Active Kinematic Modelling for Precise Manipulation of Unseen Articulated Objects

Anonymous

***Abstract*— Precisely manipulating an unseen articulated object is a common yet challenging task, primarily due to the ambiguity in predicting the affordance and the articulation model of the object. Current data-driven methods learn spurious correlations through passive observation, which limits their adaptability to new objects and precision in manipulation tasks. To achieve robust generalization and precise manipulation, we propose a novel Active Kinematic Modelling method (AKM), where robots actively interact with objects to overcome spurious correlations and achieve precise kinematic modelling. Our method is composed of three stages. First, hypothesis-driven exploration identifies contact points where interaction with the object triggers detectable relative motion between its parts. Next, unsupervised articulation modelling analyzes the recorded motion frames to construct an articulation model of the object. Finally, this articulation model facilitates closed-loop manipulation with on-the-fly relocalization at any joint configuration and replanning of trajectories, ensuring precise execution. We validate AKM in simulation across a diverse benchmark of 116 articulated objects and demonstrate its generalization ability by deploying it to a real-world robotic system, consistently achieving sub-centimeter manipulation accuracy. The full code is available at https://anonymous.4open.science/r/AKM.**

I. INTRODUCTION

Robots operating in human environments must precisely manipulate unseen articulated objects, such as doors, drawers, or cabinets, to perform tasks safely and effectively. For instance, placing a wine glass in a novel cabinet requires precise and smooth operation to avoid potential collisions. However, in complex real-world environments, uncertainties are common. It is challenging to accurately determine where to grasp an articulated object to open it, what kinematic model it follows, and its current joint configuration. These factors are difficult to predict solely from passive observations [1]. The uncertainties limit the ability of the robot to achieve precise manipulation of novel objects.

To address this challenge, researchers have investigated two primary methodologies: passive and active approaches. Passive approaches, such as Ditto[2], GeneralFlow[3], GapartNet[4], and Where2Act[5], aim to understand an articulated object from passive observations, typically utilizing point clouds or RGB images as input. However, these methods often capture spurious correlations [1], such as incorrectly identifying a sliding joint as a rotating one based solely on the location of the handle. Additionally, the model may incorrectly conclude that a component resembling a handle can be used to open the object, even when that component is non-functional. These errors limit their ability to handle new objects effectively, making precise manipulation of new objects difficult. To overcome these limitations, active approaches, like GAMMA[6] and Sim2Real2[7], [8], have been developed. These approaches mitigate spurious correlations by physically interacting with objects. Nevertheless, these approaches often rely on task-specific models to generate preliminary articulation model predictions, incorporating on-the-fly fine-tuning of joint parameters. However, these methods fail to work out when the kinematics of the target object substantially deviate from those in the training dataset.

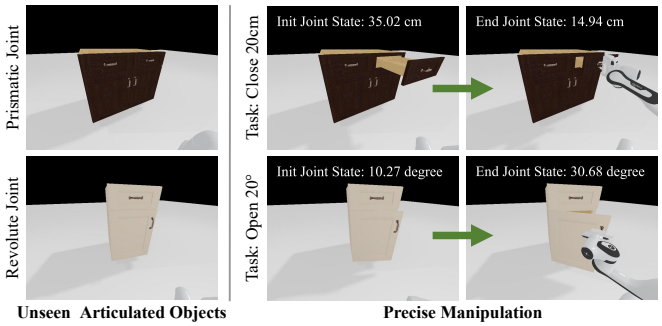

Fig. 1. **Task Settings.** Our goal is to accurately manipulate an unseen articulated object to achieve a target state, such as closing a drawer from 35 cm to 15 cm or opening a cabinet from 10° to 30°.

Humans do not instinctively recognize the kinematic patterns of unfamiliar articulated objects; instead, they rely on active interactions, leveraging visual perception and sensorimotor coordination to effectively model novel objects. Similarly, we propose that robots actively engage in physical interactions to reduce uncertainties and spurious correlations. Furthermore, robust and precise kinematic modelling can be achieved using only general-purpose visual perception and sensorimotor abilities, bypassing the need for specialized models trained to predict kinematic patterns. This is enabled by recent advancements in visual and grasping foundation models [9], [10], [11], [12], which exhibit strong generalization capabilities. Therefore, we aim to integrate these capabilities into a structured process, building precise kinematic models in a generalizable way.

To this end, we introduce Active Kinematic Modelling (AKM), a method that constructs precise kinematic models through a process of active interactions. The process begins with the robot actively exploring the object, guided by generated hypotheses and failure suppression mechanisms. Exploration concludes when a contact point that produces observable relative motion of object parts is identified. These observed motion frames enable unsupervised articulation modeling, resulting in a precise kinematic model of the object. This model further enables closed-loop manipulation of the articulated object by frequently relocalizing its current joint configuration. This allows the robot to continuously

refine planned trajectories to accurately achieve the target joint state. We extensively test our method in a simulated environment with diverse articulated objects, load poses, movable joints, and manipulation tasks. Qualitative results demonstrate the effectiveness and robustness of AKM. Real-world experiments also confirmed its high transferability.

In summary, our main contributions are fourfold:

1) We propose a hypothesis-driven exploration strategy to efficiently detect an effective interaction point on an unseen articulated object.
2) We design a fully unsupervised pipeline for articulation modeling that uses a coarse-to-fine strategy to achieve sub-centimeter modelling accuracy without requiring any category-specific priors.
3) We develop a closed-loop strategy that leverages the constructed model to perform state relocalization and iterative refinement, enabling precise control.
4) We extensively validate our method in both simulation and real-world settings, demonstrating its effectiveness in precisely manipulating unseen articulated objects.

II. Related Work

We review related work in three key areas—affordance learning (Section(II-A)), articulation modeling (Section(II-B)), and articulated object manipulation (Section(II-C))—to tackle core challenges of our method. These areas focus on identifying valid interactions, constructing articulation models, and achieving precise control for robust generalization to unseen objects.

A. Affordance for Articulated Objects

Affordance learning aims to identify potential interactions from sensory inputs. For articulated objects, this entails identifying permissible motions. Passive methods infer affordances from static observations, using contact-based [13], [14], [15], flow-based [16], [17], [3], or part-based [4], [18] approaches to predict contact points, motion flows, or actionable parts, respectively. However, these methods face challenges due to the ambiguity of static views, which obscure kinematic constraints. In contrast, active methods overcome this ambiguity through physical interaction with objects [19], [20], [5], [21], [22]. Our work falls into this category, structuring active interaction as a hypothesis-testing process, using contact analogy for initial hypothesis generation and failure suppression for hypothesis updates.

B. Articulation Modelling

Estimating an articulation model involves identifying the joint parameters (type, axis, pivot). Learning-based methods [2], [23] train on large datasets [24], [25] to regress these parameters from visual input. While effective for in-distribution objects, they struggle with novel categories due to their reliance on learned priors. Optimization-based methods derive models from motion. Classic approaches either use handcrafted features and clustering [26], [27], [28] or probabilistic graphical models [29], but can be brittle. Our approach is also optimization-based but leverages modern visual trackers [9] for robust motion analysis. By modeling observed motion based on kinematic constraints, our approach overcomes the generalization limitations of learning-based methods that depend on appearance-based correlations.

C. Articulated Object Manipulation

Manipulating articulated objects extends beyond static grasp poses [11], requiring plans that account for the object's kinematic constraints. Most methods generate open-loop trajectories from affordance predictions [14], [16], [4], but these approaches lack error correction, leading to imprecise manipulation. Closed-loop methods improve precision by incorporating feedback [30], [31]. However, some rely on mesh reconstruction and control techniques like MPC [7], which are computationally complex and expensive. Our approach achieves precise closed-loop manipulation without complex reconstruction, leveraging a lightweight articulation model for rapid state relocalization and iterative refinement.

III. Problem Formulation

We aim to address the problem of precise manipulation for unseen articulated objects. The object's category, placement, movable joint locations, and joint types are unknown. The algorithm needs to identify the joints that can be manipulated and precisely open or close them by a specified distance. The algorithm leverages continuous RGB-D observations from a fixed-position RGB-D camera. We focus on the two most common types of joints: prismatic and revolute. A prismatic joint is defined by its axis, a unit vector $\mathbf{a}^p \in \mathbb{R}^3$. A revolute joint is characterized by its rotation axis, a unit vector $\mathbf{a}^r \in \mathbb{R}^3$, and a pivot point on that axis, $\mathbf{c}_p \in \mathbb{R}^3$. The joint state, s, represents the displacement or angle relative to the fully closed configuration.

IV. Method

Our method consists of three stages, as shown in Figure(2). Stage 1: hypothesis-driven exploration discovers the affordances of an object by testing interaction hypotheses. Stage 2: unsupervised articulation modeling builds an articulation model from the interaction data. Stage 3: closed-loop manipulation uses this articulation model for precise control.

A. Hypothesis-Driven Exploration

To identify object affordances, the robot conducts hypothesis-driven exploration, where each hypothesis corresponds to a candidate contact point $\mathbf{u} = (u, v)$ in a $H \times W$ image, predicted to induce relative motion of object parts upon force application. The hypothesis space is represented as an actionability map $\mathcal{A} \in \mathbb{R}^{H \times W}$, where $\mathcal{A}(\mathbf{u})$ denotes the probability of successful interaction at point $\mathbf{u}$. The robot iteratively tests high-probability hypotheses while suppressing regions associated with failed interactions.

Hypothesis Generation through Contact Analogy. We initialize the actionability map using an analogy-based approach. For a given target image I_{tgt}, we employ RAM [14] to retrieve a similar reference image I_{ref} along with its

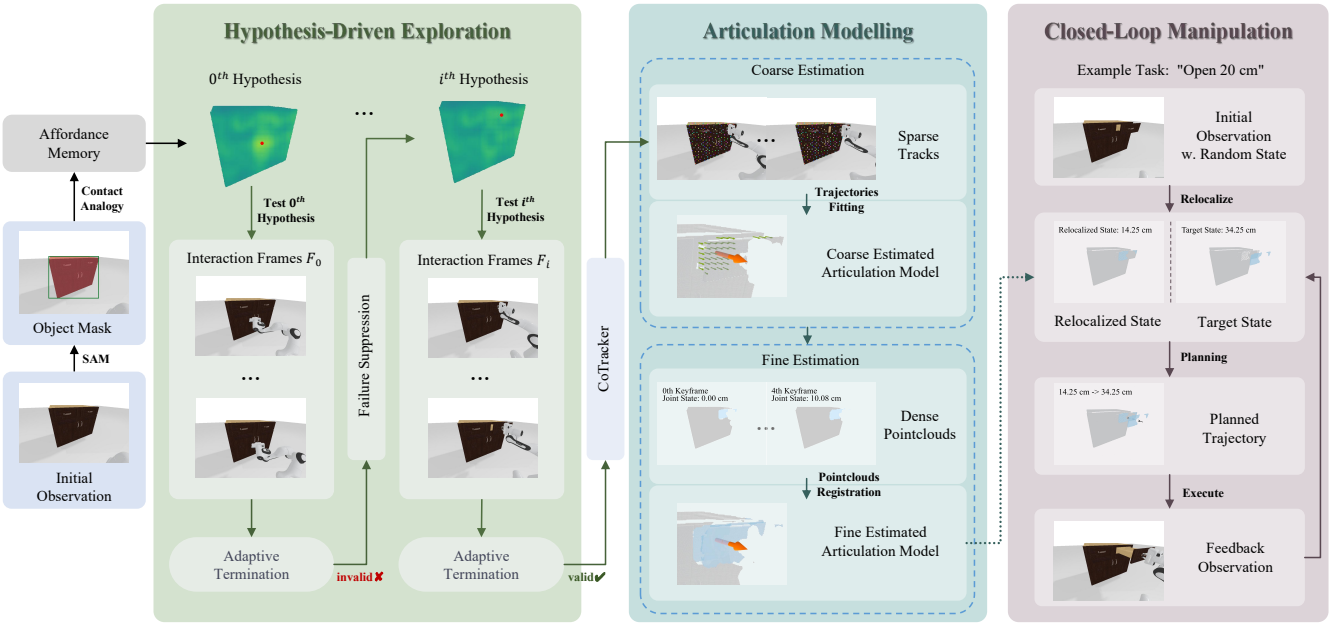

Fig. 2. **AKM Overview.** Our method follows a three-stage pipeline. **1) Hypothesis-Driven Exploration:** The robot generates an actionability map via contact analogy and iteratively tests contact points. Failed attempts lead to the suppression of the local area (failure suppression), guiding the search until a successful interaction is found. **2) Unsupervised Articulation Modeling:** Using the recorded motion sequence, the system tracks sparse points, clusters them into static/mobile parts, and performs coarse-to-fine optimization to estimate the articulation model. **3) Closed-Loop Manipulation:** The robot uses the constructed model to relocalize the current state of the object and executes iterative actions to reach the target state precisely.

associated successful contact point $\mathbf{u}^*_{ref}$. The initial score for each contact point $\mathbf{u}$ in the target image is calculated by comparing dense local features with those at $\mathbf{u}^*_{ref}$:

$$\mathcal{A}(\mathbf{u}) = \text{sim}_{\cos}(\phi_{\text{DIFT}}(I_{tgt}, \mathbf{u}), \phi_{\text{DIFT}}(I_{ref}, \mathbf{u}^*_{ref})), \quad (1)$$

where $\phi_{\text{DIFT}}(I, \mathbf{u})$ represents the DIFT [10] feature vector extracted from image I at pixel $\mathbf{u}$, and $\text{sim}_{\cos}(\cdot, \cdot)$ denotes the cosine similarity function.

Hypothesis Testing with Failure Suppression. The robot iteratively tests the hypothesis with the highest actionability score. If an interaction attempt at pixel μ fails (i.e., induces no part motion), the likelihood of revisiting that region is reduced to encourage exploration of untested areas. This is achieved by applying a localized, subtractive Gaussian kernel to the actionability map. For each pixel $\mathbf{u}$ in the map, the update rule is:

$$\mathcal{A}(\mathbf{u}) \leftarrow \mathcal{A}(\mathbf{u}) - \alpha \cdot \exp\left(-\frac{1}{2}(\mathbf{u}-\mu)^T \mathbf{\Sigma}^{-1}(\mathbf{u}-\mu)\right), \quad (2)$$

where α is a suppression factor (set empirically to 0.5) and $\mathbf{\Sigma}$ is a diagonal covariance matrix that governs the spatial extent of suppression. This "lose-shift" strategy prevents the robot from fixating on incorrect hypotheses. The complete exploration pipeline is detailed in Algorithm(1). To execute a pull action, a grasp pose is first selected from those generated by the AnyGrasp detector [11], choosing the one nearest to c_{3d}. The pulling direction $\boldsymbol{d}$ is aligned with the outward-pointing surface normal at c_{3d}, with a default pulling distance Δs of 0.1 m. The threshold α is set to 0.5 by default.

We determine whether an interaction is successful by evaluating the relative displacement of different parts of an object (Algorithm(2)). To achieve this, we uniformly sample points on the initial object mask and track them across interaction frames using CoTracker [9]. The resulting 2D tracks $\mathcal{T}$ are lifted to 3D trajectories, denoted as $\mathcal{J}$. To segment $\mathcal{J}$ into mobile and static parts, we apply spectral clustering [32]. For each 3D trajectory $\mathbf{j} \in \mathcal{J}$, represented as a sequence of points $\{\mathbf{j}_0, \mathbf{j}_1, \ldots, \mathbf{j}_{T-1}\}$, we compute a feature vector $\mathbf{v}_\mathbf{j} \in \mathbb{R}^{3\times(T-1)}$ that captures its displacement relative to the starting point $\mathbf{j}_0$:

$$\mathbf{v}_\mathbf{j} = [\mathbf{j}_1 - \mathbf{j}_0, \ldots, \mathbf{j}_{T-1} - \mathbf{j}_0]. \quad (3)$$

The clustering algorithm groups these trajectories based on the similarity of their displacement features, separating mobile parts from static ones. An interaction is deemed successful if the mean trajectory length of the mobile set

Algorithm 1 Hypothesis-Driven Exploration

Require: Actionability Map $\mathcal{A} \in \mathbb{R}^{H\times W}$, Maximum Number of Iteractions N_{max}, Pulling Direction $\boldsymbol{d}$, Pulling Distance Δs, Current Depth Frame $\mathcal{D}$, Threshold α

for $i \in [1, N_{max}]$ **do**
  Select contact point c_{2d} from $\mathcal{A}$ with highest score
  Backproject c_{2d} to a 3D contact point c_{3d} with $\mathcal{D}$
  Execute a pull action at c_{3d} along $\boldsymbol{d}$ for distance Δs
  Record RGB-D frames of the pulling action as $\mathcal{F}$
  (terminate, $\mathcal{A}'$) $\leftarrow$ UpdateAndCheck($\mathcal{F}, \mathcal{A}, \Delta s, \alpha$)
  if terminate **then**
    break
  end if
end for

Algorithm 2 Adaptive Termination Check

function UPDATEANDCHECK($\mathcal{F}, \mathcal{A}, \Delta s, \alpha$)
Require: RGB-D sequence $\mathcal{F}$, Actionability Map $\mathcal{A}$, Pulling distance Δs, Threshold α
Ensure: A boolean termination flag, an updated $\mathcal{A}'$
  Segment object mask $\mathcal{M}_0$ from the first frame of $\mathcal{F}$
  Uniformly sample N initial pixels within $\mathcal{M}_0$
  Track initial pixels across RGB frames of $\mathcal{F}$ as $\mathcal{T}$
  Lift 2D tracks $\mathcal{T}$ to 3D trajectories $\mathcal{J}$ with $\mathcal{F}$
  Compute trajectory features $\mathbf{v}$ using Equation(3)
  Cluster $\mathcal{J}$ into two groups with $\mathbf{v}$
  Classify the group with larger variance as mobile group $\mathcal{J}^m$, whereas the other as static group $\mathcal{J}^s$
  if mean_length($\mathcal{J}^m$) $> \alpha \cdot \Delta s$ **then**
    return (true, $\mathcal{A}$)
  else
    $\mathcal{A}' \leftarrow$ Update $\mathcal{A}$ using Equation(2)
    return (false, $\mathcal{A}'$)
  end if
end function

Algorithm 3 Articulation State Relocalization

Require: Current observation $\mathcal{F}_c$, Articulation model, Keyframes $\mathcal{K} = \{\mathcal{F}_{t_i}, s_{t_i}, \mathcal{P}^m_{t_i}\}_{i=0}^{K-1}$
Ensure: Relocalized Joint State $\hat{s}_c$ for $\mathcal{F}_c$
Segment object mask $\mathcal{M}_c$ from $\mathcal{F}_c$ and unproject to 3D as point cloud $\mathcal{P}_c$
Uniformly sample candidate states s_c from valid joint range $[0, s_{\max}]$
Compute loss for each candidate s_c as Equation(7)
Select candidate with minimal computed loss L_r as s_{init}
Optimize L_r w.r.t. s_{init} to find the relocalized state $\hat{s}_c$

exceeds a predefined threshold.

B. Unsupervised Articulation Modeling

The articulation modeling stage constructs an articulation model of the object using the RGB-D frames from a successful exploration. This is necessary because raw point trajectories from visual trackers are often too sparse or noisy for direct use in precise control. We therefore fit a parameterized model using a coarse-to-fine strategy.

Coarse Estimation from Sparse Trajectories. We first estimate the articulation model from the sparse 3D trajectories $\mathcal{J}^m$ of the mobile part. The points on a rigid mobile part must obey a kinematic constraint: for any two timesteps i and j, the point set $\mathcal{J}^m_i$ transforms to $\mathcal{J}^m_j$ via a rigid transformation $(\mathbf{R}^j_i, \mathbf{t}^j_i)$ parameterized by the articulation model, as shown in Equation(4),

$$\mathcal{J}^m_j = \mathbf{R}^j_i \cdot \mathcal{J}^m_i + \mathbf{t}^j_i. \tag{4}$$

For a prismatic joint, $\mathbf{R}^j_i = \mathbf{I}$ and $\mathbf{t}^j_i = (s_j - s_i)\mathbf{a}^p$. For a revolute joint, $\mathbf{R}^j_i$ is a rotation around axis $\mathbf{a}^r$ by angle $(s_j - s_i)$, and $\mathbf{t}^j_i = (\mathbf{I} - \mathbf{R}^j_i) \cdot \mathbf{c}_p$. We formulate a loss function L_c that minimizes the error between the observed points and the points transformed from the initial frame:

$$L_c = \frac{1}{T}\sum_{i=0}^{T-1} \|\mathcal{J}^m_i - (\mathbf{R}^i_0 \cdot \mathcal{J}^m_0 + \mathbf{t}^i_0)\|^2. \tag{5}$$

In this formulation, we minimize L_c with respect to the joint parameters (e.g., $\mathbf{a}^p$ or $\mathbf{a}^r, \mathbf{c}_p$) and the sequence of joint states $\{s_i\}$. We optimize L_c for both prismatic and revolute assumptions and choose the one with the lower final loss.

To facilitate convergence, we initialize joint parameters using geometric heuristics. For a prismatic joint, the axis $\mathbf{a}^p$ is initialized as the normalized vector aligned with the line connecting the centroids of the initial and final frames of $\mathcal{J}^m$. For a revolute joint, the axis $\mathbf{a}^r$ is initialized as the normal to the plane fitted to the centroids of $\mathcal{J}^m$ over time. The pivot point $\mathbf{c}_p$ is initialized as the point on $\mathcal{J}^m$ with the shortest trajectory length over time, as it is likely to be near the axis of rotation. To further address local minima issues, particularly for revolute joints, we randomly sample a set of candidate parameters around the initial geometric estimate and conduct parallel optimizations, selecting the candidate that yields the lowest loss on L_c.

Fine Estimation from Dense Point Clouds. To refine the coarse joint parameter estimates, we select K keyframes $\mathcal{F}_{t_0}, ..., \mathcal{F}_{t_{k-1}}$ from $\mathcal{F}$ with the most varied joint configurations. For each keyframe, we segment the mobile part by first generating an object mask with SAM2 [12] and then classifying masked pixels as mobile or static based on their proximity to classified 2D tracks (as done in Algorithm(2)). The resulting mobile masks are projected to 3D using depth observation to obtain dense point clouds $\mathcal{P}^m_{t_i}$. We then refine the joint parameters by minimizing an ICP [33]-like registration loss L_f over these dense clouds:

$$L_f = \frac{1}{C^2_k}\sum_{0=t_i<t_j}^{K-1} \frac{1}{|\mathcal{P}^m_{t_i}|}\sum_{\mathbf{p}_s \in \mathcal{P}^m_{t_i}} \|\mathbf{p}_s - (\mathbf{R}^{t_i}_{t_j} \cdot \mathbf{p}_t + \mathbf{t}^{t_i}_{t_j})\|^2, \tag{6}$$

where $\mathbf{p}_t$ is the closest point in $\mathcal{P}^m_{t_j}$ to the transformed $\mathbf{p}_s$. We optimize L_f using the coarse joint parameters as initial values to obtain a more precise estimate.

C. Closed-Loop Manipulation with Iterative Refinement

The final stage utilizes the derived articulation model to execute closed-loop manipulation. It comprises two key components: articulation state relocalization and iterative refinement.

Articulation State Relocalization. To manipulate an articulated object from an arbitrary configuration, the robot must first estimate the current joint state, s_c. This is achieved by relocalizing the current observation $\mathcal{F}_c$ against the keyframes $\mathcal{K}$ selected during the modeling stage, as outlined in Algorithm(3). The process involves sampling candidate states s_c uniformly from the range $[0, s_{\max}]$ and evaluating each candidate using the loss function defined in Equation(7). This loss measures the registration error between the current point cloud $\mathcal{P}_c$ and the transformed

TABLE I

QUANTITATIVE COMPARISON WITH BASELINE METHODS. METRICS ESR, ASR, AND MSR ARE DEFINED IN SECTION(V-A). CLASSIF.: JOINT TYPE CLASSIFICATION ACCURACY, AT: ADAPTIVE TERMINATION, FS: FAILURE SUPPRESSION, CA: CONTACT ANALOGY, CL: CLOSED-LOOP.

Method	Exploration			Articulation Modelling				Manipulation		
	$\text{ESR}_5^*\uparrow$	$\text{ESR}_5\uparrow$	#Interact↓	Classif.↑	$\text{ASR}_1\uparrow$	$\text{ASR}_{10}\uparrow$	$\text{ASR}_{20}\uparrow$	$\text{MSR}_{10}\uparrow$	$\text{MSR}_{20}\uparrow$	$\text{MSR}_{50}\uparrow$
GeneralFlow [3]	92.2±0.9	89.1±0.9	5.4±0.2	-	0.0±0.0	1.4±1.8	15.2±1.8	20.8±1.5	34.2±1.1	54.8±0.8
GAPartNet [4]	92.2±0.9	89.1±0.9	5.4±0.2	-	2.9±1.4	32.4±4.7	45.5±3.4	27.8±1.8	34.6±1.8	46.5±1.9
Ours w/o AT	28.4±1.8	28.3±1.7	-	27.4±1.7	18.4±1.3	27.1±1.6	27.2±1.6	21.3±1.6	24.0±1.5	25.4±1.5
Ours w/o FS	80.3±1.9	74.5±0.9	8.7±0.4	73.3±1.2	49.1±0.5	72.9±0.9	72.9±0.9	58.5±2.3	63.2±2.0	66.7±1.6
Ours w/o CA	90.9±0.7	84.7±1.7	8.4±0.2	84.7±1.7	54.7±1.5	83.8±2.6	84.1±2.2	64.8±2.5	70.7±2.2	74.7±2.0
Ours zero-shot	92.1±1.5	86.9±2.4	6.7±0.3	86.0±2.3	55.7±3.3	85.9±2.5	85.9±2.5	67.0±0.6	72.7±0.8	76.4±0.9
Ours w/o CL	92.2±0.9	89.1±0.9	5.4±0.2	88.6±0.6	59.3±2.1	88.6±0.6	88.6±0.6	61.2±1.1	67.8±1.1	73.3±1.1
Ours (Full)	**92.2±0.9**	**89.1±0.9**	**5.4±0.2**	**88.6±0.6**	**59.3±2.1**	**88.6±0.6**	**88.6±0.6**	**71.1±1.0**	**77.2±0.8**	**81.4±1.3**

keyframe mobile point clouds to the candidate state s_c:

$$L_r = \frac{1}{K}\sum_{i=0}^{K-1}\frac{1}{|\mathcal{P}_{t_i}^m|}\sum_{\mathbf{p}_s\in\mathcal{P}_{t_i}^m}\|\mathbf{p}_s - (\mathbf{R}_c^{t_i}\cdot\mathbf{p}_t + \mathbf{t}_c^{t_i})\|^2, \quad (7)$$

where $(\mathbf{R}_c^{t_i}, \mathbf{t}_c^{t_i})$ represents the transformation derived from the articulation model for a state transition from s_c to s_{t_i}, and $\mathbf{p}_t$ is the point in $\mathcal{P}_c$ closest to the transformed $\mathbf{p}_s$. The candidate state with the minimal loss L_r is selected as the initial guess, s_{init}. Subsequently, gradient-based optimization refines s_{init} to yield the final state estimate $\hat{s}_c$. This two-stage approach of sampling followed by optimization ensures efficient and accurate state estimation.

Iterative Refinement. Given the relocalized current joint state s_c and a target state s_t, the robot generates a motion plan. First, the successful grasp pose g_f from the exploration phase at state s_f is transformed to the current state s_c as g_c. The robot then plans an end-effector trajectory by uniformly sampling intermediate joint states between s_c and s_t and computing corresponding grasp poses, constrained by the estimated articulation model to maintain a constant relative pose between the gripper and the grasped mobile part. A motion planner [34] generates joint coordinates to execute this trajectory. After execution, the robot captures a new frame, relocalizes the resulting state s_e, computes the residual error $|s_t - s_e|$, and plans subsequent motions. This process iterates, minimizing the error until the target state is achieved within a specified tolerance or a maximum number of attempts is reached.

V. EXPERIMENTS

A. Task Settings

We employ the SAPIEN [24] environment for evaluation in simulation. Articulated object assets are sourced from the dataset used in RGBManip [18], comprising 116 objects with single movable joints (60 prismatic and 56 revolute). During the exploration stage, each object is initialized in a fully closed state ($s = 0$). For manipulation testing, the joint ranges are standardized: prismatic joints span 0 to 40 cm, and revolute joints span 0 to 40°. Each range is divided into four equal intervals, denoted $R_0, \ldots, R_3$. Manipulation tasks involve transitioning from a random initial state $s_{start} \in R_i$ to a target state $s_{target} \in R_j$, where $i, j \in \{0, 1, 2, 3\}$ and $i \neq j$. The target state is calculated as $s_{target} = s_{start} + (j - i)\cdot\delta$, where δ is the interval size (10 cm or 10°). This setup creates a diverse set of tasks with varying initial states and target displacements. Each task is repeated five times, and the mean and standard deviation of the results are computed for reporting.

Evaluation Metrics. To evaluate the exploration stage, we use ESR_5^* and ESR_5 to measure the success rate of achieving joint motion exceeding a 5 cm or 5° threshold for articulated objects triggered by the robot. ESR_5^* is based on algorithm predictions, while ESR_5 uses ground-truth data. We also report the number of interactions required to complete this stage. For articulation modeling, we assess the accuracy of joint type classification and the Articulation Success Rate (ASR_x), which measures the success rate when the estimated model's angle and pivot errors are within x degrees and x cm, respectively. For the manipulation stage, we report the Manipulation Success Rate (MSR_y), which indicates the success rate when the error from the target state is within $y\%$ of the total manipulation distance.

B. Comparison with Baseline Methods

We compare our method against two generalizable, open-source baselines, GAPartNet [4] and GeneralFlow [3], using our proposed exploration strategy for all methods to ensure fairness, as the baselines lack their exploration approach. For GeneralFlow, we input its predicted 3D flow into our coarse estimation module. For GAPartNet, we predict object parts in the first and last frames of the interaction sequence, match them, and estimate transformations between part bounding boxes to infer the articulation model. As neither baseline includes a relocalization module, they rely on ground-truth states and operate in an open-loop manner.

Quantitative and qualitative results are shown in Table(I) and Figure(3). The shared exploration strategy achieved 89.1% success rate with an average of 5.4 interactions. In articulation modeling, our method integrates unsupervised joint type prediction and coarse-to-fine estimation, achieving 88.6% joint type prediction accuracy across 89.1% of valid explorations. We evaluate joint axis and pivot point estimation using error thresholds of (1°, 1 cm), (10°, 10 cm), and (20°, 20 cm), requiring both angle and position errors to be below these thresholds, consistent with Ditto [2]. Our approach outperforms baselines, achieving 59.3% success at (1°, 1 cm) compared to near-zero for GeneralFlow and GAPartNet, and 88.6% at (20°, 20 cm). This success stems from optimizing joint parameters for kinematic consistency,

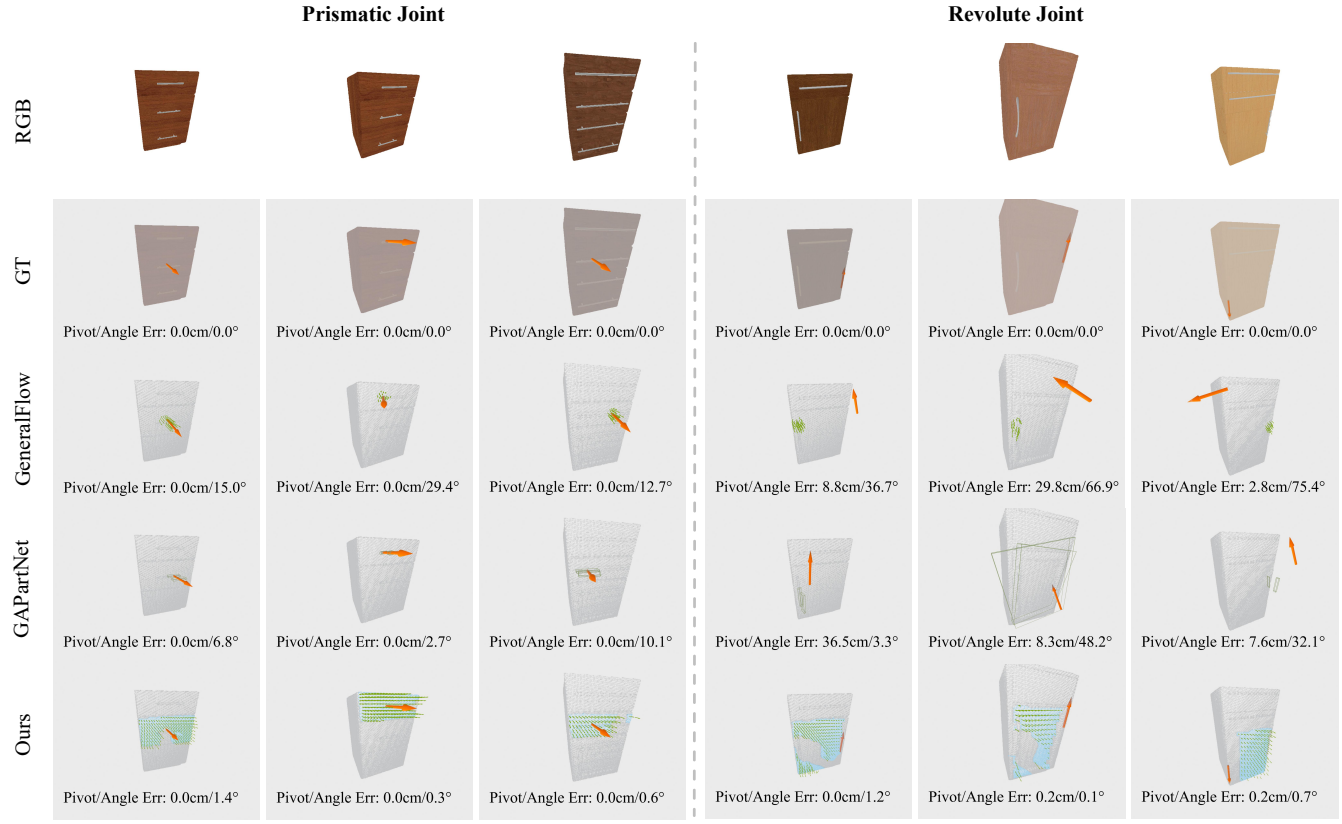

Fig. 3. **Qualitative Comparison of our method and baselines.** GeneralFlow leverages predicted 3D flows around contact points to estimate articulation models. GAPartNet relies on matched part bounding boxes for estimation. In contrast, our approach combines tracked sparse trajectories for coarse estimation with dense depth registration for fine refinement, achieving superior robustness and accuracy compared to other methods.

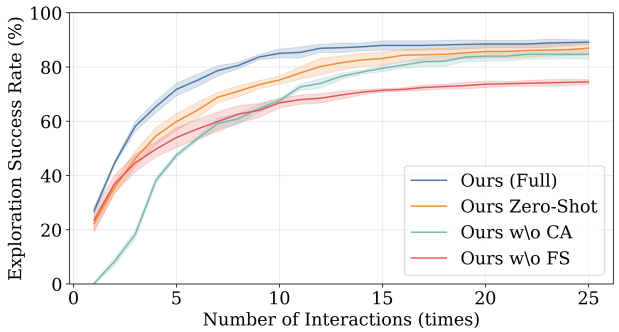

Fig. 4. **Impact of Exploration Components.** Contact analogy provides a strong initial guess, improving early success rates. The failure-suppression mechanism prevents repeated failures, ensuring a high final success rate.

unlike GeneralFlow's inconsistent per-point predictions or GAPartNet's coarse approximations. In the manipulation stage, our closed-loop approach, leveraging state relocalization and replanning, achieves a 71.1% success rate at a 10% relative error threshold, compared to 20.8% and 27.8% for GeneralFlow and GAPartNet. This demonstrates the precision of our method, enabled by accurate articulation modeling and closed-loop control, compared to the baselines' open-loop execution.

C. Ablation Studies

Ablation of Exploration Stage Components. To evaluate the effectiveness of our hypothesis-driven exploration with adaptive termination, we ablate its three key components—Contact Analogy (CA), Failure Suppression (FS), and Adaptive Termination (AT)—in Table(I). Removing AT prevents the algorithm from judging interaction validity, causing it to quit after one attempt; under high exploration uncertainty, the success rate drops from 89.1% to 28.3%. As Figure(4) shows, CA alone supplies a strong prior that boosts early success even when the analogy stems from different object classes (Ours Zero-Shot). Yet without FS, the rate plateaus later because repeated errors are never suppressed; with FS, the curve keeps climbing, demonstrating its role in recovering from poor initializations. Combining CA, FS, and AT simultaneously improves both efficiency and final effectiveness of our method.

TABLE II

ABLATION OF COARSE-TO-FINE ESTIMATION.

Stage	ASR_1 ↑	ASR_5 ↑	ASR_{10} ↑	ASR_{20} ↑
Coarse	38.6±3.6	81.9±0.8	86.9±1.0	88.4±0.4
Fine (Full)	**59.3±2.1**	**85.9±0.7**	**88.6±0.6**	**88.6±0.6**

Ablation of Coarse-to-Fine Estimation. Table(II) compares articulation model performance for the coarse stage alone versus the full coarse-to-fine pipeline. Success rates (ASR) are evaluated at (1°, 1 cm), (5°, 5 cm), (10°, 10 cm), and (20°, 20 cm). The coarse stage achieves 38.6% ASR at (1°, 1 cm), while the full pipeline boosts this to 59.3% and improves ASR across all thresholds. Coarse estimation uses sparse 2D tracks, which are susceptible to

TABLE III

QUANTITATIVE RESULTS OF REAL-WORLD EXPERIMENTS.

Metric	Prismatic (cm)					Revolute (°)				
	+10cm	+15cm	-10cm	-15cm	Average	+15°	+20°	-15°	-20°	Average
Init Joint State	27.90	23.06	36.95	39.10	-	3.00	3.60	21.20	30.10	-
End Joint State	37.68	37.90	26.95	23.85	-	18.00	23.20	6.15	11.40	-
Absolute Error	0.23	0.16	0.00	0.25	0.16	0.00	0.40	0.05	1.30	0.44
Relative Error (%)	2.25	1.07	0.00	1.67	1.25	0.00	2.00	0.33	6.50	2.21

TABLE IV

ABLATION OF CLOSED-LOOP MANIPULATION.

Metric	1st	2nd	3rd	4th	5th
$\mathbf{MSR}_{10}$	61.2±1.1	68.1±1.0	69.8±1.0	70.7±1.0	**71.1±1.0**
$\mathbf{MSR}_{20}$	67.8±1.1	74.2±0.7	75.8±0.9	76.8±0.6	**77.2±0.8**
$\mathbf{MSR}_{50}$	73.3±1.1	78.8±0.8	80.4±1.0	81.0±1.1	**81.4±1.3**

local motion contamination from robot kinematics, biasing joint-axis estimates due to correlated feature shifts. The fine stage employs dense depth-to-depth registration, using all depth pixels to refine axis and pivot estimates with sub-centimeter precision, reducing tracker drift. Dense registration also enables robust articulation state relocalization in novel configurations without pre-computed tracks, relying on global point cloud alignment.

Ablation of Closed-Loop Manipulation. Manipulation errors may arise from sensor noise (e.g., inaccurate depth data), relocalization inaccuracies, gripper slippage due to friction, or robot motor imprecision. These errors compound in long-distance tasks, risking gripper detachment or task failure. Table(IV) evaluates our relocalization module and closed-loop manipulation through an ablation study, reporting success rates at 10%, 20%, and 50% relative error thresholds over five iterations. Each iteration refines end-effector trajectories using updated joint states. While a single iteration yields moderate success, the fifth iteration improves success rates by 9.13% per threshold, as the relocalization module corrects cumulative errors, ensuring precision in long-range tasks. Error stabilization occurs in later iterations.

Zero-Shot Generalizability. We test the zero-shot generalizability of our approach with RAM's affordance memory [14] containing no objects from test categories, ensuring no prior exposure. As shown in Table(I), exploration success drops slightly by 2.2% (89.1% to 86.9%) with 6.7 average interactions (up from 5.4). Figure(4) shows our zero-shot variant underperforms compared to our method using same-class analogies, but outperforms methods without Contact Analogy. Our Failure Suppression mechanism ensures robust exploration, demonstrating strong adaptability. Articulation modeling and manipulation success rates dip slightly but remain highly competitive, showcasing effectiveness for unseen object categories.

D. Real-World Transferability

We validated our method in a real-world environment using a Franka Emika Robot Arm [35] and a RealSense D455 depth camera [36]. Camera-to-hand calibration was performed using OpenCV [37]. We test our approach on a prismatic drawer and a revolute cabinet across various scales of operation, as shown in Figure(5). Due to significant noise

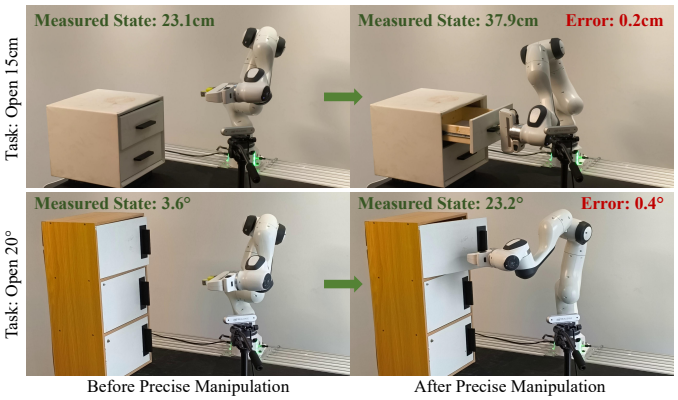

Fig. 5. **Real-World Deployment.** Our method seamlessly adapts to real-world environments with minimal modifications, achieving sub-centimeter and sub-degree manipulation accuracy.

in the RealSense depth raw data, we applied the built-in filter for processing [38]. Our method transfers to real-world environments with minimal adaptation effort, achieving results consistent with simulations, as demonstrated in Table(III).

E. Failure Case Analysis

We analyze the causes of unsuccessful cases reported in Table(I). In the exploration phase, failures primarily stem from erroneous hypothesis updates. For instance, correct grasp locations may fail due to unstable grasps, yielding invalid interaction data and suppressing valid hypotheses. Mitigation includes reducing hypothesis suppression strength or allowing multiple attempts per hypothesis. In articulation modeling, failures often result from tracker drift or misclassifying rotational joints as translational due to small motions. Solutions include using stronger priors. Manipulation failures mainly result from inaccurate articulation model estimates, leading to incorrect grasp poses and gripper slippage, or collisions between the object's rigid parts and the robot during large-distance manipulations. In real-world settings, depth sensor noise and grasp pose stability significantly impact the success rate.

VI. CONCLUSION

In this work, we presented a novel method for precisely manipulating unseen articulated objects. By enabling a robot to build the articulation model of the object through a three-stage process of exploration, modeling, and manipulation, our approach moves beyond brittle data-driven methods. Rigorous experiments validate the effectiveness of our method across simulated and real-world settings.

Despite its effectiveness, our method has limitations. First, computing grasp poses for new joint states relies on the

estimated articulation model, which may introduce error and lead to grasp failures. Second, our approach does not further leverage observations beyond the exploration stage to refine the articulation model, missing opportunities to enhance model accuracy during manipulation. Looking ahead, we aim to extend our method to handle multi-joint objects, moving beyond the single-joint objects tested in this study. Additionally, integrating tactile feedback could enhance the robustness and precision of manipulation by providing richer sensory data for real-time model refinement and control.

REFERENCES

[1] Z. Zhao, Y. Li, W. Li, Z. Qi, L. Ruan, Y. Zhu, and K. Althoefer, "Tac-man: Tactile-informed prior-free manipulation of articulated objects," IEEE Transactions on Robotics, 2024.

[2] Z. Jiang, C.-C. Hsu, and Y. Zhu, "Ditto: Building digital twins of articulated objects from interaction," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 5616–5626.

[3] C. Yuan, C. Wen, T. Zhang, and Y. Gao, "General flow as foundation affordance for scalable robot learning," arXiv preprint arXiv:2401.11439, 2024.

[4] H. Geng, H. Xu, C. Zhao, C. Xu, L. Yi, S. Huang, and H. 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, 2023, pp. 7081–7091.

[5] K. Mo, L. J. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani, "Where2act: From pixels to actions for articulated 3d objects," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 6813–6823.

[6] Q. Yu, J. Wang, W. Liu, C. Hao, L. Liu, L. Shao, W. Wang, and C. Lu, "Gamma: Generalizable articulation modeling and manipulation for articulated objects," in 2024 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2024, pp. 5419–5426.

[7] L. Ma, J. Meng, S. Liu, W. Chen, J. Xu, and R. Chen, "Sim2real 2: Actively building explicit physics model for precise articulated object manipulation," in 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 11 698–11 704.

[8] T. Jiang, L. Ma, Y. Guan, J. Meng, W. Chen, Z. Zeng, L. Li, D. Wu, J. Xu, and R. Chen, "Dexsim2real: Building explicit world model for precise articulated object dexterous manipulation," arXiv preprint arXiv:2409.08750, 2024.

[9] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht, "Cotracker: It is better to track together," in European Conference on Computer Vision. Springer, 2024, pp. 18–35.

[10] L. Tang, M. Jia, Q. Wang, C. P. Phoo, and B. Hariharan, "Emergent correspondence from image diffusion," Advances in Neural Information Processing Systems, vol. 36, pp. 1363–1389, 2023.

[11] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu, "Anygrasp: Robust and efficient grasp perception in spatial and temporal domains," IEEE Transactions on Robotics, vol. 39, no. 5, pp. 3929–3945, 2023.

[12] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson et al., "Sam 2: Segment anything in images and videos," arXiv preprint arXiv:2408.00714, 2024.

[13] R. Wu, Y. Zhao, K. Mo, Z. Guo, Y. Wang, T. Wu, Q. Fan, X. Chen, L. Guibas, and H. Dong, "Vat-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects," arXiv preprint arXiv:2106.14440, 2021.

[14] Y. Kuang, J. Ye, H. Geng, J. Mao, C. Deng, L. Guibas, H. Wang, and Y. Wang, "Ram: Retrieval-based affordance transfer for generalizable zero-shot robotic manipulation," arXiv preprint arXiv:2407.04689, 2024.

[15] K. Fang, T.-L. Wu, D. Yang, S. Savarese, and J. J. Lim, "Demo2vec: Reasoning object affordances from online videos," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2139–2147.

[16] B. Eisner, H. Zhang, and D. Held, "Flowbot3d: Learning 3d articulation flow to manipulate articulated objects," arXiv preprint arXiv:2205.04382, 2022.

[17] H. Zhang, B. Eisner, and D. Held, "Flowbot++: Learning generalized articulated objects manipulation via articulation projection," arXiv preprint arXiv:2306.12893, 2023.

[18] B. An, Y. Geng, K. Chen, X. Li, Q. Dou, and H. Dong, "Rgbmanip: Monocular image-based robotic manipulation through active object pose estimation," in 2024 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2024, pp. 7748–7755.

[19] C. Ning, R. Wu, H. Lu, K. Mo, and H. Dong, "Where2explore: Few-shot affordance learning for unseen novel categories of articulated objects," Advances in Neural Information Processing Systems, vol. 36, pp. 4585–4596, 2023.

[20] Y. Wang, R. Wu, K. Mo, J. Ke, Q. Fan, L. J. Guibas, and H. Dong, "Adaafford: Learning to adapt manipulation affordance for 3d articulated objects via few-shot interactions," in European conference on computer vision. Springer, 2022, pp. 90–107.

[21] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme, "Interactive perception: Leveraging action in perception and perception in action," IEEE Transactions on Robotics, vol. 33, no. 6, pp. 1273–1291, 2017.

[22] S. Y. Gadre, K. Ehsani, and S. Song, "Act the part: Learning interaction strategies for articulated object part discovery," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 15 752–15 761.

[23] S. Huang, H. Chang, Y. Liu, Y. Zhu, H. Dong, P. Gao, A. Boularias, and H. Li, "A3vlm: Actionable articulation-aware vision language model," arXiv preprint arXiv:2406.07549, 2024.

[24] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang et al., "Sapien: A simulated part-based interactive environment," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 11 097–11 107.

[25] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 909–918.

[26] J. Yan and M. Pollefeys, "Automatic kinematic chain building from feature trajectories of articulated objects," in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1. IEEE, 2006, pp. 712–719.

[27] D. A. Ross, D. Tarlow, and R. S. Zemel, "Unsupervised learning of skeletons from motion," in Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III 10. Springer, 2008, pp. 560–573.

[28] R. M. Martin and O. Brock, "Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2014, pp. 2494–2501.

[29] K. Hausman, S. Niekum, S. Osentoski, and G. S. Sukhatme, "Active articulation model estimation through interactive perception," in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 3305–3312.

[30] Z. Xu, Z. He, and S. Song, "Universal manipulation policy network for articulated objects," IEEE robotics and automation letters, vol. 7, no. 2, pp. 2447–2454, 2022.

[31] J. Lv, Q. Yu, L. Shao, W. Liu, W. Xu, and C. Lu, "Sagci-system: Towards sample-efficient, generalizable, compositional, and incremental robot learning," in 2022 International Conference on Robotics and Automation (ICRA). IEEE, 2022, pp. 98–105.

[32] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," Advances in neural information processing systems, vol. 14, 2001.

[33] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-9, no. 5, pp. 698–700, 1987.

[34] R. K. Guo, X. Lin, M. Liu, J. Gu, and H. Su, "MPlib: a Lightweight Motion Planning Library." [Online]. Available: https://github.com/haosulab/MPlib

[35] S. Haddadin, "The franka emika robot: A standard platform in robotics research," IEEE Robotics & Automation Magazine, 2024.

[36] V. Tadic, "Intel realsense d400 series product family datasheet," New Technologies Group, Intel Corporation, pp. 337 029–005, 2019.

[37] G. Bradski, "The opencv library." Dr. Dobb's Journal: Software Tools for the Professional Programmer, vol. 25, no. 11, pp. 120–123, 2000.

[38] G. Anders and D. Tong, "Depth post-processing for intel realsense d400 depth cameras," 2019.