FITR: FROM ARTICULATED OBJECT IMAGES TO INTERACTIVE 3D ASSETS FOR REAL-WORLD MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Collecting real-world training data for articulated object manipulation is costly, motivating scalable data generation in simulation. Recent advances in 3D asset generation have improved visual fidelity and geometric quality, but supporting robot interaction also requires appropriate physical dimensions, contact points and directions, and joint dynamics. To this end, we introduce FITR, a pipeline that constructs interactive 3D assets from images of articulated objects for policy learning in simulation and zero-shot deployment on real robots. We construct FITR-Assets by augmenting articulated assets with absolute scale, affordances, and calibrated joint drive parameters, and evaluate annotation quality on FITR-Bench. Using these assets, we combine motion planning with domain randomization to automatically collect approximately 60k manipulation demonstrations in Isaac Sim. Then, we fine-tune vision-language-action (VLA) policies via imitating the demonstrations and RL-driven simulated post-training, enabling zero-shot transfer to the real world. Through systematic ablation studies and comparative experiments in simulation and on real robots, we demonstrate that FITR enables zero-shot sim-to-real transfer for articulated object manipulation. We release FITR-Assets, the simulation demonstration dataset, policy checkpoints, and source code.

1 INTRODUCTION

Simulation environments can generate observation–action trajectories in parallel at low cost and provide privileged information that is difficult to obtain in real systems, such as object poses, contact forces, and task success signals, making them an important source of data for training embodied policies (Ye et al., 2026b). Recent work constructs simulation environments using 3D assets, generates manipulation demonstrations at scale, and transfers policies trained on these demonstrations zero-shot to real robots (Wang et al., 2025; Deshpande et al., 2026). Demonstrations with GPT-6 Astra show that general-purpose models can construct visually detailed 3D scenes[1]. For articulated objects, recent methods recover part geometry and joint kinematic structure from a single image to construct simulation-ready digital twins, providing the assets needed to generate manipulation demonstrations at scale (Wu et al., 2026).

Such advances broaden the possibilities for asset creation, but visual and geometric quality alone do not ensure that generated assets reproduce the physical interactions required for robotic manipulation. Specifically, the required attributes include **(i) absolute scale**—asset geometry is normalized to a canonical space and lacks real-world physical dimensions; **(ii) affordances**—contact points and contact directions that can be used directly for robotic arm motion planning are missing; and **(iii) joint drive parameters**—parameters such as stiffness, damping, and effort limits are absent, making it difficult to reproduce the equivalent interaction responses of real joints. Prior work has begun to augment assets with scale, material, and interaction attributes, while JODA (Gao et al., 2026) further represents conservative forces, dry friction, and damping effects at joints using structured force fields. These efforts provide a foundation for modeling the physical properties of assets, but discrepancies between simulated and real joint responses affect downstream policy transfer.

[1] OpenAI Developers Blog: Architectural visualization with Astra

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.

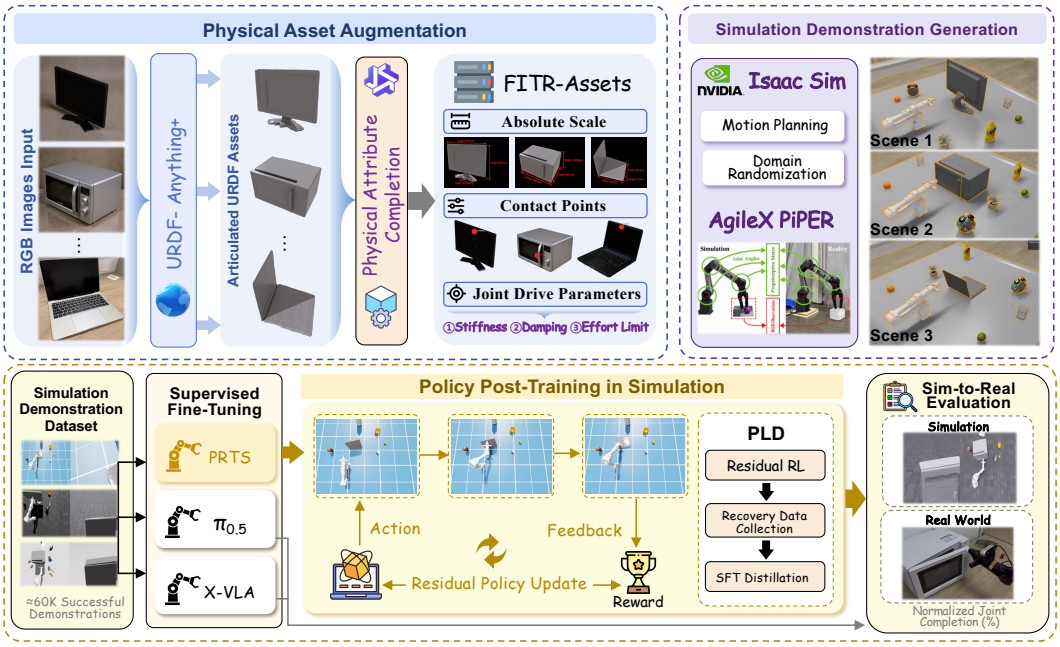


Figure 1: Overview of FITR. **Top left:** Image-generated articulated assets are augmented with absolute scale, contact points, and joint drive parameters to form FITR-Assets. **Top right:** Motion planning and domain randomization in Isaac Sim produce approximately 60k successful demonstrations. **Bottom:** The demonstrations support supervised fine-tuning (SFT) of PRTS, $\pi_{0.5}$, and X-VLA. Simulation-based post-training combines residual reinforcement learning (RL), recovery data collection, and SFT distillation to improve policy performance.

To address these missing attributes, we introduce FITR, a pipeline that augments image-generated articulated assets for robot interaction and connects them to zero-shot real-world policy deployment (Figure 1). Our physical attribute completion combines reference-guided scale recovery, contact point derivation from geometry and joint kinematics, and calibration of equivalent joint drive parameters using measured operating forces. The resulting FITR-Assets provides metric geometry, actionable contacts, and joint drive parameters for manipulation in simulation. Building on these assets, we generate demonstrations through motion planning with domain randomization (Wang et al., 2025; Chen et al., 2026). We use these demonstrations for SFT and subsequently refine the policy by distilling trajectories collected through residual RL in simulation (Xiao et al., 2026). This pipeline supports zero-shot deployment without real-world policy fine-tuning. Our main contributions are as follows:

1. **Physical attribute completion for articulated assets.** We combine reference-guided absolute scale recovery, geometry-based contact point derivation, and calibration of equivalent joint drive parameters to augment generated articulated assets for robotic manipulation. Category-level drive baselines are calibrated against measured operating forces and guide instance-level parameter estimation.

2. **FITR-Assets and simulation demonstration data.** We construct FITR-Assets with 2,772 articulated objects across 12 categories, together with FITR-Bench, a 240-object subset for attribute evaluation. Using the augmented assets, we automatically collect approximately 60k manipulation demonstrations in Isaac Sim.

3. **Validation through asset evaluation and zero-shot real-world manipulation.** We evaluate scale and contact-point accuracy and classification of handle presence on FITR-Bench, and compare three VLA models across six manipulation tasks in simulation and on real robots. These experiments demonstrate the pipeline's ability to support zero-shot sim-to-real transfer without real-world policy fine-tuning.

2 Related Work

2.1 3D Asset Generation and Simulation Data Collection

3D repositories and articulated object datasets support policy training in simulation (Xiang et al., 2020; Deitke et al., 2023; Geng et al., 2023), while a recent survey identifies physical annotation and interaction validity as remaining challenges for embodied 3D generation (Ye et al., 2026a). Recent methods generate articulated geometry and joint structure from visual inputs (Le et al., 2025; Li et al., 2026b; Wu et al., 2026; Zhang et al., 2026a). Physical asset modeling further incorporates scale, material properties, and interaction attributes (Cao et al., 2026a;b;c), while Scalable Real2Sim recovers geometry and inertial parameters through robotic measurements (Pfaff et al., 2025). For joint dynamics, Infinigen-Articulated specifies sampleable physical parameter priors (Joshi et al., 2026), and JODA models conservative forces, dry friction, and damping (Gao et al., 2026). We augment generated assets with absolute scale and calibrate equivalent joint drive parameters using measured operating forces and simulation responses.

Contact prediction methods learn actionable regions and articulation-aware affordances (Mo et al., 2021; Huang et al., 2025), or use predicted articulation flow and joint information to guide manipulation (Eisner et al., 2022; Zhang et al., 2023). We instead derive contact positions analytically from known part geometry, joint kinematics, and moment-arm rules, using a vision-language model (VLM) to provide handle semantics that cannot be inferred from geometry.

Simulation demonstrations can be generated through grasp sampling and motion planning (Wang et al., 2025), model-generated task code and simulation feedback (Hua et al., 2025), or affordance correspondences across objects (Zhang et al., 2026b). Sim2Real-VLA (Zhao et al., 2026) and MolmoB0T (Deshpande et al., 2026) further demonstrate zero-shot real-world deployment using synthetic training data. Domain randomization broadens visual and scene variation to improve sim-to-real transfer (Tobin et al., 2017; Chen et al., 2026). We use kinematics-guided demonstration generation (Wang et al., 2025), with planning branches for different joint types and randomization of camera parameters, object poses, and initial joint configurations to cover variations in articulated object manipulation.

2.2 Reinforcement Learning Post-Training in Simulation

Behavior cloning (BC) can suffer from compounding execution errors outside the demonstration distribution (Ross et al., 2011). Direct RL post-training uses task rewards to improve VLA policies in simulation (Li et al., 2026a; Zang et al., 2026). Alternatively, RLDG uses trajectories from task-specific RL policies to fine-tune generalist policies (Xu et al., 2025). Probe, Learn, Distill (PLD) (Xiao et al., 2026) freezes the base VLA, trains residual policies to explore its failure regions, and distills recovery trajectories into the base policy through SFT. Object-Centric Residual RL deploys simulation-trained residual policies zero-shot on real robots (Kim et al., 2026), whereas Phys2Real combines VLM priors with online adaptation using real-world interactions (Wang et al., 2026). Using PRTS as the base VLA, we adopt PLD's residual learning and data distillation procedure with generated articulated assets and joint-state feedback, keeping demonstration generation and post-training entirely in simulation for zero-shot real-world transfer.

3 Preliminaries

3.1 Articulated Asset Representation

Define a base articulated asset by $\mathcal{A} = \left(\{\mathcal{X}^{(k)}\}_{k=1}^{K},\ \{\mathcal{J}^{(k)}\}_{k=1}^{K}\right)$, where K is the number of rigid links and $\mathcal{X}^{(k)}$ is the mesh geometry of the k-th rigid link. The asset geometry is normalized to the canonical space $[-1,1]^3$ by uniform scaling. The joint between link k and its parent is $\mathcal{J}^{(k)} = \left(\mathbf{o}^{(k)},\ \mathbf{a}^{(k)},\ \tau^{(k)},\ \ell_{\text{low}}^{(k)},\ \ell_{\text{up}}^{(k)}\right)$, where $\tau^{(k)}$ is the joint type, $\mathbf{o}^{(k)} \in \mathbb{R}^3$ is the joint origin, $\mathbf{a}^{(k)} \in \mathbb{S}^2$ is the unit joint axis, and $\ell_{\text{low}}^{(k)}$ and $\ell_{\text{up}}^{(k)}$ are the motion limits. The sources and category statistics of the base assets are provided in Appendix A.

3.2 JOINT-KINEMATICS-BASED MOTION PLANNING FOR ARTICULATED OBJECTS

End-effector (EE) trajectories keep the relative pose between the EE and the manipulated link fixed during construction (Wang et al., 2025). The EE establishes contact with link k at joint configuration q_{init}. Let $T_{\text{eef}}^{\text{init}}$ and $T_{\text{link}}^{(k)}(q_{\text{init}})$ denote the EE pose and the link pose at initial contact, respectively. When the joint is driven to configuration q, the desired EE pose is $T_{\text{eef}}^{(k)}(q) = T_{\text{link}}^{(k)}(q)\big(T_{\text{link}}^{(k)}(q_{\text{init}})\big)^{-1} T_{\text{eef}}^{\text{init}}$. Section 4.2 applies this relation to revolute and prismatic joints.

4 FITR

As shown in Figure 1, FITR connects physical asset augmentation, demonstration generation, and policy post-training for zero-shot deployment on real robots. Starting from images of articulated objects, we obtain base 3D assets with part geometry and joint kinematic structure (Wu et al., 2026). Physical augmentation combines reference-based scale recovery, geometric contact point derivation, and calibration of equivalent joint drive parameters against measured operating forces to construct FITR-Assets (Section 4.1). Using these assets, joint-type-aware motion planning with online inverse kinematics (IK) tracking and multidimensional domain randomization generates demonstrations for VLA SFT (Section 4.2). We then train residual experts with joint-state feedback from the same assets to recover from execution deviations, collect recovery trajectories, and distill them into the base VLA (Xiao et al., 2026) (Section 4.3). Both demonstration generation and policy post-training remain in simulation, enabling deployment without real-world fine-tuning.

4.1 PHYSICAL AUGMENTATION OF ARTICULATED ASSETS

The base asset $\mathcal{A}$ is mapped to an augmented asset:

$$\Phi:\ \mathcal{A} \longmapsto \mathcal{A}^{+}, \qquad \mathcal{A}^{+} = \Big(s,\ \{\mathcal{X}^{(k)}\}_{k=1}^{K},\ \{\mathcal{J}^{+(k)}\}_{k=1}^{K},\ \mathcal{P}\Big). \tag{1}$$

Its output augments $\mathcal{A}$ with three types of physical quantities: absolute scale $s \in \mathbb{R}_{>0}$, a set of actionable contact points $\mathcal{P} = \{\mathbf{p}_c^{(m)}\}_{m=1}^{M}$, and drive dynamics parameters for each movable joint. The specification of each movable joint is augmented from $\mathcal{J}^{(k)}$ to $\mathcal{J}^{+(k)} = \big(\mathbf{o}^{(k)}, \mathbf{a}^{(k)}, \tau^{(k)}, \ell_{\text{low}}^{(k)}, \ell_{\text{up}}^{(k)}, k_p^{(k)}, k_d^{(k)}, e_{\max}^{(k)}\big)$, where $k_p^{(k)}$, $k_d^{(k)}$, and $e_{\max}^{(k)}$ are the drive parameters of the k-th joint. The operator Φ comprises three steps: absolute scale recovery Φ_{scale}, contact point derivation Φ_{aff}, and drive dynamics calibration Φ_{dyn}. Through these three steps, we construct FITR-Assets. Inspired by the attribute-level evaluation in PhysX-Bench, we select assets from each category of FITR-Assets and perform manual measurement and verification to construct FITR-Bench, an attribute evaluation subset for assessing the accuracy of absolute scale, affordance, and semantic annotations (Figure 2).

4.1.1 ABSOLUTE SCALE RECOVERY Φ_{SCALE}

Geometry normalized to $[-1, 1]^3$ does not determine absolute dimensions. Φ_{scale} recovers a metric scale by combining a VLM's dimensional priors with a reference asset from the same category (Cao et al., 2026b;c). Given rendered images of the reference and target, their category labels, and the reference dimensions, Qwen3-VL estimates the target's longest bounding-box edge. Converting this estimate to meters, $\hat{L}_{\text{real}}$, gives $s = \hat{L}_{\text{real}}/\hat{L}_{\text{norm}}$, where $\hat{L}_{\text{norm}}$ is the corresponding normalized length and $s \in \mathbb{R}_{>0}$ is measured in meters per normalized coordinate unit. All link meshes, joint origins, and prismatic joint motion limits are uniformly transformed using the same scale factor, while the angular limits of revolute joints remain unchanged. The full prompt is provided in Appendix B.

4.1.2 CONTACT POINT DERIVATION Φ_{AFF}

For downstream motion planning, Φ_{aff} derives at least one contact point $\mathbf{p}_c$ and its contact direction per movable joint from the part meshes $\mathcal{X}^{(k)}$ and joint parameters $\mathcal{J}^{(k)}$, using separate rules for each joint type:

(1) Revolute joints ($\tau^{(k)} = \texttt{revolute}$). We select the point on the part's candidate surface $\mathcal{S}^{(k)}$ with the largest moment arm about the joint axis $\mathbf{a}^{(k)}$ passing through the joint origin $\mathbf{o}^{(k)}$:

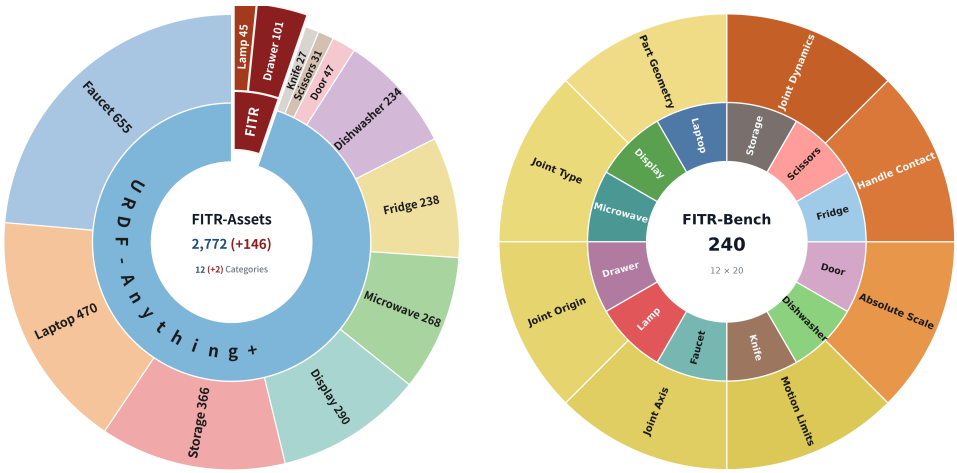


Figure 2: Composition of FITR-Assets and FITR-Bench. (a) FITR-Assets contains 2,772 assets across 12 categories, including 146 additional drawer and lamp assets. (b) FITR-Bench contains 240 assets, with 20 samples per category.

$\mathbf{p}_c^{(k)} = \arg\max_{\mathbf{x}\in\mathcal{S}^{(k)}} \left\|(\mathbf{x}-\mathbf{o}^{(k)}) - \left((\mathbf{x}-\mathbf{o}^{(k)})\cdot\mathbf{a}^{(k)}\right)\mathbf{a}^{(k)}\right\|$, where $\mathbf{x}$ ranges over this surface, and the candidate set $\mathcal{S}^{(k)} \subseteq \partial\mathcal{X}^{(k)}$ is the candidate contact region on the part surface. The contact direction is the outward surface normal at the selected point, so that the front of the gripper aligns with the surface.

(2) Prismatic joints ($\tau^{(k)} = \texttt{prismatic}$). Prismatic joints move along a fixed axis, so contact point selection primarily considers accessibility for manipulation along this axis. We therefore approximate a candidate contact position on the front side using the link's center of mass and its extent along the joint axis $\mathbf{a}^{(k)}$: $\mathbf{p}_c^{(k)} = \mathbf{c}^{(k)} + \frac{1}{2}\, r^{(k)}\, \mathbf{a}^{(k)}$, where $\mathbf{c}^{(k)}$ is the link's center of mass and $r^{(k)}$ is its extent along $\mathbf{a}^{(k)}$. The contact direction is set to the same direction as the joint axis $\mathbf{a}^{(k)}$.

The geometric candidates are validated by executing the planned contact in simulation and retaining successful trajectories. Stored contacts are mapped to $\mathbf{p}_c^{\text{world}}$ at runtime; visualizations are provided in Appendix F.

Geometry does not determine whether a protruding grasp handle is present. A VLM therefore predicts this attribute for each movable joint (Huang et al., 2025). Contact points and handle attributes are manually verified and stored in FITR-Assets; verified FITR-Bench annotations serve as evaluation labels.

4.1.3 Drive Dynamics Calibration Φ_{DYN}

We calibrate equivalent joint drive parameters by matching simulated operating forces to measurements from representative physical objects in each category. These calibrated parameters define category-level baselines for stiffness, damping, and effort limits.

For assets without physical measurements, Qwen3-VL uses calibrated references, multi-view renderings, absolute scale, movable-part geometry, and joint structure to infer parameter adjustments and confidence scores. We check the predictions for nonnegativity, unit consistency, and category-specific bounds, reverting to the category baseline when confidence is low or the parameters are invalid. Accepted parameters are written into $\mathcal{J}^+$. Calibration and parameter completion details are provided in Appendix C.

4.2 Demonstration Generation in Simulation

We generate demonstrations in two stages: the EE reaches $\mathbf{p}_c^{\text{world}}$ and then drives the joint to q^g under the relative pose constraint in Section 3.2, with online differential IK in Isaac Sim. The link pose $T_{\text{link}}^{(k)}(q)$ depends on the joint type:

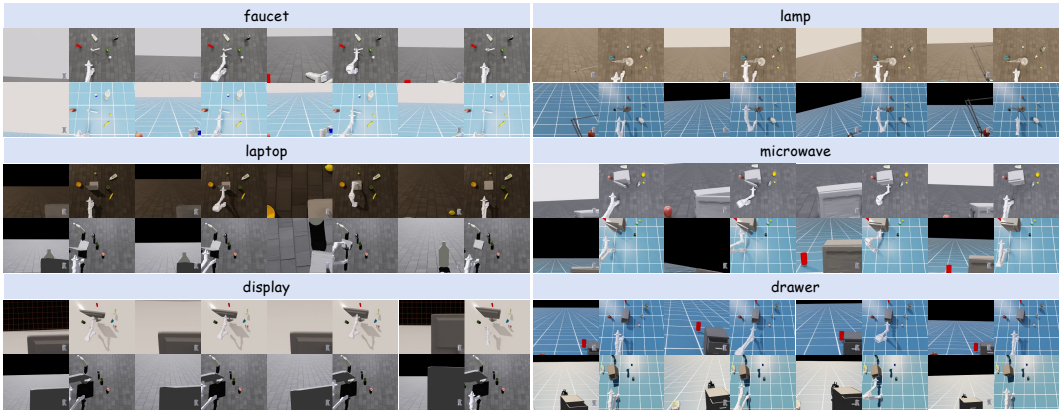


Figure 3: Examples of automatically generated simulation demonstrations for six manipulation tasks. Each task shows two demonstrations under different domain-randomized conditions, with each row depicting four successive time steps from left to right. At each time step, the left and right images show observations from the wrist and external cameras, respectively, forming the two visual inputs to the policy. The two demonstrations illustrate variations in lighting, camera viewpoints, and object placement.

(1) Revolute joints ($\tau^{(k)} = \texttt{revolute}$). Rotation about $(\mathbf{o}^{(k)}, \mathbf{a}^{(k)})$ yields an EE trajectory that traces a circular arc in a plane perpendicular to $\mathbf{a}^{(k)}$, with radius equal to the EE's distance from the axis.

(2) Prismatic joints ($\tau^{(k)} = \texttt{prismatic}$). The link translates along $\mathbf{a}^{(k)}$ by $q - q_{\text{init}}$, yielding a straight-line EE trajectory with fixed orientation: $\mathbf{p}_{\text{eef}}^{(k)}(q) = \mathbf{p}_{\text{eef}}^{\text{init}} + (q - q_{\text{init}})\mathbf{a}^{(k)}$ and $R_{\text{eef}}^{(k)}(q) = R_{\text{eef}}^{\text{init}}$.

A demonstration succeeds when the target joint reaches q^g. Successful episodes (Figure 3) are stored for policy training.

We apply domain randomization to each demonstration, varying scene clutter, background textures, and lighting, together with **(i) camera parameters**: main-camera extrinsics and focal length; **(ii) object poses**: translation and rotation about the Z-axis within the workspace; and **(iii) initial joint configurations**: q_{init} sampled from the valid range of motion (Chen et al., 2026). The randomization ranges are provided in Appendix D.

4.3 Policy Post-Training in Simulation for Zero-Shot Transfer

The base VLA is obtained by SFT on PRTS using the demonstrations from Section 4.2, and its parameters are frozen during residual learning. At each step, the first predicted action is executed with a bounded residual:

$$a_t = a_t^b + \Delta a_t, \qquad \Delta a_t = \mathbf{b} \odot u_t, \qquad u_t \sim \pi_\theta(\cdot \mid x_t), \tag{2}$$

where a_t^b is the base action and $\mathbf{b}$ is the residual scaling vector. The residual policy takes $x_t = (o_t, a_t^b, \eta_t)$ as input, where o_t contains visual observations, proprioceptive states, and language instructions, and η_t is the normalized remaining execution budget. The residual corrects only the robotic arm's actions.

One residual expert is trained per task in simulation. The base policy runs independently for a randomly sampled number of steps T_{probe}, after which the residual expert learns to recover from the encountered states. After training, the frozen base policy and residual experts generate hybrid trajectories containing initial attempts and recovery behaviors. The executed actions from successful trajectories are used as SFT targets; the resulting distilled policy is deployed alone. Implementation details are provided in Appendix E.

Using the joint states of the augmented assets, we apply the success criterion in Section 4.2 to residual learning and trajectory filtering. Define the reward by $r_t = \mathbf{1}\big[\big|q^{\text{obj}}(s_{t+1}) - q^g\big| \le \varepsilon_g\big]$, where s_t is the simulation state, $q^{\text{obj}}(s_{t+1})$ is the target object's joint configuration at s_{t+1}, and q^g

is the same target joint configuration as in motion planning. Within the prescribed number of steps, task completion is recorded as reward 1; otherwise the reward is 0.

5 EXPERIMENTS

Our experiments address four questions: **Q1:** Does physical augmentation improve asset annotation accuracy? **Q2:** Can policies trained on simulation demonstrations transfer zero-shot to real robots? **Q3:** How do VLA backbones differ in task performance and execution behavior? **Q4:** How do domain randomization and simulation-based RL post-training affect manipulation performance?

5.1 EXPERIMENTAL SETUP

Tasks. Using FITR-Assets, we design six manipulation tasks: laptop, display, microwave, drawer, lamp, and faucet manipulation; the first four are standard tasks, and the last two are challenging.

Policies. We evaluate $\pi_{0.5}$, X-VLA, and PRTS (Black et al., 2025; Zheng et al., 2026; Zhang et al., 2026c), using each model's SFT result as its BC baseline. For the main comparison, all models are trained for 30k steps on fully randomized FITR data and deployed zero-shot.

Metrics. On FITR-Bench, we measure absolute-scale error with symmetric mean absolute percentage error (sMAPE, %), contact point error with mean absolute error (MAE, cm), and handle classification accuracy (%). Manipulation is evaluated by Normalized Joint Completion (NJC) for both joint types and closing tasks (Wang et al., 2025): $C_r = \text{clip}\left(1 - \frac{|q_r^T - q_r^g|}{|q_r^0 - q_r^g|}, 0, 1\right)$, where q_r^0, q_r^g, and q_r^T are the target joint's initial, target, and terminal positions in rollout r. We report mean NJC (%) over 50 simulation rollouts or 20 real-world trials per condition. We conduct a separate set of evaluation trials for each comparison. Task, training, and evaluation details are provided in Appendices D and E; ablation budgets are specified in Section 5.4.1.

Table 1: Attribute augmentation results on FITR-Bench. Bold indicates lower errors. Dashes denote unreported handle accuracy; its mean is computed over the five reported categories.

	Absolute scale sMAPE (%) ↓		Contact point MAE (cm) ↓		Handle accuracy (%) ↑
Category	VLM	VLM + scale references (Ours)	VLM	Geometry (Ours)	`has_handle`
Laptop	17.21	**10.94**	11.90	**0.66**	—
Display	10.13	**6.45**	15.89	**0.84**	—
Microwave	5.45	**2.96**	23.80	**1.07**	95.10
Drawer	57.80	**18.10**	5.01	**0.94**	—
Lamp	4.51	**3.92**	9.76	**1.17**	—
Faucet	29.98	**10.81**	5.43	**0.48**	—
Knife	9.43	**9.13**	4.54	**0.33**	—
Dishwasher	30.69	**5.51**	36.66	**0.92**	94.30
Door	65.63	**2.05**	74.48	**5.54**	97.50
Refrigerator	31.30	**4.22**	49.47	**4.33**	97.10
Scissors	18.66	**8.87**	7.64	**0.31**	—
Storage cabinet	52.19	**12.16**	12.82	**1.21**	93.80
Mean	27.75	**7.93**	21.45	**1.48**	95.56

5.2 ASSET AUGMENTATION QUALITY

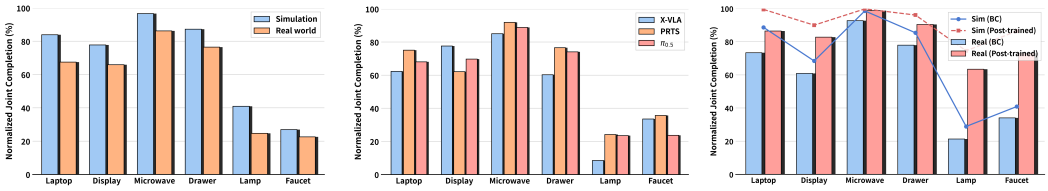
Table 1 reports attribute accuracy on FITR-Bench. Direct VLM estimation yields a mean absolute-scale sMAPE of 27.75%; introducing scale references derived from physical measurements reduces the error to 7.93%, a relative reduction of 71.4%. The largest absolute reductions occur for doors, storage cabinets, and drawers, supporting the use of scale references to reduce scale estimation errors.

For affordances, direct VLM prediction yields a mean contact-point MAE of 21.45 cm, while FITR's geometric derivation reduces it to 1.48 cm, a relative reduction of 93.1%. For categories with handle annotations, the mean classification accuracy of `has_handle` is 95.56%. These results show that, when joint axes and movable-part geometry are known, explicit kinematic constraints can provide more accurate interaction positions than direct visual inference.

5.3 Manipulation Policies and Sim-to-Real Transfer

In Figure 4(a), mean NJC decreases from 68.95% in simulation to 57.25% on real robots, a gap of 11.70 percentage points. The policy achieves nonzero progress on all six tasks without real-world fine-tuning, but lamp and faucet manipulation remain difficult in both environments.

In Figure 4(b), PRTS achieves the highest mean NJC of 60.93%, followed by $\pi_{0.5}$ at 57.99% and X-VLA at 54.55%. PRTS leads on five tasks, while X-VLA performs best on display manipulation, indicating that the overall ranking does not hold uniformly across tasks.



(a) Sim-to-Real Transfer (b) Real-World VLA Comparison (c) Effect of RL Post-Training

Figure 4: Manipulation performance measured by NJC (%, higher is better). (a) $\pi_{0.5}$ in simulation and the real world. (b) Three VLA backbones on real robots. (c) PRTS before and after RL post-training; lines denote simulation and bars denote real-world evaluation.

5.4 Ablation Studies

5.4.1 Multidimensional Domain Randomization

We compare $\pi_{0.5}$ trained with full randomization (approximately 10k demonstrations per task, 30k training steps) and without randomization (300 demonstrations per task, 5k steps), keeping other training settings unchanged. As shown in Table 2, full randomization yields higher NJC on all six tasks in both environments: mean NJC increases from 19.38% to 68.78% in simulation and from 13.08% to 57.68% on real robots. These gains reflect the combined effects of randomization, dataset size, and training budget.

Table 2: Randomization ablation for $\pi_{0.5}$, measured by mean NJC (%, ↑). Full/None denote full/no randomization; bold indicates higher NJC for each task within each environment.

		Standard tasks				Challenging tasks	
Environment	Randomization	Laptop	Display	Microwave	Drawer	Lamp	Faucet
Simulation	Full	**81.7**	**76.3**	**95.3**	**83.2**	**45.7**	**30.5**
	None	40.6	21.9	22.7	15.4	6.8	8.9
Real world	Full	**68.3**	**66.8**	**89.4**	**75.3**	**20.4**	**25.9**
	None	25.5	19.9	18.8	8.9	1.9	3.5

5.4.2 Reinforcement Learning Post-Training in Simulation

In Figure 4(c), residual RL and SFT distillation improve PRTS on all six tasks, raising mean NJC from 68.48% to 91.05% in simulation and from 60.05% to 82.50% on real robots, corresponding to gains of 22.57 and 22.45 percentage points, respectively. The largest real-world gains occur on lamp and faucet manipulation: NJC increases from 21.3% to 63.4% and from 34.1% to 73.1%,

respectively, yielding improvements of 42.1 and 39.0 percentage points. The combined post-training procedure thus provides the largest absolute gains on tasks where the BC baseline performs poorly. These results support the transfer of simulation-based post-training gains without real-world fine-tuning, although NJC on these two tasks remains lower than on the standard tasks.



Figure 5: Representative rollout snapshots for six articulated object manipulation tasks in simulation and the real world. **Top:** Simulated rollouts in Isaac Sim. **Bottom:** Real-world rollouts of policies trained in simulation and deployed without real-world fine-tuning. Each column shows the corresponding task in both environments, illustrating robot–object interactions during execution.

5.5 Experimental Analysis

Physical augmentation improves scale and contact-point accuracy and provides accurate handle annotations, while simulation-trained policies make progress on real-world tasks without real-world fine-tuning. PRTS achieves the highest mean NJC, whereas X-VLA performs best on display manipulation. The fully randomized training setting improves performance on all tasks through combined changes in randomization, dataset size, and training budget. Residual RL and SFT distillation further improve completion in both environments, with the largest gains on challenging tasks.

Figure 5 shows representative rollouts in Isaac Sim and the real world. In real-world trials, differences in lighting, materials, and color complicate visual sim-to-real transfer. Policies sometimes miss effective contact regions, apply force in unsuitable directions, or stop at intermediate object joint positions. Unseen objects, initial configurations, and scene changes can also cause trajectory deviations, motivating training that covers diverse appearances and states alongside correction and recovery behaviors. X-VLA generally produces smoother action sequences than $\pi_{0.5}$, but performs less consistently in target identification, instruction following, and recovery after deviations. These observations highlight the importance of effective contact and closed-loop correction alongside action smoothness.

6 Conclusion

We present FITR, which combines physical augmentation of image-generated articulated assets, simulation demonstration generation, and policy post-training for zero-shot real-world manipulation. Experiments demonstrate improved scale and contact point accuracy and accurate handle annotations. Fully randomized training improves manipulation across all six tasks, while residual RL and distillation further improve the BC baseline in simulation and the real world, without real-world policy training or fine-tuning.

Limitations and future work. The current method uses category-level equivalent drive parameters calibrated from measured operating forces, rather than identifying each instance's internal dynamics; real-world evaluation covers only six single-object articulated tasks. Future work will expand the asset and task range, study instance-level physical calibration, and test distilled VLA policies on more complex articulations.

AI USE STATEMENT

In this work, we used generative AI tools to assist with editing the manuscript for grammar, clarity, and readability, to consult related literature and conference formatting conventions, and to assist with editing and debugging research code.

The overall system architecture, engineering design, and algorithmic ideas were determined by the authors; AI-assisted code was reviewed, tested, and verified by the authors against the actual implementation and experimental results.

We have not used generative AI tools to generate datasets, develop theoretical models, formulate or prove mathematical claims, design the methodology or experiments, translate the paper, or interpret results.

All AI-assisted text was reviewed and verified by the authors, and the methodological details, claims, experimental settings, and reported results were independently checked against the actual implementation and experiments. We take responsibility for the final content of this work, including all text, claims, and other artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken the following steps to support reproducibility. The asset-augmentation procedure, including scale recovery, contact derivation, and joint-drive calibration, is described in Section 4.1, with category baselines, the VLM prompt, and the measurement protocol in Appendices A–C. Task definitions, language prompts, domain-randomization ranges, and demonstration filtering are specified in Appendix D. Observation–action interfaces, SFT configurations, residual-RL execution, and the evaluation protocol are specified in Section 5.1 and Appendix E. Contact-point examples are provided in Appendix F. We will include anonymous source code and configuration files with the supplementary material to reproduce the simulation pipeline, training, and evaluation.

REFERENCES

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025.

Ziang Cao, Zhaoxi Chen, Liang Pan, and Ziwei Liu. Physx-3d: Physical-grounded 3d asset generation. *Advances in Neural Information Processing Systems*, 38:93771–93784, 2026a.

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

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

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A Scalable Data Generator and Benchmark with Strong Domain Randomization for Robust Bimanual Robotic Manipulation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13142–13153, 2023.

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. In *10th Annual Conference on Robot Learning*, 2026. Accepted for publication.

Ben Eisner, Harry Zhang, and David Held. FlowBot3D: Learning 3D Articulation Flow to Manipulate Articulated Objects. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi: 10.15607/RSS.2022.XVIII.018.

Tianhong Gao, Cheng Yu, Yinghao Xu, and Mengyu Chu. Joda: Composable joint dynamics for articulated objects. *arXiv preprint arXiv:2605.09954*, 2026.

Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7081–7091. IEEE, 2023.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018.

Pu Hua, Minghuan Liu, Annabella Macaluso, Yunfeng Lin, Weinan Zhang, Huazhe Xu, and Lirui Wang. GenSim2: Scaling Robot Data Generation with Multi-modal and Reasoning LLMs. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 5030–5066. PMLR, 06–09 Nov 2025.

Siyuan Huang, Haonan Chang, Yuhan Liu, Yimeng Zhu, Hao Dong, Abdeslam Boularias, Peng Gao, and Hongsheng Li. A3VLM: Actionable Articulation-Aware Vision Language Model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 1675–1690. PMLR, 06–09 Nov 2025.

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. In *10th Annual Conference on Robot Learning*, 2026. Accepted for publication.

Kinam Kim, Namiko Saito, Heecheol Kim, Katsushi Ikeuchi, Jaegul Choo, and Yasuyuki Matsushita. Object-Centric Residual RL for Zero-Shot Sim-to-Real VLA Enhancement. In *10th Annual Conference on Robot Learning*, 2026. Accepted for publication.

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

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Yang Zhaohui, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. Simplevla-rl: Scaling vla training via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 125846–125869, 2026a.

Zhe Li, Xiang Bai, Jieyu Zhang, Zhuangzhe Wu, Che Xu, Ying Li, Chengkai Hou, and Shanghang Zhang. Urdf-anything: Constructing articulated objects with 3d multimodal language model. *Advances in Neural Information Processing Systems*, 38:94974–95002, 2026b.

Kevin M Lynch and Frank C Park. Modern robotics. *Mechanics, Planning, and Control*, 2017.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Munoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

Kaichun Mo, Leonidas J Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6813–6823, 2021.

Nicholas Pfaff, Evelyn Fu, Jeremy Binagia, Phillip Isola, and Russ Tedrake. Scalable real2sim: Physics-aware asset generation via robotic pick-and-place setups. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6296–6303. IEEE, 2025.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Maggie Wang, Stephen Tian, Aiden Swann, Ola Shorinwa, Jiajun Wu, and Mac Schwager. Phys2Real: Fusing VLM Priors with Interactive Online Adaptation for Uncertainty-Aware Sim-to-Real Manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

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 *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.156.

Zhuangzhe Wu, Yue Xin, Chengkai Hou, Minghao Chen, Yaoxu Lyu, Jieyu Zhang, and Shanghang Zhang. Urdf-anything+: End-to-end generation for simulation-ready articulated assets. *arXiv preprint arXiv:2603.14010*, 2026.

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.

Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Zhengyi Luo, Yuqi Xie, Fengyuan Hu, Jim Fan, Guanya Shi, et al. Self-improving vision-language-action models with data generation via residual rl. In *International Conference on Learning Representations*, volume 2026, pp. 13209–13236, 2026.

Charles Xu, Qiyang Li, Jianlan Luo, and Sergey Levine. RLDG: Robotic Generalist Policy Distillation via Reinforcement Learning. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.028.

Tianwei Ye, Yifan Mao, Minwen Liao, Jian Liu, Chunchao Guo, Dazhao Du, Quanxin Shou, Fangqi Zhu, and Song Guo. 3d generation for embodied ai and robotic simulation: A survey. *arXiv preprint arXiv:2604.26509*, 2026a.

Yifan Ye, Yankai Fu, Yaoxu Lv, Bohan Hou, Jun Cen, Lingdong Kong, Duo Zheng, Tianxing Chen, Jiaming Liu, Ziang Cao, et al. Data pyramid for embodied manipulation: A survey. *arXiv preprint arXiv:2607.24744*, 2026b.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Peihong Wang, Hua Yuan, Yixian Zhang, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, Zhihao Liu, Kang Chen, Wenhao Tang, Quanlu Zhang, Weinan Zhang, Chao Yu, and Yu Wang. RLux-VLA: A Unified and Efficient Framework for Reinforcement Learning of Vision-Language-Action Models. In *Proceedings of Robotics: Science and Systems*, 2026.

Chuanrui Zhang, Minghan Qin, Yuang Wang, Baifeng Xie, Hang Li, and Ziwei Wang. SimArt: Decomposing Monolithic Meshes into Sim-ready Articulated Assets via MLLM. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12. ACM, 2026a. doi: 10.1145/3799902.3811173.

Harry Zhang, Ben Eisner, and David Held. FlowBot++: Learning Generalized Articulated Objects Manipulation via Articulation Projection. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1222–1241. PMLR, 06–09 Nov 2023.

Jiawei Zhang, Kaizhe Hu, Yingqian Huang, Yuanchen Ju, Zhengrong Xue, and Huazhe Xu. Affordgen: Generating diverse demonstrations for generalizable object manipulation with affordance correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15966–15975, 2026b.

Yang Zhang, Jiangyuan Zhao, Chenyou Fan, Fangzheng Yan, Tian Li, Haitong Tang, Sen Fu, Xuan'er Wu, Qizhen Weng, Weinan Zhang, et al. Prts: A primitive reasoning and tasking system via contrastive representations. *arXiv preprint arXiv:2604.27472*, 2026c.

Runyi Zhao, Sheng Xu, Ruixing Jin, Yueci Deng, Yunxin Tai, Kui Jia, and Guiliang Liu. Sim2real vla: Zero-shot generalization of synthesized skills to realistic manipulation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026.

A FITR-ASSETS STATISTICS AND CATEGORY BASELINES

Of the 2,772 assets in FITR-Assets, 2,626 across 10 categories come from URDF-Anything+, and two-tier drawers and lamps from PartNet-Mobility (Xiang et al., 2020). Operating-force measurements set category-level drive baselines and are not used as batch evaluation metrics. Table 3 lists the scale and drive baselines.

Table 3: Category-specific scale and drive dynamics baseline parameters.

Asset category	Scale s (m/normalized unit)	Damping k_d	Effort limit $e_{\max}$ (N/N·m)	Stiffness k_p	Count
Laptop	0.15	90	0.8	0	470
Display	0.275	1500	2.4	0	290
Microwave	0.21	1	0.01	0	268
Drawer	0.163	30	1	0	101
Lamp	0.4	120	1.5	0	45
Faucet	0.175	50	0.04	0	655
Knife	0.075	10	0.3	50	27
Dishwasher	0.298	500	7.5	0	234
Door	1.075	70	1.8	0/100	47
Refrigerator	0.885	60	0.675	0	238
Scissors	0.085	5	0.05	0	31
Storage cabinet	0.235	8	3	0	366

The scale baselines in Table 3 map the canonical space $[-1, 1]^3$ to physical space in meters. For example, a laptop asset with a normalized longest edge of 2 and $s = 0.15$ has a real-world longest edge of 0.30 m. The remaining values are category calibration baselines rather than values shared by every instance; for doors, $k_p = 0/100$ denotes free hinges and elastically returning joints, respectively.

B ABSOLUTE SCALE ANNOTATION AND VERIFICATION PROTOCOL

The VLM receives rendered images of same-category reference and target assets together with the reference dimensions. We call Qwen3-VL-Flash (`qwen3-vl-flash`) via DashScope and require JSON only; `longest_edge_cm` is the target's longest bounding-box edge in centimeters.

B.1 VLM PROMPT TEMPLATE

```
[SYSTEM]
You analyze rendered images of articulated objects. Estimate the longest
↪ bounding-box edge of the whole object in cm. For handles, only mark
↪ has_handle true when a clearly protruding grasp handle is visible.
↪ Reply with JSON only.

[USER]
Category: {category}
Target asset ID: {asset_id}

Scale references (same category; known longest edge in cm):
Image 1: asset {ref_asset_id_1}, longest edge = {L_1} cm
Image 2: asset {ref_asset_id_2}, longest edge = {L_2} cm
...
Target (Image {N+1}): asset {asset_id}
Compare the target to the references and estimate its longest edge in cm.

Movable joints:
- {joint_name} / {link_name} ({motion_type})

For each joint, set has_handle true only when you see a clearly
↪ protruding grasp handle on that movable part.
```

```
Reply with JSON only. No markdown:
{"longest_edge_cm": <float>, "has_handle": {<joint_name>: <boolean>},
 ↪  "door_type": <"normal"|"self_closing"|null>, "confidence": <float>}

- longest_edge_cm: longest bounding-box edge of the target (Image {N+1})
 ↪  in cm.
- has_handle: boolean per joint listed above, or {} if none.
- confidence: 0.0-1.0.
```

After receiving the model output, we divide `longest_edge_cm` by 100 to obtain the length in meters, $\hat{L}_{\text{real}}$, and compute $s = \hat{L}_{\text{real}}/\hat{L}_{\text{norm}}$ using the target asset's longest edge in the canonical space.

B.2 Benchmark Annotations and Evaluation Metrics

On FITR-Bench, the ground-truth annotation L_i^* is the longest bounding-box edge of the corresponding physical object. Absolute scale estimation is evaluated with sMAPE (Cao et al., 2026c):

$$\text{sMAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{L}_i - L_i^*|}{(|\hat{L}_i| + |L_i^*|)/2} \times 100\%, \tag{3}$$

where $\hat{L}_i$ is the estimated longest edge of the i-th asset.

C Joint Drive Parameters and Calibration

C.1 Implicit Joint Drive Control Law

Both data collection and policy training are conducted in Isaac Sim, where the simulator drives the movable joints of articulated objects using a proportional–derivative (PD) control law (Mittal et al., 2025). For the k-th joint, the drive output is

$$\begin{aligned} e_{\text{act}}^{(k)} = \text{clip}\Big[& k_p^{(k)}\big(q_{\text{tgt}}^{(k)} - q^{(k)}\big) \\ & + k_d^{(k)}\big(\dot{q}_{\text{tgt}}^{(k)} - \dot{q}^{(k)}\big), -e_{\max}^{(k)}, e_{\max}^{(k)}\Big], \end{aligned} \tag{4}$$

where $q^{(k)}$ and $\dot{q}^{(k)}$ are the current joint position and velocity, $q_{\text{tgt}}^{(k)}$ and $\dot{q}_{\text{tgt}}^{(k)}$ are the drive targets, and $k_p^{(k)}$, $k_d^{(k)}$, and $e_{\max}^{(k)}$ are stiffness, damping, and the effort limit. The drive target $q_{\text{tgt}}^{(k)}$ need not equal the task target q^g.

C.2 Parameter Mapping and Operating Force Conversion

OpenUSD's `UsdPhysicsDriveAPI` models a joint drive as an implicit spring–damper with a maximum output limit. In Isaac Lab, `stiffness` and `damping` are written to the physics engine, and `effort_limit_sim` caps the solver effort.

Parameter	OpenUSD/Isaac Lab field	Revolute joint unit	Prismatic joint unit
Drive stiffness k_p	`stiffness`	N·m/rad	N/m
Drive damping k_d	`damping`	N·m·s/rad	N·s/m
Effort limit $e_{\max}$	`maxForce` / `effort_limit_sim`	N·m	N

Drive stiffness k_p is the position P-gain; drive damping k_d is the velocity D-gain and resists motion when the target velocity is zero; and $e_{\max}$ is the maximum joint torque (revolute) or axial force (prismatic). The solver clips the combined stiffness–damping output, so $e_{\max}$ is not the EE operating force.

For a revolute joint, treating the measured operating force as a concentrated force at the contact point, the torque along the joint axis is $\tau_{\text{joint}} = \mathbf{a}^\top(\mathbf{r} \times \mathbf{F})$, where $\mathbf{a}$ is the unit joint axis, $\mathbf{r}$ is the

vector from a point on the axis to the contact point, and $\mathbf{F}$ is the force acting on the manipulated link, all expressed in the same coordinate frame (Lynch & Park, 2017). When the operating force acts tangentially to the contact point's motion about the joint axis, the torque magnitude simplifies to $|\tau_{\text{joint}}| = F_{\text{op}} r_{\perp}$, where $r_{\perp}$ is the perpendicular distance from the contact point to the joint axis. For prismatic joints, we use the component of the operating force along the direction of joint motion. The resulting force or torque initializes $e_{\max}$; later adjustments are described in Appendix C.3. These parameters describe the joint's equivalent interaction response, not separately identified springs, friction, or transmission.

C.3 Operating Force Measurement and Parameter Calibration

Operating force measurement setup. For each category, a Piper arm manipulates a representative physical object along a predefined trajectory, and a digital force gauge records the operating force (N) along the task direction. Each object is measured 10 times with a matched initial joint configuration and force application position; the mean reading is the operating-force baseline, and the initial effort limit follows Appendix C.2. In simulation we match the contact position, EE orientation, motion direction, joint travel, and speed, then adjust the drive parameters based on the simulated contact force. Calibration trajectories vary joint position and speed to isolate restoring, damping, and saturation effects. For ordinary hinges or slides we set $k_p^{(k)} = 0$, let $k_d^{(k)}$ capture velocity-dependent resistance, and calibrate $e_{\max}^{(k)}$ from the measured force.

Drive parameter calibration objective. To formalize the alignment of real and simulated operating forces, let ξ_j denote the j-th calibration trajectory, with J trajectories in total. The error minimization objective for the category baseline parameters is written as

$$\left(k_p^{(k)}, k_d^{(k)}, e_{\max}^{(k)}\right) = \underset{k_p, k_d, e_{\max}}{\arg\min} \sum_{j=1}^{J} \left\| F_{\text{sim}}(\xi_j; k_p, k_d, e_{\max}) - F_{\text{meas}}(\xi_j) \right\|_2^2, \tag{5}$$

where F_{meas} and F_{sim} denote the real and simulated operating forces measured under the same manipulation conditions, respectively. This is the calibration objective; in practice, parameters are adjusted individually based on operating-force discrepancies.

Reference-guided instance parameter completion. For unmeasured assets in the same category, Qwen3-VL infers adjustments relative to the category baseline from calibrated references, multi-view renderings, scale, movable-part geometry, and joint structure, together with a confidence score. We reject predictions that violate nonnegativity, unit consistency, or category bounds, or whose confidence is too low, and fall back to the category baseline. Accepted parameters are written into $\mathcal{J}^+$.

D Randomization Parameters for Simulation Assets and Scenes

D.1 Task Configurations and Asset Randomization

The initial joint configuration and object pose randomization ranges, together with the language prompts for each task, are listed below. Angular parameters are in degrees (°), while prismatic joint displacement and object translations along the X and Y axes are in meters (m).

1. **Closing a laptop:** opening angle $[65°, 100°]$, X $[0.34, 0.54]$, Y $[-0.15, 0.15]$, yaw $[160°, 200°]$. Prompt: `close the laptop lid.`
2. **Adjusting a display:** opening angle $[-22°, -12°]$, X $[0.35, 0.55]$, Y $[-0.15, 0.15]$, yaw $[160°, 200°]$. Prompt: `adjust the display.`
3. **Closing a microwave:** opening angle $[-30°, -10°]$, X $[0.65, 0.75]$, Y $[-0.1, 0.2]$, yaw $[-15°, +15°]$. Prompt: `close the microwave door.`

4. **Closing a drawer:** opening displacement $[0.04, 0.16]$, X $[0.60, 0.70]$, Y $[-0.15, 0.15]$, yaw $[-25°, +25°]$. Prompt: `close the top drawer.`
5. **Adjusting the height of a lamp:** opening angle $[-19°, -15°]$, X $[0.35, 0.45]$, Y $[-0.13, -0.03]$, yaw $[80°, 100°]$. Prompt: `lower the desk lamp.`
6. **Adjusting a faucet:** opening angle $[20°, 40°]$, X $[0.5, 0.6]$, Y $[-0.15, 0.15]$, yaw $[75°, 105°]$. Prompt: `adjust the faucet.`

D.2 Environment Randomization Options

Parameter	Current value / options
Lighting mode	Random: `stage` / `camera` / `rig`
Light Rig	Random: `Colored Lights` / `Default` / `Grey Studio`
Intensity scale	0.65–1.05 (relative to the baseline)
Exposure offset	-0.6 to $+0.15$
Color temperature	4500–7500 K
Background scale	0.35
Candidate scenes	`ground_plane` (untextured ground), `grid_default`, `gridroom_black`, `terrain_flat_plane`
Main camera X/Y/Z	0.0–0.5 m / -0.6 to -0.2 m / 0.7–0.9 m
Focal length	6.9–8.9 cm (`PinholeCameraCfg.focal_length`)
Scene clutter	5 of 15 candidates; 5 slots (right $\times$ 3 + front $\times$ 1 + left $\times$ 1); yaw $[-180°, +180°]$

The 15 candidates are apple, retro alarm clock, lemon, utility jug, succulent plant, yellow sorting bowl, red sorting beaker, tomato soup can, banana, bleach cleanser, mug D1, NVIDIA cube, Boston round bottle, block, and mug A2.

D.3 Data Collection in Simulation and Trajectory Filtering

Dataset sizes are specified in Section 5.4.1; the fully randomized setting varies clutter, textures, lighting, camera parameters, object poses, and initial joint configurations. Data collection uses Isaac Sim 5.1.0 and Isaac Lab 0.54.3. The physics step is $1/60$ s; robot control and the two RGB streams run at 30 Hz. The dataset keeps the overhead and left wrist cameras (Appendix E.1). Each Isaac Sim process runs one environment; the main collection launches three processes on a single RTX 5090 D v2, at about 100 successful trajectories per process per hour.

Completed trajectories are filtered in two stages: the target joint must reach the target joint configuration within the specified tolerance before exhausting the control-step budget, and episodes with non-finite joint states, timestamp or frame discontinuities, missing frames, or corrupted images are discarded.

E Policy Training and Evaluation Details

E.1 Observation and Action Interfaces and SFT Configurations

We initialize $\pi_{0.5}$, X-VLA, and PRTS from `pi05_base`, `xvla-base`, and `PRTS-4B`, and follow each model's official fine-tuning configuration. X-VLA uses the new-embodiment setting with Soft Prompt and jointly fine-tunes the vision encoder, language encoder, and policy Transformer. All SFT runs on NVIDIA A100-SXM4-80GB GPUs. In the main comparison, all models train for 30k steps on fully randomized data; action chunks have length 50 for $\pi_{0.5}$ and PRTS, and 30 for X-VLA. $\pi_{0.5}$ and PRTS use two GPUs (global batch 32) and X-VLA one GPU (global batch 16).

Manipulation uses a single arm with 224×224 RGB from the overhead and left wrist cameras. Proprioception and actions are 7-dimensional (six left-arm joints and one gripper); right-arm channels and the right wrist image are dropped rather than zero-padded. PRTS predicts relative joint increments. The residual is 6-dimensional and corrects only the arm joints; the gripper channel is zero in the training data and is excluded from residual correction.

Backbone	Initial model	Batch per GPU / GPUs / gradient accumulation
$\pi_{0.5}$	`pi05_base`	16 / 2 / 1 (global batch 32)
PRTS	`PRTS-4B`	8 / 2 / 2 (global batch 32)
X-VLA	`xvla-base`	16 / 1 / 1 (global batch 16)

The randomization ablation follows the $\pi_{0.5}$ configuration above, with budgets in Section 5.4.1.

E.2 PRTS Execution and Residual Replay

Successful base-policy trajectories form an offline replay buffer; the critic is initialized with calibrated Q-learning (Cal-QL). After a base-policy warm-up, soft actor-critic (SAC) updates the residual actor and critic using equal numbers of samples from the offline success buffer and the online replay buffer (Haarnoja et al., 2018). The probing prefix only establishes the takeover state and is omitted from the online buffer; both success and failure after takeover are used. During BC evaluation, expert data collection, and distilled-policy evaluation, actions are replanned at every step under the same actuator constraints.

PRTS predicts a chunk of length 50, executes only the first row (`ACTION_EXEC_HORIZON=1`), then observes and replans. Control runs at 30 Hz for at most 600 steps (20 s of simulation time). During collection, inference takes about 270 ms and communication about 100 ms, so one control step is roughly 400 ms of wall-clock time. The step count includes the probing prefix and is not reset at takeover. Flow-matching sampling uses a fixed seed so the prefix is reproducible.

The remaining budget is $\eta_t = (600 - t)/600$. The residual $u_t \in (-1, 1)^6$ is scaled by $\mathbf{b} \in \mathbb{R}^6_{>0}$, added to the six arm components of the base action, converted to joint targets, and clipped. SAC uses the density after tanh and before physical scaling, including the tanh change-of-variables term. Each replay step stores the current input, executed base action, normalized residual, reward, next input, and termination flag; residuals on successful base-policy trajectories are zero. The time limit is part of the finite-horizon objective (no bootstrap after timeout); batch boundaries are not terminals, and numerically invalid transitions are dropped.

E.3 Recovery Trajectories and Policy Distillation

The base policy is frozen during expert training and hybrid collection; collection uses the deterministic residual mean after tanh. Successful hybrid trajectories keep the probing prefix and the post-takeover sequence; online RL replay drops the prefix.

Distillation labels are the actions executed after the control interface, including residual corrections. Chunks are consecutive actions from one episode; relative actions are defined relative to the joint state at the observation time. Normalization, end-of-trajectory padding, and masking follow the base model. Only the subsequent SFT stage updates VLA parameters; deployment loads the distilled VLA alone.

F CONTACT POINT VISUALIZATION EXAMPLES

Figure 6 shows contact point visualization examples across the 12 asset categories.

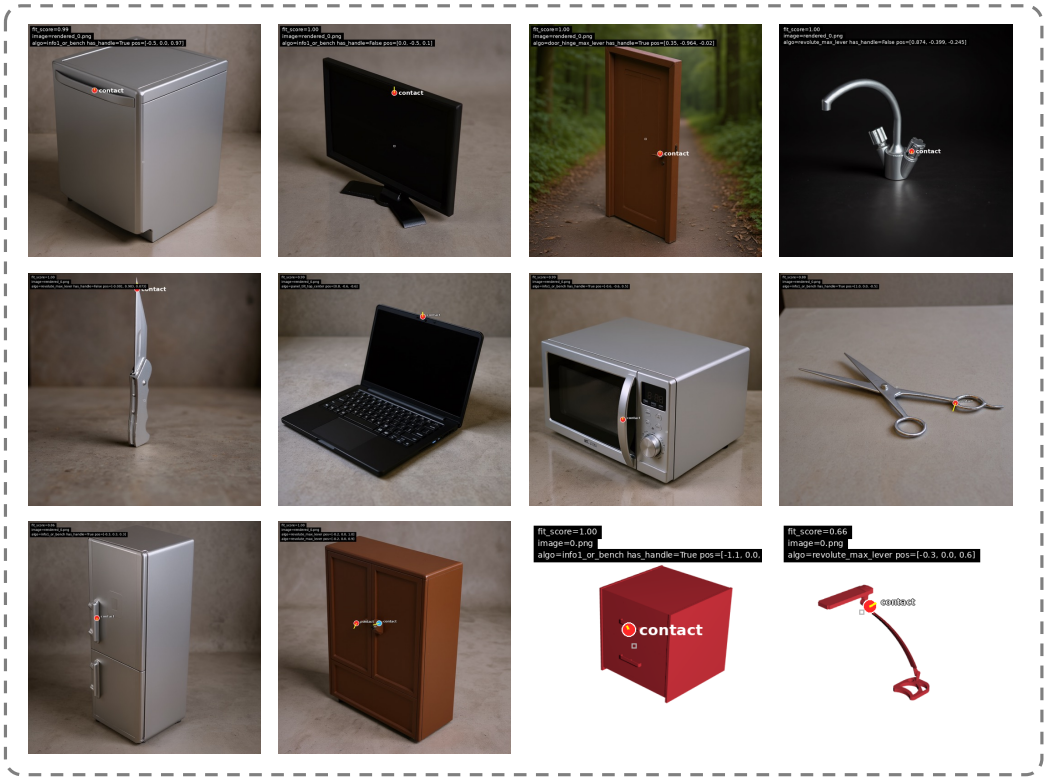


Figure 6: Contact point visualization examples across the 12 categories of FITR-Assets. Marked points indicate the contact locations on the articulated objects.