finally, the VLM identifies part materials to support fine-grained, part-wise and joint-wise visual and physical randomization (Sec. 3.3).

Motion generation. We generate trajectories based on the annotated action frames. In collision-free space, we use cuRobo (Sundaralingam et al., 2023) for motion planning and perturb the resulting trajectories to introduce motion variations. For in-contact interactions, motion generation is subject to tighter constraints, as the end-effector must maintain contact while following the articulation constraints throughout the interaction. To address this, we leverage simulation privileged information, including the exact joint types, positions, and valid motion ranges at runtime, to formulate trajectory optimization with these quantities as explicit constraints (see App. B.5 for details). The resulting trajectories conform to the underlying articulation constraints, improving interaction stability in simulation. Since variations in action frames may introduce unstable or infeasible motions, we leverage the high parallelism of *SMART-Sim* to efficiently filter out such trajectories. Together, these designs enable *SMART-Sim* to support diverse articulated interactions while maintaining stable and high-quality motions.

Manipulation skills. To enable efficient agentic task planning, we group joint types and manipulation semantics into 23 manipulation skills. Each skill is tailored with specific constraint types and parameterizations according to its typical motion patterns and grasping modes, enabling better adaptation to different manipulation tasks. During agentic planning for articulated-object manipulation, skills are selected based on the manipulation semantics of the target parts, enabling fine-grained manipulation behaviors and more effective motion generation.

3.3 Task Domain Randomization

SMART-Sim supports the following domain randomization aspects to increase demonstration diversity: **1) Spatial randomization.** In each episode, we randomize object poses, robot initial configurations, and camera extrinsics. For articulated objects containing other interactable objects, we first randomize them as a group and then independently randomize each contained object. **2) Physical randomization.** Articulated objects consist of multiple movable parts with distinct physical properties, making part-wise randomization important for capturing dynamics variations. We therefore independently randomize the friction and density of each part, while perturbing joint stiffness and damping within small ranges. **3) Visual randomization.** Articulated objects consist of multiple parts with distinct functional semantics, and coarse object-level randomization may obscure these part-level visual distinctions. To diversify visual appearances while preserving the semantic structure of each object in visual observations, we adopt fine-grained, part-level randomization by independently sampling materials and textures for different components from a pre-designed library. We further randomize local and global illumination by perturbing light placement, intensity, color, and temperature. The randomization effects are shown in Fig. 10.

4 Data Synthesis

4.1 Agentic Task Generation

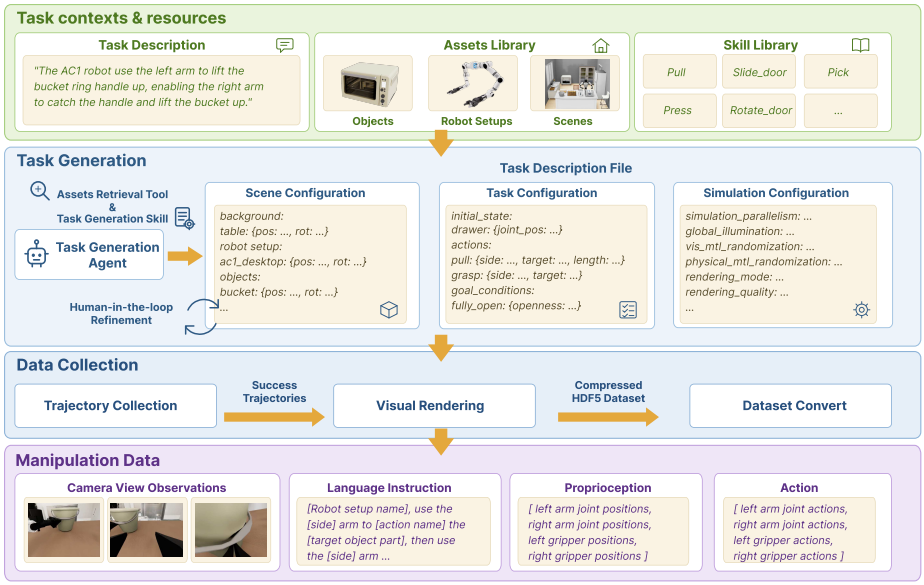
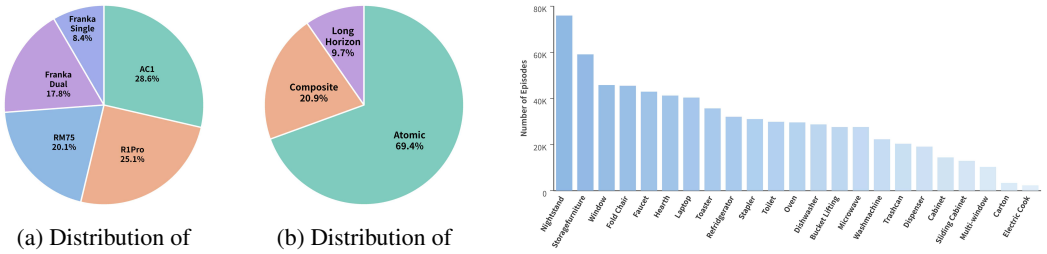


Figure 3: Illustration of our agentic task generation and distributed data synthesis.

We implement a task generation agent to convert natural-language task descriptions into executable simulation configurations (Fig. 3), with the following components: **1) Unified task specification.** Inspired by task specification languages such as BDDL (Li et al., 2023), we introduce a unified task format covering environment configuration (scenes, objects, and robot embodiments), task definition (skill sequences, initial states, and success conditions), and simulation settings (rendering, physics, randomization, and parallelization), bridging high-level task intent and low-level task execution. This structured representation serves as explicit context for the LLM, enabling it to translate abstract instructions into grounded, executable simulator configurations. App. B.7 gives an example. **2) Scene grounding and skill program generation.** Given a task description, the agent first retrieves the required robot, objects, and scene from the resource library using an asset retrieval tool. It then leverages a predefined task generation skill, querying the manipulation annotations of the target object part (Sec. 3.2) and translating task semantics into a parameterized program of skills introduced in Sec. 3.2. **3) Simulation configuration setting.** The agent then configures the simulation settings according to task contents and expected simulation overhead, including domain randomiza-



(a) Distribution of trajectories across five robot setups.

(b) Distribution of trajectories over three task complexity levels.

(c) Distribution of trajectories over articulated object categories.

Figure 4: Statistics of **SMART-Data** over different dimensions.

tion described in Sec. 3.3, rendering mode, parallelism, etc. Human-in-the-loop refinement can optionally be performed through the simulator UI.

4.2 Data Collection

Efficient large-scale demonstration generation is bottlenecked by rendering wasted on failed trajectories and by the mismatch between CPU-bound planning and GPU-intensive rendering. Inspired by Nimbus (He et al., 2026), we address this with a distributed system of a scheduler and three workload types: **1) Trajectory collection.** Workers run environments in parallel, retain only successful trajectories, and apply quality-control conditions to trigger early resets, reaching about 1.25 trajectories per second with 500 parallel environments. **2) Visual rendering.** Workers replay successful trajectories, rendering them into HDF5 samples via tiled cameras for parallel multi-view rendering, with incremental encoding to keep CPU utilization stable. **3) Dataset conversion.** Workers convert the rendered HDF5 files into the LeRobot (Cadene et al., 2026) format with redundant fields filtered and streaming video encoding, running conversion alongside collection and rendering.

By decoupling these CPU- and GPU-intensive stages, our system is highly scalable with more workers and can generate about 144 hours of data per day on a single 8-GPU RTX 4090 server.

4.3 Dataset Statistics

Using this approach, we construct **SMART-Data**, comprising over 1M episodes across 5 robot setups, approximately 500M frames, and over 4,600 hours of manipulation data (Fig. 4). **1) Tasks and skills.** Task design follows a hierarchical taxonomy comprising 69.4% atomic, 20.9% composite, and 9.7% long-horizon tasks (Fig. 4b), covering 23 skills and 44 task types across 10k task YAML configurations. **2) Robots.** The dataset spans five robot setups constructed from the embodiments supported by *SMART-Sim*: ARX AC One, R1Pro, Dual RM75, Dual Franka, and Single Franka, whose proportions are shown in Fig. 4a. **3) Object diversity.** *SMART-Data* covers 23 of the 88 functional categories in *SMART-Sim*'s asset library (Sec. 3.1), spanning objects with revolute, prismatic, and compound joint mechanisms (Fig. 4c), with diversity extending beyond geometry to procedurally generated textures and surface properties.

5 Simulation Experiments

In this section, we evaluate the effectiveness of *SMART-Data* using two simulation benchmarks: LIBERO (Liu et al., 2023) and RoboCasa365 (Nasiriany et al., 2026).

5.1 Experiment Setup

We directly adopt the model architecture proposed in PRTS (Zhang et al., 2026b) as the base VLA architecture for our experiments and remove its Contrastive Reinforcement Learning (CRL) objectives to isolate the effect of pretraining data and enable a controlled comparison. In the first stage, we train the VLM backbone exclusively on *SMART-Data* using a standard cross-entropy objective. In the second stage, we introduce a DiT-based action expert and train the resulting VLA model on the downstream dataset of each target benchmark using a flow-matching objective. Across all controlled variants, we keep the policy architecture and downstream training recipe fixed, so that the primary difference lies in the pretraining data.

Table 1: **Evaluation results (success rates). Left:** LIBERO; **right:** RoboCasa365. Training budgets are shown in gray parentheses. The best result in each column is shown in **bold**.

LIBERO

Method	Spatial	Object	Goal	Long	Average
GR00T N1	94.4	97.6	93.0	90.6	93.9
π_0 (bs=32, 30K steps)	96.8	98.8	95.8	85.2	94.2
Qwen3-VL-π (bs=32, 30K steps)	95.2	99.0	96.2	88.4	94.7
InternVLA-M1	98.0	99.0	93.8	92.6	95.9
$\pi_{0.5}$ (bs=256, 30K steps)	**98.8**	98.2	**98.0**	92.4	96.9
SMART-VLA(Real) (bs=32, 30K steps)	97.8	**99.8**	**98.0**	**95.6**	**97.8**
SMART-VLA(w/o S1) (bs=32, 30K steps)	95.8	99.2	95.8	88.0	94.7
SMART-VLA (bs=32, 30K steps)	97.6	98.8	96.8	94.0	96.8

RoboCasa365

Method	Atomic Seen	Composite Seen	Composite Unseen	Average
π_0	34.6	6.1	1.1	14.8
$\pi_{0.5}$	39.6	7.1	1.2	16.9
GR00T N1.6	51.1	9.4	1.7	21.9
GR00T N1.5	50.7	14.8	2.7	23.9
SMART-VLA(w/o S1)	43.3	9.4	4.4	20.0
SMART-VLA	**52.8**	**23.1**	**7.5**	**28.8**

We denote the model whose first-stage pretraining uses exclusively *SMART-Data* as **SMART-VLA**. For comparison, we reuse the *PRTS w/o CRL* ablation checkpoint evaluated in PRTS (Zhang et al., 2026b) report, which is pretrained on real-world data, and denote it as **SMART-VLA(Real)**. We additionally train a variant that skips first-stage pretraining and undergoes only second-stage training on the downstream benchmark dataset, denoted as **SMART-VLA(w/o S1)**. See App. D.2 for model and training details.

5.2 Key Findings

Pretraining improves the model's downstream manipulation capability, showing the effectiveness of *SMART-Data* in facilitating downstream task learning. On LIBERO, *SMART-VLA* demonstrates faster convergence than *SMART-VLA(w/o S1)* (Fig. 13), reaching 94.5% average success at 15k post-training steps, while *SMART-VLA(w/o S1)* requires 30k steps to achieve comparable performance (94.7%). At 30k steps, *SMART-VLA* further improves to 96.8%, outperforming *SMART-VLA(w/o S1)* by 2.1 points. As shown in Tab. 1, this performance also surpasses π_0 (94.2%) and Qwen3-VL-π (94.7%), while essentially matching $\pi_{0.5}$ (96.9%) despite using a significantly smaller batch size (32 vs. 256). On RoboCasa365, *SMART-VLA* achieves an average score of 28.8%, outperforming the architecturally similar $\pi_{0.5}$ by 11.9 points and showing superior performance to all other baselines that are pretrained on real-world data. Despite *SMART-Data* consisting entirely of synthetic data focused primarily on articulated-object manipulation, pretraining on it yields consistent improvements across a broader range of downstream tasks. This suggests that *SMART-Data* provides transferable supervision for VLA pretraining rather than benefiting only tasks closely aligned with its synthetic training distribution.

Notably, *SMART-VLA* shows a particularly large advantage on longer-horizon and more compositional tasks across both benchmarks, achieving 94.0% versus 88.0% for *SMART-VLA(w/o S1)* on LIBERO and reaching 23.1% and 7.5% on Composite Seen and Composite Unseen, respectively, on RoboCasa365. This pattern is consistent with the multi-stage nature of articulated-object manipulation in *SMART-Data* and suggests that pretraining on such demonstrations may improve temporal generalization.

6 Sim-to-Real Experiments

In this section, we conduct sim-to-real experiments on three robot platforms to further evaluate the effectiveness of *SMART-Data* and our data synthesis method. We investigate the following three research questions: **1)** How effective is pretraining on *SMART-Data* at improving downstream performance on real-world articulated-object manipulation tasks? **2)** Can *SMART-Sim* synthesize effective data to enable zero-shot sim-to-real policy transfer in articulated-object manipulation, and is this agnostic to different robot embodiments? **3)** Is the performance of VLA models on articulated-object manipulation tasks scalable with the amount of synthesized data?

6.1 Experiment Setup

Our real-world evaluations use three platforms: a dual-arm RealMan RM75 platform, an R1Pro platform, and an ARX AC-1 platform, illustrated in App. 7. We construct twelve articulated-object manipulation tasks (8 types and 4 variations) for evaluation, including bucket handle lifting, drawer pulling, dispenser pressing, cabinet-door sliding, toaster-lever pressing, microwave door closing, microwave knob rotating, and toolbox lid flipping. We adopt the same model notation as in Sec. 5, using only demonstrations generated by *SMART-Sim* for the second training phase. We also post-

train a $\pi_{0.5}$-base model on the same synthesized dataset for comparison. See App. D.2 for details. We evaluate policy performance using two metrics: 1) success rate (SR), and 2) action chunk quality, characterized by normalized root-mean-square joint jerk (NRMSJ) for local motion smoothness and normalized mean action variation (NMAV) for global assessment. See App. D.3 for detailed definitions.

6.2 Sim-to-Real Experiment Results

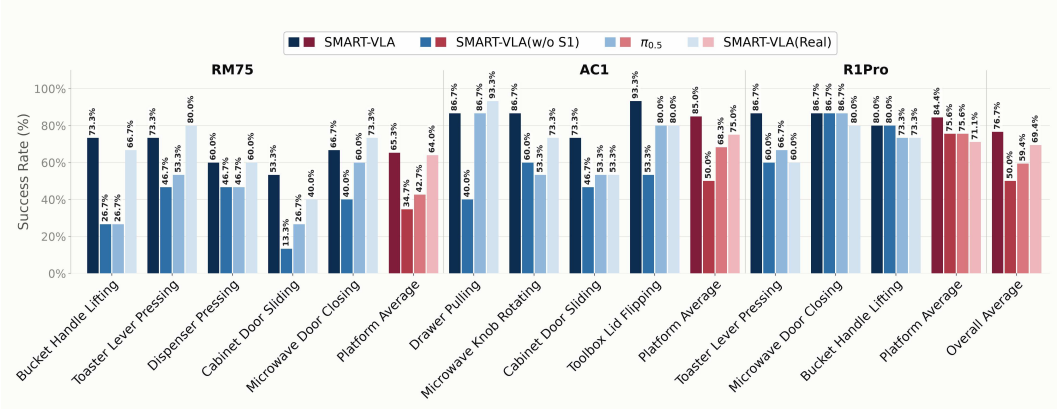


Figure 5: Sim-to-real evaluation across twelve articulated-object manipulation tasks.

Demonstrations synthesized by *SMART-Sim* can enable zero-shot sim-to-real policy transfer in articulated-object manipulation across different robot platforms. Across all evaluated settings, *SMART-VLA* reaches an overall average SR of 76.7%, peaking at 85.0% on AC1 and 84.4% on R1Pro. It also achieves 93.3% SR on toolbox lid flipping with AC1 and 86.7% on microwave door closing with R1Pro, showing strong performance across different tasks. This effectiveness holds across policy models, all of which achieve an overall average SR of at least 50.0%. Together, these results demonstrate that synthetic data enables effective VLA post-training for articulated-object manipulation and zero-shot sim-to-real transfer across robot platforms, tasks, and policy architectures without real-world demonstrations. This further demonstrates that *SMART-Sim* can generate demonstrations that provide effective supervision for real-world articulated-object manipulation.

***SMART-VLA* matches or outperforms comparable policies on real-world articulated-object manipulation tasks across three embodiments, indicating the effectiveness of *SMART-Data* as a VLA pretraining dataset.** Across the twelve tasks, *SMART-Data* pretraining raises SR by 26.7 points on average (76.7% vs. 50.0%), while *SMART-VLA* matches or outperforms *SMART-VLA(w/o S1)* on every task and attains its largest advantage of 46.6 points on RM75 bucket-handle lifting (73.3% vs. 26.7%). This advantage holds across embodiments despite variations in improvement magnitude, suggesting that *SMART-Data* pretraining provides embodiment-agnostic benefits. Against the two baselines pretrained on real-world data, *SMART-VLA* attains an average SR of 76.7%, exceeding $\pi_{0.5}$ and *SMART-VLA(Real)* by 17.3 and 7.3 points and matching or outperforming $\pi_{0.5}$ on ten of the twelve tasks. Together, these results indicate that *SMART-Data* pretraining is competitive with real-world pretraining across the evaluated articulated-object manipulation tasks.

Pretraining on *SMART-Data* enables smoother action generation and stronger robustness to task variations. As shown in Tab. 7, *SMART-VLA* achieves the lowest NRMSJ (0.0581) and NMAV (0.0673), substantially lower than the corresponding values for *SMART-VLA(w/o S1)* (0.1592 and 0.1674), and generates smoother, more stable action chunks. Meanwhile, under an unseen deployment height on AC1 (10 cm higher than during data synthesis), *SMART-VLA* maintains at least 60% SR across all four tasks with an average drop of only 7.5 points, compared with 19.2 and 45.0 points for $\pi_{0.5}$ and *SMART-VLA(w/o S1)*, respectively (Fig. 15). These results suggest that the smooth trajectories and diverse task configurations introduced by motion generation and randomization in *SMART-Data* provide effective supervision for learning stable actions and improving robustness to out-of-distribution deployment conditions.

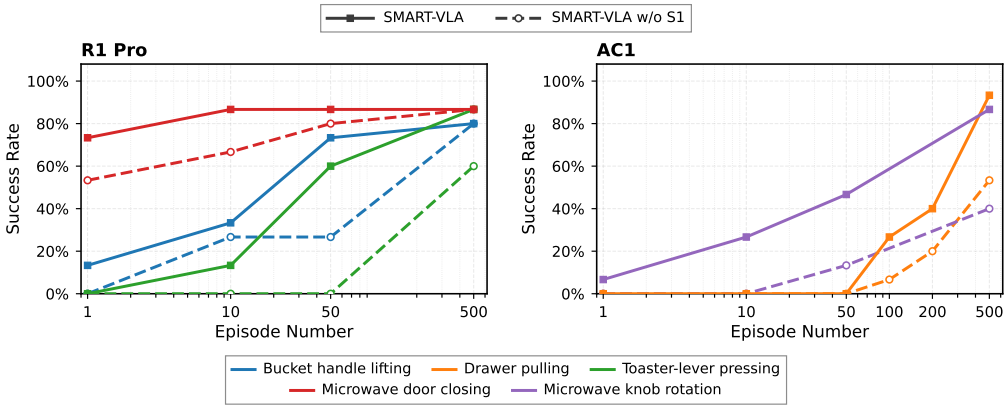


Figure 6: Performance scaling experiment results on AC1 and R1Pro tasks.

6.3 Performance Scaling Experiment

Positive performance scaling shows that *SMART-Sim* provides increasingly rich supervision for articulated-object manipulation. Across all five tasks, both *SMART-VLA* and *SMART-VLA(w/o S1)* improve consistently with more synthesized demonstrations. For *SMART-VLA*, scaling from 1 to 50 episodes increases SR from 13.3% to 73.3% on R1Pro bucket-handle lifting, from 0% to 60.0% on toaster-lever pressing, and from 6.7% to 46.7% on AC1 microwave-knob rotation. More challenging tasks require further scaling for *SMART-VLA*: drawer pulling shows no success below 50 episodes but reaches 40.0% at 200 episodes, while all five tasks reach at least 80.0% at 500 episodes. Scaling rates track task complexity, with loosely constrained tasks (microwave-door closing, bucket-handle lifting) largely saturating by 50 episodes, whereas toaster-lever pressing, drawer pulling, and microwave-knob rotation keep improving at larger scales. Together with spatial and lighting randomization, these results show that *SMART-Sim* provides scalable, diverse supervision, with larger synthetic datasets most beneficial for tasks requiring precise contacts and constrained trajectories.

Pretraining on *SMART-Data* enables more effective and data-efficient downstream adaptation for sim-to-real policy transfer. As shown in Fig. 6, *SMART-VLA* consistently outperforms *SMART-VLA(w/o S1)* at the same post-training data budget. With just one post-training episode, *SMART-VLA* already shows non-trivial manipulation capabilities, reaching 73.3%, 13.3%, and 6.7% SR on microwave-door closing, bucket-handle lifting, and microwave-knob rotation. Pretraining also cuts the task-specific data needed for strong performance, as 50 episodes bring 73.3% on R1Pro bucket-handle lifting and 60.0% on toaster-lever pressing, matching *SMART-VLA(w/o S1)* at 500 episodes (80.0% and 60.0%). The gap is widest on challenging tasks such as microwave-knob rotation, where *SMART-VLA(w/o S1)* records no success with up to 10 episodes while *SMART-VLA* succeeds with one. Overall, these results show that *SMART-Data* pretraining enables rapid adaptation from limited task-specific data while maintaining consistent gains as downstream data scales.

7 Conclusion

In this work, we present *SMART*, a scalable system for synthesizing articulated-object manipulation demonstrations for VLA training. At its core, *SMART-Sim* supports automatic motion generation, diverse robot embodiments, and extensive domain randomization. Building on *SMART-Sim*, agentic task generation and our distributed data synthesis system enable scalable construction of *SMART-Data* with diverse objects and manipulation skills. Experiments show that pretraining on *SMART-Data* improves VLA performance and data efficiency, enables zero-shot sim-to-real transfer, and exhibits strong scaling trends. Together, *SMART* demonstrates the potential of synthetic data as a scalable complement to real-world demonstrations for articulated-object manipulation.

Limitations. Our work has three limitations. First, synthesis throughput decreases for long-horizon tasks due to lower planning and execution success rates. Second, agentic task generation is susceptible to LLM hallucinations and depends on the underlying LLM/VLM capabilities. Third, articulated assets require substantial post-processing for geometry and articulation annotations, motivating future work on jointly generating articulated objects and manipulation annotations.

REPRODUCIBILITY STATEMENT

The implementation details of *SMART-Sim*, the asset annotation method details, and the details of *SMART-Data* are shown in the Appendix. We also provide a video in the supplementary material as an illustration and visual explanation of our framework, our scalable data synthesis system, our dataset, and our sim2real experiments.

AI USE STATEMENT

In this work, we used generative AI tools for automatic asset annotation and the generation of task files for asset manipulation, as described in Sec. 3. We did not use generative AI tools to draft any sections of the paper, and the use of such tools for research ideation or execution was not applicable to this work. Additionally, we used generative AI tools to polish the writing and identify relevant prior work. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REFERENCES

Qingwen AgiBot-World-Contributors, Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Robotics: Science and Systems XXI*, volume 21. Robotics: Science and Systems Foundation, 2025. doi: 10.15607/RSS.2025.XXI.010. URL https://doi.org/10.15607/RSS.2025.XXI.010.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H. Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Robotics: Science and Systems XIX*, volume 19. Robotics: Science and Systems Foundation, 2023. doi: 10.15607/RSS.2023.XIX.025. URL https://doi.org/10.15607/RSS.2023.XIX.025.

Remi Cadene, Simon Alibert, Francesco Capuano, Michel Aractingi, Adil Zouitine, Pepijn Kooijmans, Jade Choghari, Martino Russi, Caroline Pascal, Steven Palma, et al. Lerobot: An open-source library for end-to-end robot learning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=CiZMMAFQR3.

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, 2025. URL https://arxiv.org/abs/2506.18088.

Yue Chen, Muqing Jiang, Kaifeng Zheng, Jiaqi Liang, Chenrui Tie, Haoran Lu, Ruihai Wu, and Hao Dong. PA3FF:learning part-aware dense 3d feature field for generalizable articulated object manipulation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=qXfRXfAHOK.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal Manipulation Interface: In-The-Wild Robot Teaching Without In-The-Wild Robots. In *Robotics: Science and Systems XX*, volume 20. Robotics: Science and Systems Foundation, 2024. doi: 10.15607/RSS.2024.XX.045. URL https://doi.org/10.15607/RSS.2024.XX.045.

Wenbo Cui, Chengyang Zhao, Songlin Wei, Jiazhao Zhang, Haoran Geng, Yaran Chen, Haoran Li, and He Wang. GAPartManip: A Large-Scale Part-Centric Dataset for Material-Agnostic Articulated Object Manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14791–14798. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11128643. URL https://doi.org/10.1109/ICRA55743.2025.11128643.

Abhay Deshpande, Maya Guru, Rose Hendrix, Snehal Jauhri, Ainaz Eftekhar, Rohun Tripathi, Max Argus, Jordi Salvador, Haoquan Fang, Matthew Wallingford, Wilbert Pumacay, Yejin Kim, Quinn Pfeifer, Ying-Chun Lee, Piper Wolters, Omar Rayyan, Mingtong Zhang, Jiafei Duan, Karen Farley, Winson Han, Eli Vanderbilt, Dieter Fox, Ali Farhadi, Georgia Chalvatzaki, Dhruv Shah, and Ranjay Krishna. MolmoB0T: Large-Scale Simulation Enables Zero-Shot Manipulation, 2026. URL https://arxiv.org/abs/2603.16861.

Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 39(5):3929–3945, 2023.

Ning Gao, Yilun Chen, Shuai Yang, Xinyi Chen, Yang Tian, Hao Li, Haifeng Huang, Hanqing Wang, Tai Wang, and Jiangmiao Pang. Genmanip: Llm-driven simulation for generalizable instruction-following manipulation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12187–12198. IEEE, 2025.

Yuhan Gao, Xueying Li, Jicong Ao, Chao Yu, Peng Liu, and Chenjia Bai. Sdgscenes: User-intent driven indoor scene generation via semantic dependency graph. *Pattern Recognition*, pp. 113674, 2026.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4D: Around the World in 3,000 Hours of Egocentric Video. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18973–18990. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01842. URL https://doi.org/10.1109/CVPR52688.2022.01842.

Zeyu He, Yuchang Zhang, Yuanzhen Zhou, Miao Tao, Hengjie Li, Hui Wang, Yang Tian, Jia Zeng, Tai Wang, Wenzhe Cai, et al. Nimbus: A unified embodied synthetic data generation framework. *arXiv preprint arXiv:2601.21449*, 2026.

Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning Dexterous Manipulation from Large-Scale Egocentric Video. In *The Fourteenth International*

Conference on Learning Representations, 2026. URL https://arxiv.org/abs/2505.11709.

Taoran Jiang, Yixuan Guan, Liqian Ma, Jing Xu, Jiaojiao Meng, Weihang Chen, Zecui Zeng, Lusong Li, Dan Wu, and Rui Chen. DexSim2Real^2: Building Explicit World Model for Precise Articulated Object Dexterous Manipulation. *IEEE Transactions on Robotics*, 41:4360–4379, 2025. doi: 10.1109/TRO.2025.3584504. URL https://doi.org/10.1109/TRO.2025.3584504.

Zhi Jing, Siyuan Yang, Jicong Ao, Ting Xiao, Yu-Gang Jiang, and Chenjia Bai. Humanoidgen: Data generation for bimanual dexterous manipulation via llm reasoning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 156210–156256. Curran Associates, Inc., 2025. doi: 10.52202/085713-5220. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/e4ef7454447baa15a424314e6284441b-Paper-Conference.pdf.

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.

Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. EgoMimic: Scaling Imitation Learning via Egocentric Video. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13226–13233. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127989. URL https://doi.org/10.1109/ICRA55743.2025.11127989.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, Mona Anvari, Minjune Hwang, Manasi Sharma, Arman Aydin, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Silvio Savarese, Hyowon Gweon, Karen Liu, Jiajun Wu, and Li Fei-Fei. BEHAVIOR-1K: A Benchmark for Embodied AI with 1,000 Everyday Activities and Realistic Simulation. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 80–93. PMLR, 2023. URL https://proceedings.mlr.press/v205/li23a.html.

Xinhai Li, Xiaotao Zhang, Yuehao Huang, Jiankun Dong, Tianhang Wang, Sunyao Zhou, Yunzi Wu, Chengnuo Sun, Yunfei Ge, Qizhen Weng, et al. Gn0: Toward a unified paradigm for generation, evaluation, and policy learning in visual-language navigation. *arXiv preprint arXiv:2606.03682*, 2026.

Suhan Ling, Yian Wang, Ruihai Wu, Shiguang Wu, Yuzheng Zhuang, Tianyi Xu, Yu Li, Chang Liu, and Hao Dong. Articulated Object Manipulation with Coarse-to-fine Affordance for Mitigating the Effect of Point Cloud Noise. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10895–10901. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10610593. URL https://doi.org/10.1109/ICRA57147.2024.10610593.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Liqian Ma, Jiaojiao Meng, Shuntao Liu, Weihang Chen, Jing Xu, and Rui Chen. Sim2Real^2: Actively Building Explicit Physics Model for Precise Articulated Object Manipulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11698–11704. IEEE,

2023. doi: 10.1109/ICRA48891.2023.10160370. URL https://doi.org/10.1109/ICRA48891.2023.10160370.

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 *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.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-Scale Simulation of Household Tasks for Generalist Robots. In *Robotics: Science and Systems XX*, volume 20. Robotics: Science and Systems Foundation, 2024. doi: 10.15607/RSS.2024.XX.050. URL https://doi.org/10.15607/RSS.2024.XX.050.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A Large-Scale Simulation Framework for Training and Benchmarking Generalist Robots. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.04356.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots, 2025a. URL https://arxiv.org/abs/2503.14734.

NVIDIA, Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Huihua Zhao, Fatima Anes, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, David Chu, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac Lab: A GPU-Accelerated Simulation Framework for Multi-Modal Robot Learning, 2025b. URL https://arxiv.org/abs/2511.04831.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient Action Tokenization for Vision-Language-Action Models. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.012.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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, 2025. URL https://arxiv.org/abs/2504.16054.

Tianhe Ren, Yihao Chen, Qing Jiang, Zhaoyang Zeng, Yuda Xiong, Wenlong Liu, Zhengyu Ma, Junyi Shen, Yuan Gao, Xiaoke Jiang, et al. Dino-x: A unified vision model for open-world object detection and understanding. *arXiv preprint arXiv:2411.14347*, 2024.

Manycore Tech Inc. SpatialVerse Research Team. Interiorgs: A 3d gaussian splatting dataset of semantically labeled indoor scenes. https://huggingface.co/datasets/spatialverse/InteriorGS, 2025.

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. Curobo: Parallelized collision-free robot motion generation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8112–8119. IEEE, 2023.

HunyuanWorld Team, Zhenwei Wang, Yuhao Liu, Junta Wu, Zixiao Gu, Haoyuan Wang, Xuhui Zuo, Tianyu Huang, Wenhuan Li, Sheng Zhang, et al. Hunyuanworld 1.0: Generating immersive, explorable, and interactive 3d worlds from words or pixels. *arXiv preprint arXiv:2507.21809*, 2025.

Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, Yaping Li, Ping Wang, Junhao Cai, Jia Zeng, Hao Dong, and Jiangmiao Pang. InternData-A1: Pioneering High-Fidelity Synthetic Data for Pre-training Generalist Policy, 2025. URL https://arxiv.org/abs/2511.16651.

Hanqing Wang, Jiahe Chen, Wensi Huang, Qingwei Ben, Tai Wang, Boyu Mi, Tao Huang, Siheng Zhao, Yilun Chen, Sizhe Yang, et al. Grutopia: Dream general robots in a city at scale. *arXiv preprint arXiv:2407.10943*, 2024a.

Xi Wang, Tianxing Chen, Qiaojun Yu, Tianling Xu, Zanxin Chen, Yiting Fu, Ziqi He, Cewu Lu, Yao Mu, and Ping Luo. Articulated Object Manipulation using Online Axis Estimation with SAM2-Based Tracking, 2024b. URL https://arxiv.org/abs/2409.16287.

Yufei Wang, Ziyu Wang, Mino Nakura, Pratik Bhowal, Chia-Liang Kuo, Yi-Ting Chen, Zackory Erickson, and David Held. ArticuBot: Learning Universal Articulated Object Manipulation Policy via Large Scale Simulation. In *Robotics: Science and Systems XXI*, volume 21. Robotics: Science and Systems Foundation, 2025. doi: 10.15607/RSS.2025.XXI.156. URL https://doi.org/10.15607/RSS.2025.XXI.156.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025a.

Ruihai Wu, Kai Cheng, Yan Zhao, Chuanruo Ning, Guanqi Zhan, and Hao Dong. Learning environment-aware affordance for 3d articulated object manipulation under occlusions. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 60966–60983. Curran Associates, Inc., 2023. doi: 10.52202/075280-2664. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/bf78fc727cf882df66e6dbc826161e86-Paper-Conference.pdf.

Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025b.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, Yiyu Ren, Kejia Zhang, Hui Yu, Jingmei Zhao, Shuai Zhou, Zhenqi Qiu, Houlong Xiong, Ziyu Wang, Zechen Wang, Ran Cheng, Yong-Lu Li, Yongtao Huang, Xing Zhu, Yujun Shen, and Kecheng Zheng. A Pragmatic VLA Foundation Model, 2026. URL https://arxiv.org/abs/2601.18692.

Yuhan Wu, Tiantian Wei, Shuo Wang, ZhiChao Wang, Yanyong Zhang, Daniel Cremers, and Yan Xia. ArtiBench and ArtiBrain: Benchmarking Generalizable Vision-Language Articulated Object Manipulation, 2025c. URL https://arxiv.org/abs/2511.20330.

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 *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11094–11104. IEEE, 2020.

Ruihan Yang, Qinxi Yu, Yecheng Wu, Rui Yan, Borui Li, An-Chieh Cheng, Xueyan Zou, Yunhao Fang, Xuxin Cheng, Ri-Zhao Qiu, Hongxu Yin, Sifei Liu, Song Han, Yao Lu, and Xiaolong Wang. EgoVLA: Learning Vision-Language-Action Models from Egocentric Human Videos, 2025. URL https://arxiv.org/abs/2507.12440.

Yunhan Yang, Xiaoyang Wu, Tong He, Hengshuang Zhao, and Xihui Liu. Sam3d: Segment anything in 3d scenes. *arXiv preprint arXiv:2306.03908*, 2023.

He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, Wanjia He, Ling Chen, Kai Huang, Jiahao Chen, Sichang Su, Xumin Yu, Ziyi Wang, Chengwei Zhu, Xiao Teng, Yuchun Guo, Yufeng Zhang, Yuandong Liu, Rui Wang, Zisheng Lu, Han Hu, and Zhengyou Zhang. Hy-Embodied-0.5-VLA: From Vision-Language-Action Models to a Real-World Robot Learning Stack, 2026a. URL https://arxiv.org/abs/2606.14409.

Yang Zhang, Jiangyuan Zhao, Chenyou Fan, Fangzheng Yan, Tian Li, Haitong Tang, Sen Fu, Xuan'er Wu, Qizhen Weng, Weinan Zhang, Xiu Li, Chi Zhang, Chenjia Bai, and Xuelong Li. PRTS: A Primitive Reasoning and Tasking System via Contrastive Representations, 2026b. URL https://arxiv.org/abs/2604.27472.

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

Zhaxizhuom Zhaxizhuoma, Kehui Liu, Chuyue Guan, Zhongjie Jia, Ziniu Wu, Xin Liu, Tianyu Wang, Shuai Liang, Pengan Chen, Pingrui Zhang, Haoming Song, Delin Qu, Dong Wang, Zhigang Wang, Nieqing Cao, Yan Ding, Bin Zhao, and Xuelong Li. FastUMI: A Scalable and Hardware-Independent Universal Manipulation Interface with Dataset. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3069–3093. PMLR, 2025. URL https://proceedings.mlr.press/v305/zhaxizhuoma25a.html.

Ruijie Zheng, Dantong Niu, Yuqi Xie, Jing Wang, Mengda Xu, Yunfan Jiang, Fernando Castañeda, Fengyuan Hu, You Liang Tan, Letian Fu, Trevor Darrell, Furong Huang, Yuke Zhu, Danfei Xu, and Linxi Fan. EgoScale: Scaling Dexterous Manipulation with Diverse Egocentric Human Data, 2026. URL https://arxiv.org/abs/2602.16710.

Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Kevin Lin, Abhiram Maddukuri, Soroush Nasiriany, and Yifeng Zhu. robosuite: A Modular Simulation Framework and Benchmark for Robot Learning, 2020. URL https://arxiv.org/abs/2009.12293.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of

Proceedings of Machine Learning Research, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Additional Related Work

VLA models and Real-world data. Vision-language-action models have recently become a prominent approach to general-purpose robot control, and their performance depends on high-quality, diverse, and large-scale demonstration datasets. Works like the π series (Black et al., 2025; Physical Intelligence et al., 2025), LingBot-VLA (Wu et al., 2026), and PRTS (Zhang et al., 2026b) heavily rely on teleoperated robot demonstrations, using more than 10,000 hours of multi-robot data, approximately 20,000 hours of real bimanual data, and 167 billion tokens of manipulation and embodied-reasoning data, respectively. However, teleoperation remains hard to scale because of its high cost and low data collection efficiency. To reduce this burden, studies on UMI (Chi et al., 2024; Zhaxizhuoma et al., 2025) enable portable robot-free demonstration collection through handheld interfaces, with Hy-Embodied-0.5-VLA (Zhang et al., 2026a) further performing effective pretraining with more than 10,000 hours of UMI data. Additionally, egocentric demonstration collection offers another robot-free alternative by capturing human wrist and hand motion, with Ego4D (Grauman et al., 2022) and EgoDex (Hoque et al., 2026) providing egocentric human interaction data and EgoMimic (Kareer et al., 2025), EgoVLA (Yang et al., 2025), and EgoScale (Zheng et al., 2026) demonstrating its value for VLA pretraining. Despite improved collection efficiency and lower cost, these methods either face a trade-off between tracking precision and portability or require pose estimation, retargeting, and robot alignment to recover executable actions. These limitations are particularly problematic for articulated-object manipulation, where small action errors can break sustained contact or violate joint constraints. Overall, current real-world demonstration collection methods struggle to simultaneously provide high-quality data, broad diversity, and high collection efficiency for articulated-object manipulation tasks.

B Framework Details

This section provides additional details of *SMART-Sim*, covering the comparison with existing simulation frameworks, the robot platforms, the annotation pipeline, the domain randomization, and the task file design.

B.1 Framework Comparison

Tab. 2 compares *SMART-Sim* with common robot simulation frameworks. Scenes count the distinct simulation environments, Embodiments count the supported robot platforms, and Objects are reported separately for rigid and articulated ones, where some frameworks only give a total number of objects without distinguishing the two types. Benefiting from our annotation and motion generation pipeline, *SMART-Sim* provides the largest articulated-object asset base among the compared frameworks and marks every listed capability, being the only one that combines AI-generated tasks, multi-environment parallelism, and distributed data synthesis in a single pipeline.

Table 2: Comparison of common robot simulation frameworks. Some frameworks only list the total number of objects and do not distinguish between rigid and articulated objects.

Feature	***SMART-Sim*** (Ours)	Robo-Casa365	Molmo-Space	Arti-Bench	RoboTwin 2.0	RL-Bench	Behavior-1K	Humanoid-Gen	Mani-Skill 2	LIBERO	InternData-A1
Scenes	1,452	120	230,000+	1	1	1	50	20	–	20	227
Embodiments	7	1	2	1	5	1	12	1	1	1	4
(Rigid) Objects	9,116	2,509	130,000+	–	687	28	9318	–	2144	–	3185
Articulated Objects	2,507	20	–	–	44	–	–	4	–	–	321
Realistic Physics	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Realistic Rendering	✓	✗	✗	✗	✓	✗	✓	✓	✓	✗	✓
Object Annotation	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗	✓
Scripted Datagen	✓	✓	✓	✓	✓	✓	✗	✓	✓	✗	✓
AI-generated Tasks	✓	✗	✓	✗	✗	✗	✗	✓	✗	✗	✓
Multi-environment Parallelism	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗
Distributed Data Synthesis	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

B.2 Robot Platform Details

SMART-Sim supports a range of real-world robot platforms through a unified setup assembly interface, so that the same task and skill definitions can be reused across different hardware configurations.



Figure 7: Overview of the real-world robot platforms, including a dual-arm RealMan-75 platform (left), an ARX AC-1 dual-arm platform (center), and an R1pro platform (right). All platforms have two wrist cameras and one center camera mounted.

We show the details of supported robot platforms in Tab. 3.

Table 3: Robot platform details of *SMART-Sim*. Franka appears as two platforms, a single-arm and a dual-arm configuration, which are counted as one robot embodiment in Sec. 3.1.

Platform Name	**RM75**	**R1Pro**	**AC1**	**Franka (Single)**	**Franka (Dual-Arm)**	**Flexiv Rizon 4s**	**Marvin M6s**	**Unitree H1-2**
Degrees of Freedom	16	16	14	8	16	8	16	36
Camera Number	3	3	3	2	3	2	3	3
End-Effector Type	Robotiq 2F-85	Self-designed Gripper	AC1 Gripper	Franka Hand	Franka Hand	Robotiq 2F-85	DAS Gripper V3	X Hand
Center Camera Type	RealSense L515	RealSense D435	RealSense D405	RealSense D435	RealSense D435	RealSense D405	RealSense L515	RealSense L515
Wrist Camera Type	RealSense D405	RealSense D405	RealSense D405	RealSense D435	RealSense D435	RealSense D405	DAS Fisheye	RealSense D405

B.3 Illustration of Annotations

Fig. 8 shows the action frames that our pipeline annotates on representative assets, following the convention introduced in Sec. B.4. Each frame is anchored at the gripper-object contact point and consists of an approach axis and a parallel axis, so a single annotation specifies both where the end effector should contact the asset and how it should be oriented. Panels (a) to (c) show this convention on three articulated assets, namely a sliding door, a microwave oven, and a drawer. In each case the frame sits on the handle or knob that the gripper grasps, the approach axis points along the direction from which the end effector reaches the part, and the parallel axis is aligned with the joint axis along which the part is actuated. The orange arrows mark the direction of the constrained motion of the corresponding part, which links the annotated frame to the articulation constraint used during motion generation. Panel (d) applies the same frame definition to a Robotiq end effector, so that asset frames and the end-effector frame share one convention and can be matched directly when a motion is generated.

B.4 Asset Annotation

We illustrate the two asset annotation paths introduced in Sec. 3.2, one for rigid objects and one for articulated objects, in Fig. 9.



(a) The action frame annotated on the round handle of a sliding door.

(b) The action frames annotated on the bar handle and the round knobs of a microwave oven.

(c) The action frame annotated on the round knob of a drawer.

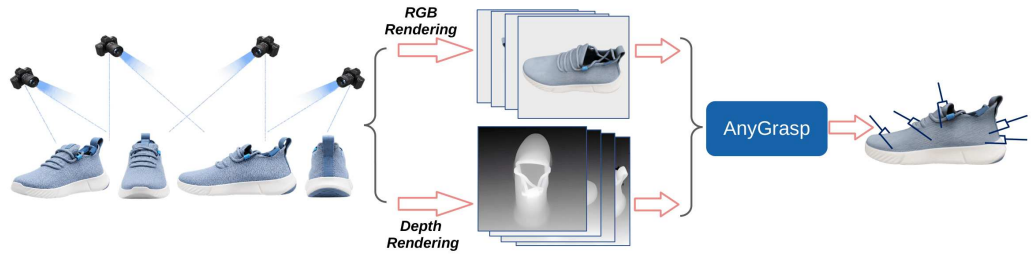
(d) The action frame annotated on a Robotiq end effector.

Figure 8: The illustration of asset annotations, with (a), (b), and (c) showing the action frames on different articulated objects, and (d) showing the annotation on a Robotiq end effector. The orange arrows show the direction of the articulation constraints between parts.

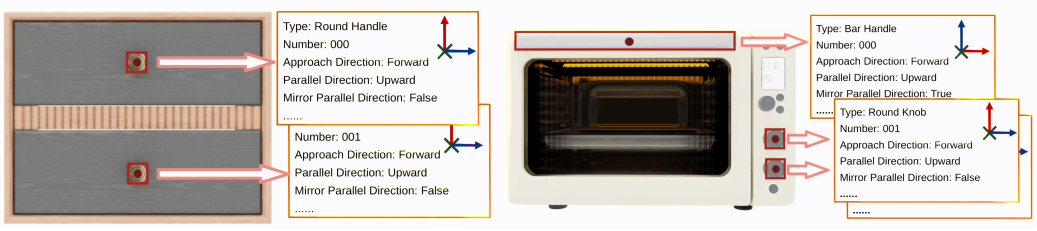
An action frame is defined by an approach axis, a parallel axis, and an origin at the gripper-object contact point, and it follows the convention of HumanoidGen (Jing et al., 2025). The approach axis is the direction along which the end effector approaches the object, so the pre-contact pose is aligned with the target part. The parallel axis is perpendicular to the approach axis and is fixed by the right-hand rule, and it specifies the rotation axis when the end effector manipulates a part. The origin marks where the gripper contacts the object, so a single frame encodes both the contact location and the orientation of the end effector. Both annotation paths produce frames under this convention, which turns grasp pose annotation into explicit pose targets for motion generation.

For rigid objects, we use AnyGrasp (Fang et al., 2023) to generate grasp annotations. Specifically, each object is rendered from four different viewpoints, and AnyGrasp is applied to predict 40 grasp poses for each object. The predicted grasp poses are then projected back to the object base coordinate frame using the extrinsic parameters of the rendering cameras. See Fig. 9a for detailed approach illustration. In addition, object categories and language descriptions are generated by a VLM, which are later used for LLM-based task generation.

For articulated objects, we follow the part-level pipeline in Sec. 3.2, where an anchor action frame is extracted for each localized part and then expanded into a set of feasible frames. See Fig. 9b for detailed approach illustration.



(a) The annotation approach for rigid objects.



(b) The annotation approach for articulated objects.

Figure 9: The illustration of annotation approaches for (a) rigid objects and (b) articulated objects.

B.5 CONSTRAINT-AWARE MOTION SOLVER

This section provides the solver-level details of the constraint-aware motion generation introduced in Sec. 3.2. Given a long-horizon task, the task generation agent produces a grounded sequence of manipulation skills and arm movements. Each skill specifies the manipulated part, an annotated action frame, and continuous parameters such as a translation distance or a joint rotation angle. *SMART-Sim* then converts each skill into a constrained motion problem.

We represent a movement process by $T+1$ discrete time steps. Let $t \in \{0, 1, \ldots, T\}$ index these steps, where $t = 0$ and $t = T$ denote the initial and final steps, respectively. At time step t, the robot joint configuration is $\theta_t \in \mathbb{R}^n$, and the pose of the end-effector action frame is ${}^{W}\mathbf{e}_t \in SE(3)$. The end-effector pose is determined by forward kinematics as ${}^{W}\mathbf{e}_t = f_{\mathrm{FK}}(\theta_t)$. We use c_j to denote a constraint on the robot joint configuration and c_e to denote a constraint on the end-effector pose. For each skill, let $\mathcal{C}_{\mathrm{goal}}$ contain the end-effector pose constraints c_e that must be satisfied at the target configuration, and let $\mathcal{C}_{\mathrm{path}}$ contain the joint constraints c_j that must be satisfied throughout the movement.

Goal configuration. The first step solves for the target Cartesian end-effector pose ${}^{W}\mathbf{e}_T^{\mathrm{tar}}$. Let Δ denote the specified motion value of the manipulated joint, such as a displacement for a prismatic joint or a rotation angle for a revolute joint. The manipulated part annotation provides the action frame ${}^{\mathcal{J}}\mathbf{F}_{\mathrm{act}}$ in the joint coordinate frame. Based on the joint type and Δ, we construct the corresponding spatial transformation $\mathbf{T}_{\mathrm{joint}}(\Delta)$ in the joint frame. The canonical target end-effector pose in the joint coordinate frame is obtained by left-multiplying the action frame with this transformation:

$$ {}^{\mathcal{J}}\bar{\mathbf{e}}_T = \mathbf{T}_{\mathrm{joint}}(\Delta)\, {}^{\mathcal{J}}\mathbf{F}_{\mathrm{act}}. \tag{1} $$

We then express this pose in the global coordinate frame using the joint frame pose ${}^{W}\mathbf{F}_{\mathcal{J}}$:

$$ {}^{W}\bar{\mathbf{e}}_T = {}^{W}\mathbf{F}_{\mathcal{J}}\, {}^{\mathcal{J}}\bar{\mathbf{e}}_T. \tag{2} $$

We use ${}^{W}\bar{\mathbf{e}}_T$ as the initial target pose for the optimization below and solve for the target end-effector pose that satisfies the skill constraints:

$$ \begin{aligned} {}^{W}\mathbf{e}_T^{\mathrm{tar},*} = \arg\min_{{}^{W}\mathbf{e}_T^{\mathrm{tar}}} \quad & w_{\mathrm{init}} d_{SE(3)}({}^{W}\mathbf{e}_T^{\mathrm{tar}}, {}^{W}\bar{\mathbf{e}}_T)^2 \\ & + \sum_{c_e \in \mathcal{C}_{\mathrm{goal}}} w_{\mathrm{pose}}\, \mathrm{dist}\big(c_e({}^{W}\mathbf{e}_T^{\mathrm{tar}}, \Delta), [c_e^{\mathrm{lower}}, c_e^{\mathrm{upper}}]\big)^2 \\ \text{s.t.} \quad & c_e({}^{W}\mathbf{e}_T^{\mathrm{tar}}, \Delta) \in [c_e^{\mathrm{lower}}, c_e^{\mathrm{upper}}], \quad \forall\, c_e \in \mathcal{C}_{\mathrm{goal}}, \end{aligned} \tag{3} $$

where $d_{SE(3)}(\cdot, \cdot)$ measures the weighted translation and orientation difference between two end-effector poses, w_{init} balances the initialization term, and w_{pose} weights the end-effector pose constraints.

After obtaining ${}^{W}\mathbf{e}_T^{\mathrm{tar},*}$, we solve for the target joint configuration using a constrained pose-matching objective:

$$ \begin{aligned} \theta_T^* = \arg\min_{\theta_T} \quad & w_{\mathrm{ik}} d_{SE(3)}(f_{\mathrm{FK}}(\theta_T), {}^{W}\mathbf{e}_T^{\mathrm{tar},*})^2 + w_{\mathrm{reg}} \|\theta_T - \theta_{\mathrm{nominal}}\|_2^2 - w_{\mathrm{manip}} \mathcal{M}(\theta_T) \\ \text{s.t.} \quad & \theta_{\min} \le \theta_T \le \theta_{\max}, \\ & \theta_T \in \Theta_{\text{collision-free}}, \end{aligned} \tag{4} $$

where $\mathcal{M}(\theta_T)$ is the manipulability evaluated from the robot Jacobian at θ_T during the optimization. This term prefers well-conditioned arm configurations while the joint configuration is being optimized, which is useful for sustained contact tasks where small changes in the end-effector pose can otherwise lead to large joint motions.

Constrained trajectory. After obtaining θ_T^*, the second step solves for the intermediate configurations $\{\theta_t\}_{t=1}^{T-1}$ that connect the initial configuration to the target configuration while preserving the

interaction constraints. We use a constrained trajectory solver of the form

$$
\begin{aligned}
\min_{\{\theta_t\}_{t=1}^{T-1}} \quad & \sum_{t=1}^{T-1} \left(w_{\mathrm{s}} \|\theta_{t+1} - 2\theta_t + \theta_{t-1}\|_2^2 + w_{\mathrm{e}} \|\theta_{t+1} - \theta_t\|_2^2 \right) \\
& + \sum_{t=1}^{T-1} w_{\mathrm{c}} \sum_{c_j \in \mathcal{C}_{\mathrm{path}}} \mathrm{dist}\big(c_j(\theta_t), [c_j^{\mathrm{lower}}, c_j^{\mathrm{upper}}]\big)^2 \\
\text{s.t.} \quad & {}^{W}\mathbf{e}_t = f_{\mathrm{FK}}(\theta_t), \\
& c_j(\theta_t) \in [c_j^{\mathrm{lower}}, c_j^{\mathrm{upper}}], \quad \forall\, c_j \in \mathcal{C}_{\mathrm{path}},\ \forall\, t \in \{1, \dots, T-1\}, \\
& \theta_t \in \Theta_{\text{collision-free}}, \quad \forall\, t \in \{1, \dots, T-1\}, \\
& \theta_0 = \theta_{\mathrm{start}}, \qquad \theta_T = \theta_T^*, \\
& \|\theta_{t+1} - \theta_t\|_\infty \le v_{\max}, \qquad \|\theta_{t+1} - 2\theta_t + \theta_{t-1}\|_\infty \le a_{\max},
\end{aligned}
\tag{5}
$$

where the three objective terms encourage smooth acceleration, efficient motion, and satisfaction of the path constraints, respectively. The path constraints $\mathcal{C}_{\mathrm{path}}$ are skill-specific joint constraints derived from the action frame and articulation model, and they restrict the robot joint trajectory to the feasible set induced by the interaction. The velocity and acceleration bounds are applied to adjacent and consecutive differences of θ_t.

We use the constrained trajectory formulation in Eq. (5) for in-contact motion, where the articulation model supplies the joint axis, origin, type, and motion-range constraints directly from the simulator. The resulting joint-space trajectory is checked for collision, joint-limit violation, contact loss, and excessive configuration deviation. Trajectories that violate these conditions are rejected. This two-stage design allows the same solver interface to support translation-based, rotation-based, and composite manipulation skills while retaining the geometric semantics of the annotated action frames.

B.6 Illustration of Domain Randomization

Fig. 10 visualizes the randomization aspects that *SMART-Sim* applies, which cover spatial configuration, physical properties, visual appearance, and illumination. The spatial aspect varies object poses, robot initial configurations, and camera extrinsics at the start of each episode, and for articulated objects that contain other interactable objects it first randomizes the container and then the contained objects independently. The physical aspect samples the friction and density of every movable part separately and perturbs joint stiffness and damping within small ranges, so that part-level dynamics differ across episodes. The visual aspect assigns materials and textures at the part level from a pre-designed library of 200+ materials and 1000+ textures, and the illumination aspect further perturbs local and global lighting through light placement, intensity, color, and temperature. Tab. 4 summarizes the parameters involved in each aspect together with the ranges that are sampled.

B.7 Task File Illustration

The following YAML file shows the task file skeleton used for the toolbox opening task.

Task file skeleton for toolbox opening task on AC1 robot

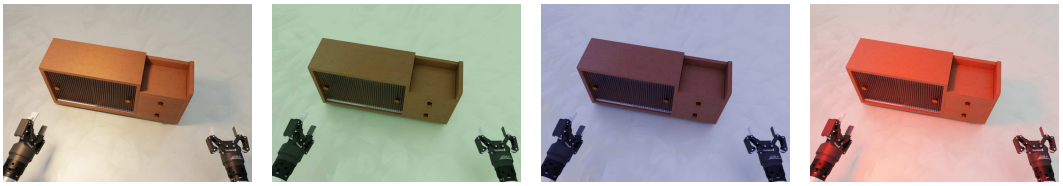
```yaml
task:
  task_name: toolbox_open_real_ac1_left_000
  task_type: revolute_door_open_horizontally
  task_description: AC1 robot, use the left arm to open the toolbox.
  detailed_task_description: The AC1 robot first uses the left arm to reach the handle
      of the toolbox upper lid. Then it opens the toolbox with locked-orientation
      revolute rotation, flipping the lid upward.

data_saving:
  DATA_SAVING_TASK_DIR: toolbox_open_real_ac1_left_000
  trajectory_collecting:
    trajectory_collecting_saving_sub_dir: trajectories
    saving_workers: 4
    task_execution_number: 1000
```



(a) Visualization of visual material randomization.



(b) Visualization of camera randomization.



(c) Visualization of illumination randomization.



(d) Visualization of spatial and robot configuration randomization.

Figure 10: The domain randomization effects supported by **SMART-Sim**.

```
  replay_demo:
    replay_demo_consuming_sub_dir: trajectories
    replay_demo_saving_sub_dir: demos
    saving_workers: 4
    task_execution_number: 9999

simulation:
  trajectory_collecting_num_envs: 100
  trajectory_collecting_time_sync_mode: E_mode
  trajectory_collecting_record_freq: 30
  replay_demo_num_envs: 16
  replay_demo_record_freq: 30
  replay_demo_time_sync_mode: E_mode
  env_spacing: 50
  enable_cameras: true
  render:
    rendering_mode: quality
    rtx.rendermode: RealTimePathTracing
  physics_material:
    static_friction: 0.2
    dynamic_friction: 0.2

lighting:
  enable_light_randomization_at_reset: true
  enable_stepwise_light_randomization: false
  global_dome_light:
    color: [..., ...]
    color_temperature: [..., ...]
    intensity: [..., ...]
    texture_format: automatic
```

Table 4: Domain randomization parameters of *SMART-Sim*.

Aspect	Parameter	Range	Purpose
Spatial	Object position (x, y, z)	± 0.1 m (x, y), ± 0.03 m (z)	Prevent the policy from memorizing a fixed placement and generalize to arbitrary object positions on the tabletop
	Object orientation (roll, pitch, yaw)	$\pm 15^\circ$ (roll, pitch), $\pm 5^\circ$ (yaw)	Cover varied grasp and contact directions so that the policy does not overfit a single pose
	Robot initial joint configuration	$\pm 5^\circ$	Make the policy robust to initial configuration offsets and narrow the gap to real deployment
	Camera extrinsics	± 0.05 m (position), $\pm 5^\circ$ (orientation)	Tolerate camera mounting errors and ease visual sim-to-real transfer
Physical	Part friction	± 0.15	Adapt to contact and slipping behavior under different surface conditions
	Part density	± 0.5	Cover the dynamics difference caused by part mass variation
	Joint stiffness	± 0.003 (prismatic), ± 0.001 (revolute)	Cover the looseness difference between hinges and sliding rails
	Joint damping	± 0.003 (prismatic), ± 0.001 (revolute)	Cover joint resistance variation and keep motion stable after contact
Visual	Part material and texture	Per part, from a library of 200+ materials and 1000+ textures	Avoid letting the policy rely on a fixed appearance
	Metallic	± 0.2	Cover the reflectance variation of metal-like surfaces
	Roughness	± 0.2	Cover gloss and highlight changes under different finishes
Illumination	Light position	± 0.1 m (x, y), ± 0.03 m (z)	Adapt to the shadow and highlight patterns produced by shifted light sources
	Light intensity	$[0.5, 1.5]\times$ nominal	Adapt to brightness variation and avoid failures under over- or under-exposure
	Light color	± 0.3 (RGB)	Cover the color casts that appear in the visual input
	Light temperature	± 2000 K	Cover both warm and cool lighting conditions

```
    texture_file: ...
  local:
    key_fill_sphere:
      type: sphere
      pos: ...
      color: [..., ...]
      color_temperature: [..., ...]
      intensity: [..., ...]
```

```yaml
      radius: [..., ...]
    rim_light_cylinder:
      type: cylinder
      pos: ...
      rot: ...
      color: [..., ...]
      color_temperature: [..., ...]
      intensity: [..., ...]
      radius: ...
      length: ...

background:
  use_default_skybox: false

robot:
  trajectory_collecting_setup: ac1_cfg_no_camera
  replay_demo_setup: ac1_cfg_camera_no_randomization
  origin_pose:
    translation: [0.05, 0, 0.9]
    eulerzyx: [0, 0, 0]

rigid_objects:
  table_brown:
    usd_path: null
    pos: [0.6074, 0, 0.6585]
    rot: [0, 0.7071, 0.7071, 0]
    scale: [2.0000, 1.8000, 1.0000]
    is_static: true
    pose_randomization_cfg: null
    mdl_material_randomization_cfg: null
    enable_sensors: false
    filter_prim_paths: null
    collision_enabled: false

  scene_lab:
    as_name: scene_lab
    usd_path: null
    pos: [-1.3889, 1.2404, 1.3347]
    rot: [0.9970, -0.0072, 0.0177, 0.0745]
    scale: [1.0, 1.0, 1.0]
    is_static: true
    pose_randomization_cfg: null
    mdl_material_randomization_cfg: null
    enable_sensors: false
    filter_prim_paths: null
    collision_enabled: false

articulated_objects:
  toolbox_real:
    as_name: toolbox_real
    usd_path: null
    is_static: true
    pos: [0.7035, 0.2666, 0.9400]
    rot: [0, -0.7071, 0.7071, 0]
    scale: [1.0, 1.0, 1.0]
    enable_self_collisions: false
    pose_randomization_cfg:
      mode: delta
      position:
        x:
          dist: uniform
          low: -0.05
          high: 0.05
        y:
          dist: uniform
          low: -0.05
          high: 0.05
        z:
          dist: uniform
          low: -0.03
          high: 0.03
      orientation:
        yaw:
          dist: uniform
          low: -0.5
          high: 0.5
    joint_initial_state_dict: null
    disable_gravity: true
```

```
    enable_sensors: false
    mdl_material_randomization_cfg: null

task_condition:
  runtime_checking_interval: 0
  duplicate_runtime_conditions_to_ending: false
  conditions:
    all_actor_ee_translation_deviation_condition:
      phase: runtime
      type: failure
      threshold: 0.05
    all_actor_joint_jerk_condition:
      phase: runtime
      type: failure
      threshold: 1
    all_actor_joint_out_of_hard_limit_condition:
      phase: runtime
      type: failure
      threshold: 3.14
    object_fully_opened_condition:
      object_name: toolbox_real
      joint_name: joint_revolute_000
      target_openness: 0.6
      threshold: 0.2
      phase: ending
      type: success

actions:
  parallel_rotate_revolute_door_by_angle_with_locked_orientation:
    target_object: toolbox_real
    side: left
    action_frame_name: grasp_001
    robot_action_frame_name: grasp_001
    rotate_angle: 65.0
    approaching_random_offset_params:
      translation_min: [-0.03, -0.02, -0.05]
      translation_max: [0.03, 0.0, -0.03]
      eulerzyx_min: [-5, 0, -5]
      eulerzyx_max: [5, 0, 5]
    approaching_method: motion_planning
    reaching_random_offset_params: null
    reaching_method: interpolation
    grasping_hand_command_name: grasp
    rotate_frame_suffix: revolute_000
    releasing_hand_command_name: grasp
    target_grasp_percentage: null
    retreating_random_offset_params:
      translation_min: [0, 0, -0.05]
      translation_max: [0, 0, -0.02]
      eulerzyx_min: [-5, -10, -5]
      eulerzyx_max: [5, 0, 5]
    retreating_method: skip
```

C Dataset Details

This section provides additional details of *SMART-Data*, including the distributed synthesis system that produces it, the comparison with existing simulation datasets, and the illustration of data samples.

C.1 Details of Distributed Synthesis

This subsection provides the implementation details of the distributed synthesis system summarized in Sec. 4.2, covering the stage decomposition, the worker configuration, and the throughput measurement.

Our synthesis system follows the decoupled design of Nimbus (He et al., 2026) and consists of a scheduler and three workload types, namely trajectory collection, visual rendering, and dataset conversion. The scheduler maintains the task queue and dispatches every validated task configuration to the corresponding workers, so that the three stages run concurrently on the same server instead of sequentially inside a single process. Trajectory collection and visual rendering are GPU-bound

and share the GPU pool, while dataset conversion is CPU-bound and keeps the CPUs busy while the collected trajectories are rendered and written.

At the system level, all workers share one storage pool over NFS, so the intermediate products of one stage are immediately visible to the next without explicit data movement. The scheduler dispatches tasks in a first-in-first-out order by default and additionally supports priority scheduling, which allows urgent tasks to be moved ahead of the queue. To relieve storage pressure, the scheduler can split a task that requires a large number of episodes into several tasks that each require only a small number of episodes and dispatch them independently, so that the intermediate products of one job do not accumulate on disk at the same time. We further provide a command-line tool that reports task status and resets failed tasks, which simplifies monitoring and recovery during long synthesis runs. The released *SMART-Data* is generated with 96 workers in total. We describe the trajectory collection, visual rendering, and dataset conversion stages in detail in the following paragraphs.

The trajectory collection workers instantiate many environments in parallel and retain only successful executions for the subsequent rendering stage. Each task is launched with a default parallelism of 500 environments, and the number of parallel environments is adjusted according to the task horizon estimated from the generated skill sequence, which balances generation throughput against GPU memory consumption. To support this parallelism, all environments of a task share a single motion planner, which lowers the GPU memory cost of running many rollouts at once. The workers also apply quality-control conditions on the joint jerk and the configuration deviation to trigger early resets, so that failed and low-quality trajectories are discarded before they reach rendering or storage. With this setting, a task achieves a collection throughput of about 1.25 trajectories per second.

The visual rendering workers replay the successful trajectories with the recorded seeds and render them into complete VLA training samples stored in HDF5 format. Tiled cameras in IsaacLab render the multi-view observations of an environment in a single pass, which reduces the rendering cost of each sample. A stream recorder writes and encodes the HDF5 files incrementally during the simulation instead of at the end of each episode, which removes the end-of-episode CPU spikes and keeps the GPU utilization stable throughout rendering.

The dataset conversion workers then convert the intermediate HDF5 files into the LeRobot (Cadene et al., 2026) v3.0 format, remove redundant fields, and apply key mapping to align trajectory and camera data across robot embodiments. The multi-camera video encoding is performed per frame instead of at the end of each episode, which flattens the CPU peaks and allows conversion workers to run alongside collection and rendering workers.

By decoupling the CPU-bound and GPU-bound stages and applying these optimizations, a single server with 8 RTX 4090 GPUs reaches a production throughput of about 144 hours of manipulation data per day.

C.2 Dataset Comparison

Tab. 5 compares *SMART-Data* with simulation datasets that contain substantial articulated-object manipulation data. Traj. counts the total number of trajectories and Traj. (Arti.) the subset that involves articulated objects, Skill and Task report the number of covered manipulation skills and tasks, Scene and Embodiment report the number of simulation scenes and robot embodiments, and Collection Method distinguishes data collected by teleoperation and augmentation from data synthesized autonomously. The Scene value reported for *SMART-Data* counts the 1,122 distinct scenes that are actually used to collect the released trajectories, which form a subset of the 1,452 scenes in the asset library listed in Tab. 2. For *SMART-Data*, we only count atomic tasks, since composite tasks can be assembled in many ways and a consistent quantitative comparison of them is difficult. These entries appear in both Tab. 2 and this table when a work provides both a simulation framework and a released dataset, and the two tables ask different questions: the framework table compares what a simulator can do, while this table compares the scale and coverage of the data it provides. Across the compared entries, *SMART-Data* provides the largest articulated-object trajectory set, the broadest skill and scene coverage, and it is the only one whose data is synthesized fully autonomously at this scale, which makes the resulting demonstrations consistently reproducible and scalable to new assets and tasks.

Table 5: Comparison of robotic simulation datasets with substantial articulated-object manipulation data. For *SMART-Data*, we only report the number of atomic tasks, as the high diversity of composite tasks makes a consistent quantitative comparison challenging.

Dataset	Traj. (All)	Traj. (Arti.)	Skill	Task	Scene	Embodiment	Collection Method
RoboCasa365 (Nasiriany et al., 2026)	655k	415k	8	100	120	2	Teleoperation & Augmentation
RoboTwin 2.0 (Chen et al., 2025)	100k	12k	–	50	1	5	Autonomous
Articubot (Wang et al., 2025)	42.3k	42.3k	2	1	1	1	Autonomous
MolmoBot (Deshpande et al., 2026)	1.7M	125.6k	2	1	1	1	Autonomous
InternData-A1 (Tian et al., 2025)	630k	74.4k	18	70	227	4	Autonomous
SMART-Data	**1M**	**1M**	**23**	**44(Atomic)**	**1,122**	**5**	**Autonomous**



Figure 11: Representative samples from **SMART-Data** across diverse scenes, objects, and manipulation tasks.

C.3 Dataset Illustration

Fig. 11 shows representative samples from *SMART-Data* across diverse scenes, objects, and manipulation tasks.

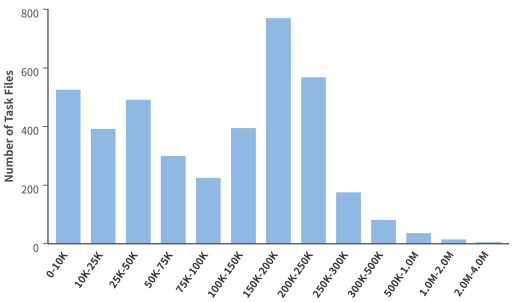
Fig. 12 shows two additional statistics of *SMART-Data* that are not reported in the main text, namely the distribution of task durations and the coverage of manipulation skills. The duration distribution confirms that the dataset is not dominated by short single-stage episodes, and a substantial portion of the trajectories covers longer horizons with many frames. The skill distribution shows that the collected trajectories spread over a broad set of atomic manipulation skills rather than concentrating on a few common ones, which supports the diversity of the dataset for pretraining.

D Experiment Details

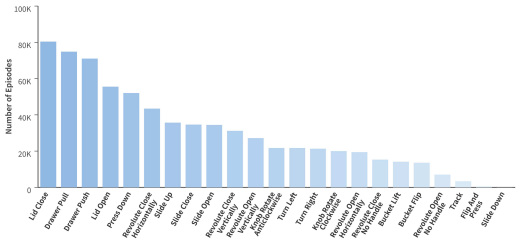
D.1 Task Details

We use Hunyuan3D (Zhao et al., 2025) to generate 3D assets from images of the target objects and then post-process the generated geometry into part-separated assets with explicit articulation constraints. Hunyuan-World (Team et al., 2025) is used to rapidly generate 3DGS scenes to serve as task backgrounds. To diversify demonstration trajectories and improve policy robustness, we randomize the initial configurations of task objects and robots within small, bounded ranges, and introduce perturbation into motion generation as mentioned in Sec. 3.2. We also randomize scene lighting at each environment reset to diversify the illumination conditions in demonstrations.

- **Bucket handle lifting**: lift the bucket by its compliant revolute handle;
- **Drawer pulling**: grasp the drawer handle and pull the drawer out of the cabinet using the left arm;
- **Dispenser pressing**: press a spring-loaded dispenser button downward;
- **Cabinet-door sliding**: slide a right-side sliding cabinet door to the left to open it;
- **Toaster-lever pressing**: push a toaster lever downward;
- **Microwave door closing**: close a horizontally hinged revolute door;
- **Microwave knob rotation**: rotate a microwave knob clockwise;



(a) Distribution of task durations measured by the number of data frames.



(b) Distribution of trajectories over manipulation skills.

Figure 12: Additional statistics of **SMART-Data** on task durations and manipulation skill coverage.

- **Toolbox lid flipping**: grasp the handle of a horizontally placed toolbox and flip the upper lid of the box up to open it.

These eight task types yield twelve evaluation tasks in total, since four of them are evaluated on two robot platforms rather than one. Specifically, bucket handle lifting, toaster-lever pressing, and microwave door closing are evaluated on both the RM75 and the R1Pro platforms, and cabinet-door sliding is evaluated on both the RM75 and the AC1 platforms. Fig. 14 shows all twelve tasks.

In the sim-to-real experiment, we train models on the full 500-episode dataset and directly evaluate them on the real-world tasks. In the performance scaling experiment, we sample 500, 200, 100, 50, 10, and 1 episode from the full 500-episode dataset recursively to construct subdatasets for post-training.

D.2 Training Details

We adopt the model architecture design from PRTS (Zhang et al., 2026b). The architecture consists of a pretrained Qwen3-VL (Bai et al., 2025) 4B backbone and a DiT-based action expert that generates continuous robot actions via flow matching. To isolate the effect of pretraining data and enable a controlled comparison with representative VLA models, we retain the architectural design of PRTS while removing its Contrastive Reinforcement Learning (CRL) objectives. In the first stage, we train the VLM backbone exclusively on *SMART-Data* using a standard cross-entropy objective to predict robot action sequences tokenized with the FAST (Pertsch et al., 2025) tokenizer. In the second stage, we introduce a DiT-based action expert and train the resulting VLA model on the downstream dataset using a flow-matching objective.

We show the hyperparameters used in pre-training and post-training in Tab. 6.

Table 6: Hyperparameters used in pre-training and post-training. For pre-training, we adopt data packing (Zhang et al., 2026b) to form training batches, where the global batch size is estimated according to the average number of samples contained in each data pack.

Hyperparameters	Pre-training	Post-training (Sim2Real)
Global Batch Size	2048	32
VLM Learning Rate	5e-5	1e-5
Action Expert Learning Rate	-	1e-4
Learning Rate Schedule	Cosine Decay	Cosine Decay
Minimum Learning Rate	5e-6	1e-6
Warmup Ratio	0.01	0.01
Training Steps	278k	25k

D.3 Evaluation Protocol Details

Here we explain the metrics we use for evaluation. The success rate is computed as the percentage of successful trials over 15 independent trials for each policy. For action chunk smoothness metrics, we take the AC1 robot as an example to show the definitions of NRMSJ and NMAV. For AC1, the action vector used for these metrics contains the 12 arm degrees of freedom; the two gripper degrees of freedom are excluded. For an action chunk, define

$$A = [\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{T-1}], \qquad \mathbf{a}_t \in \mathbb{R}^D, \qquad T = 50,\ D = 12. \tag{6}$$

To normalize action dimensions with different scales, we apply task-level quantile normalization. Let $q_{0.01,d}$ and $q_{0.99,d}$ denote the 1st and 99th percentiles of the d-th action dimension, respectively. The normalized action is defined as

$$\tilde{a}_{t,d} = 2\frac{a_{t,d} - q_{0.01,d}}{q_{0.99,d} - q_{0.01,d} + \varepsilon_{\mathrm{q}}} - 1, \qquad \varepsilon_{\mathrm{q}} = 10^{-8}. \tag{7}$$

All metrics below are computed using the quantile-normalized actions $\tilde{\mathbf{a}}_t$.

Normalized RMS Joint Jerk (NRMSJ). The third-order finite difference of the normalized action trajectory is defined as

$$j_{t,d} = \Delta^3 \tilde{a}_{t,d} = \tilde{a}_{t+3,d} - 3\tilde{a}_{t+2,d} + 3\tilde{a}_{t+1,d} - \tilde{a}_{t,d}, \qquad t = 0, \ldots, T-4, \quad d = 1, \ldots, D. \tag{8}$$

The normalized RMS joint jerk is computed as

$$\mathrm{NRMSJ}(A) = \sqrt{\frac{1}{(T-3)D} \sum_{t=0}^{T-4} \sum_{d=1}^{D} j_{t,d}^2}. \tag{9}$$

Normalized Mean Action Variation (NMAV). Let

$$\tilde{\mathbf{a}}_t = [\tilde{a}_{t,1}, \ldots, \tilde{a}_{t,D}]^\top \tag{10}$$

denote the quantile-normalized action vector at time step t. The normalized mean action variation is defined as the mean L2 distance between consecutive quantile-normalized action vectors:

$$\mathrm{NMAV}(A) = \frac{1}{T-1} \sum_{t=0}^{T-2} \left\| \tilde{\mathbf{a}}_{t+1} - \tilde{\mathbf{a}}_t \right\|_2. \tag{11}$$

D.4 Simulation Evaluation Details

Fig. 13 shows the post-training convergence of *SMART-VLA* and *SMART-VLA(w/o S1)* on LIBERO under the same 30K-step budget. While the main text reports the final success rates on both benchmarks, this curve isolates the convergence speed: *SMART-VLA* reaches 94.5% average success at 15K steps, whereas *SMART-VLA(w/o S1)* needs the full 30K steps to reach a comparable 94.7%. The two methods therefore converge to a similar final level, but *SMART-VLA* gets there in roughly half the post-training budget, which indicates that pretraining on *SMART-Data* provides a better initialization for downstream simulation tasks.

D.5 Real-world Experiment Details

Fig. 14 shows the twelve real-world evaluation tasks, which are collected on the three robot setups introduced in Fig. 7, namely the dual-arm RM75, the R1Pro, and the ARX AC-1 platforms. Tab. 3 lists the hardware configuration of each platform, and App. B.7 shows the task file format that specifies the objects, the skills, and the success conditions of a task. The post-training demonstrations of all twelve tasks are synthesized by *SMART-Sim*, and App. D.2 reports the corresponding data synthesis and training settings.

Each task is defined by the manipulation process and the condition that counts as a success, as summarized below.

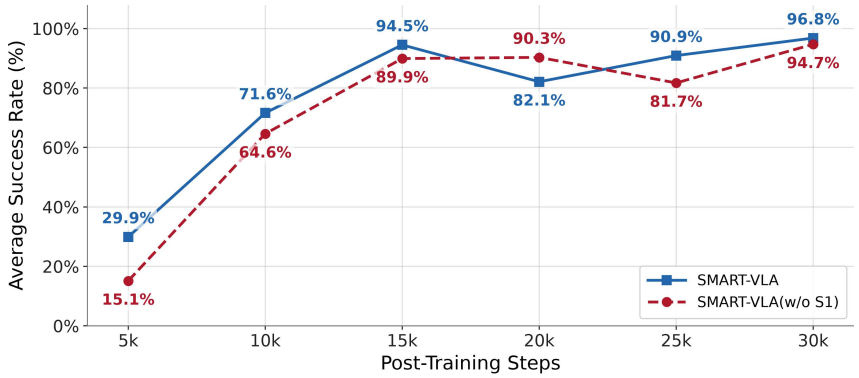
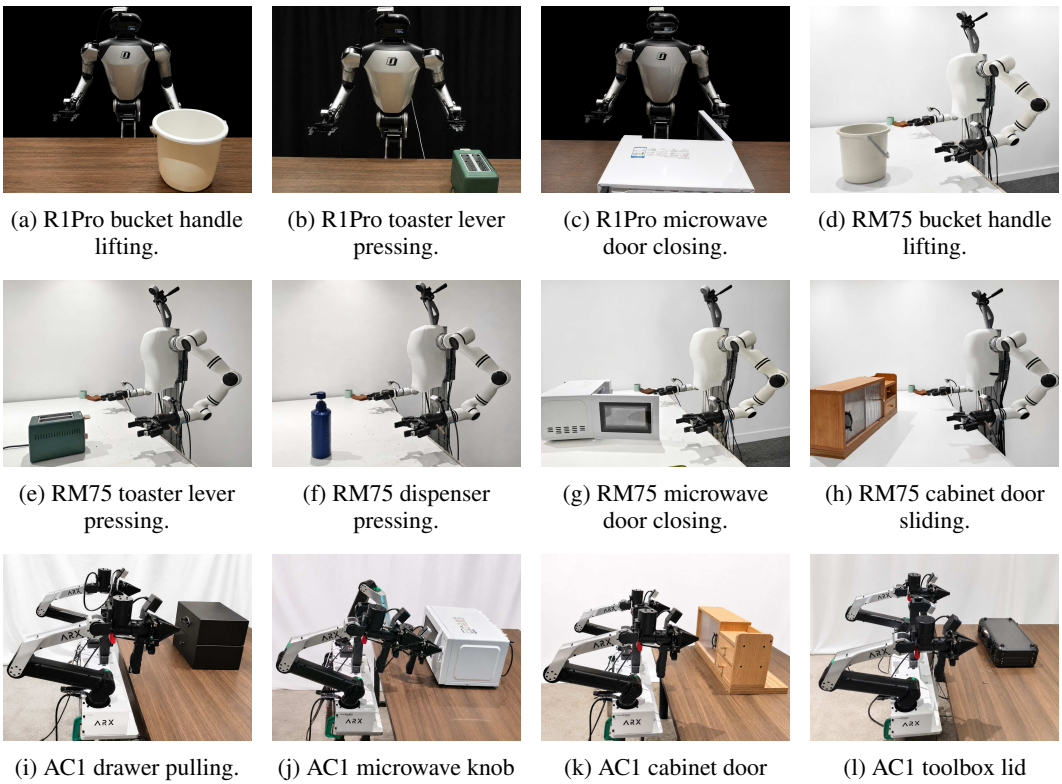


Figure 13: **Post-training performance curve of SMART-VLA and SMART-VLA(w/o S1) on LIBERO.** Average success rates are reported every 5K post-training steps up to the standard 30K-step training budget.



(a) R1Pro bucket handle lifting. (b) R1Pro toaster lever pressing. (c) R1Pro microwave door closing. (d) RM75 bucket handle lifting.

(e) RM75 toaster lever pressing. (f) RM75 dispenser pressing. (g) RM75 microwave door closing. (h) RM75 cabinet door sliding.

(i) AC1 drawer pulling. (j) AC1 microwave knob rotating. (k) AC1 cabinet door sliding. (l) AC1 toolbox lid flipping.

Figure 14: Overview of the twelve real-world articulated-object manipulation tasks.

- **Bucket handle lifting (RM75 and R1Pro)**: the robot's left arm reaches the compliant handle and lifts it above the bucket, and a trial succeeds when the handle is raised at least 30° above the horizontal.
- **Drawer pulling (AC1)**: the robot's left arm grasps the drawer handle and pulls the drawer out of the cabinet, and a trial succeeds when the drawer is extended by at least 20 cm.
- **Dispenser pressing (RM75)**: the robot presses the spring-loaded dispenser button downward, and a trial succeeds when water is dispensed from the container.
- **Cabinet door sliding (RM75 and AC1)**: the robot grasps the round handle on RM75 or the vertical bar handle on AC1 and slides the door to the left, and a trial succeeds when the door is moved by at least 20 cm.

- **Toaster lever pressing (RM75 and R1Pro)**: the robot presses the lever from the top all the way down, and a trial succeeds when the lever reaches the bottom without the gripper losing contact during the motion.
- **Microwave door closing (RM75 and R1Pro)**: the robot's left arm pushes the horizontally hinged door shut from an initial opening angle between 60° and 80°, and a trial succeeds when the door is fully closed and the latch is engaged.
- **Microwave knob rotation (AC1)**: the robot's left arm grasps the knob, which points straight up in the initial state, and rotates it clockwise, and a trial succeeds when the knob has been turned by at least 70°.
- **Toolbox lid flipping (AC1)**: the robot's left arm grasps the handle on the upper lid and flips the lid upward to open the box, and a trial succeeds when the lid opening angle exceeds 70°.

D.6 Evaluation Results of Action Chunk Smoothness

Tab. 7 reports the average NRMSJ and NMAV of the four methods across the three platforms and over all twelve tasks. Lower values indicate smoother and more temporally consistent action chunks, so the overall columns summarize the trajectory quality of each policy. *SMART-VLA* achieves the lowest overall NRMSJ (0.0581) and NMAV (0.0673), which are less than half of the corresponding values for *SMART-VLA(w/o S1)* (0.1592 and 0.1674) and lower than those for *SMART-VLA(Real)* (0.0637 and 0.0721) and $\pi_{0.5}$ (0.0913 and 0.1625). The gap is most pronounced on AC1, where the NMAV of $\pi_{0.5}$ and *SMART-VLA(w/o S1)* reaches 0.2445 and 0.3757, while *SMART-VLA* remains at 0.1046. The remaining per-platform numbers are mixed, for example *SMART-VLA(Real)* obtains a lower NRMSJ on R1Pro and $\pi_{0.5}$ a lower NRMSJ on AC1, but neither reduces the two metrics together as consistently as *SMART-VLA* does. Since *SMART-VLA* is post-trained on the same policy backbone as the other three methods and differs mainly in the pretraining data, the smoother action chunks suggest that the demonstrations synthesized by our motion generation pipeline provide trajectory-level supervision of higher quality for articulated-object manipulation.

Table 7: The average of Normalized Root-Mean-Square Jerk (NRMSJ, normalized using quantiles) and Normalized Mean Action Variation (NMAV, normalized using quantiles) across AC1, RM75, R1Pro, and over all twelve tasks. Lower values (↓) indicate smoother and more consistent trajectories.

	NRMSJ ↓				NMAV ↓			
Method	RM75	AC1	R1Pro	Overall	RM75	AC1	R1Pro	Overall
SMART-VLA	0.0258	0.1006	0.0448	0.0581	0.0416	0.1046	0.0520	0.0673
SMART-VLA(Real)	0.0456	0.1177	0.0157	0.0637	0.0584	0.1209	0.0253	0.0721
SMART-VLA(w/o S1)	0.0357	0.3703	0.0424	0.1592	0.0481	0.3757	0.0486	0.1674
$\pi_{0.5}$	0.1636	0.0705	0.0225	0.0913	0.0868	0.2445	0.1539	0.1625

D.7 Evaluation Under Height Variation

Fig. 15 reports the success rates of *SMART-VLA*, *SMART-VLA(w/o S1)*, and $\pi_{0.5}$ on the four AC1 tasks at a seen height and an unseen higher deployment height. For the AC1 demonstrations, the robot base is placed at the same height as the tabletop of the task environment, whereas in the height-variation experiment the base is raised to 10 cm above the tabletop while all other conditions are kept unchanged. Across the four tasks, *SMART-VLA* keeps at least 60% success rate at the unseen height and shows the smallest average drop, which indicates that the demonstrations synthesized by *SMART-Sim* cover the height variation well enough for the policy to transfer to a higher base.

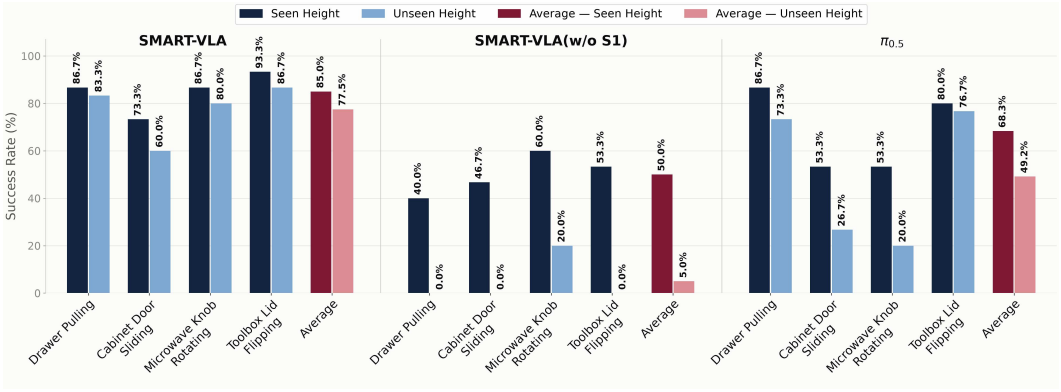


Figure 15: The success rates of **SMART-VLA**, **SMART-VLA(w/o S1)**, and $\pi_{0.5}$ on AC1 tasks at a seen height and an unseen higher deployment height.