Track2Art: Motion-Centric Articulated Object Model Recovery from 2D Point Trackers

Xiaotong Li, Yixiong Jing, Guangming Wang, and Brian Sheil

***Abstract*— Understanding articulated objects is fundamental for robots to interact with the physical world, which requires accurate segmentation of object parts and their kinematic relations. Motion provides a strong cue for understanding articulated structure: points on the same rigid part exhibit coherent motion, while relative trajectories between parts directly expose their kinematic constraints. However, many existing approaches recover articulation indirectly by first reconstructing object geometry across a small number of articulation states and subsequently inferring motion through cross-state alignment or per-instance optimization. Such reconstruction-first pipelines can underutilize rich temporal evidence and couple articulation estimation to errors in geometry reconstruction and correspondence.**

We present Track2Art, a motion-centric framework for recovering structured articulated object models from RGB-D interaction videos. Track2Art leverages a pretrained point tracker to obtain feature-rich 4D point trajectories. A motion-aware slot model groups trajectories into rigid object parts based on tracker appearance embeddings and explicit trajectory geometry. Given the recovered parts, a pairwise relation module predicts the directed kinematic graph and joint types, while a rotation-equivariant geometry module fuses motion-derived directional proposals and performs constrained axis-line refinement to recover joint geometry. Experimental results demonstrate the effectiveness of reasoning directly from persistent point motion for recovering structured articulated object models from dynamic visual observations.

I. INTRODUCTION

Understanding articulated objects is a fundamental capability for robots that interact with the physical world. Everyday objects such as drawers, doors, and appliances are composed of multiple rigid parts connected through constrained joints, and successful interaction requires reasoning not only about their geometry, but also about how their parts can move. Recovering such structured object models—including part decomposition, kinematic connectivity, joint types, and joint geometry—provides a compact representation that can support downstream tasks.

Early approaches to articulated-object understanding often adopt a part-pose-centric formulation. The observations are first decomposed into rigid components, whose time-varying poses are estimated from visual correspondences; articulation is then inferred by fitting kinematic models to the relative transformations between component poses [1], [2]. While intuitive, this decomposition introduces an intermediate part-pose bottleneck, where errors in motion segmentation, correspondence estimation, and rigid registration directly perturb the relative transformations used for kinematic inference. Sequential pose estimation may additionally accumulate drift, while partial visibility or weakly constrained observations can make the pose of an individual component difficult to estimate reliably. Consequently, useful point-level motion evidence can be corrupted or discarded before kinematic reasoning is performed.

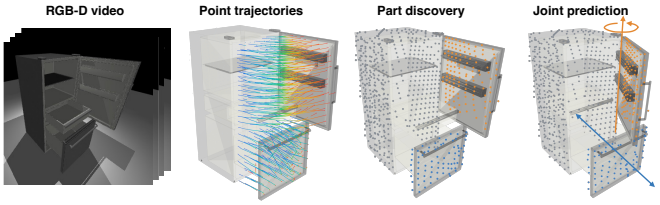

Fig. 1: Overview of Track2Art. Track2Art transforms an RGB-D interaction video into persistent 4D point trajectories, whose coherent motion reveals articulated part structure. The recovered parts and their relative motion are then used to infer a structured articulated object model with a directed kinematic graph, joint types, and joint geometry.

More recent approaches reduce the reliance on sequential part-pose estimation through joint optimization or learned feed-forward prediction. ReArt jointly recovers articulated parts, their motions, and kinematic relations from temporally varying 3D observations [3], while methods such as Ditto and LARM infer articulated representations from sparse articulation states without requiring sequential pose tracking [4], [5]. A substantial body of work, however, follows a reconstruction-centric formulation, in which object geometry is reconstructed from one or a small number of articulation states and articulation is recovered jointly with or subsequent to the underlying geometric representation [6], [7], [8], [9], [10]. Although effective in recovering high-quality geometric representations under favorable observations, such pipelines couple articulation estimation to reconstruction and correspondence quality and do not fully exploit the dense temporal motion evidence available throughout an interaction. These limitations become particularly pronounced for complex multi-part objects, where multiple moving components must be consistently associated and their relative motions jointly explained.

In contrast, we avoid introducing an explicit part-pose or reconstruction bottleneck before kinematic reasoning and instead take a motion-centric view of articulated object recovery. Our key observation is that articulation is directly revealed by persistent point motion: points belonging to the same rigid part exhibit coherent motion over time, while relative motion between parts exposes the constraints imposed by their connecting joints. Based on this observation, we present

Track2Art, a framework that recovers structured articulated object models directly from RGB-D interaction videos, as illustrated in Fig. 1. Track2Art establishes persistent point correspondences and lifts them with depth into feature-rich 4D trajectories, preserving point-level motion evidence throughout the interaction. A motion-aware slot model organizes these trajectories into rigid object parts, after which pairwise reasoning over the recovered parts predicts directed kinematic connectivity and joint types, while a rotation-equivariant geometry module recovers joint geometry from their observed motion. Experiments on PartNet-Mobility demonstrate stronger part decomposition than representative reconstruction- and optimization-based baselines under an object-aligned evaluation protocol. Additional evaluations examine robustness to camera, appearance, and occlusion shifts, as well as cross-dataset generalization. (TBD, waiting for robustness matrix result).

Our main contributions are:

- We introduce feature-rich persistent 4D point trajectories as the primary representation for articulated object recovery, leveraging pretrained tracking features and explicit metric motion to preserve strong correspondence and motion cues throughout an RGB-D interaction.
- We develop a motion-aware part discovery model that combines persistent point correspondences, pretrained per-track visual features, and explicit 3D motion to group trajectories into a variable number of rigid object parts.
- We develop a part-level kinematic reasoning module that combines learned relation prediction with a rotation-equivariant geometry design. Joint axes are recovered by sign-invariant fusion of explicit motion-derived directional proposals, while revolute axis lines are estimated using motion-based initialization and constrained orthogonal refinement.

II. Related Work

A. Articulated Object Modeling

Recovering articulated object models from visual observations has been studied through a range of geometry- and reconstruction-centric formulations. Early approaches such as Ditto [4] estimate part-level geometry and articulation from observations before and after interaction. PARIS [6] similarly considers two static articulation states and jointly optimizes part-level implicit geometry and motion parameters. These formulations provide compact articulated reconstructions, but represent motion primarily through differences between a small number of discrete states.

More recent methods incorporate stronger 3D representations and richer observations. ReArt [3] reasons over temporally varying 3D point clouds to jointly recover articulated parts and their kinematic relationships. With the emergence of 3D Gaussian Splatting, methods such as SplArt [7], ArtGS [8], GaussianArt [9], and ArtPro [10] formulate articulated modeling through differentiable reconstruction and motion optimization, progressively extending reconstruction to more complex articulated structures. DTA [11] similarly recovers articulated models from multi-state observations, but relies on a known part count. Despite strong reconstruction quality, these approaches generally couple articulation estimation with reconstruction, correspondence, or per-instance optimization of an underlying geometric representation.

TABLE I: Comparison with representative articulated object modeling methods. **2-State** denotes observations of sparse articulation states. $\mathscr{P}$ denotes automatic part discovery, and $\mathbf{1}^+$ denotes support for multi-joint articulation. **Geo.** and **GS** denote geometry- and Gaussian-based motion representations, respectively. **Track*** indicates point tracks used only for initialization. ✓ denotes explicit support, △ denotes partial support or additional required priors, and × denotes unsupported functionality.

Method	Obs.	Motion Cue	$\mathscr{P}$	$\mathbf{1}^+$	Inference
Ditto [4]	2-State	Geo.	✓	×	Feed-forward
PARIS [6]	2-State	Geo.	△	×	Optimization
SplArt [7]	2-State	GS	△	×	Optimization
ArtGS [8]	2-State	GS + Geo.	△	✓	Optimization
GaussianArt [9]	2-State	GS	△	✓	Optimization
ArtPro [10]	2-State	Geo.	△	✓	Optimization
DTA [11]	2-State	Geo.	△	✓	Optimization
ReArt [3]	Video	3D Motion	✓	✓	Optimization
iTACO [12]	Video	3D Motion	✓	✓	Hybrid
VideoArtGS [13]	Video	GS + Track*	△	✓	Optimization
AiM [14]	Video	GS Motion	✓	✓	Optimization
LARM [5]	2-State	Geo.	✓	✓	Feed-forward
Particulate [15]	Static 3D	Geo.	✓	✓	Feed-forward
Track2Art	**Video**	**Point Motion**	✓	✓	**Feed-forward**

Recent works increasingly exploit richer interaction sequences. VideoArtGS [13] reconstructs articulated digital twins from monocular videos and uses filtered 3D tracks to initialize articulation parameters before Gaussian optimization. AiM [14] models interaction videos with dynamic and static Gaussian components to discover moving parts and recover articulation without assuming a predefined number of parts, while iTACO [12] builds interactable digital twins from casually captured RGB-D videos. These approaches exploit richer temporal observations, but motion reasoning remains closely coupled to an optimized scene representation.

A complementary line of work avoids per-instance optimization through learned feed-forward articulation prediction. LARM [5] directly predicts articulated representations from sparse multi-state observations. More recently, Particulate [15] uses a transformer to infer parts, multi-joint kinematic structure, and motion constraints directly from a static 3D mesh in a single feed-forward pass. While efficient, such methods infer articulation primarily from learned geometric priors rather than observed object motion.

In contrast, Track2Art treats persistent point motion as the primary intermediate representation for articulation reasoning. Rather than first reconstructing geometry and subse-

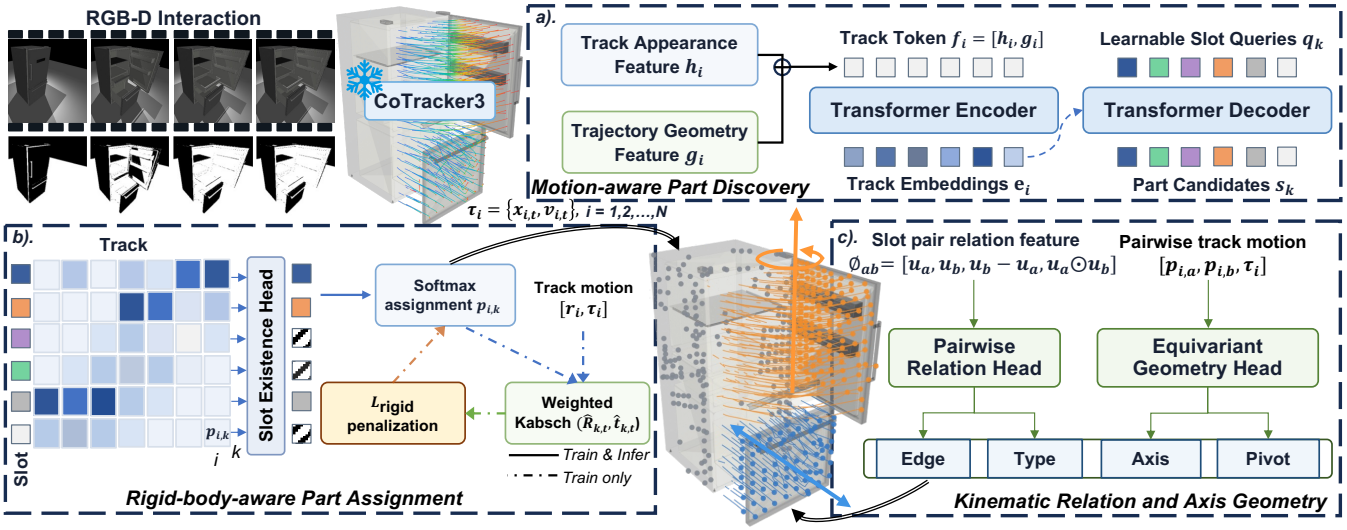

Fig. 2: Overview of Track2Art. Given an RGB-D interaction video and an object mask, Track2Art extracts persistent CoTracker trajectories and lifts them into 3D using calibrated depth. Each track token combines a frozen appearance descriptor with an explicit trajectory-geometry encoding. A set-prediction slot model groups the tracks into a variable number of rigid parts, supervised by a weighted-Kabsch motion-consistency objective. For every ordered pair of active part slots, a relation head predicts directed edges and joint types, while an equivariant geometry head combines analytic motion proposals to recover axes and pivot points, followed by a bounded correction.

quently extracting motion, we directly organize feature-rich 4D point trajectories into rigid parts and reason over their relative motion to recover kinematic structure. This motion-centric formulation preserves temporal evidence throughout an interaction and naturally supports structured multi-part reasoning.

B. Point Tracking and Motion Representations

Long-term point tracking provides explicit temporal correspondence across video and has recently benefited from large-scale learned representations. TAPIR [16] combines matching and temporal refinement for tracking arbitrary points over long video sequences, while CoTracker [17] jointly tracks multiple points to exploit dependencies between trajectories. CoTracker3 [18] further improves robustness through large-scale pseudo-labeled training. Recent 3D tracking methods such as TAPIP3D [19] extend persistent correspondence reasoning from image space to metric 3D dynamic scenes.

Beyond correspondence estimation, motion has long served as a cue for segmenting independently moving regions. Classical motion segmentation methods often build long-term point trajectories by integrating frame-to-frame optical flow and group them according to coherent or low-dimensional motion [20]. In contrast to local two-frame flow, a point trajectory preserves the identity and motion history of a point over an extended temporal window, providing a more direct representation of long-term motion coherence.

Modern learned point trackers make such trajectory representations substantially more robust to large displacement, appearance variation, and occlusion. Karazija et al. [21] exploit long-term tracks from CoTracker as supervision for motion-based video segmentation, demonstrating that correlated trajectories provide a strong grouping cue. VideoArtGS [13] also uses tracked points, but primarily to initialize articulation parameters before subsequent Gaussian optimization.

Track2Art builds on this trajectory-based view of motion, but targets a different level of structure. Rather than using trajectories only for object-level segmentation, correspondence, or optimization initialization, we lift persistent tracks into metric 3D and treat them as the primary entities for discovering rigid object parts and recovering their kinematic relationships.

III. Method

A. Problem Formulation and Overview

Given an RGB-D interaction video $\mathcal{V} = \{(I_t, D_t)\}_{t=1}^{T}$ of an articulated object, our goal is to recover its rigid part decomposition, kinematic graph, and joint parameters. As illustrated in Fig. 2, Track2Art reasons about articulation directly from persistent point motion. We first convert the interaction into feature-rich 4D point trajectories using pretrained point trackers. A motion-aware part module groups these trajectories into rigid components, after which a relation module reasons over the recovered parts to predict their parent–child relationships and joint properties. Together, these predictions form a structured articulated object model. We next describe the trajectory representation, part discovery, and kinematic reasoning modules in detail.

B. Feature-Rich 4D Point Trajectories

We aim to represent an RGB-D interaction as a set of persistent, feature-rich 4D point trajectories. Foreground queries are sampled from a category-agnostic object mask and tracked using the frozen CoTracker3. The resulting 2D

trajectories are lifted into the world frame using synchronized depth and calibrated camera poses, yielding world-space positions $\{\mathbf{x}_{i,t}\}_{t=1}^{T}$, $\mathbf{x}_{i,t} \in \mathbb{R}^3$, together with visibility indicators $\{v_{i,t}\}_{t=1}^{T}$. From each trajectory $\tau_i = \{\mathbf{x}_{i,t}, v_{i,t}\}$, we compute the 32-D explicit geometric descriptor $g_i(\tau_i)$, which contains canonical reference position, endpoint displacement, maximum motion magnitude, visibility ratio, and eight uniformly sampled displacement vectors. We compute the coordinate-wise median $\mathbf{c}$ of the observed track reference positions and the diagonal length s of their axis-aligned bounding box. The reference position is centered by $\mathbf{c}$, while all metric position and displacement terms are scaled by s. Both quantities are computed online solely from the observed tracks at training and inference time, without ground-truth geometry. We concatenate its frozen per-track embedding h_i with a 32-D explicit trajectory geometry descriptor $g_i(\tau_i)$:

$$\mathbf{f}_i = [\mathbf{h}_i; \mathbf{g}_i]. \tag{1}$$

C. Motion-Aware Part Discovery

Points on the same rigid component should be explainable by a shared time-varying rigid transformation. We use a DETR-style set predictor [22] to group tracks into a variable number of rigid part slots. As shown in Fig. 2(a)., a Transformer encoder [23] first models interactions among the track tokens, and $K_{\max}$ learnable part queries $\mathbf{q}_k$ are decoded through cross-attention:

$$\begin{aligned} \{\mathbf{e}_i\} &= \mathrm{Enc}(\mathrm{Proj}(\{\mathbf{f}_i\})), \\ \{\mathbf{s}_k\} &= \mathrm{Dec}(\{\mathbf{q}_k\}, \{\mathbf{e}_i\}), \\ p_{ik} &= \mathrm{softmax}_k\left(\frac{\mathbf{e}_i^\top \mathbf{s}_k}{\sqrt{d}}\right). \end{aligned} \tag{2}$$

Here, $\mathbf{s}_k$ is a candidate part representation, p_{ik} is the soft assignment of track i to slot k, and d denotes the feature dimension shared by the encoded track tokens and part slots. In Fig. 2(b), a separate existence head selects the active subset, so the number of parts is inferred rather than provided as input. Because slot identities are unordered, ground-truth parts are aligned to predicted slots through bipartite Hungarian matching, following set-prediction formulations [22].

To encourage each predicted part to exhibit coherent rigid motion, we regularize the soft track-to-slot assignments with a training-time rigidity loss. Let p_{ik} denote the probability of assigning track i to slot k, and let $v_{i,t} \in \{0,1\}$ indicate whether track i is visible at frame t. We take the query position in the reference frame as $\mathbf{r}_i = \mathbf{x}_{i,1}$, while $\mathbf{x}_{i,t}$ denotes its world-space position at frame t.

For each active slot k and frame t, we estimate a rigid transformation $(\hat{\mathbf{R}}_{k,t}, \hat{\mathbf{t}}_{k,t})$ from $\{\mathbf{r}_i\}$ to $\{\mathbf{x}_{i,t}\}$ using weighted Kabsch alignment [24], with weights $w_{i,k,t} = p_{ik} v_{i,t}$. We then penalize the normalized weighted re-projection error:

$$\mathscr{L}_{\mathrm{rigid}} = \frac{1}{|\Omega|} \sum_{(k,t)\in\Omega} \frac{\sum_i w_{i,k,t} \left\| \mathbf{x}_{i,t} - \left(\hat{\mathbf{R}}_{k,t}\mathbf{r}_i + \hat{\mathbf{t}}_{k,t}\right)\right\|_2^2}{s^2 \sum_i w_{i,k,t} + \varepsilon}, \tag{3}$$

where Ω is the set of valid slot–frame pairs, s is the object bounding-box scale used for scale normalization, and ε is a small constant for numerical stability. This loss encourages tracks assigned to the same slot to be explained by a single rigid motion.

D. Kinematic Relations and Rotation-Equivariant Geometry

Joint geometry should transform consistently with the coordinate frame in which the object is observed. Directly regressing an axis as an unconstrained Cartesian vector, however, may entangle the prediction with orientation biases present in the training data. We therefore introduce a rotation-equivariant axis fusion operator that combines explicit motion-derived directional cues using learned scalar weights. The key idea is to separate which geometric cue is reliable from how the predicted axis transforms under a change of coordinates.

Pairwise relation head. For every ordered non-self slot pair (a,b), we construct the pairwise representation

$$\phi_{ab} = [\mathbf{u}_a, \mathbf{u}_b, \mathbf{u}_b - \mathbf{u}_a, \mathbf{u}_a \odot \mathbf{u}_b], \tag{4}$$

where $\mathbf{u}_k$ combines the learned slot representation with an assignment-weighted trajectory summary. A shared relation module predicts directed edge existence and joint type from ϕ_{ab}, while joint geometry is estimated in parallel by the equivariant geometry branch described below.

Motion-derived directional proposals. Using the soft part assignments, we estimate the time-varying rigid motion of each slot by weighted Kabsch alignment. For an ordered parent–child pair, these motions define a sequence of relative rigid transformations. From the relative motion and reference geometry, we construct four directional proposals: (i) the dominant direction of the temporal child–parent centroid motion, (ii) the dominant rotation direction extracted from relative rotation logarithms, (iii) the reference inter-part centroid direction, and (iv) the cross product between the reference centroid direction and the translational-motion direction. These proposals provide complementary translational, rotational, and structural cues to the joint axis.

Rotation-equivariant axis fusion. Let $\{\bar{\mathbf{a}}_j\}_{j\in\mathscr{V}}$ denote the valid normalized directional proposals. A learned scalar branch conditions on the part-pair representation and geometric diagnostics to predict proposal logits ℓ_j, from which

$$\alpha_j = \mathrm{softmax}_{j\in\mathscr{V}}(\ell_j). \tag{5}$$

Rather than directly averaging the proposal vectors, which is ill-defined because several geometric directions are sign ambiguous, we represent each proposal by its second-order moment:

$$\mathbf{M} = \sum_{j\in\mathscr{V}} \alpha_j \bar{\mathbf{a}}_j \bar{\mathbf{a}}_j^\top, \qquad \hat{\mathbf{a}} = \mathrm{eigvec}_{\max}(\mathbf{M}). \tag{6}$$

This representation removes the sign ambiguity because $\mathbf{a}\mathbf{a}^\top = (-\mathbf{a})(-\mathbf{a})^\top$. Moreover, for proposals transformed by a global rotation $\mathbf{Q} \in \mathrm{SO}(3)$ and unchanged scalar weights,

$$\mathbf{M}' = \mathbf{Q}\mathbf{M}\mathbf{Q}^\top. \tag{7}$$

When the dominant eigenspace is one-dimensional, the recovered axis therefore satisfies

$$\hat{\mathbf{a}}' \equiv \mathbf{Q}\hat{\mathbf{a}}, \tag{8}$$

where $\equiv$ accounts for the sign ambiguity of an unoriented joint axis. This equivariance property applies to the geometric fusion operator. The full prediction pipeline additionally depends on learned part assignments and proposal weights, whose response to rotation is not analytically constrained; we evaluate the resulting end-to-end rotation robustness separately.

Axis-consistent line refinement. For revolute joints, the axis direction alone does not determine the axis line. A point $\mathbf{p}$ on the axis satisfies

$$(\mathbf{I} - \mathbf{R}_{ab,t})\mathbf{p} = \mathbf{t}_{ab,t}. \tag{9}$$

We obtain a motion-based initialization $\mathbf{p}_{\text{ana}}$ by solving this constraint over valid frames. Since points along the axis represent the same physical line, we restrict the learned correction to the plane orthogonal to the predicted axis. With

$$\mathbf{P} = \mathbf{I} - \hat{\mathbf{a}}\hat{\mathbf{a}}^{\top}, \tag{10}$$

we construct an orthonormal local basis $\{\mathbf{u}_1, \mathbf{u}_2\}$ in this plane and predict only two bounded scalar coefficients:

$$\hat{\mathbf{p}} = \mathbf{p}_{\text{base}} + \gamma_{ab} \sum_{j=1}^{2} \frac{1}{2} \tanh(\beta_j)\mathbf{u}_j, \tag{11}$$

where γ_{ab} is a bounded pair-local geometric scale derived from the spatial extent of the parent–child configuration. This constrains the learned correction to remain local to the candidate joint while adapting its range to the geometry of each part pair. We set $\mathbf{p}_{\text{base}} = \mathbf{p}_{\text{ana}}$ when a valid rotational estimate is available, and otherwise use the midpoint of the reference part centroids.

Motion-consistency supervision. We further require the recovered joint geometry to explain the observed relative part motion. Given the predicted joint parameters, we fit the scalar joint coordinate q_t for each frame and replay the corresponding rigid motion. For a reference point $\mathbf{r}_i$, the predicted trajectory is

$$\hat{\mathbf{x}}_{i,t} = \begin{cases} \mathbf{r}_i + q_t\hat{\mathbf{a}}, & \text{prismatic}, \\ \mathbf{R}(\hat{\mathbf{a}}, q_t)(\mathbf{r}_i - \hat{\mathbf{p}}) + \hat{\mathbf{p}}, & \text{revolute}. \end{cases} \tag{12}$$

The normalized discrepancy between the replayed and observed trajectories defines $\mathscr{L}_{\text{joint}}$. This loss is used only as a training regularizer and introduces no iterative optimization at inference. The relation objective is

$$\mathscr{L}_{\text{rel}} = \mathscr{L}_{\text{edge}} + \lambda_t\mathscr{L}_{\text{type}} + \lambda_a\mathscr{L}_{\text{axis}} + \lambda_l\mathscr{L}_{\text{line}} + \lambda_j\mathscr{L}_{\text{joint}}. \tag{13}$$

E. Training and Inference *(waiting for robustness, cross/real data result)*

Training proceeds in three phases. We first train the part-discovery module using segmentation and rigidity supervision. We then train the relation and geometry modules on the recovered slot representations. Finally, the complete model is jointly fine-tuned, using a reduced learning rate for the previously trained slot backbone. Random geometric perturbations and track dropout are used during part training. Motion-derived geometric proposals and analytic line initializations are treated as stop-gradient quantities for numerical stability; geometry supervision therefore trains the proposal weighting and line-refinement modules without back-propagating through the underlying Kabsch, PCA, or analytic fitting operations.

At inference, slots whose existence scores exceed a threshold are retained, and each trajectory is assigned to its most likely active slot. All ordered active-slot pairs are then evaluated to recover directed edges, joint types, and joint geometry. The resulting part decomposition and directed kinematic graph form a URDF-compatible articulated object model. After pretrained trajectory extraction, Track2Art requires no test-time clustering, graph search, or per-object joint optimization.

IV. Experiments

A. Experimental Setup

a) Dataset and protocol: We evaluate Track2Art on articulated objects from PartNet-Mobility [25]. Each interaction contains 120 synchronized timesteps recorded over eight seconds at 15 FPS, with three calibrated RGB-D views at every timestep. At inference, Track2Art receives only the RGB-D observations, camera intrinsics and extrinsics, and a category-agnostic object mask. Semantic part labels and joint annotations are used only for training supervision and evaluation.

b) Evaluation metrics: For part decomposition, all methods are evaluated on the same method-independent source-frame reference points after collapsing fixed-connected-body chains into kinematic parts. Predicted and ground-truth part IDs are aligned by one-to-one Hungarian matching. We report object-averaged Point IoU, Adjusted Rand Index (ARI), and ordinary Rand Index (RI). In Table II, Seg. *N* is the number of objects with valid segmentation output, and GT J. is the number of ground-truth joints on those objects under the collapsed observable ontology. Match counts ground-truth joints whose parent and child parts are associated with a recovered part pair and whose predicted edge passes the method's edge gate. Type accuracy is evaluated on these matched joints and is shown together with its type-correct/matched count. Axis *N* counts matched, type-correct joints with a valid predicted axis. Axis error is the undirected angle $\arccos(|\hat{\mathbf{a}}^{\top}\mathbf{a}|)$ and is reported as mean/median over Axis *N*. Line is the distance between predicted and ground-truth revolute axis lines, normalized by the object bounding-box diagonal, and is evaluated only when a valid type-correct revolute prediction is available. J@20 is the fraction of GT J. recovered with the correct type and axis error at most 20°; therefore unmatched edges and type errors count as failures. Because external methods retain their native observation protocols and oracle inputs, the comparison is object- and metric-aligned but not input-

TABLE II: External comparison on the object-aligned 20-object PartNet-Mobility suite under the collapsed observable ontology. Axis and line errors are conditional on valid matched, type-correct joints, whereas J@20 uses all eligible GT joints. Method-specific priors are listed under Qualification.

Method	Seg. N	IoU↑	ARI↑	RI↑	GT J.	Match↑	Type Acc.↑	Axis N	Axis mean/med.↓	Line↓	J@20↑	Qualification
Ours	20	**0.581**	**0.635**	**0.861**	61	**56**	0.839 (47/56)	47	40.02/40.60	**0.020**	0.279	No GT oracle
AiM	20	0.259	0.082	0.551	61	15	0.467 (7/15)	7	43.57/47.85	–	0.033	Native motion decomposition
ReArt	20	0.411	0.332	0.717	61	16	0.688 (11/16)	11	50.76/53.65	0.283	0.033	Converted native SE(3) trajectories
DTA + replay	17	0.476	0.425	0.784	41	24	0.458 (11/24)	11	48.79/60.09	0.978	0.073	GT part count; no-GT replay type selector
ArtGS	20	0.500	0.495	0.793	61	38	0.711 (27/38)	27	46.91/46.07	0.586	0.180	GT part count
VideoArtGS	14	0.352	0.214	0.596	41	21	0.952 (20/21)	20	**25.56/11.02**	0.210	**0.341**	Joint inventory and parent topology
PARIS	4	0.532	0.199	0.673	4	4	**1.000** (4/4)	4	42.82/34.40	0.156	0.250	Native two-part scope

equivalent; these differences are stated in the Qualification column.

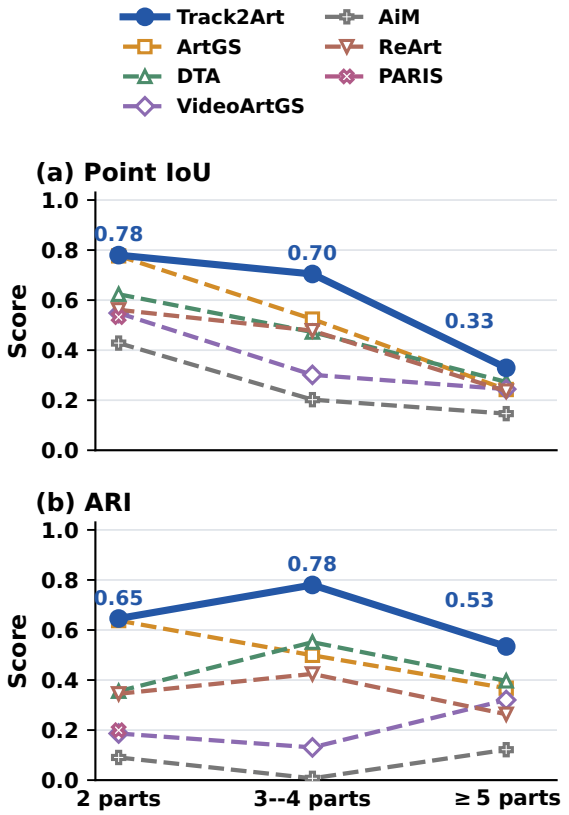

Fig. 3: Part decomposition across increasing articulation complexity on the object-aligned external suite. Track2Art uses the validation-selected CoTracker-only representation. All methods retain their native observation protocols and oracle assumptions; the comparison is metric-aligned but not input-equivalent. PARIS is shown only for two-part objects, its native scope.

B. Main Performance and Comparison with Existing Methods

a) Overall comparison: We compare Track2Art with representative reconstruction- and optimization-based approaches on the object-aligned 20-object suite. As shown in Table II, Track2Art produces valid output for all 20 objects and obtains the strongest common point-domain decomposition, with 0.581 Point IoU, 0.635 ARI, and 0.861 RI. It recovers 56 of the 61 observable GT joints after edge gating, of which 47 have the correct type (0.839 matched-joint type accuracy). Its conditional axis mean/median are 40.02°/40.60°, while its normalized revolute line error is 0.020 and its end-to-end J@20 is 0.279. These kinematic columns must be interpreted with their denominators: VideoArtGS has lower conditional axis error and higher J@20 on the 14 relatively easy objects for which it succeeds. However, its 14-object success subset is substantially easier in the common point domain than the six missing objects (mean best-achieved Point IoU for all methods: 0.639 vs. 0.360). Its axis result should therefore be interpreted as conditional success-subset performance, additionally aided by the provided joint inventory and parent topology, rather than as a full-suite ranking. DTA and ArtGS use the ground-truth part count, PARIS is restricted to four native two-part outputs, and AiM and ReArt retain their native motion representations. Unsupported outputs are left unreported rather than inferred or assigned zero scores.

b) Generalization with articulation complexity: Figure 3 groups the aligned external objects by their number of kinematic parts. Track2Art obtains Point IoU values of 0.78, 0.70, and 0.33 on objects with 2, 3–4, and at least 5 parts, respectively. The corresponding ARI values are 0.65, 0.78, and 0.53. Track2Art remains the strongest common point-domain method in every bucket, showing that persistent motion cues remain useful as articulation becomes more complex. However, its under-segmentation rate rises to 0.833 in the at-least-five-part bucket, revealing that small or weakly excited components remain difficult to recover.

c) Qualitative comparison: Figure 4 visualizes representative objects with simple, medium, and complex articulation. On the two-part example, most methods recover a coarse moving/static decomposition, although their joint geometry varies in stability. Differences become clearer on the three-part object, where several baselines merge moving components or predict inconsistent joint geometry. On the complex example, Track2Art preserves more of the major motion-consistent components, while reconstruction-centric baselines commonly merge multiple parts or produce fragmented predictions. The same example also exposes Track2Art's main failure mode: small or weakly excited parts may still be absorbed by larger motion groups.

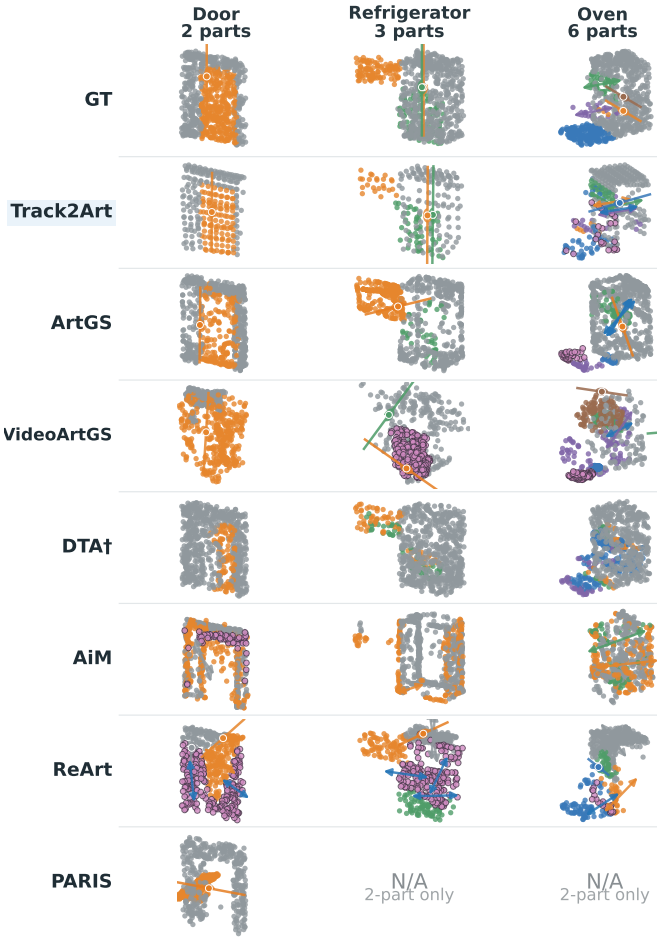

Fig. 4: Qualitative comparison on object-aligned articulated objects of increasing complexity. Part colors are matched to the ground truth only for visualization and do not modify the underlying predictions. Joint geometry is overlaid only when natively supported; DTA† shows segmentation only because it does not provide compatible joint-geometry outputs under our common visualization contract. Track2Art preserves clearer motion-consistent part decomposition across all three examples, especially on medium and complex objects, although small or weakly excited parts remain challenging.

C. *Robustness Matrix*

Camera jitter,Appearance/texture randomization,Moving occlusion

D. *Cross-Dataset and Real Observations*

Out-of-training-distribution dataset like lightwheel

V. Conclusion and Limitations

Conclusion. We presented Track2Art, a motion-centric framework for recovering structured articulated object models from RGB-D interaction videos. Rather than treating articulation as a by-product of geometry reconstruction and cross-state alignment, Track2Art uses persistent, feature-rich 4D point trajectories to infer the articulation model from RGB-D observations. Experiments on an object-aligned PartNet-Mobility benchmark demonstrate accurate part decomposition and end-to-end kinematic recovery. Track2Art also achieves the strongest common point-domain decomposition in the object-aligned external comparison and maintains a clear advantage as articulation complexity increases. These results support persistent point trajectories as an effective bridge between dynamic visual observations and structured, physically meaningful articulated object models.

Limitations and future work. Track2Art remains dependent on trajectory quality and motion observability: small, occluded, or weakly excited parts may be absorbed into larger motion groups. Although object-centric normalization and geometric augmentation reduce sensitivity to scale and viewpoint, the learned representation remains affected by appearance, camera-viewpoint, and track-distribution shifts. Consequently, models trained primarily on simulation exhibit degraded performance on real RGB-D sequences, where depth noise, calibration errors, imperfect foreground masks, occlusion, and tracking failures jointly corrupt the lifted 4D trajectories. The independently decoded relations may also produce globally invalid kinematic graphs, while a small tail of catastrophic joint-axis errors remains. Future work will investigate stronger sim-to-real appearance and sensor randomization, globally constrained graph decoding, confidence-aware learned–analytic estimation, active interaction for improved part observability, and adaptation to noisy real-world RGB-D observations.

References

[1] J. ürgen Sturm, V. Pradeep, C. Stachniss, C. Plagemann, K. Konolige, and W. Burgard, "Learning kinematic models for articulated objects," in *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[2] D. Katz, M. Kazemi, J. A. Bagnell, and A. Stentz, "Interactive segmentation, tracking, and kinematic modeling of unknown 3d articulated objects," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5003–5010.

[3] S. Liu, S. Gupta, and S. Wang, "Building rearticulable models for arbitrary 3d objects from 4d point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 138–21 147.

[4] Z. Jiang, C.-C. Hsu, and Y. Zhu, "Ditto: Building digital twins of articulated objects from interaction," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2022, pp. 5606–5616.

[5] S. Yuan, R. Shi, X. Wei, X. Zhang, H. Su, and M. Liu, "Larm: A large articulated object reconstruction model," in *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, 2025, pp. 1–12.

[6] J. Liu, A. Mahdavi-Amiri, and M. Savva, "Paris: Part-level reconstruction and motion analysis for articulated objects," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 352–363.

[7] S. Lin, J. Fang, M. Z. Irshad, V. C. Guizilini, R. A. Ambrus, G. Shakhnarovich, and M. R. Walter, "Splart: Articulation estimation and part-level reconstruction with 3d gaussian splatting," in *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2025, pp. 8841–8851.

[8] Q. Yu, X. Yuan, Y. Jiang, J. Chen, D. Zheng, C. Hao, Y. You, Y. Chen, Y. Mu, L. Liu, *et al.*, "Artgs: 3d gaussian splatting for interactive visual-physical modeling and manipulation of articulated objects," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 13 170–13 177.

[9] L. Shen, S. Zhang, H. Li, P. Yang, Z. Huang, Z. Zhang, and H. Zhao, "Gaussianart: Unified modeling of geometry and motion for articulated objects," in *2026 International Conference on 3D Vision (3DV)*. IEEE, 2026, pp. 384–395.

[10] X. Li, Z. Wang, X. Wang, L. Wu, M. Li, and C. Tu, "Artpro: Self-supervised articulated object reconstruction with adaptive integration of mobility proposals," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026, pp. 13 897–13 907.

[11] Y. Weng, B. Wen, J. Tremblay, V. Blukis, D. Fox, L. Guibas, and S. Birchfield, "Neural implicit representation for building digital twins of unknown articulated objects," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024, pp. 3141–3150.

[12] W. Peng, J. Lv, C. Lu, and M. Savva, "itaco: Interactable digital twins of articulated objects from casually captured rgbd videos," in *2026 International Conference on 3D Vision (3DV)*. IEEE, 2026, pp. 520–531.

[13] Y. Liu, B. Jia, R. Lu, C. Gan, H. Chen, J. Ni, S.-C. Zhu, and S. Huang, "Videoartgs: Building digital twins of articulated objects from monocular video," *arXiv preprint arXiv:2509.17647*, 2025.

[14] H. Ai, W. Chang, J. Jiao, A. Leonardis, and E. Ofek, "Articulation in motion: Prior-free part mobility analysis for articulated objects by dynamic-static disentanglement," in *International Conference on Learning Representations (ICLR)*, vol. 2026, 2026, pp. 87 749–87 774.

[15] R. Li, Y. Yao, C. Zheng, C. Rupprecht, J. Lasenby, S. Wu, and A. Vedaldi, "Particulate: Feed-forward 3d object articulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026, pp. 27 708–27 718.

[16] C. Doersch, Y. Yang, M. Vecerik, D. Gokay, A. Gupta, Y. Aytar, J. Carreira, and A. Zisserman, "Tapir: Tracking any point with per-frame initialization and temporal refinement," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2023, pp. 10 027–10 038.

[17] 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.

[18] N. Karaev, Y. Makarov, J. Wang, N. Neverova, A. Vedaldi, and C. Rupprecht, "Cotracker3: Simpler and better point tracking by pseudo-labeling real videos," in *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2025, pp. 1–10.

[19] B. Zhang, L. Ke, A. Harley, and K. Fragkiadaki, "Tapip3d: Tracking any point in persistent 3d geometry," *Advances in Neural Information Processing Systems*, vol. 38, pp. 135 284–135 303, 2026.

[20] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *European conference on computer vision*. Springer, 2010, pp. 282–295.

[21] L. Karazija, I. Laina, C. Rupprecht, and A. Vedaldi, "Learning segmentation from point trajectories," *Advances in neural information processing systems*, vol. 37, pp. 112 573–112 597, 2024.

[22] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[24] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Foundations of Crystallography*, vol. 32, no. 5, pp. 922–923, 1976.

[25] 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 *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, pp. 11 094–11 104.