# Act to See: Structured Articulated Representations through Robot Interaction

Yifeng Liu[1,*] Yuzhen Chen[2,*] Bingyang Wang[3,*] Chengqi Li[4] Jingyi Lu[3]
Han Li[3] Kaichen Zhou[1,2] Mengyu Wang[2] Fangneng Zhan[3]
[1]Massachusetts Institute of Technology (MIT) [2]Harvard University
[3]The Hong Kong University of Science and Technology (HKUST) [4]Eindhoven University of Technology
lauyihong.github.io/act2see/

***Abstract*—Articulated objects cannot be fully understood from passive observation alone, as their hidden joints, motion constraints, and manipulation-relevant structure are revealed faithfully only through physical interaction. Prior work largely treats such interaction as a one-off measurement, processed offline after the full observation sequence has been collected. We instead treat interaction as an online signal for reconstruction: as the robot moves a part, our framework Act2See incrementally fuses each streaming RGB-D frame into an explicit URDF, driven by the motion signal and without waiting for the interaction to finish. The recovered articulation is estimated primarily from the end-effector trajectory traced while the part is in motion, with sparse visual tracks as a fallback; views captured during motion are aligned by the estimated articulation and fused into per-part geometry. When newly arriving observations become inconsistent with the current axis hypothesis, Act2See rolls back and re-estimates from cached frames, allowing the reconstruction to self-correct during interaction rather than committing to an early erroneous estimate. The resulting explicit model is interpretable and directly reusable; we further show that it can be consumed by a VLM-based planner to drive downstream manipulation. In the SAPIEN simulator with a Franka Panda arm, Act2See achieves strong revolute-joint accuracy without access to asset priors, while remaining competitive on prismatic joints. On a real UFactory xArm 6 robot, we demonstrate the full closed-loop system on desktop cabinets as a proof of concept, successfully operating prismatic drawers in all trials while revolute doors remain substantially harder, which is a failure mode consistent with our simulation results. We report geometric reconstruction qualitatively and analyze its current limitations, leaving a quantitative geometric evaluation to future work.**

## I. Introduction

In everyday environments, articulated objects are crucial for robotic manipulation. Objects such as cabinets, drawers, appliances, and tools require manipulation strategies that depend not merely on their visual appearance, but also on an understanding of their kinematic structure, specifically, which parts move, around which axes they rotate, and what their range of motion entails. Reconstructing such structures has emerged as a central objective in the field of robotic perception [16, 10, 13, 19, 11, 15, 23]. However, only relying on passive visual observation often proves insufficient for fully recovering this structural information. A tightly shut drawer may appear visually indistinguishable from a solid panel; similarly, hidden hinges reveal their rotational axes only when the cabinet door is set in motion. In all such scenarios, the structural information that a robot attempts to capture through reconstruction becomes apparent only after physical actions—that is, interactions—have been applied to the object.

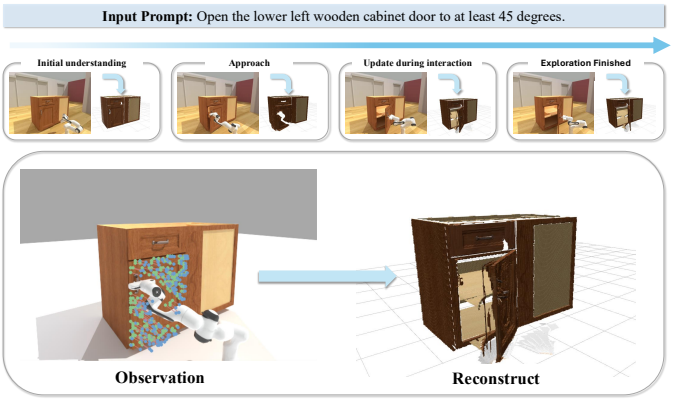



Fig. 1: **Act2See** extracts structures from vision-language models and actions history, to compose URDF representation, which are used by motion planners to zero-shot synthesize trajectories for everyday manipulation tasks.

Although interactive manipulation can yield valuable clues about an object's articulated structure, existing systems rarely exploit them effectively. The observations gathered during interaction are typically treated as transient measurements—scattered across separate interaction episodes—rather than being integrated into a persistent model that can support subsequent manipulation. Consequently, existing systems typically either rely on recorded video feeds as input to perform reconstruction in an offline manner, or they lack a structured memory mechanism during the manipulation process. This leads to highly redundant exploration strategies, causing the systems to be particularly vulnerable in the presence of object occlusions, and precludes the reuse of previously acquired structural knowledge.

Accordingly, in this work, we conceptualize interactive reconstruction as a "closed-loop" process, wherein actions serve as the active source of reconstruction signals rather than merely as downstream consumers of perceptual information. We introduce Act2See, a framework designed to progressively construct an "explicit URDF" (Unified Robot Description Format) for articulated objects through interactive exploration,

as illustrated in Fig. 1. As the robot executes actions, Act2See receives each RGB-D observation and updates the object model in real-time—incorporating details such as the geometry of each component (including oriented bounding boxes), joint types, joint axes, and the current joint states. Crucially, the maintained URDF is not a passive output; rather, at every step, a VLM-based planner integrates encoding of the current URDF—comprising renderings of the current state and a textual list of joints—to determine the next component or hypothesized joint to interact with, whether further manipulation is required, and which structural cues (such as joint axes or handle masks) should be utilized to constrain downstream trajectory policies. Once constructed, the URDF can be directly employed to drive manipulation tasks, whether within the same scene or for novel tasks involving the same object. Leveraging a foundation VLM, our framework generalizes to cabinets unseen during development.

Our contributions are summarized as follows: (1) We cast articulated-object reconstruction as an online, interaction-driven process: visual signals produced during interaction are fused on the fly into an incrementally updated explicit URDF, rather than processing a fixed, pre-collected observation set offline. (2) We present a reconstruct-while-moving mechanism that jointly recovers articulation parameters and dense part masks from sparse visual tracks, aligns views captured during motion, and fuses them via per-part TSDF; an axis-triggered recomputation rolls back and re-estimates when the axis hypothesis conflicts with new observations, letting the model self-correct mid-interaction. (3) We evaluate Act2See in SAPIEN, where it achieves the best articulation accuracy among methods that reconstruct joints from observations without asset priors, and further deploy the full closed-loop system on a real xArm 6 as a proof of concept; the resulting explicit URDF is directly reusable by a VLM planner for downstream manipulation.

In §II we discuss related work; in §III we present the Act2See framework, including the online reconstruction module (§III-D) and the closed-loop VLM planner (§III-B); and in §IV we report simulation and real-world experiments, ablations, and quantitative reconstruction analysis.

## II. Related Works

### A. Articulated Objects Understanding

Manipulating articulated objects requires more than recognizing their shape; a robot must infer which regions are movable parts and how those parts move. Existing approaches primarily formulate articulated understanding as a passive inference problem from static observations. Early category-level approaches estimate part pose or joint parameters directly, including ANCSH [16] for normalized part coordinates and ScrewNet [10] for screw-parameter articulation. A second line replaces explicit joint regression with action-oriented representations that are easier for a manipulation policy to consume.

Eisner et al. [7] predicts dense 3D articulation flow, indicating how object points should move under an articulated motion and providing local motion cues for interaction. Zhang et al. [33] projects this flow onto articulation-consistent directions to improve manipulation across object instances. Geng et al. [8] instead segments objects into generalizable actionable parts, making part-level affordances the interface between perception and manipulation. Recent work has pushed this line in two complementary directions. The first studies articulation inference under sparse visual evidence: DailyArt [32] estimates joint parameters from cross-state discrepancies after synthesizing a maximally articulated opened state from a single closed-state image. LARM [30] reconstructs detailed articulated objects from sparse-view images with a large feed-forward model, PAct [20] generates part-decomposed articulated 3D assets from a single image, and KineDiff3D [29] uses kinematic-aware diffusion for category-level articulated shape reconstruction, generation, and pose estimation.

A complementary line makes outputs more directly usable by simulators or robot pipelines: URDFormer [4] generates articulated simulation environments from real-world images, Articulate-Anything [15] builds interactable articulated assets from multimodal inputs, URDF-Anything [17] targets articulated-object reconstruction with a 3D multimodal language model for URDF-style digital twins, AutoURDF [18] constructs simulator-compatible robot description files from point-cloud frames, Real2Code [23] represents articulated objects through generated code, and VideoArtGS [21] reconstructs part-aware articulated digital twins from monocular video. Related 4D perception models such as PAGE-4D [35] further show that dynamic image sequences can support joint camera-pose, depth, and point-map estimation in non-static scenes.

These methods substantially improve visual priors, dynamic geometry estimation, and output representations, but they mostly solve offline reconstruction or perception from fixed, pre-collected visual sequences. They do not generally close the loop in which the current articulated model determines the next interaction and the resulting observation updates the model. Our method instead treats articulated understanding as an intermediate state in a closed-loop manipulation process, where the reconstruction can be tested and updated through interaction.

### B. Active Perception and Interactive Reconstruction

For articulated objects, manipulation itself acts as a sensing mechanism that reveals otherwise unobservable structure and constraints. Active perception therefore turns perception into a decision problem, where the agent chooses physical interactions that are expected to reveal useful structure.

Hausman et al. [9] maintains a belief over rigid, prismatic, revolute, and free-body models and selects actions that disambiguate these hypotheses. A complementary line uses interaction or multi-state evidence after it has been collected: DITTO [13] reconstructs articulated digital twins from pre- and post-interaction point clouds, PARIS [19] jointly reconstructs parts and estimates motion, and AutoURDF [18] recovers simulator-compatible descriptions from point-cloud

frames. More recent robot systems close part of the loop by actively selecting contacts or exploration policies, as in Sim2Real$^2$ [22], DexSim2Real$^2$ [12], Where2Explore [25], and AKM [31]. Online estimators then show how such evidence can be fused incrementally, from factored belief propagation [6] to factor-graph estimation [2, 1], human-motion priors [26], and single-axis online refinement [28]. These methods demonstrate that interaction is a reconstruction signal. Our difference is the role of the signal: our method uses each interaction to update a persistent multi-part URDF memory, and that memory determines what interaction should be attempted next.

## III. Methodology

### A. Problem Formulation and System Overview

We present **Act2See**, a closed-loop real-time framework that incrementally reconstructs an explicit URDF of an unknown articulated object through robot interaction, as illustrated in Fig. 2. A robot arm (a Franka Panda in simulation, a UFactory XArm 6 in the real world) with a single fixed RGB-D camera faces an articulated object and is given a single high-level objective—*"fully explore the object in front of you."* No prior CAD model, URDF, or category label is assumed. The agent must recover an explicit kinematic model in which each part carries fused geometry and an oriented bounding box (OBB), and each joint carries a type (revolute or prismatic), a 3D axis, an origin, and a current articulation state.

The defining difficulty is observability: a closed drawer and a hinged door are geometrically near-identical until one is moved, and a joint's axis and range are expressed only through the part's motion under contact. Passive single-view reconstruction is therefore under-determined, and we cast the task as *active perception*: the robot must choose actions that simultaneously advance exploration and render the latent kinematics observable. A central design commitment follows: **every control decision uses only vision and robot proprioception**. RGB-D observations, joint positions, and the end-effector pose computed from forward kinematics; simulator ground truth is used only for offline evaluation and never enters the loop.

*a) Closed-loop formulation.:* At iteration $t$ the agent maintains a partial URDF $\mathcal{M}_t = \{\mathcal{P}_t, \mathcal{J}_t\}$, where $\mathcal{P}_t$ is the set of reconstructed parts (each with fused geometry and OBB) and $\mathcal{J}_t$ the set of estimated joints, and observes RGB-D $\mathcal{O}_t$. Three tightly coupled modules advance the model:

$$\begin{aligned} d_t &= \pi(\mathcal{O}_t, \mathcal{M}_t), \\ \xi_t &= \tau(d_t, \mathcal{M}_t), \\ \mathcal{M}_{t+1} &= \textsc{Reconstruct}(\mathcal{O}_{t+1}, \mathcal{M}_t, \xi_t), \end{aligned} \tag{1}$$

a VLM **mission planner** $\pi$ (Sec. III-B) that selects the next interaction from the current belief, a **action grounder module** $\tau$ (Sec. III-C) that realizes that decision as motion respecting the estimated structure, and an **online reconstruction module** (Sec. III-D) that fuses the resulting observations back into the URDF. Executing $\xi_t$ yields the next observation $\mathcal{O}_{t+1}$, and the loop repeats until the planner judges the object explored. The decision is the structured tuple

$$d_t = \langle target_t,\ action_t,\ constraint_t,\ cue_t \rangle, \tag{2}$$

detailed in §III-B. What makes the system *active* rather than a feed-forward detect-then-reconstruct pipeline are two coupling edges within this loop: a *feed-forward* structural cue passed from action into reconstruction, and a *feed-back* belief rendered from reconstruction into the planner.

We refer to the full $\pi \rightarrow \tau \rightarrow$ Reconstruct loop as *closed-loop sequential planning*, and to the maintained URDF $\mathcal{M}_t$ as the *structured representation* (equivalently, *structured memory*) on which the planner conditions. Ablating either ingredient is the basis of our planning experiments in §IV.

We assume that each interaction moves a single articulated part relative to the rest of the object, and that the induced motion is a single-degree-of-freedom rigid motion, either revolute or prismatic. This assumption gives each interaction a well-defined motion signal for joint estimation and pose-aligned fusion. We discuss extensions to simultaneous multi-part motion and movable bases in §V.

### B. VLM Mission Planner

The mission planner $\pi$ is the outer loop. It conditions on a structured multi-modal prompt and emits the decision $d_t$ of Eq. (2). Crucially, the maintained URDF is used not only as output but as a *planning input*: the planner reasons over the current belief rather than the raw image alone, which realizes the feed-back coupling edge of Eq. (1). Concretely, at each step the planner receives (i) the current RGB-D observation, (ii) the target part under interaction, (iii) the joint type and axis estimated between that part and the static base, and (iv) a log of past attempts.

*a) Explicit URDF as persistent memory.:* The model $\mathcal{M}_t$ is the system's persistent memory. It is initialized with a single static root part representing the object base, i.e., the non-moving body relative to which articulated parts move; as interaction proceeds, the reconstruction module (§III-D) appends each newly moved component as a part node with its fused mesh and OBB, and each confirmed articulation as a joint with type, axis, and state. Because the representation is an explicit kinematic graph rather than a latent embedding, it is interpretable to the planner and directly consumable by the trajectory module. The persistent record lets the planner spend its finite interaction budget on unresolved structure rather than re-exploring confirmed parts.

*b) Decision space.:* The decision $d_t$ of Eq. (2) bundles four fields: $target_t$, the part or handle to interact with next; $action_t$, the interaction primitive (below); $constraint_t$, whether the motion is unconstrained (first contact, no axis known) or constrained by an already-estimated axis; and $cue_t$, the structural cue handed to the action grounder, i.e. a 2D handle mask before an axis is known, or the estimated joint axis once it is available. The $action$ field selects one of two interaction primitives or termination:

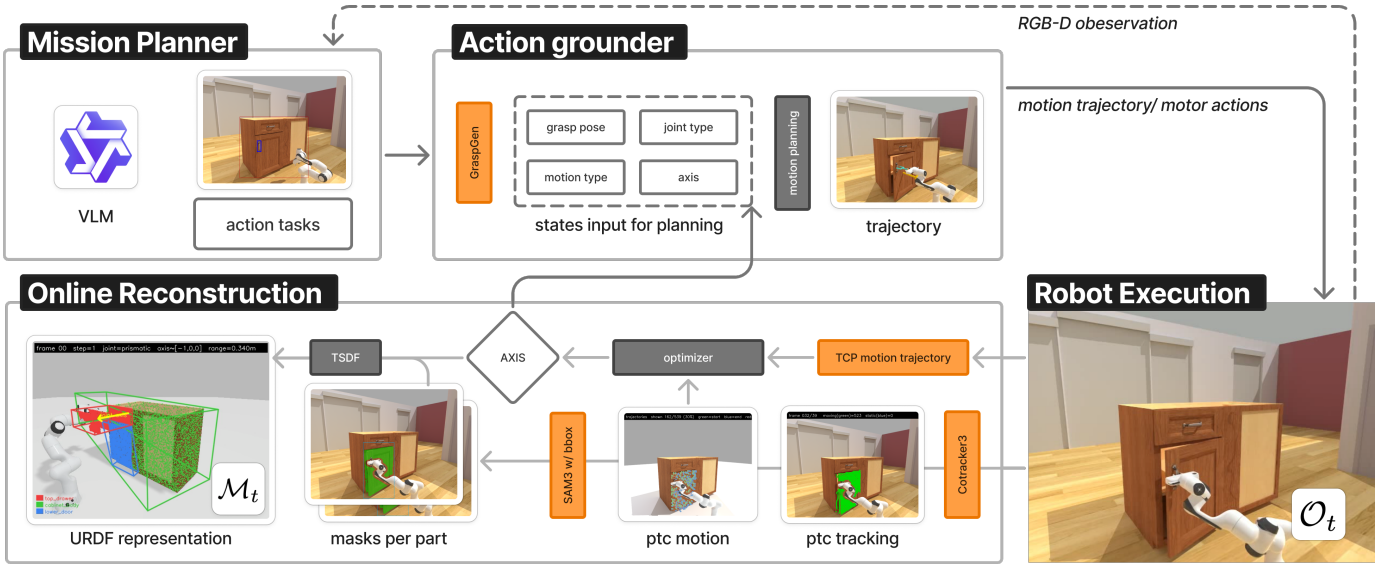



Fig. 2: Overview of the Act2See system. A robot arm aims to fully explore the object(s) through interaction and reconstruct an articulated digital twin in an online, closed-loop manner.

- PROBE — a short *probe* that excites the target just enough to (re-)estimate its joint axis and type;
- EXECUTE — a larger *pull/arc* along the confirmed axis that exploits the estimated structure to drive the part and accumulate geometry;
- SKIP — terminates the current target, triggered either when the success criterion stated in the prompt is met or when the interaction budget (five attempts) is exhausted.

### C. Action Grounder

The action grounder $\tau$ converts the planner's discrete decision $d_t$ into executable robot motion. Its input is the planner-selected target, the current RGB-D observation, and the current articulated axis information when available; its output is an executed trajectory $\xi_t$, together with interaction feedback such as grasp success, slip, stall, and the recorded TCP trajectory. The module does not decide what object should be explored next; instead, it grounds the mission planner's intent and currently perception information into grasping, probing, or kinematically constrained manipulation.

*a) Grasp selection and grip confirmation.:* Given a handle or part mask from the planner, the action grounder combines the mask with the RGB-D observation to obtain a target handle point cloud $\mathcal{H}$. This point cloud defines the contact region but not a grasp pose. We therefore query GraspGen [24] on $\mathcal{H}$ to generate candidate parallel-jaw grasps, then re-rank them using lightweight geometric criteria that balance approach direction, centering on the handle, and finger alignment. After selecting the best grasp, we refine its depth along the grasp approach axis so that the fingertips close slightly behind the handle front face. The output of this stage is either a confirmed grasp pose or a failed-grasp signal; an empty closure is treated as a missed grasp and returned to the planner for retry.

*b) Probe and exploit motions.:* Once a grasp is confirmed, the action grounder chooses the motion primitive according to the current joint belief. If no reliable joint type or axis is available, it executes a short probing pull along the local outward-facing direction of the target part. The purpose of this probe is to elicit motion and provide a point cloud motion information and TCP trajectory to the reconstruction module, rather than to fully actuate the object. If a joint type and axis have already been estimated, the action grounder instead executes an exploit motion constrained by the recovered kinematics: a linear pull along the translation axis for prismatic joints, or an arc motion around the hinge axis for revolute joints. The output of this stage is the executed trajectory $\xi_t$ and the corresponding robot proprioceptive trace.

*c) Online interaction monitoring.:* During execution, the action grounder monitors low-level feedback to detect ineffective contacts. A slip condition is inferred from an unexpected increase in gripper width, indicating that the handle has escaped the fingers; a stall condition is inferred from near-zero TCP displacement while the gripper remains closed, indicating that the part is jammed, blocked, or already at its limit. On either signal, the module halts execution and returns a failure status to the planner, which may request another grasp or probe. Otherwise, the completed trajectory and TCP trace are passed to the reconstruction loop for model update.

### D. Online Reconstruction

The reconstruction module implements RECONSTRUCT$(\cdot)$ in Eq. (1), converting the RGB-D stream from each interaction into an incremental URDF update. It operates online: rather than fusing raw frames in the camera frame, the module first estimates how the grasped part moves, then uses the recovered kinematics to align observations into a common object frame before TSDF fusion. The same motion that reveals the hidden

articulation thus also provides the transformation needed to reconstruct the part while it is moving. We organize the loop into four stages: (i) extracting motion evidence and estimating the joint axis; (ii) confirming the joint type; (iii) obtaining dense moving-part segmentation from sparse motion cues; and (iv) pose-aligned fusion and URDF update.

*a) Proprioceptive axis estimation.:* The joint geometry is estimated primarily from the end-effector (TCP) trajectory recorded during the probe. As the part is pushed open, the TCP passively traces the true hinge, yielding a clean, millimeter-accurate articulation signal—substantially more reliable than visual tracking, which we retain only as a fallback. We fit the axis from the probe motion alone and never fold in the subsequent exploit arc, which is driven along the already-estimated axis and corrupted by the commanded direction and by slip. Because a single grasp sweeps only a short arc, we estimate the revolute axis by *accumulating* TCP traces from several segments grasped at different contact points: within a segment the grasp is rigid and traces a circular arc, and across segments all arcs share the same hinge, so they become concentric once expressed in the axis frame. Each additional grasp therefore tightens the estimate.

*b) Joint-type confirmation.:* Each newly moved part is initially marked with an *unknown* joint type. The observed trajectory is tested against both prismatic and revolute models; when the evidence is insufficient to separate them, the module keeps the type unknown and returns this uncertainty to the planner, which triggers additional probing. Once enough motion has accumulated, the type is fixed and remains unchanged for the rest of the interaction, while later observations refine only the continuous kinematic parameters. This prevents unstable type switching while still letting the recovered URDF improve with further interaction.

*c) Motion-guided segmentation and pose-aligned fusion.:* On first contact, before the part has moved, a detection keyed to the interaction handle localizes the target. Thereafter, segmentation is motion-guided: **CoTracker3** [14] tracks salient surface points, the moving points define a bounding box, and that box prompts **SAM** [3, 27] to return a complete moving-part mask. Intersecting the mask with the depth frame yields a dense per-part point cloud. Because the part moves during interaction, raw clouds lie in different body frames; we register each back to the rest configuration with the estimated articulation transform $T_t^{-1}$ and fuse the aligned clouds into a per-part **TSDF** volume [5], from which marching cubes extracts the mesh. Fusion proceeds incrementally: when the motion residual indicates the current axis no longer explains incoming frames, the axis is re-estimated and the affected TSDF volume rebuilt, keeping the online model consistent without requiring the full sequence in advance. The refined geometry and joint parameters are written back into $\mathcal{M}_{t+1}$ and rendered for the planner, closing the loop.

## IV. Experiment and Analysis

In this section, we describe the experimental setup and evaluation methods, as illustrated in Fig. 3. We build a benchmark in the SAPIEN simulator with different types of assets and implement our method on a real-world cabinet setup using a UFactory XArm 6 robot arm and an Intel RealSense D435i depth camera.

We perform a reconstruction experiment comparing Act2See with other SOTA articulation object reconstruction methods. We also design an ablation study to quantify the contribution of each module on sequential manipulation tasks. We evaluate the full system with quantitative results on success rate and manipulation efficiency, together with a qualitative analysis of failure cases.

### A. Experimental Setup and Benchmark

We set up several cabinets from the PartNet-Mobility dataset within the SAPIEN simulator. The primary types are classified as drawers and cabinet doors with different orientations. Each cabinet case contains one to five individual interactive objects arranged in two opposing combinations.

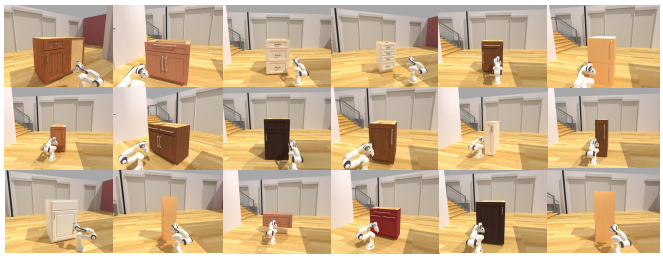
In our simulator experiments, once the environment is set up, we provide the VLM with only one high-level prompt: ”Explore the cabinet in front of you.” Task completion is defined by two distinct scenarios based on the type of interaction: For pull-based interaction objects, the policy must pull the object to its limit and then close it; for cabinet door objects, the policy will be asked to open the door to at least 45 degrees or its limitation, and then close it.(Information regarding joint constraints can be found in the original URDF of the dataset, which can be used as the ground truth for verification)

We have also expanded our experiments to real-world robotic arm interaction scenarios. The real environment experimental setup used the following equipment: A UFactory xArm 6 robotic arm for interaction and an front global view Intel RealSense depth camera for perception. Two desktop cabinets bought from Amazon act as interaction objects. We used this configuration to test our Act2See system within the real interaction environment. More detailed test results in real world will be presented in a future extended article and project website.

### B. Online Reconstruction Performance

We compare Act2See with two state-of-the-art reconstruction methods, ArticulateAnything and VideoArtGS, on articulation-axis accuracy, joint-type classification, and latency. For a controlled comparison, a Panda arm manipulates a single object in SAPIEN and the recorded RGB-D stream is fed to every method; since the clips come from successful Act2See interactions, the comparison isolates reconstruction quality given a usable interaction rather than each method's ability to elicit motion.

*a) Online vs. offline latency:* Our protocol is *online*: frames stream in at $1\,\mathrm{Hz}$ and reconstruction runs from the first frame, whereas the offline baselines must wait for the full sequence. We report two timings in Table I: $133.11\,\mathrm{s}$ is the full-episode wall-clock time (almost all of it the interaction itself, since reconstruction runs concurrently), while $4.23\,\mathrm{s}$ is the marginal latency to finalize a reconstruction once the



(a) Simulation tasks

(b) Cabinet A

(c) Cabinet B

(d) Cabinet A+B

Fig. 3: Simulation and real-world task setups.

stream ends. The baselines have no analogue to $4.23\,\mathrm{s}$—they can only start after the full stream is available—so for post-stream latency $4.23\,\mathrm{s}$ is the like-for-like number, one to two orders of magnitude below the baselines.

*b) Reconstruction quality:* Act2See returns a usable estimate on every test case ($100\%$ valid rate) for both joint types. On prismatic joints its axis error is competitive with the strongest baseline. On revolute joints, ArticulateAnything reports $0.00°$ axis error because it is a retrieval-based method, and on our PartNet-Mobility test set its retrieval step recovers the ground-truth asset itself, so its revolute numbers reflect an oracle-like upper bound rather than reconstruction from observations. Among methods that estimate joint parameters from observations without access to the asset library, Act2See attains the best revolute axis and position accuracy. Robustness to early mis-typing comes from axis-triggered recomputation (§III-D): when incoming frames stop agreeing with the current hypothesis, the estimate is rolled back and the dual prismatic/revolute fits are re-arbitrated, so an early wrong type is corrected within the same interaction rather than committed to.

### C. Ablation Study: Reconstruction as Memory for VLM Planning

We design a four-setting ablation to isolate the two ingredients introduced in §III: *closed-loop sequential planning*—the $\pi \to \tau \to$ RECONSTRUCT feedback loop of Eq. (1)—and *structured memory*—the maintained URDF $\mathcal{M}_t$ on which the planner conditions. All settings share the same VLM backbone (Qwen3-VL-32B) , action space, and task specification, and differ only in which of these two ingredients is available to the planner at each step.

**(1) Act2See w/o sequential planning.** The VLM receives only the initial RGB-D frame and emits a complete action sequence that is executed without revision, removing the feedback loop and with it any closed-loop correction. **(2) Act2See w/o reconstruction.** The feedback loop is restored, but the planner re-plans at every step from the *current RGB-D frame alone*: the reconstruction module is disabled, so no persistent structured memory is maintained. Since the maintained URDF *is* the product of reconstruction, ablating reconstruction here is precisely what removes structured memory while keeping closed-loop feedback intact. **(3) Act2See (full).** At every step the reconstruction module ingests the latest RGB-D frame and updates the articulated representation—per-part geometry, joint-type posterior, joint axis with covariance $(\sigma_p, \sigma_\theta)$, and current articulation state—on which the VLM re-plans, conditioned jointly on the current observation and a rendered view plus structured prompt derived from $\mathcal{M}_t$; this is our full method. **(4) Act2See (full) w/ transferred memory.** After a complete rollout under (3), we reset the environment to its initial state but retain the full URDF in memory, re-rendering it and including the structured prompt at the start of a second pass that otherwise proceeds identically to (3).

Read in sequence, the four settings introduce one ingredient at a time: closed-loop feedback ($1 \to 2$), structured memory ($2 \to 3$), and cross-episode transfer of that memory ($3 \to 4$). The $(2) \to (3)$ gap is the load-bearing comparison for the value of structured, belief-aware memory over raw closed-loop feedback, and the $(3) \to (4)$ gap tests whether the reconstruction captures object-level rather than rollout-specific information. We report all settings under two goal configurations: *same-goal*, which measures policy improvement under accumulated reconstruction, and *cross-goal*, which measures the transferability of the reconstruction itself, decoupled from goal-specific learning.

Two patterns in Table II are worth noting. First, on prismatic joints the *w/o sequential planning* setting attains the fewest attempts ($1.00$) yet a lower completion rate ($81.8$ vs. $90.9$): a single open-loop shot succeeds when it succeeds, but fails outright otherwise, whereas the full loop trades a few extra attempts for substantially higher reliability. Second, revolute joints collapse to $0$ completion without sequential planning, confirming that closed-loop feedback is a prerequisite—not a refinement—for the harder articulation type.

### D. Real World Implementation

In our real-robot setup, we deploy the proposed system on a UFACTORY xArm6. After hand–eye calibration, the robot performs autonomous cabinet exploration using only a single third-person RGB-D camera. As shown in Fig. 4, this setup enables the arm to perceive the cabinet, localize interaction targets, and execute exploration actions without relying on additional external sensors or multi-camera observations.

## V. Discussion

**Limitations and future works.** Three limitations bound the present system. First, the single-part, single-DoF, stationary-base assumption of §III-A keeps reconstruction and memory tractable but fails in two regimes: under a *kinematic chain* or multi-segment linkage—e.g., a nested panel carried by an

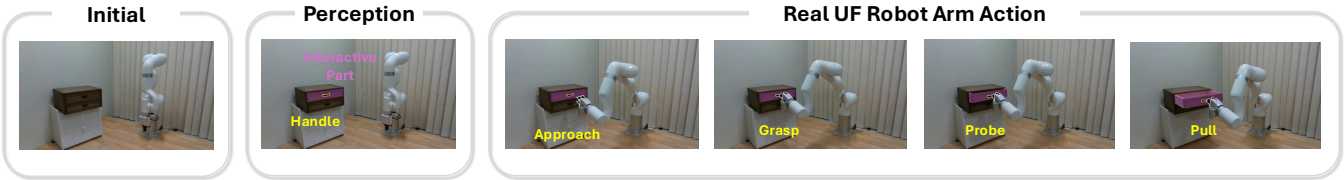



Fig. 4: Robot deployment in the real world setting

TABLE I: **Quantitative evaluation of URDF reconstruction quality.** Metrics are reported as mean $\pm$ std over all test videos. Lower ($\downarrow$) is better on all metrics, and the best results are highlighted in bold. N/A stands for "Not Applicable," as ArticulateAnything is a retrieval-based reconstruction method. The average speed of our method is presented as follows: values outside the parentheses represent the additional reconstruction convergence time required after the manipulation finishes its operations, which is the same with other offline method; while values inside the parentheses represent the total system runtime for both the reconstruction operations and the algorithm as a whole.

| Method | Prismatic Joint Estimation | | Revolute Joint Estimation | | | Speed (s)$\downarrow$ |
|---|---|---|---|---|---|---|
| | Axis (°) | Valid Rate (%) | Axis (°) | Position (cm) | Valid Rate (%) | |
| ArticulateAnything | 9.00$\pm$27.00 | 76.9 | 0.00 | N/A | 100 | 181.62 |
| VideoArtGS (RGB input) | **1.84$\pm$1.68** | 76.9 | – | – | 0 | 874.30 |
| **Ours** | 2.60$\pm$1.69 | **100** | 13.15$\pm$16.60 | **10.06$\pm$14.31** | **100** | **4.23(133.11)** |

TABLE II: Quantitative results of the ablation study across planning tasks in simulation. We ablate the two ingredients introduced in §III: *closed-loop sequential planning* (the $\pi \rightarrow \tau \rightarrow$ RECONSTRUCT feedback loop) and *structured memory* (the maintained URDF $\mathcal{M}_t$). A: task completion rate (%, $\uparrow$); B: number of interaction attempts ($\downarrow$).

| Method | Prismatic Joint | | Revolute Joint | |
|---|---|---|---|---|
| | A $\uparrow$ | B $\downarrow$ | A $\uparrow$ | B $\downarrow$ |
| Act2See w/o sequential planning | 81.8 | **1.00** | 0 | **1.00** |
| Act2See w/o reconstruction | 72.7 | 3.73 | 45.5 | 4.45 |
| Act2See (full) w/ transferred memory | 81.1 | 1.45 | **81.9** | 2.25 |
| **Act2See (full)** | **90.9** | 2.18 | **81.9** | 2.18 |

outer door—a single rest-pose alignment no longer holds; and when the *base itself moves*—e.g., the cabinet body is pushed—the stationary-base premise breaks and world-frame fusion is corrupted. A complementary future direction is to connect Act2See with geometry-enhanced video world models such as GEM-4D [34]: given the same initial third-person RGB-D observation, our interaction-driven URDF reconstruction could provide real and simulated data, as well as structured validation signals, for evaluating whether generated rollouts respect articulated kinematics.

Second, revolute joints remain substantially harder than prismatic ones, in both simulation and on the real robot: while drawers are operated reliably, hinged doors are estimated and manipulated far less consistently, and the lowest door in our real cabinet could not be opened at all. We attribute this gap to the larger and more constrained workspace required by an arc motion and to the greater sensitivity of revolute axis estimation to contact slip; closing it is an open challenge rather than a solved case, and the bold revolute entries in Table I should be read against this caveat.

Third, parts of our evaluation remain incomplete. Our geometric reconstruction is currently reported only qualitatively: a quantitative simulator evaluation (e.g., Chamfer distance against ground-truth meshes) is not yet available, and on the real robot the motion-guided segmentation becomes unstable during dynamic contact—the per-frame mask is noisy while the part is moving—which degrades the fused per-part geometry. Likewise, several mechanisms that we motivate by design—axis-triggered rollback and the accumulation of reconstruction accuracy over successive interactions—are not yet isolated by dedicated ablations; quantifying rollback's contribution (on/off and trigger frequency) and reporting error-versus-interaction-count curves, along with stabilizing motion-time segmentation and a quantitative geometric benchmark, are immediate next steps.

**Conclusion.** We presented Act2See, a closed-loop framework that incrementally reconstructs an explicit URDF of an unknown articulated object through physical interaction, where each action serves as an active source of reconstruction signal rather than a downstream consumer of perception. By maintaining the URDF as persistent, structured memory, a VLM-based planner allocates its interaction budget toward unresolved structure and reasons about parts that are not currently visible, while an online reconstruction module fuses streaming RGB-D observations into per-part geometry and is designed to self-correct its joint estimates through axis-triggered recomputation. In the SAPIEN simulator, structured memory yields gains over open-loop and representation-free baselines on prismatic tasks. On a real UFactory xArm 6, the full system operates prismatic drawers (3/3) while revolute

doors remain harder (1/3), mirroring the simulation trend. Two honest gaps frame the road ahead: revolute articulation is not yet robust, and geometric reconstruction is so far only qualitative. Extending Act2See to multi-DoF kinematic chains, movable bases, and a broader range of articulated categories under heavier occlusion together with a quantitative geometric benchmark is a natural next direction.

## REFERENCES


[1] Russell Buchanan, Adrian Röfer, João Pousa de Moura, Abhinav Valada, and Sethu Vijayakumar. Online estimation and manipulation of articulated objects. *Autonomous Robots*, 50(13):1–19, 2026. doi: 10.1007/s10514-025-10233-4.

[2] Russell Buchanan et al. Online estimation of articulated objects with factor graphs. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[3] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

[4] Zoey Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Karthikeya Vemuri, Alan Wu, Dieter Fox, and Abhishek Gupta. URDFormer: A pipeline for constructing articulated simulation environments from real-world images. *arXiv preprint arXiv:2405.11656*, 2024.

[5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*, pages 303–312. ACM, 1996. doi: 10.1145/237170.237269.

[6] Karthik Desingh, Shiyang Lu, Anthony Opipari, and Odest Chadwicke Jenkins. Factored pose estimation of articulated objects using efficient nonparametric belief propagation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7221–7227, 2019.

[7] Ben Eisner, Harry Zhang, and David Held. FlowBot3D: Learning 3D articulation flow to manipulate articulated objects. In *Robotics: Science and Systems (RSS)*, 2022.

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

[9] Karol Hausman, Scott Niekum, Sarah Osentoski, and Gaurav S. Sukhatme. Active articulation model estimation through interactive perception. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3305–3312, 2015. doi: 10.1109/ICRA.2015.7139655.

[10] Ajinkya Jain, Rudolf Lioutikov, Caleb Chuck, and Scott Niekum. ScrewNet: Category-independent articulation model estimation from depth images using screw theory. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 13670–13677, 2021.

[11] Hanxiao Jiang, Binghao Huang, Ruihai Wu, Zhuoran Li, Shubham Garg, Hooshang Nayyeri, Shenlong Wang, and Yunzhu Li. Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2024.

[12] Taoran Jiang, Yixuan Guan, Liqian Ma, Jing Xu, Jiaojiao Meng, Weihang Chen, Zecui Zeng, Lusong Li, Dan Wu, and Rui Chen. DexSim2Real$^2$: Building explicit world model for precise articulated object dexterous manipulation. *arXiv preprint arXiv:2409.08750*, 2024.

[13] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. DITTO: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5616–5626, 2022.

[14] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. 2024.

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

[16] Xiaolong Li, He Wang, Li Yi, Leonidas J. Guibas, A. Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3706–3715, 2020.

[17] 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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[18] Jiong Lin, Lechen Zhang, Kwansoo Lee, Jialong Ning, Judah Goldfeder, and Hod Lipson. AutoURDF: Unsupervised robot modeling from point cloud frames using cluster registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[19] Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. PARIS: Part-level reconstruction and motion analysis for articulated objects. *arXiv preprint arXiv:2308.07391*, 2023.

[20] Qingming Liu, Xinyue Yao, Shuyuan Zhang, Yueci

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

[21] Yu Liu, Baoxiong Jia, Ruijie Lu, Chuyue Gan, Huayu Chen, Junfeng Ni, Song-Chun Zhu, and Siyuan Huang. VideoArtGS: Building digital twins of articulated objects from monocular video. *arXiv preprint arXiv:2509.17647*, 2025.

[22] Liqian Ma, Jiaojiao Meng, Shuntao Liu, Weihang Chen, Jing Xu, and Rui Chen. Sim2Real$^2$: Actively building explicit physics model for precise articulated object manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 11698–11704, 2023. doi: 10.1109/ICRA48891.2023.10160370.

[23] Zhao Mandi, Yijia Weng, Dominik Bauer, and Shuran Song. Real2Code: Reconstruct articulated objects via code generation. *arXiv preprint arXiv:2406.08474*, 2024.

[24] Adithyavairavan Murali, Balakumar Sundaralingam, Yu-Wei Chao, Jun Yamada, Wentao Yuan, Mark Carlson, Fabio Ramos, Stan Birchfield, Dieter Fox, and Clemens Eppner. GraspGen: A diffusion-based framework for 6-dof grasping with on-generator training. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2026.

[25] Chuanruo Ning, Ruihai Wu, Haoran Lu, Kaichun Mo, and Hao Dong. Where2Explore: Few-shot affordance learning for unseen novel categories of articulated objects. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[26] Adrian Pfisterer, Xing Li, Vito Mengers, and Oliver Brock. A helping (human) hand in kinematic structure estimation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[27] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[28] Xi Wang, Tianxing Chen, Qiaojun Yu, Tianling Xu, Zanxin Chen, Yiting Fu, Ziqi He, Cewu Lu, Yao Mu, and Ping Luo. Articulated object manipulation using online axis estimation with SAM2-based tracking. *arXiv preprint arXiv:2409.16287*, 2025.

[29] WenBo Xu, Liu Liu, Li Zhang, Ran Zhang, Hao Wu, Dan Guo, and Meng Wang. KineDiff3D: Kinematic-aware diffusion for category-level articulated object shape reconstruction and generation. *arXiv preprint arXiv:2510.17137*, 2025.

[30] Sylvia Yuan, Ruoxi Shi, Xinyue Wei, Xiaoshuai Zhang, Hao Su, and Minghua Liu. LARM: A large articulated-object reconstruction model. *arXiv preprint arXiv:2511.11563*, 2025.

[31] Boyuan Zhang, Yuxuan Wang, Yizhou Wang, Wei Wang, and Zhenliang Zhang. Active kinematic modeling for precise manipulation of unseen articulated objects. *IEEE Robotics and Automation Letters*, 11(3):3254–3261, 2026. doi: 10.1109/LRA.2026.3656781.

[32] Hang Zhang, Qijian Tian, Jingyu Gong, Daoguo Dong, Xuhong Wang, Yuan Xie, and Xin Tan. DailyArt: Discovering articulation from single static images via latent dynamics. *arXiv preprint arXiv:2604.07758*, 2026.

[33] Harry Zhang, Ben Eisner, and David Held. FlowBot++: Learning generalized articulated objects manipulation via articulation projection. In *Conference on Robot Learning (CoRL)*, 2023.

[34] Kaichen Zhou, Yuzhen Chen, Fangneng Zhan, Hang Hua, Grace Chen, Xinhai Chang, Ao Qu, Yilun Du, Zhuang Liu, Paul Pu Liang, and Mengyu Wang. Gem-4d: Geometry-enhanced video world models for robot manipulation. *arXiv preprint arXiv:2605.22882*, 2026.

[35] Kaichen Zhou, Yuhan Wang, Grace Chen, Xinhai Chang, Gaspard Beaudouin, Fangneng Zhan, Paul Pu Liang, and Mengyu Wang. Page-4d: Vggt-4d perception via disentangled pose and geometry estimation. *arXiv preprint arXiv:2510.17568*, 2026.