

# RoboCousin: Build Your Own Simulation Playground for Robust Bimanual Robotic Manipulation

Jingxuan Zhu<sup>\*†</sup>, Jingyi Li<sup>\*</sup>, LiangLiang Chen, Zhiyuan Jing, Jidong Zhang, Hongming Li<sup>†</sup>

E-surfing Digital Life Technology Co., Ltd., China Telecom

<sup>\*</sup> Equal contribution    <sup>†</sup> Corresponding authors



Figure 1: Overview of **RoboCousin**

## Abstract

Bimanual manipulation policies require large and diverse training datasets, yet collecting demonstrations on physical robots is expensive and difficult to scale. Simulation can generate data efficiently, but existing pipelines typically operate within closed asset libraries and predefined scenes: adding a newly observed object or environment still requires substantial effort to reconstruct geometry, specify physical and semantic properties, annotate interactions, and integrate the result into executable tasks. We present **RoboCousin**, an extensible simulation-based data-generation platform that turns user-provided observations into reusable assets, scenes, and expert trajectories for bimanual manipulation. Built on RoboTwin 2.0, RoboCousin converts object images into simulation-ready assets with visual and collision geometry, semantic and physical metadata, and automatically generated grasp-contact candidates. It further constructs digital cousins that vary compatible objects, backgrounds, layouts, and language instructions while preserving

task-relevant affordances and spatial relations. The same asset system supports tabletop and room-level scene construction, with collision-aware base control for interaction beyond a fixed workspace. We release **RoboCousin-OBD**, containing more than 3,000 annotated object instances and 50 background environments, and use RoboCousin to generate over one million expert trajectories across 50 tasks. Simulation and real-robot experiments show that the automatically generated interaction annotations are comparable to curated annotations, generated assets provide effective sim-to-real supervision, and tabletop cousins can improve transfer beyond training on a single reconstructed scene. RoboCousin therefore provides a practical path for expanding both the scale and coverage of synthetic bimanual manipulation data.

## 1 Introduction

Bimanual manipulation is essential for complex real-world activities such as folding clothes, opening containers, and carrying bulky objects. Learning policies that perform these tasks robustly requires demonstrations spanning diverse objects, layouts, environments, instructions, and robot embodiments. This demand is particularly acute for general-purpose vision-language-action models, whose performance depends strongly on the scale and coverage of their training data [20, 8]. Collecting such data on physical robots, however, is expensive, slow, and difficult to scale safely. Simulation provides a compelling alternative by enabling automated supervision and controlled variation at substantially lower cost.

Yet scalable rollout generation does not by itself yield a scalable manipulation data distribution. Most existing systems generate large amounts of data within a *closed simulation world*: they rely on a curated asset library, a predefined set of scenes, and task templates designed around those resources. Introducing a newly observed household object or a user-defined workspace still requires substantial manual effort to reconstruct geometry, create collision models, estimate physical properties, annotate interaction regions, and integrate the result into executable tasks. Consequently, increasing the number of trajectories may improve coverage within the original simulator distribution while leaving its object and environment support fundamentally fixed. Our central motivation is therefore that scalable robot data generation should make the simulated world itself extensible, not merely accelerate rollouts within an existing world.

This objective exposes three coupled challenges. **First, assets must be interaction-ready rather than merely visually plausible.** Broad 3D collections and recent generative models provide abundant geometry, but robot learning additionally requires accurate scale, collision geometry, physical attributes, and manipulation annotations. Without these properties, newly generated objects cannot be reliably used by motion planners or expert policies. **Second, diversity must preserve task semantics.** Conventional domain randomization improves robustness by perturbing appearance, lighting, geometry, or physical parameters [27], but arbitrary perturbations can be weakly related to the functional structure of the target scene. Exact digital twins preserve that structure but are costly to construct and provide only a narrow training distribution. Digital cousins offer a useful middle ground by varying object and scene instances while preserving relevant geometric and semantic affordances [5]. Moreover, most synthetic manipulation trajectories begin from a fixed pose at a predefined tabletop, leaving the surrounding environment outside the data-generation process. Mobile bimanual platforms make it possible to extend the same tabletop task with approach trajectories that vary across room layouts, initial poses, and paths, thereby adding spatial and temporal diversity beyond object-level manipulation [4, 16].

We introduce **RoboCousin**, a simulation-based platform that addresses these challenges through a unified observation-to-data pipeline. Built on RoboTwin 2.0 [3], RoboCousin allows users to expand the simulator from visual and textual inputs and then generate manipulation data with the resulting assets and scenes. Its *Digitalize Anything* module converts object images into textured, simulator-ready assets with visual and collision geometry, semantic and physical metadata, and automatically generated grasp-contact candidates. It also supports background and room construction, enabling object-level and environment-level expansion within the same asset system. RoboCousin then constructs tabletop digital cousins by preserving task-relevant semantic relations and support structures while varying compatible objects, backgrounds, layouts, and language instructions. These assets and cousin scenes are connected directly to automated bimanual expert trajectory collection.

At the room level, RoboCousin uses the generated scene layout and the collision meshes of both the environment and robot to plan executable routes from varied initial poses to the tabletop collection station. These navigation segments can be prepended to existing tabletop demonstrations, extending fixed-workspace manipulation data into longer navigation-and-manipulation trajectories.

The resulting platform supports two primary data-generation workflows: asset-centric generation for task-specific objects and tabletop cousin generation from real observations or specified layouts. Generated room layouts additionally support navigation-augmented data collection, in which the same manipulation task can be paired with different environments, starting poses, approach paths, and egocentric observation sequences. We release **RoboCousin-OB**, an extensible object-background library containing more than 3,000 annotated object instances and 50 background environments, together with a unified interface for adding assets, constructing scenes, and collecting data. Using RoboCousin, we generate over one million expert trajectories across 50 bimanual tasks. Experiments show that the automatically generated contact points are comparable to curated annotations, policies trained with generated assets match or outperform their RoboTwin 2.0 counterparts across the evaluated real-robot tasks, and tabletop cousins can improve transfer over training on a single reconstructed scene while remaining competitive on longer grasp-and-place tasks.

Our contributions are summarized as follows:

- We develop an extensible real-to-sim asset pipeline that converts user-provided observations into interaction-ready object and background assets with semantic, physical, collision, and grasp-contact annotations.
- We introduce a structured digital-cousin generation framework that expands object, scene, and language diversity while preserving task-relevant affordances, spatial relations, and support structures.
- We provide a unified asset-to-scene-to-data workflow and release RoboCousin-OB, containing more than 3,000 annotated object instances and 50 background environments, together with over one million expert trajectories across 50 tasks.
- We validate the automatically generated interaction annotations, assets, and cousin distributions through controlled simulation and real-robot experiments.

## 2 Method

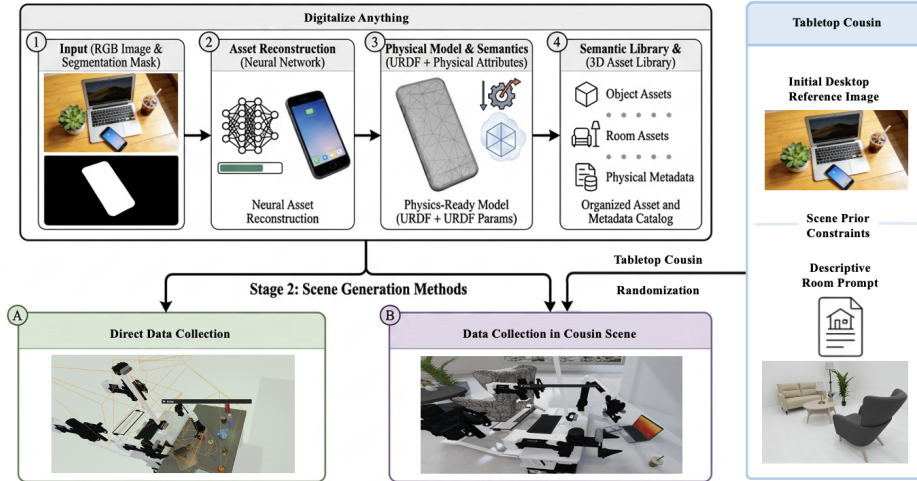


Figure 2: **RoboCousin Pipeline.**

We illustrate the overall RoboCousin pipeline in Fig. 2. The framework begins with a 3D property generation module that leverages 3D object reconstruction models, 3D Gaussian Splatting (3DGS), and large language models (LLMs) to automatically synthesize interactable objects and background environments from images. This module serves as the foundation for constructing a large-scale object-background asset library, enabling the digitization of user-defined instances across any object categories and workplace environments.

To ensure diverse demonstrations, we further integrate this automated 3D property generation pipeline with RoboCousin’s comprehensive digital-cousin randomization scheme. Beyond diversifying observations along linguistic, visual, and spatial dimensions, this strategy also introduces semantically related but non-identical objects into the replicated environments. Together, these components support the generation of diverse and realistic training data, facilitating the development of manipulation and lightweight navigation policies that are robust to real-world environmental variability.

## 2.1 Digitalize Anything

To address the bottlenecks in large-scale simulation data generation, we developed an automated Real-to-Sim 3D asset production pipeline that converts visual and textual inputs into simulation-ready digital assets. The pipeline covers three types of assets required by our simulator: object assets for manipulation, interaction annotations for robotic control, and scene-level assets for background construction. Given an RGB image, the system can extract a target object, reconstruct its 3D geometry and texture, infer simulator-compatible physical and semantic metadata, and organize the result into the asset structure used by RoboTwin 2.0 [3]. Given a high-level room prompt, the system can also generate a structured scene layout, which is later instantiated by sampling assets from the corresponding semantic categories. This design reduces the dependence on manually curated assets and annotations, and enables rapid construction of diverse interactive environments for upper-limb manipulation data generation.

**Image to 3D Asset Generation** Our end-to-end asset generation pipeline, as illustrated in Fig. 3, is built upon EmbodiedGen [30] and extends it with simulator-oriented interaction and asset-conversion modules. For object extraction, we support both point-based interactive segmentation with SAM [15] and text-guided segmentation with SAM 3 [2]. Compared with pure click-based segmentation, text prompts provide semantic constraints on the target category, color, or spatial relation, making it easier to isolate complete multi-part objects and reduce interference from cluttered backgrounds.

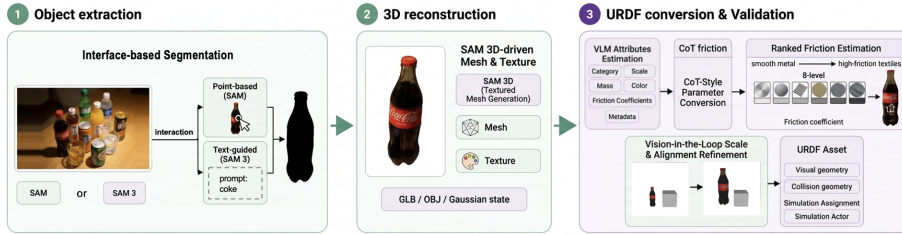


Figure 3: 3D Asset Generation Pipeline

After segmentation, the extracted object is reconstructed into a textured 3D mesh using SAM 3D [24] and converted into a URDF asset. During conversion, a vision-language model estimates semantic and physical attributes from multi-view renderings, including category, scale, mass, friction coefficients, color, and descriptive metadata. We further develop a CoT [32]-style converter to improve physical parameter consistency. In particular, it employs a ranked friction estimation mechanism, which maps objects to a 8-level standardized friction scale (ranging from smooth metal to high-friction textiles) to ensure numerical consistency across diverse categories. A vision-in-the-loop refinement step is adopted to further ensure spatial alignment in simulators, which compares the generated asset against a virtual 10cm reference cube to detect and correct scale-level hallucinations. The resulting asset contains both visual geometry and collision geometry, and can be synchronized into the simulator as an actor, static object, room asset, or general asset according to its intended function.

**Contact Point Inference** To bridge the gap between static 3D reconstructions and interactive simulation, we implement an automated contact point synthesis annotator module that converts generated assets into manipulation-ready objects. Instead of relying on manually annotated grasp poses, the module first normalizes the asset geometry and extracts both Axis-Aligned Bounding Box (AABB) and Oriented Bounding Box (OBB) representations to estimate spatial occupancy, principal axes, and grasp-relevant object dimensions. The overall pipeline is illustrated in Fig. 4. For thin or unusually scaled objects, additional alignment and scaling operations are applied to make the asset compatible with the simulator’s gripper conventions and tabletop manipulation settings.



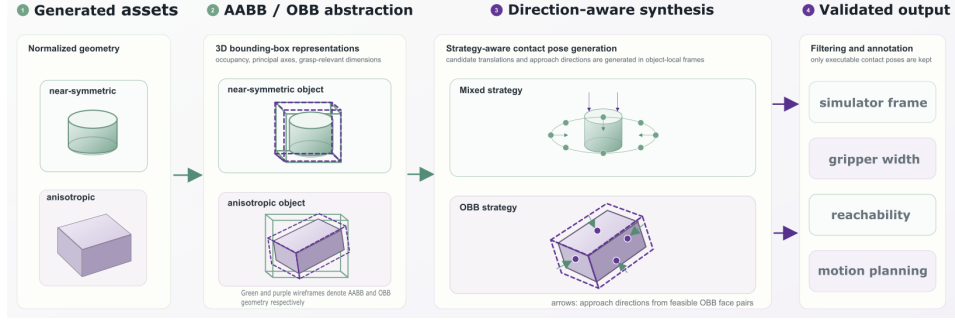


Figure 4: **Contact Point Automated Generation Pipeline**

Based on the normalized geometry, the system generates grasp candidates using two primary strategies:

- **Mixed Strategy:** For objects with approximate rotational symmetry, such as bottles or cans, the *mixed* strategy samples a ring of side grasps around the object together with a small set of vertical approach poses.
- **OBB Strategy:** For asymmetric or box-like objects, an *OBB* strategy identifies potentially graspable OBB faces and generates opposing contact pairs whose local frames are aligned with the corresponding face normals. Faces are pre-filtered according to gripper-width constraints, so that contact pairs are generated only on geometrically feasible surfaces.

A symmetry-aware selector automatically chooses between these two strategies by measuring the variation of the projected object extent under rotation, assigning near-symmetric objects to the mixed strategy and more anisotropic objects to the OBB strategy. Since purely geometric candidates may still contain physically invalid or kinematically unsuitable grasps, especially for OBB faces whose normals induce infeasible approach directions, we further apply candidate filtering before data collection. The filtering stage removes contact poses that violate gripper-width constraints, simulator frame conventions, reachability requirements, or motion-planning feasibility. Episodes without valid grasp candidates after filtering are discarded rather than used for expert trajectory generation. This produces assets with diverse but executable contact annotations, enabling large-scale automated manipulation data collection with minimal human intervention.

**Scene Asset Generation.** Beyond object-level assets, we further prepare scene-level assets for household background construction. Given a room prompt, the system generates a structured JSON layout that describes the room in terms of furniture categories, anchor objects, relative placement relations, spacing, rotations, and static attributes. This layout serves as an intermediate scene specification: it defines the semantic and spatial organization of the room, while leaving the concrete mesh instance of each furniture category to be resolved during simulator instantiation.

During scene construction, each category-level entry is matched to assets from the corresponding room-asset directory and loaded into the simulator with its specified pose and physical state. This representation separates high-level room planning from low-level asset selection, allowing generated rooms to be instantiated consistently while remaining compatible with later scene randomization. Visual generation results are illustrated in Fig. 5., showing diverse household scenes automatically constructed under different room prompts. Each scene strictly adheres to its corresponding JSON specification, achieving realistic spatial layout and asset matching.

## 2.2 Generate Digital Cousins

In contrast to digital twins—which aim to explicitly construct fully interactive replicas of specific real-world environments capable of capturing fine-grained details—recent approaches based on the concept of digital cousins generate virtual assets or scenes that do not directly replicate a real-world counterpart, yet still preserve similar geometric and semantic affordances. Such representations are not merely low-cost alternatives to digital twins; they can also improve policy robustness to real-world variability and distribution shifts.

In this work, we introduce digital-cousin-style randomization along two dimensions: Tabletop Digital Cousins and Background Asset Digital Cousins. This systematic augmentation strategy expands the

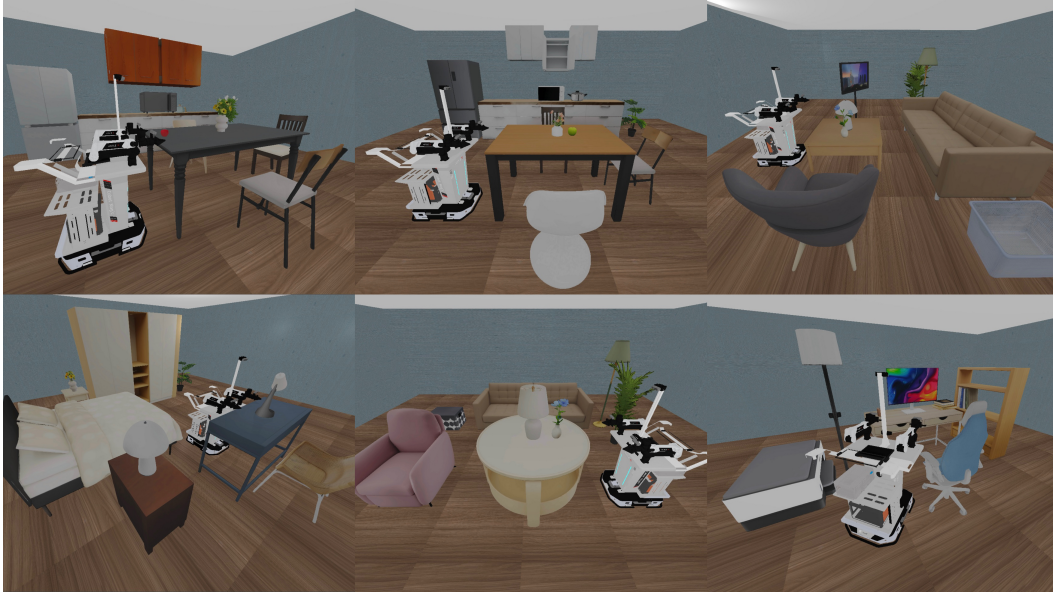


Figure 5: **Visualization of generated 3D household scenes**

training distribution in a semantically meaningful manner, thereby substantially improving generalization to unseen objects, environments, and task configurations. The effects of these randomization strategies are illustrated in Fig. 6.



Figure 6: **Visualization of randomization of tabletop cousins**

**Tabletop Digital Cousin.** For tabletop manipulation, we generate digital cousins from real desktop images rather than reconstructing an exact digital twin. Given a single RGB image of a tabletop scene, the pipeline first extracts object-level information using segmentation, depth estimation, and recaptioning, and then estimates a relative tabletop layout. The extracted objects are matched to assets in the RoboTwin asset library, using both manipulation-relevant actor assets and background non-actor assets. To make the resulting layout executable in simulation, the pipeline further estimates per-instance object orientation through camera-pose-aware snapshot rendering and records the selected orientation as a best snapshot index.

The generated tabletop cousin is represented as a structured layout, which contains object labels, relative positions, matched asset classes, candidate instances, and support relations. In particular, the pipeline constructs an explicit on-top support graph to preserve stacking and parent-child placement relations, allowing objects to be placed not only on the table but also on top of other objects when appropriate. This layout can then be directly consumed by RoboTwin for preview, task execution, and

demonstration collection, enabling policies to train on tabletop scenes that are semantically grounded in real observations while varying object identity and geometry.

**Background Asset Digital Cousin** Building on the scene-level asset representation, we introduce background digital cousins by randomizing the concrete mesh instances assigned to each category-level layout entry. Unlike a digital twin that fixes a particular reconstructed room and its exact assets, our background layout specifies semantic constraints, such as placing lamps on tables, chairs near desks, or nightstands beside beds, while allowing each furniture category to be instantiated with different assets from the same category. This preserves the functional and spatial affordances of the room while varying its visual appearance and geometric details. As a result, the policy observes manipulation tasks under diverse but semantically coherent household contexts without requiring manually authored digital twins for every background scene.

### 2.3 Robots Should Move

The room-scale environments generated by RoboCousin provide more than visual context: their structured layouts can be used directly for navigation planning. This creates a natural connection between room-scale scene generation and tabletop data collection. Given a collision-free initial pose, the robot first navigates through the generated room and docks at the embodiment-specific pose used for tabletop trajectory collection. It can then execute the same upper-body manipulation tasks as in the fixed-tabletop setting. In this way, scene construction, mobile navigation, and tabletop manipulation share a common coordinate frame and form a continuous data-generation workflow rather than two disconnected simulation modes.

**Scene-consistent navigation map.** The planner takes as input the JSON layout produced by the scene-generation module. For every furniture or room object, it resolves the referenced mesh and applies the same scale, orientation correction, semantic anchor relation, and world translation used when instantiating the scene in simulation. The transformed mesh vertices are projected onto the ground plane and converted into conservative obstacle polygons; when available, simulator metadata are additionally used to enlarge these footprints. The table, fixed wall geometry, and planning bounds are incorporated into the same map. To account for the physical extent of the robot, we parse the collision meshes from the embodiment URDF, project the selected base links onto the ground plane, and derive a navigation footprint. Each obstacle is then inflated by the footprint radius and a configurable safety margin, yielding a two-dimensional occupancy map that remains geometrically consistent with the rendered scene.

**Planning and table docking.** We discretize the free space into an eight-connected grid and use A\* [10] to plan from a user-specified collision-free start pose to a target pose. Diagonal transitions that cut across obstacle corners are disallowed. The resulting grid path is simplified by retaining only collision-free line-of-sight waypoints, and each segment is assigned a heading so that the robot rotates toward the next waypoint before translating. By default, the target is read from the embodiment configuration and corresponds to the nominal tabletop data-collection pose, although other reachable targets can also be specified. Because this pose lies close to the table, we use a staged docking procedure: the robot first approaches under the normal safety margin and then performs the final approach while retaining its physical footprint but removing only the additional margin.

**Simulation replay.** The planned route is serialized as a sequence of planar poses  $(x, y, \theta)$  and replayed in the corresponding scene by moving the complete Aloha-AgileX embodiment through alternating in-place rotations and forward translations. This deterministic replay leaves the arm configuration and tabletop controllers unchanged, allowing navigation to terminate directly at the pose expected by the existing manipulation-data pipeline. Figure 7 shows representative routes in three generated rooms. In the top-down map in the first column, the green dot, red star, and blue line denote the initial position, target tabletop data-collection position, and planned path, respectively; the subsequent frames show the robot following this path and arriving at the manipulation station. Thus, a tabletop task that originally contributes only a fixed-workspace arm trajectory can additionally yield room-scale episodes that vary in starting pose, path length, obstacle context, and visual observations. RoboCousin thereby connects room generation to executable navigation and ultimately to manipulation-data collection, expanding both the spatial coverage and temporal composition of the generated data.

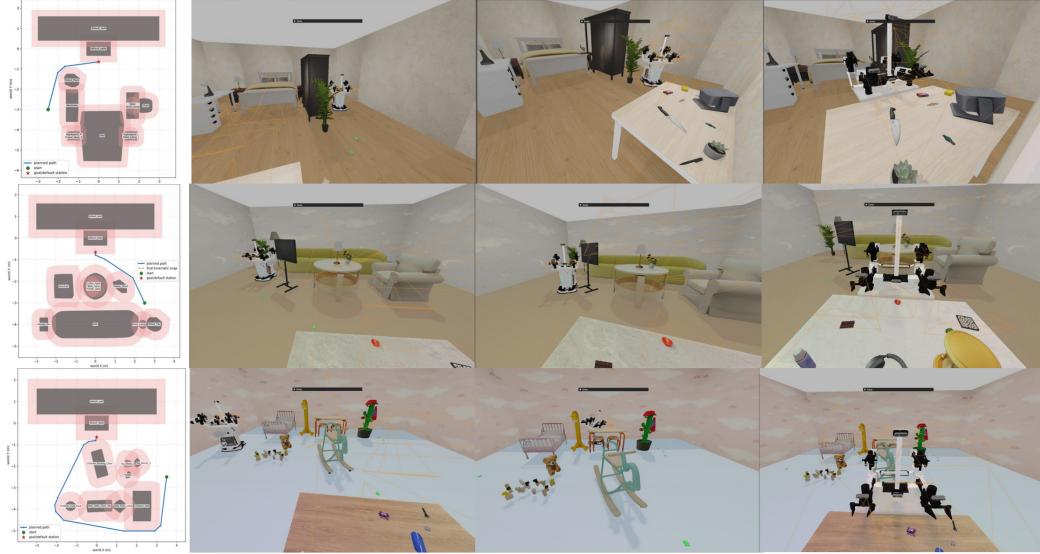


Figure 7: Representative navigation-augmented trajectories in three generated RoboCousin environments. Each row shows a top-down planning result followed by simulation frames along the corresponding route.

### 3 RoboCousin Data Generator, User Interface and Large Scale Dataset

#### 3.1 RoboCousin-OB: RoboCousin Object-Background Dataset

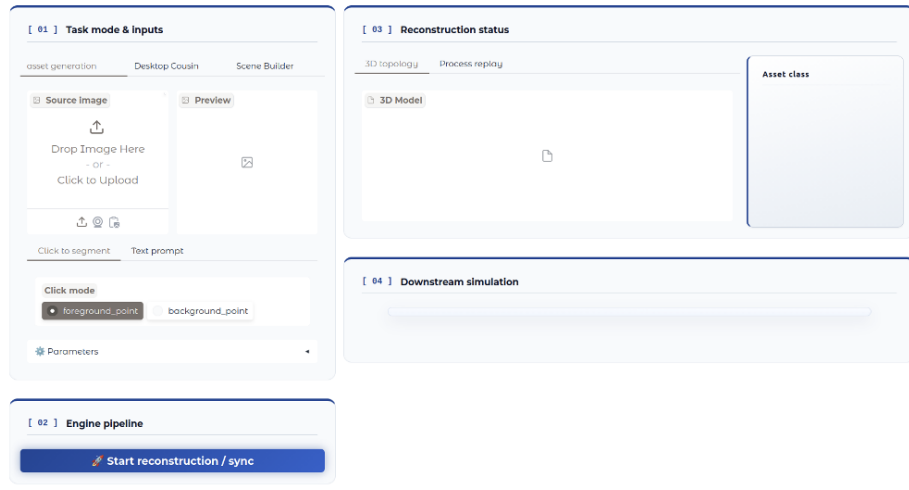


Figure 8: User interface

To support scalable bimanual manipulation research, we introduce RoboCousin-OB, a comprehensive object-background dataset with rich semantic, physical, and interaction-oriented annotations. RoboCousin-OB contains over 300 object categories and 3000 object instances, together with 50 ready-to-use digital-cousin-style background scenes. All these object instances are reconstructed from publicly available internet images using the generative asset pipeline described in Section 2.1. Each generated object is annotated with simulation-relevant physical attributes, including dimensions, mass, and friction coefficients. To support mobile manipulation in realistic household contexts, the dataset further provides 50 background environments covering 6 household scenarios, offering varied yet functionally meaningful simulation settings.



To enable object-centric interaction and physically grounded data generation, RoboCousin-OBd provides automated annotation scripts that encode object affordances for manipulation. Rather than treating assets as static visual meshes, our pipeline infers contact points and interaction regions from geometric structure and semantic priors. These annotations include graspable regions, placement surfaces, and task-relevant functional parts when applicable. Combined with the CoT-based URDF generator, each asset is assigned physically consistent parameters and can be directly instantiated in the simulator. This design enables scalable generation of grasping and manipulation trajectories across diverse objects, layouts, and background environments.

RoboCousin-OBd is designed to support flexible scene construction. Background assets can be loaded through JSON configurations or GLB binaries, enabling both pre-configured scene usage and customized environment synthesis. Beyond the 50 provided backgrounds, users can combine the object and room-asset libraries to construct new digital-cousin environments. Our scene generator further supports automatic generation of JSON scene configurations, allowing high-level room descriptions to be converted into diverse, simulation-ready household layouts.

### 3.2 Interactive Multi-Mode Pipeline for Scalable Trajectory Generation

Building on generative asset reconstruction and digital-cousin environment synthesis, we develop an end-to-end pipeline for scalable trajectory generation. The pipeline is organized around an interactive interface that allows users to configure inputs, inspect intermediate assets, monitor simulation execution, and control the data collection process. It supports three complementary operational modes:

- **Interactive Asset-Centric Generation:** This mode targets task-specific object manipulation. Users can generate object assets, inspect their reconstructed geometry and physical attributes, and collect trajectories that emphasize object-level interaction fidelity.
- **Tabletop Digital-Cousin Randomization:** This mode constructs randomized tabletop scenes from real-world desktop observations or user-specified layouts. By varying object instances and spatial arrangements while preserving semantic affordances, it supports robust data collection under diverse tabletop configurations.
- **Scene-Scale Mobile Manipulation:** This mode extends data generation beyond tabletop settings to room-level environments. It supports mobile manipulation scenarios in which the robot must navigate within diverse household layouts before executing functional upper-limb manipulation.

All three modes are integrated into a unified user-facing interface, as illustrated in Fig. 8. The interface includes multi-modal input panels for images and text prompts, a 3D reconstruction quality-check viewport, and real-time monitors for execution logs and simulation status. This design enables human-in-the-loop verification at key stages of the pipeline, including asset generation, scene construction, and trajectory collection, while preserving the scalability of automated data generation. Using this pipeline, we pre-collect over 1,000,000 dual-arm manipulation trajectories in RoboCousin.

## 4 Experiment

We evaluate whether the components of RoboCousin produce simulation assets and training distributions that are useful for real-world bimanual manipulation. Our experiments address three questions: (1) Can the automatically generated contact points support executable grasps in simulation? (2) Can policies trained with automatically reconstructed assets transfer to their corresponding real objects? (3) Does training with multiple semantically consistent tabletop cousins improve real-world robustness compared with training on a single digital twin? The first experiment isolates the quality of interaction annotations in simulation, whereas the latter two evaluate the complete sim-to-real pipeline. Unless otherwise stated, policies are post-trained from the  $\pi_{0.5}$  base checkpoint with a batch size of 32 for 30,000 optimization steps. Simulation data are generated using the Aloha-AgileX embodiment, and the resulting policies are deployed on an ARX AC-One real-robot setup with a 0.5m separation between the two arm bases. A snapshot of our real-world experiment is shown in Fig. 9.

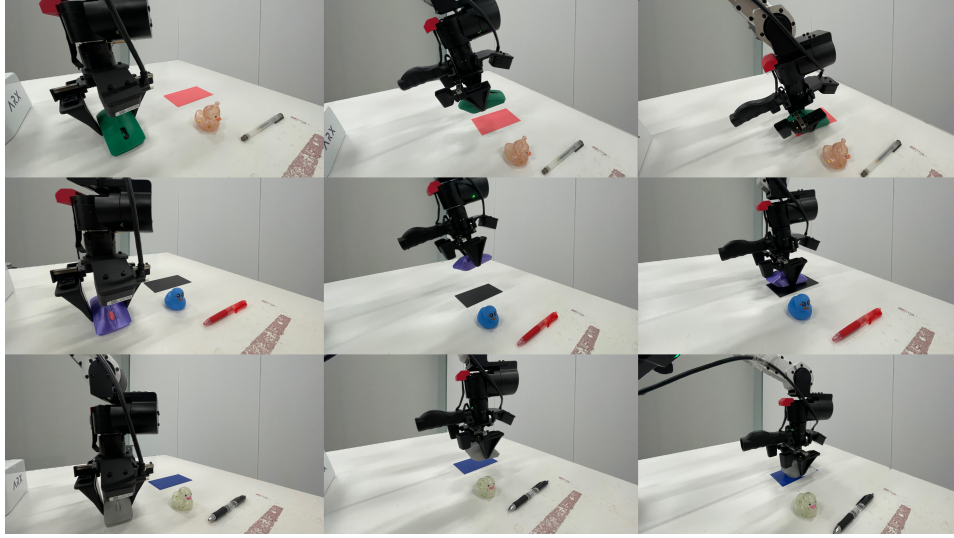


Figure 9: **Real-world evaluation on ARX AC-One.** The snapshot shows the bimanual robot performing the place mouse pad task.

#### 4.1 Evaluation of Automatic Contact-Point Generation

We first evaluate the automatic contact-point generation pipeline. Because many assets in the original RoboTwin library are used only as static scene objects, we select a subset that appear as manipulable actors in existing RoboTwin tasks. The selected objects span diverse geometries and categories, including bottles, cups, food items, electronic devices, stationery, toys, and common household objects, as shown in Table. 1.

Table 1: Selected manipulable objects from the RoboTwin library for evaluation.

| Category                | Asset ID & Name                       |
|-------------------------|---------------------------------------|
| <b>Drinkware</b>        | 001_bottle, 021_cup, 071_can          |
| <b>Food Items</b>       | 006_hamburg, 035_apple, 075_bread     |
| <b>Electronics</b>      | 046_alarm-clock, 047_mouse, 077_phone |
| <b>Stationery</b>       | 048_stapler, 058_markpen              |
| <b>Toys &amp; Games</b> | 057_toycar, 081_playingcards          |
| <b>Household</b>        | 107_soap, 112_tea-box, 113_coffee-box |

To separate contact-point quality from long-horizon task planning, we use a simplified pick-up task in which the robot grasps a target object and lifts it by a fixed distance. This task directly tests whether an annotation yields a reachable and stable grasp while minimizing confounding effects from object rearrangement and task-specific constraints. For every selected asset, we compare the original RoboTwin contact annotations with those produced by our automatic pipeline, keeping the asset, robot configuration, and evaluation protocol unchanged.

The original annotations achieve a success rate of 72%, whereas our automatically generated contact points achieve 76%, evaluated across 100 trajectories randomly sampled from the selected object actors. Thus, without manual contact annotation, our pipeline matches and slightly exceeds the performance of the curated annotations. These results show that the generated contact points are sufficiently reliable to turn diverse reconstructed meshes into interaction-ready assets, removing an important manual bottleneck in scaling object libraries and expert-trajectory generation.

#### 4.2 Sim-to-Real Evaluation of Generated 3D Assets

We next evaluate whether assets reconstructed by our image-to-3D pipeline provide effective supervision for real-world manipulation. The central comparison is between policies trained with our automatically generated assets and policies trained with the corresponding assets supplied by RoboTwin 2.0. For each task and asset condition, we collect 250 successful expert trajectories

Table 2: Real-world success rates using RoboCousin-generated and RoboTwin 2.0 assets. Each policy uses 250 successful simulated trajectories.

| Task                       | Ours (%)  | RoboTwin (%) |
|----------------------------|-----------|--------------|
| Pick Up Bottle             | <b>80</b> | 40           |
| Place Microphone in Basket | <b>75</b> | <b>75</b>    |
| Place Mouse Pad            | <b>40</b> | 30           |
| Place Marker Box           | <b>65</b> | 50           |

in simulation. Starting from the same  $\pi_{0.5}$  base checkpoint, we post-train a separate policy for each condition using a batch size of 32 for 30,000 steps. All demonstrations are generated with Aloha-AgileX in simulation, and the resulting policies are evaluated on the ARX AC-One using the corresponding physical objects. The two real arm bases are separated by 0.5 m.

We evaluate four tasks of increasing manipulation complexity. `Pick_Up_Bottle` is our simplified grasp-and-lift task and contains only grasping and lifting. `Place_Microphone_in_Basket` is adapted from RoboTwin 2.0’s `place_object_basket` task. We remove the original object-category restriction, allowing an arbitrary target category to be specified, and omit the final basket-lifting stage. The resulting action sequence consists only of grasping the microphone and placing it in the basket. `Place_Mouse_Pad` is retained unchanged from RoboTwin 2.0 and requires accurate object transport and placement. `Place_Marker_Box` is adapted from RoboTwin 2.0’s `place_object_stand` task: the robot moves a marker onto a box and must place it with the longest edge of the marker parallel to the longest edge of the box.

As shown in Table 2, RoboCousin assets match or outperform the original RoboTwin 2.0 assets on all four tasks. The largest gain occurs on `Pick_Up_Bottle`, where success increases from 40% to 80%. Performance is identical on `Place_Microphone_in_Basket` (75%), while `Place_Mouse_Pad` improves from 30% to 40% and the orientation-constrained `Place_Marker_Box` improves from 50% to 65%. These results show that automatically generated assets provide effective sim-to-real supervision across grasping, transport, and placement tasks.

### 4.3 Sim-to-Real Evaluation of Tabletop Digital Cousins

Finally, we test whether training on a distribution of tabletop digital cousins improves transfer relative to training on a single digital twin. Starting from the same real tabletop observation, the `Twin` condition constructs one simulation scene that closely matches the observed object instances, layout, and spatial relations. The `Cousin` condition instead constructs multiple scenes that preserve task-relevant semantic affordances and relational structure while varying compatible object instances, appearances, and physical properties. For each task and scene-generation condition, we collect 400 successful expert trajectories. We then post-train separate policies from the same  $\pi_{0.5}$  base checkpoint with a batch size of 32 for 30,000 steps. The policy architecture and all other training settings are identical across the `Cousin` and `Twin` conditions, so the comparison isolates the effect of the training-scene distribution. As in Sec. 4.2, data are generated with Aloha-AgileX, and the policies are deployed on the ARX AC-One setup with a 0.5m arm-base separation.

We evaluate four tasks. `Pick_Up_Eyeglass_Case` uses the simplified grasp-and-lift sequence described in Sec. 4.1. `Place_Spray_Bottle_in_Basket` uses the same modified `place_object_basket` protocol as Sec. 4.2, with a spray bottle as the manipulated object and without the basket-lifting stage. `Adjust_Bottle` is retained unchanged from RoboTwin 2.0 and requires the robot to grasp the bottle with the appropriate arm and place it upright. `Place_Mouse_Pad` follows the same unmodified RoboTwin 2.0 protocol used in Sec. 4.2.

As shown in Table 3, cousin training improves `Pick_Up_Eyeglass_Case` from 0% to 55% and `Adjust_Bottle` from 50% to 70%. It also improves `Place_Mouse_Pad` from 20% to 25%, while achieving 35% compared with 40% for `Twin` on `Place_Spray_Bottle_in_Basket`. Overall, tabletop cousins improve transfer on three of the four evaluated tasks and remain competitive with single-scene twin training on the remaining task.

Table 3: Real-world success rates using tabletop cousins or a single digital twin. Each policy uses 1,000 successful simulated trajectories.

| Task                         | Cousin (%) | Twin (%)  |
|------------------------------|------------|-----------|
| Pick Up Eyeglass Case        | <b>55</b>  | 0         |
| Place Spray Bottle in Basket | 35         | <b>40</b> |
| Adjust Bottle                | <b>70</b>  | 50        |
| Place Mouse Pad              | <b>25</b>  | 20        |

## 5 Related Work

### 5.1 Simulation Platforms and Scalable Manipulation Data Generation

Physics-based simulation has become an important foundation for scalable robot learning. SAPIEN supports physically grounded interaction with large collections of articulated objects [33], while RL Bench provides a standardized suite of vision-guided manipulation tasks [12]. ManiSkill2 [9] and ManiSkill3 [26] further improve object-level diversity, physical simulation, rendering efficiency, and support for heterogeneous robot embodiments. At the household scale, Habitat 2.0 [25] and BEHAVIOR-1K [17] introduce interactive indoor environments and long-horizon mobile-manipulation tasks, whereas RoboCasa [21] provides diverse kitchen scenes, assets, tasks, and demonstrations for training generalist manipulation policies. BiGym more specifically provides a demo-driven benchmark for mobile bimanual manipulation in household environments [4].

Several systems reduce the cost of collecting manipulation demonstrations. MimicGen adapts a small number of human demonstrations to new object poses, scenes, and robot embodiments [19], while DexMimicGen extends automated demonstration generation to coordinated bimanual dexterous manipulation [13]. RoboTwin [20] combines generative digital twins with automatically generated expert trajectories for dual-arm manipulation. RoboTwin 2.0 [3] substantially extends with a larger annotated asset library, automated task generation, multiple embodiments, and structured domain randomization. For mobile bimanual settings, MoMaGen generates multi-step demonstrations by explicitly optimizing reachability and visibility constraints [16].

These platforms provide valuable simulation infrastructure and standardized evaluation protocols. Nevertheless, the practical distributions represented by their released benchmarks are generally centered on curated asset collections and predefined scene or task templates. Although simulator-specific interfaces may permit manual extension, converting an arbitrary user-provided observation into a physically annotated and interaction-ready asset remains a separate engineering process. RoboCousin complements these platforms with a modular asset-to-data workflow in which newly observed objects and environments can be reconstructed, assigned physical properties, annotated with manipulation-relevant contact information, organized into extensible asset libraries, and directly incorporated into existing expert trajectory-generation pipelines.

### 5.2 Generative Assets and Automated Simulation Construction

Generative methods have been used to automate different stages of simulation construction. GenSim uses large language models to generate task programs and expert demonstrations [29], and GenSim2 extends this direction to long-horizon tasks with articulated objects using multimodal reasoning models [11]. Gen2Sim jointly generates assets, task descriptions, temporal decompositions, physical parameters, and rewards [14], while RoboGen proposes a self-guided propose-generate-learn loop for automated robot learning [31]. These approaches primarily target task-level generation and automated learning curricula.

At the asset level, Objaverse offers broad visual and categorical coverage, but most assets are not directly equipped with reliable collision geometry, physical parameters, or manipulation annotations [6]. EmbodiedGen generates scaled and physically grounded URDF assets for robotics simulation [30]. Real-to-sim methods such as RialTo construct digital twins for policy improvement [28], while DreMa combines Gaussian Splatting and physics simulation to augment manipulation demonstrations through imagined object configurations [1]. SplatSim uses Gaussian Splatting to reduce the visual sim-to-real gap for RGB manipulation policies [23]. More recently, RoboSimGS combines Gaussian-



Splatting backgrounds with physics-enabled object representations and multimodal-model-based physical-property inference [34].

RoboCousin differs in its emphasis on the complete path from a visual observation to reusable manipulation data. It produces both visual and collision representations, estimates semantic and physical attributes, generates grasp-contact candidates, and organizes assets by their functional roles in simulation. The same pipeline handles manipulable objects and background components, allowing newly generated assets to be used directly in expert trajectory collection. At the scene level, RoboCousin separates semantic layout planning from concrete asset instantiation. A single relational layout can therefore be populated with different compatible assets, providing a scalable basis for both scene construction and digital-cousin generation.

### 5.3 Domain Randomization, Digital Twins, and Digital Cousins

Domain randomization improves sim-to-real transfer by exposing policies to variations in rendering, textures, lighting, camera parameters, object appearance, and physical properties [27]. ProcTHOR scales environmental diversity through procedural scene composition [7], while THE COLOSSEUM evaluates manipulation policies under systematic visual, geometric, physical, and environmental perturbations [22]. RoboTwin 2.0 further introduces structured visual, spatial, and linguistic randomization for bimanual manipulation [3].

Digital cousins occupy the space between unconstrained randomization and exact digital twins. ACDC constructs cousin scenes by retrieving semantically and geometrically related assets from an existing collection and shows that training across multiple cousins can improve robustness over a single twin [5]. From Seeing to Simulating extends this idea to high-fidelity, room-scale environments using editable 3D Gaussian Splatting scenes, collision meshes, and semantic world editing [18].

RoboCousin is complementary to these approaches but targets a different bottleneck: scalable bimanual manipulation data rather than high-fidelity scene replication alone. Unlike retrieval-only cousin pipelines, RoboCousin can generate new interaction-ready objects with collision geometry, physical properties, and contact annotations, so its cousin distribution is not bounded by a fixed asset collection. It also reconstructs tabletop relations and support structures, then varies compatible object and background instances while preserving task-relevant semantics. By coupling this extensible cousin representation with automated expert trajectory generation, RoboCousin turns semantic scene variation directly into training data for real-world bimanual manipulation.

## 6 Conclusion

We presented **RoboCousin**, an extensible simulation-based data-generation platform for bimanual manipulation. RoboCousin addresses a central limitation of existing synthetic-data pipelines: scaling the number of rollouts does not expand the underlying object and environment distribution. By connecting real-to-sim asset generation, interaction annotation, digital-cousin construction, and expert trajectory collection, the platform turns user-provided observations into reusable robot-learning data. Its structured cousin generation varies concrete assets, layouts, backgrounds, and language instructions while preserving task-relevant affordances and spatial relations. The same framework supports both tabletop and room-level environments, including base movement beyond a fixed workspace. Using this pipeline, we construct RoboCousin-OBD with more than 3,000 annotated object instances and 50 background environments and generate over one million expert trajectories.

Our experiments validate the key stages of this pipeline. Automatically generated contact points perform comparably to curated annotations, and policies trained with RoboCousin assets match or outperform their RoboTwin 2.0 counterparts across the evaluated real-robot tasks. Tabletop cousins further improve transfer on the eyeglass-case pick-up task while remaining competitive with single-scene twin training on the longer spray-bottle placement task. Together, these results show that simulator extensibility can translate into useful real-robot supervision rather than merely greater synthetic-data volume. Future work will broaden evaluation across tasks, environments, and robot embodiments and improve physical calibration and task-aware cousin generation for contact-sensitive and long-horizon manipulation.

## References

- [1] Leonardo Barcellona, Andrii Zadaianchuk, Davide Allegro, Samuele Papa, Stefano Ghidoni, and Efstratios Gavves. Dream to manipulate: Compositional world models empowering robot imitation learning with imagination. In *International Conference on Learning Representations*, volume 2025, pages 56729–56763, 2025.
- [2] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.
- [3] Tianxing Chen, Zhanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.
- [4] Nikita Chernyadev, Nicholas Backshall, Xiao Ma, Yunfan Lu, Younggyo Seo, and Stephen James. BiGym: A demo-driven mobile bi-manual manipulation benchmark, 2024.
- [5] Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Automated creation of digital cousins for robust policy learning. *arXiv preprint arXiv:2410.07408*, 2024.
- [6] 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*, pages 13142–13153, 2023.
- [7] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Proctor: Large-scale embodied ai using procedural generation. *Advances in neural information processing systems*, 35:5982–5994, 2022.
- [8] Shengliang Deng, Mi Yan, Songlin Wei, Haixin Ma, Yuxin Yang, Jiayi Chen, Zhiqi Zhang, Taoyu Yang, Xuheng Zhang, Heming Cui, et al. Graspvla: a grasping foundation model pre-trained on billion-scale synthetic action data. *arXiv preprint arXiv:2505.03233*, 2025.
- [9] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023.
- [10] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
- [11] 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, 2024.
- [12] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.
- [13] Zhenyu Jiang, Yuqi Xie, Kevin Lin, Zhenjia Xu, Weikang Wan, Ajay Mandlekar, Linxi Fan, and Yuke Zhu. DexMimicGen: Automated data generation for bimanual dexterous manipulation via imitation learning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.
- [14] Pushkal Katara, Zhou Xian, and Katerina Fragkiadaki. Gen2sim: Scaling up robot learning in simulation with generative models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6672–6679. IEEE, 2024.
- [15] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023.
- [16] Chengshu Li, Mengdi Xu, Arpit Bahety, Hang Yin, Yunfan Jiang, Huang Huang, Josiah Wong, Sujay Garlanka, Cem Gokmen, Ruohan Zhang, et al. Momagen: Generating demonstrations under soft and hard constraints for multi-step bimanual mobile manipulation. In *International Conference on Learning Representations*, volume 2026, pages 112425–112446, 2026.
- [17] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023.

- [18] Jasper Lu, Zhenhao Shen, Yuanfei Wang, Shugao Liu, Shengqiang Xu, Shawn Xie, Jingkai Xu, Feng Jiang, Jade Yang, Chen Xie, et al. From seeing to simulating: Generative high-fidelity simulation with digital cousins for generalizable robot learning and evaluation. *arXiv preprint arXiv:2604.15805*, 2026.
- [19] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Ireteyo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. *arXiv preprint arXiv:2310.17596*, 2023.
- [20] Yao Mu, Tianxing Chen, Zanzin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, et al. Robotwin: Dual-arm robot benchmark with generative digital twins. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2025.
- [21] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.
- [22] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024.
- [23] Mohammad Nomaan Qureshi, Sparsh Garg, Francisco Yandun, David Held, George Kantor, and Abhisesh Silwal. SplatSim: Zero-shot sim2real transfer of RGB manipulation policies using gaussian splatting, 2024.
- [24] SAM 3D Team, Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J. Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, et al. Sam 3d: 3dfy anything in images. *arXiv preprint arXiv:2511.16624*, 2025.
- [25] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266, 2021.
- [26] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, et al. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *arXiv preprint arXiv:2410.00425*, 2024.
- [27] 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)*, pages 23–30. IEEE, 2017.
- [28] Marcel Torme, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkrit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *arXiv preprint arXiv:2403.03949*, 2024.
- [29] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. Gensim: Generating robotic simulation tasks via large language models. In *International Conference on Learning Representations*, volume 2024, pages 4890–4924, 2024.
- [30] Xinjie Wang, Liu Liu, Yu Cao, Ruiqi Wu, Wenkang Qin, Dehui Wang, Wei Sui, and Zhizhong Su. Embodiedgen: Towards a generative 3d world engine for embodied intelligence. *arXiv preprint arXiv:2506.10600*, 2025.
- [31] Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Katerina Fragkiadaki, Zackory Erickson, David Held, and Chuang Gan. Robogen: Towards unleashing infinite data for automated robot learning via generative simulation, 2023.
- [32] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022.
- [33] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11097–11107, 2020.
- [34] Haoyu Zhao, Cheng Zeng, Linghao Zhuang, Yaxi Zhao, Shengke Xue, Hao Wang, Xingyue Zhao, Zhongyu Li, Kehan Li, Siteng Huang, Mingxiu Chen, Xin Li, Deli Zhao, and Hua Zou. High-fidelity simulated data generation for real-world zero-shot robotic manipulation learning with gaussian splatting, 2025.