2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

PokeNet: Learning Kinematic Models of Articulated Objects from Human Observations

Anmol Gupta[1], Weiwei Gu[1], Omkar Patil[1], Jun Ki Lee[2], Nakul Gopalan[1]

***Abstract*— Articulation modeling enables robots to learn joint parameters of articulated objects for effective manipulation which can then be used downstream for skill learning or planning. Existing approaches often rely on prior knowledge about the objects, such as the number or type of joints. Some of these approaches also fail to recover occluded joints that are only revealed during interaction. Others require large numbers of multi-view images for every object, which is impractical in real-world settings. Furthermore, prior works neglect the order of manipulations, which is essential for many multi-DoF objects where one joint must be operated before another, such as a dishwasher. We introduce PokeNet, an end-to-end framework that estimates articulation models from a single human demonstration without prior object knowledge. Given a sequence of point cloud observations of a human manipulating an unknown object, PokeNet predicts joint parameters, infers manipulation order, and tracks joint states over time. PokeNet outperforms existing state-of-the-art methods, improving joint axis and state estimation accuracy by an average of over 27% across diverse objects, including novel and unseen categories. We demonstrate these gains in both simulation and real-world environments. Code and dataset are available on our webpage[+].**

I. INTRODUCTION

Articulation modeling is the task of learning structured object models that capture joints, degrees of freedom (DoF), and motion constraints to provide a feasible direction for the manipulation of an articulated joint. Such models can be used by planners or policies across robot embodiments to generate feasible manipulation strategies even on unseen objects.

Many prior works on articulation modeling rely on a single RGB-D image [1][2][3][4][5]. However, static views are often insufficient: joints may be occluded or revealed only through interaction, for example, a drawer hidden inside a refrigerator. Moreover, a single image provides no information about the range or limits of joint motion, which are critical for safe manipulation, for example, opening a laptop hinge without exceeding its allowable range. Other approaches, such as [1][6][7] achieve promising results but rely on strong assumptions, such as known number of joints or type of joints. Recent works, inspired by 3D reconstruction methods like NeRF [8], NeuS[9] and Gaussian Splatting[10], use reconstruction to predict the articulation parameters of objects [11][12][13][14]. While effective, these techniques require a large number of multi-view images and often use multistage pipelines. Importantly, none of these methods accounts for the sequence of manipulations in multi-DoF objects, which is critical when one joint must be operated before another, such as opening a dishwasher door before pulling out its rack.

To address these limitations, we propose PokeNet, an end-to-end method for estimating articulation parameters from a single human demonstration without requiring prior knowledge of the object. By observing a person manipulate an object through a sequence of point cloud frames, PokeNet captures not only the articulation structure, including joint types and parameters of occluded joints, but also the intermediate state of each joint at every time step. In addition, it infers the order of joint operations from demonstration, which is essential for many multi-DoF objects. A key challenge is that every novel object has an unknown number and hierarchy or ordering of joints. Traditional models assume a fixed number of outputs, making them unsuitable for this task. To tackle this problem, we adopt a set prediction formulation: the network directly outputs a set of joint hypotheses and uses a permutation invariant loss to align them with ground truth. This design avoids assumptions about joint count, while enabling end-to-end learning of articulation parameters. A qualitative comparison of PokeNet with prior works is given in Table I. In summary, our contributions are as follows:

- In this work, we propose a novel framework for learning articulated objects with multiple degrees of freedom (DoFs) from a single human demonstration captured from one viewpoint. Our method jointly estimates the articulation structure (joint types and parameters), the state of each joint at every time step, and the correct sequence of joint manipulations required to reach a target configuration. **To our knowledge, this is the first method to simultaneously model articulation structure and manipulation order from human input for multi-DoF objects.**
- We release a comprehensive real-world dataset of articulated objects, spanning seven objects from four categories and comprising $5{,}500$ human–object interaction data points. **To our knowledge, this is the largest annotated real-world articulated object corpus.**
- For evaluation, we conduct extensive experiments on simulated and real-world multi-DoF objects. Our model generalizes to four unseen categories in simulation, remains robust to variations in object sizes, and extends to unseen real-world instances. **PokeNet achieves state-of-the-art performance, improving joint axis and**

[1]School of Computation and AI, ASU, Tempe {anmolgupta, weiweigu, opatil3, ng}@asu.edu
[2]AI Institute, Seoul National University, Seoul, South Korea junkilee@snu.ac.kr
[+]Webpage: https://sequential-joints.github.io/

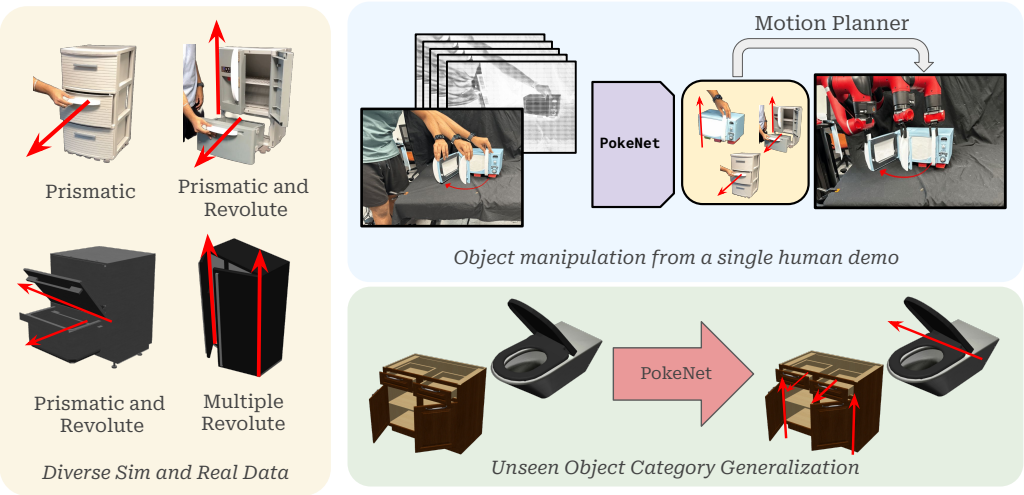


Fig. 1: We propose a novel framework that learns the joint parameters and manipulation order of articulated objects directly from human demonstrations. Leveraging human interactions allows our method to reason about occluded joints, while requiring no prior object knowledge. It generalizes to unseen object categories and achieves state-of-the-art performance in both simulation and real-world settings.

TABLE I: Capability comparison of articulation modeling methods. PokeNet is the only method that jointly handles multi-joint objects, recovers occluded or shut joints, and predicts both joint states and manipulation order.

Method	Sng.	Multi	Occ.	State	Order
ScrewNet	✓	✗	✗	✓	✗
GAPartNet	✓	✓	✗	✗	✗
PokeNet	✓	✓	✓	✓	✓

state estimation by up to 25% **on** 8,000 **simulated data points and by** 30% **on** 1,600 **real-world data points.** Finally, we demonstrate successful manipulation on both seen and unseen object classes using a motion planner.

II. Related Work

Estimation from Visual Data. Several approaches estimate the articulation properties of objects directly from visual input [1], [6], [7]. However, these methods often assume prior object category knowledge or are limited to single-DoF objects. Other approaches [15], [16] rely on predefined models or DoF assumptions, restricting generalization. More recent methods [2], [3], [5] take a single partial point cloud as input, but such approaches cannot reliably recover occluded or fully closed joints and provide no information on joint motion limits, posing safety risks when deployed on real robots. In contrast, our method makes no assumptions about object categories or joint types, requiring only that the links be rigid and that a maximum number of joints be specified. Using full motion sequences from a single human demonstration, PokeNet is able to recover occluded joints, capture motion limits, and enable safer deployment in real-world settings.

Interactive Methods. Interactive perception approaches [17], [18], [19], [20] involve robot-driven exploration to estimate articulation. However, these methods require object textures and rely on primitive actions like pushes or pokes, limiting their effectiveness in complex settings. We use human demonstrations to overcome these limitations, capturing more realistic interaction dynamics without requiring object texture or category priors. Our method generalizes across diverse object instances and handles occluded configurations more robustly.

Reconstruction-Based Methods. Several approaches recover articulation models through reconstruction frameworks such as NeRF and Gaussian Splatting [21], [22], [13]. These methods often rely on strong priors about object categories or structures, limiting their generalization. More recent work such as ScrewSplat [11] removes some of these assumptions but still requires large numbers of multi-view images to estimate articulation parameters. In contrast, our method requires no prior object knowledge and predicts articulation parameters from human demonstration. PokeNet is designed to infer articulation parameters with sample efficiency, without reconstructing the full visual representation of the object.

III. Problem Formulation

Articulated objects have components that are connected at joints. These components are called “links” that can be rigid. The “joints” afford relative motion between these links. The joints of these objects can be classified into two broad classes – revolute and prismatic joints. These joints can be defined using a direction and an anchor point in space. For prismatic joints, the link moves along the direction of the axis, while for revolute joints, the link moves in a direction perpendicular to the axis. The axis and position of the joint together are called articulation parameters.

In our problem formulation, we assume the links are rigid and the maximum number of joints are K. The input is a sequence of 3D point clouds of the object while it is being manipulated. Formally, each data point is a sequence $\mathcal{P} = \{P_1, P_2, \ldots, P_T\}, \quad P_t \in \mathbb{R}^{N\times 3}$, where T is the number of observed frames and N is the number of 3D points per frame.

The objective is to recover the underlying articulation model directly from $\mathcal{P}$. We define a prediction function parameterized by θ:

$$f_\theta : \mathcal{P} \mapsto \mathcal{S} = \{s_1, s_2, \ldots, s_K\},$$

where $\mathcal{S}$ is a set of K slots. Each slot corresponds to a hypothesized joint and is defined as

$$s_k = \{c_k, \tau_k, \mathbf{d}_k, \mathbf{p}_k, o_k\}.$$

Here, $c_k \in [0, 1]$ is a confidence score that determines if slot s_k corresponds to a valid joint; $\tau_k \in \{0, 1\}$ denotes the joint type (0 revolute, 1 prismatic); $\mathbf{d}_k \in \mathbb{R}^3$ is a unit axis direction; $\mathbf{p}_k \in \mathbb{R}^3$ is an anchor point on this axis; and $o_k \in [0, 1]$ encodes the relative manipulation order.

In addition to the articulation parameters, we predict per-timestep joint states aligned with the same slots. Let $\mathcal{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_T\}, \quad \mathbf{y}_t \in \mathbb{R}^{K\times 3}$, denote the auxiliary output, where the k-th component $y_{t,k}$ represents the displacement of slot s_k at time t. Its physical interpretation depends on the type: for $\tau_k = 0$ (revolute), $y_{t,k} = (\sin\theta_{t,k}, \cos\theta_{t,k}, 0)$ encodes angular displacement in a wraparound-free form, while for $\tau_k = 1$ (prismatic), $y_{t,k} = (0, 0, \rho_{t,k})$ encodes linear displacement.

The final output is obtained by keeping only slots with high confidence,

$$\hat{\mathcal{S}} = \{s_k \in \mathcal{S} \mid c_k > \mu\},$$

and taking their associated per-timestep states $\hat{\mathcal{Y}} = \{y_{t,k} \mid s_k \in \hat{\mathcal{S}},\ t = 1 \ldots T\}$. The retained set is sorted by o_k to provide an ordered articulation model together with its time-varying joint states.

IV. Dataset

For the proposed model, we needed an extensive range of objects to learn from. For this, we used two distinct datasets: a simulated dataset and a real-world dataset.

A. Simulated Dataset

The simulated dataset was created using the Partnet-Mobility dataset [23], [24], [25], which we chose for its diversity and ease of use, allowing us to test the robustness and generalizability of our approach under controlled conditions. Each object was rendered in simulation and its articulated links were moved to simulate realistic joint motions. We captured this motion as sequences of point clouds in the camera coordinate frame, which were then used to predict articulation parameters.

In total, we collected 110,000 sequences spanning eleven categories: microwave, laptop, washing machine, fridge, drawer, bucket, door, fan, scissors, window, and trashcan. These objects were chosen for their variability in size, geometry, and degrees of freedom, as well as their practical relevance in real-world settings. We additionally reserved four categories—furniture, box, toilet, and pliers—exclusively for testing. This set of objects is comparable to those used in prior baselines.

B. Real World Object Dataset with Ground Truth

Due to the absence of large-scale real-world 3D video datasets for articulated objects, we collected our own. We selected four commonly encountered household objects—microwave, dishwasher, refrigerator, and drawer—chosen for their availability and diversity in joint structure. These categories collectively cover a wide range of articulation types, including both revolute and prismatic joints. For example, microwaves exhibit standard revolute joints, while dishwashers and refrigerators often feature combinations of revolute and prismatic links.

Each object was physically manipulated by three of the authors by opening and closing its movable parts, and the resulting joint displacements and angles were recorded. To obtain accurate ground-truth articulation data, we attached ArUco markers [26] to object parts and tracked their motion with a dual calibrated camera setup, ensuring each marker was visible from at least one camera throughout the sequence. Note that this two-camera setup was used only for annotation; the demonstration videos used for training were recorded from a single camera view.

In total, we collected 5,500 annotated point cloud sequences across four object categories. Of these, 3,900 samples were used for training and 1,600 for testing, making this, to the best of our knowledge, the largest real-world dataset of 3D videos of articulated objects annotated with ground-truth joint parameters. To assess generalization, we also recorded 10 physical demonstrations each for two additional unseen objects: a slider knife and a stapler.

V. Methods

In this work, we propose a novel end-to-end framework for estimating the articulation parameters and manipulation order of articulated objects with multiple degrees of freedom (DoFs) using sequential point cloud observations from a *single camera view*. For a novel object, the robot cannot know in advance either the number of joints or their manipulation order. This requires predictions to be made over *sets* of joints with no predefined order. To address this, we adopt a DETR-style decoder [27] with Hungarian matching, which aligns predicted slots with ground-truth joints in a permutation-invariant manner.

Our method begins by encoding the input sequence of point clouds with a PointNet++ encoder [28] and a lightweight transformer encoder [29], which produce spatial embeddings for each frame. The sequence of embeddings is then processed by another transformer encoder to capture temporal dependencies across frames. The resulting temporal features are passed into a transformer decoder that maintains a fixed set of learnable queries, each of which attends to the

sequence and specializes into a "slot" representing a potential joint.

Each slot outputs: a *confidence score* indicating whether it corresponds to a valid joint, a *joint type* (revolute or prismatic), a normalized 3D *axis direction*, a 3D *anchor point* defining the axis position in space, and an *order score* representing the relative sequence in which joints are manipulated. During inference, slots with high confidence scores are kept and sorted by their predicted order scores, producing an ordered set of joints that parametrizes the articulation model of the object.

This design provides several advantages: (i) it removes the need for prior knowledge of object categories or exact joint counts, requiring only a specified maximum number of joints, (ii) it uses temporal reasoning by attending to embeddings across frames, and (iii) it uses a structured slot-based representation that is generalizable across objects with varying articulations.

A. Architecture

In this section, we describe the architecture of **PokeNet**, which consists of three main components: (i) spatial encodings, (ii) temporal encodings, and (iii) decoders for joint parameter prediction.

Spatial Encodings. Each frame P_t is a 3D point cloud. We encode it with PointNet++ [28] to obtain Q point-level embeddings $X_t = \{x_{t,1}, \ldots, x_{t,Q}\} \in \mathbb{R}^{Q \times D}$. We append a learnable [CLS] token c_t and pass $\{c_t\} \cup X_t$ through a lightweight transformer encoder [29]. The updated [CLS] output $\hat{c}_t \in \mathbb{R}^D$ serves as the *frame-level representation*, summarizing the frame's information via attention over the local point tokens.

Temporal Encodings. The sequence $\{\hat{c_0}, \hat{c_1}, \ldots, \hat{c_{T-1}}\}$ is then processed by a transformer encoder [29] to capture temporal dependencies. These temporal features form the memory that is accessed by the decoders in the next stage.

Joint Set Decoder. The first decoder is responsible for predicting the articulation parameters. We adopt a DETR-style formulation [27], where a fixed set of K learnable queries attends to the temporal memory. Each query is updated through cross attention and specializes in a "slot" corresponding to a potential joint. For every slot, we attach prediction heads that output: a confidence score, a joint type (revolute or prismatic), a normalized axis direction $\mathbf{d} \in \mathbb{R}^3$, an anchor point $\mathbf{p} \in \mathbb{R}^3$, and an order score $o \in [0, 1]$ indicating the relative sequence in which the joint is manipulated. The slot-based formulation allows the model to predict a variable number of joints under K.

Auxiliary Decoder. In addition to the joint set decoder, PokeNet includes an auxiliary decoder that tracks the per-frame state of each joint across the observed sequence. For every slot s_k and time step t, the decoder outputs a state vector $\mathbf{z}_{t,k} \in \mathbb{R}^3$. The three components are interpreted according to the slot's joint type: for revolute joints, $\mathbf{z}_{t,k} = (\sin\theta_{t,k}, \cos\theta_{t,k}, 0)$ encodes angular displacement in a wraparound-free form, while for prismatic joints, $\mathbf{z}_{t,k} = (0, 0, \rho_{t,k})$ provides linear translation. This unified representation avoids discontinuities for angles while maintaining a consistent $K \times T \times 3$ output structure across all slots. In the final output, only states associated with slots that exceed the confidence threshold $c_k > \mu$ are retained, ensuring that per-frame states are defined solely for valid joints.

B. Loss Function

We train PokeNet with a composite objective that supervises all aspects of the predicted joint slots. Since our formulation treats joint estimation as a set prediction task, we first align predicted slots with ground-truth joints using the Hungarian algorithm to find the assignment that minimizes a cost composed of axis direction, anchor consistency, and joint type classification. The matched pairs are then used to compute the loss.

For each predicted slot, we supervise five components. First, an *confidence loss* is applied using binary cross-entropy, with matched slots labeled as positive and unmatched slots as negatives. Second, joint *type classification* (revolute vs. prismatic) is trained with cross-entropy loss. Third, the predicted axis direction is supervised using an angular loss $L_{\text{axis}} = \mathbb{E}[1 - |\langle \hat{\mathbf{d}}, \mathbf{d} \rangle|]$, which encourages alignment between normalized predicted and ground-truth directions while being invariant to axis sign. Fourth, anchor point prediction is trained with mean squared error, minimizing the point-to-line distance between predicted anchors and ground-truth joint axes. Fifth, we supervise the *order score* predicted by each slot. Since ground-truth joint order is given as a discrete index $r \in \{0, 1, \ldots, M-1\}$ for M joints, we map it to a continuous target in $[0, 1]$ to enable gradient-based regression. Specifically, we define the normalized order as

$$\tilde{r} = \frac{r}{\max(1,\, M-1)}. \tag{1}$$

This ensures that the first joint always maps to 0, the last joint to 1, and intermediate joints are evenly spaced between them, regardless of M. The predicted order score $\hat{o}_i \in [0, 1]$ for joint i is then trained with two complementary losses. First, an L_1 regression loss anchors the prediction to its normalized target:

$$L_{\text{L1}} = \frac{1}{M} \sum_{i=1}^{M} \left| \hat{o}_i - \tilde{r}_i \right|. \tag{2}$$

Second, a pairwise ranking hinge loss enforces relative ordering: for any two ground-truth joints $i \prec j$ we require $\hat{o}_i + m \leq \hat{o}_j$ with margin m:

$$L_{\text{rank}} = \frac{1}{|\mathcal{X}|} \sum_{(i,j) \in \mathcal{X}} \max\big(0,\, m - (\hat{o}_j - \hat{o}_i)\big), \tag{3}$$

where $\mathcal{X}$ denotes the set of all ordered joint pairs.

Finally, we introduce a state loss L_{state} that applies mean squared error over the predicted per-timestep joint displacements. L_{state} is computed on matched slots only, with supervision applied according to the ground-truth joint type. For revolute joints, the auxiliary decoder outputs $(\hat{u}, \hat{v})$, normalized to unit length and compared against the ground-truth

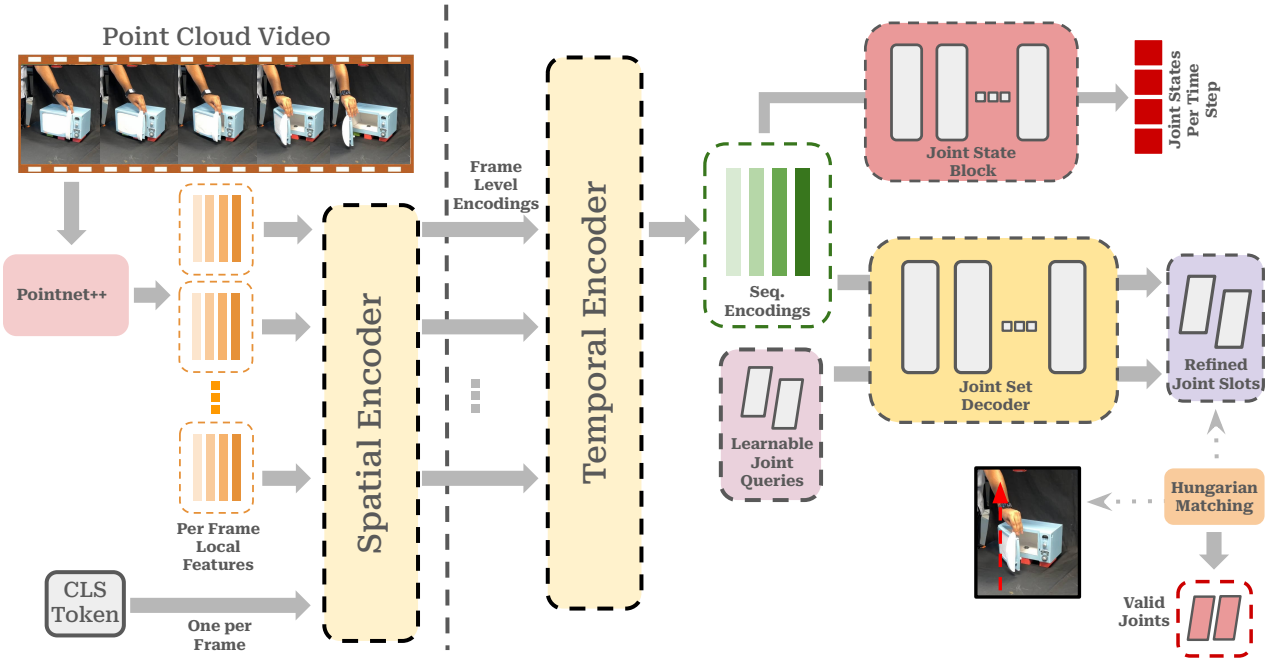


Fig. 2: Overview of **PokeNet**. Our model takes a sequence of point clouds as input. Each frame is encoded with PointNet++ to extract spatial features, and [CLS] tokens are passed through a transformer encoder to capture temporal dependencies. A DETR style joint decoder with learnable queries attends to these temporal features to predict an ordered kinematic slot model, including joint type, axis direction, anchor point, confidence, and manipulation order. An auxiliary decoder additionally predicts per-frame joint states.

$(\sin\theta, \cos\theta)$. For prismatic joints, the third channel encodes the scalar displacement ρ, which is directly regressed. The unused channels are masked in each case.

The total loss is a weighted combination:

$$\mathcal{L} = \lambda_{\text{conf}} L_{\text{conf}} + \lambda_{\text{type}} L_{\text{type}} + \lambda_{\text{axis}} L_{\text{axis}} + \lambda_{\text{point}} L_{\text{point}} + \lambda_{\text{order}} L_{\text{L1}} + \lambda_{\text{rank}} L_{\text{rank}} + \lambda_{\text{state}} L_{\text{state}}. \quad (4)$$

VI. Experiments

The following sections present a comprehensive evaluation of PokeNet on simulated and real-world datasets. We assess performance across four key aspects: (1) comparison with existing SOTA baselines of GAPartNet [5] and an extended multi-joint version of ScrewNet [6] for joint parameters; (2) generalization to *unseen object instances*; (3) generalization to *unseen object categories* in both simulation and the real world; and (4) robustness to variations in *object scales*, including test time scaling not seen during training.

A. Simulated Dataset Results

From the simulated data, $88,000$ sequences were used for training, and $2,000$ per category were set aside for testing. To evaluate category level generalization, we additionally held out four categories from training: furniture, box, toilet, and plier, with 2,000 sequences per category for testing. To evaluate robustness under distribution shift, we varied object scale during training and testing. During training, the scales of the objects were randomly perturbed by $\pm 5\%$. At test time, we used unseen scales of up to $\pm 7\%$, allowing us to assess robustness to scale variations beyond the training range.

Across $30,000$ test samples, PokeNet significantly outperforms baseline methods in both axis alignment and anchor point accuracy. As shown in Table II, it generalizes well to unseen object instances, categories, and appearance variations. On simulated data, PokeNet surpasses GAPartNet by 25% in joint axis prediction accuracy.

B. Real-World Dataset Results

Across $1,600$ test samples, PokeNet consistently outperforms both baselines across all the object categories. PokeNet improves the accuracy of the prediction of the joint axis, exceeding GAPartNet by 30%. We further evaluate PokeNet on two unseen objects, a slider knife and a stapler, using 10 test demonstrations, and calculate the mean error in predicted axes. In these objects, PokeNet achieves an improvement of 40% compared to GAPartNet. The results for real world data are shown in Table III

To further validate the applicability of our method, we integrated PokeNet's output into a planning pipeline. The robot uses the predicted articulation parameters and provided contact points to generate manipulation trajectories. For prismatic joints, the planner traces linear paths in 1 cm increments, while for revolute joints, it traces arcs in 1° increments. Fig. 3 shows an example of the robot successfully opening a refrigerator using parameters inferred by PokeNet.

Object	Method	Axis Orientation Error (°)	Axis Displacement Error (cm)	Joint State Error (deg/cm)
Microwave	Screwnet	21.13 ± 1.07	14.28 ± 1.03	18.47 ± 0.96
	GAPartNet	9.71 ± 1.12	6.37 ± 1.21	-
	Ours	**7.26 ± 1.08**	5.26 ± 0.97	**9.31 ± 1.09**
Washing Machine	Screwnet	21.78 ± 0.88	11.64 ± 1.22	21.36 ± 1.27
	GAPartNet	10.57 ± 1.33	7.92 ± 1.08	-
	Ours	**7.48 ± 1.01**	**6.23 ± 1.15**	**7.11 ± 1.34**
Laptop	Screwnet	22.69 ± 1.14	12.37 ± 0.91	17.42 ± 1.38
	GAPartNet	7.28 ± 0.95	5.72 ± 1.44	-
	Ours	6.14 ± 1.23	4.68 ± 1.07	**8.21 ± 1.07**
Fridge	Screwnet	26.87 ± 1.25	13.58 ± 1.09	24.21 ± 1.11
	GAPartNet	11.74 ± 1.18	8.83 ± 0.86	-
	Ours	**8.92 ± 0.97**	**6.77 ± 1.31**	**9.48 ± 0.91**
Drawer	Screwnet	23.58 ± 0.93	10.96 ± 1.36	20.66 ± 0.84
	GAPartNet	10.07 ± 1.09	6.77 ± 1.12	-
	Ours	**7.36 ± 1.14**	**4.59 ± 0.88**	**6.91 ± 1.12**
Trashcan	Screwnet	24.72 ± 1.31	14.61 ± 0.99	19.18 ± 1.43
	GAPartNet	9.34 ± 0.86	7.37 ± 1.27	-
	Ours	8.51 ± 1.08	**4.92 ± 1.13**	**10.03 ± 1.36**
Window	Screwnet	20.93 ± 1.17	12.83 ± 1.02	21.57 ± 0.98
	GAPartNet	9.39 ± 1.24	6.58 ± 1.14	-
	Ours	**7.16 ± 0.89**	5.73 ± 1.41	**6.88 ± 1.05**
Door	Screwnet	24.05 ± 1.08	13.47 ± 1.20	23.62 ± 1.26
	GAPartNet	9.88 ± 1.43	8.11 ± 0.92	-
	Ours	9.03 ± 1.12	**5.06 ± 1.33**	**7.95 ± 1.14**
Fan	Screwnet	22.41 ± 0.99	11.72 ± 1.28	20.84 ± 1.21
	GAPartNet	9.93 ± 1.02	7.68 ± 1.34	-
	Ours	**7.84 ± 0.95**	**4.51 ± 1.07**	**6.59 ± 1.18**
Scissors	Screwnet	21.62 ± 1.34	10.73 ± 0.90	24.39 ± 1.32
	GAPartNet	8.85 ± 1.20	8.21 ± 1.25	-
	Ours	**6.12 ± 1.06**	**5.87 ± 0.82**	**8.36 ± 1.06**
Bucket	Screwnet	20.74 ± 0.92	12.42 ± 1.35	22.53 ± 1.09
	GAPartNet	10.66 ± 1.26	6.97 ± 1.10	-
	Ours	**8.21 ± 0.96**	**4.38 ± 1.22**	**7.44 ± 1.03**
Plier*	Screwnet	27.84 ± 1.19	13.66 ± 0.88	19.47 ± 1.31
	GAPartNet	10.63 ± 1.08	6.94 ± 1.12	-
	Ours	**8.14 ± 1.23**	6.58 ± 1.29	**11.37 ± 0.99**
Toilet*	Screwnet	29.31 ± 1.15	14.12 ± 1.33	21.66 ± 0.97
	GAPartNet	12.08 ± 0.96	7.76 ± 1.18	-
	Ours	**8.52 ± 1.01**	**5.23 ± 0.93**	**9.97 ± 1.09**
Furniture*	Screwnet	26.47 ± 1.28	11.95 ± 1.07	25.88 ± 1.44
	GAPartNet	17.24 ± 1.11	8.62 ± 0.91	-
	Ours	**13.15 ± 0.90**	**5.87 ± 1.36**	**10.94 ± 1.27**
Box*	Screwnet	25.72 ± 0.91	12.31 ± 1.24	26.18 ± 0.84
	GAPartNet	19.37 ± 1.36	7.23 ± 1.06	-
	Ours	**11.42 ± 1.07**	**5.11 ± 0.95**	**13.82 ± 1.38**

TABLE II: Results for simulated dataset. We report the mean error (with 95% confidence intervals) for joint axis directions (in degrees), positions (in centimeters), and joint state estimation (in deg/cm), averaged over the available axes per object. All the objects were in partially opened state for GAPartNet. * represents objects unseen during training.

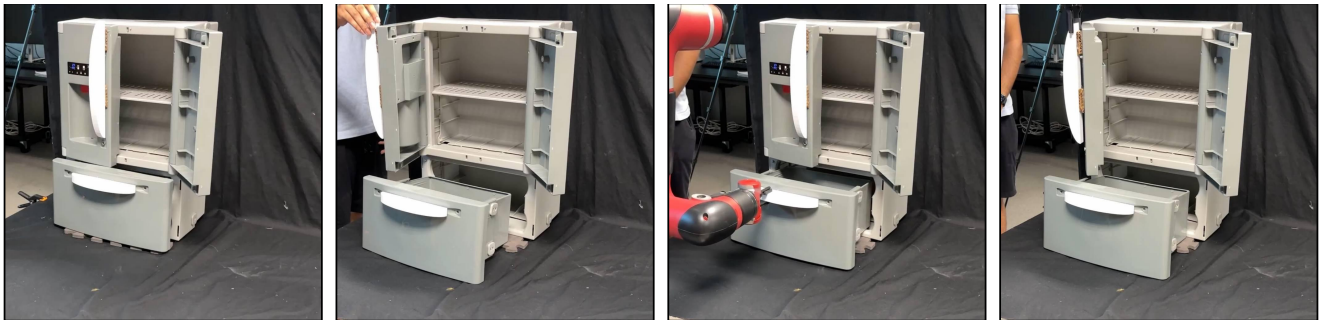
The accompanying video and website further demonstrate the robot manipulating common kitchen appliances in real world using our model's predictions.

C. Discussion

Fig. 4 presents a radar chart comparison between GAPartNet and PokeNet across object conditions. As shown in

Object	Method	Axis Orientation Error (°)	Axis Displacement Error (cm)	Joint State Error (deg/cm)
Microwave	Screwnet	28.42 ± 1.36	17.58 ± 1.22	29.11 ± 1.09
	GAPartNet	18.62 ± 1.12	12.77 ± 1.31	-
	Ours	**13.88 ± 0.96**	11.42 ± 1.24	**13.15 ± 1.12**
Fridge	Screwnet	30.97 ± 1.07	31.66 ± 1.38	27.54 ± 1.41
	GAPartNet	21.41 ± 1.46	20.82 ± 0.98	-
	Ours	**16.97 ± 1.05**	**16.34 ± 1.27**	**12.59 ± 1.16**
Drawer	Screwnet	35.33 ± 1.28	33.85 ± 1.11	25.62 ± 0.94
	GAPartNet	22.88 ± 0.91	18.41 ± 1.32	-
	Ours	**17.62 ± 1.39**	**13.04 ± 1.18**	**12.71 ± 1.08**
Dishwasher	Screwnet	30.84 ± 0.99	28.87 ± 1.42	32.18 ± 1.22
	GAPartNet	23.73 ± 1.18	26.28 ± 1.45	-
	Ours	**20.44 ± 1.12**	**21.72 ± 1.05**	**20.06 ± 1.30**
Slider Knife*	Screwnet	28.22 ± 3.43	17.47 ± 3.18	-
	GAPartNet	58.64 ± 4.09	9.42 ±3.82	-
	Ours	**14.18 ± 3.25**	**6.38 ± 4.14**	-
Stapler*	Screwnet	36.81 ± 4.32	16.34 ± 3.41	-
	GAPartNet	19.17 ± 5.15	13.04 ± 3.92	-
	Ours	16.26 ± 4.98	**8.19 ± 3.36**	-

TABLE III: Results for real world dataset. We report the mean error (with 95% confidence intervals) for joint axis directions (in degrees), positions (in centimeters), and joint state estimation (in deg/cm), averaged over the available axes per object. All the objects were in partially opened state for GAPartNet. * represents objects unseen during training.



(a) (b) (c) (d)

Fig. 3: This figure shows Sawyer robot manipulating the two joints of the fridge in the order of demonstration estimated by Pokenet. (a) and (b) shows human demonstrations while (c) and (d) shows robot manipulating the object.

Table I, PokeNet is the only method capable of jointly predicting both single and multi DoF objects, recovering occluded or fully shut joints, and estimating joint states and manipulation order by using human demonstrations. To the best of our knowledge, PokeNet is the first unified framework that integrates all these aspects of articulated object understanding.

In contrast, GAPartNet performs poorly in unseen categories with fully closed links. Its reliance on a single partial point cloud often causes it to miss parts when joints are occluded or objects are completely shut. This issue is especially worse in high-DoF objects such as storage furniture and boxes, or in objects with unusual geometries like slider knives, leading to missed segmentations and incorrect articulation estimates. PokeNet avoids these failures by using human demonstrations, which naturally reveal occluded joints and allow the model to capture articulation structure more reliably.

VII. Limitations and Future Work

Although our method demonstrates state-of-the-art performance in estimating joint parameters, it has several limitations. First, it does not estimate contact points for manipulation. Moreover, the model does not consider obstacles, relying solely on inverse kinematics for robot motion, which increases the risk of collisions. Finally, it does not recover full object geometry, which can be useful for generating digital twins. Future work will address these gaps by detecting contact points and integrating collision-aware planning.

VIII. Conclusion

In this work, we present a novel framework that learns the kinematic constraints and manipulation sequences of multi-

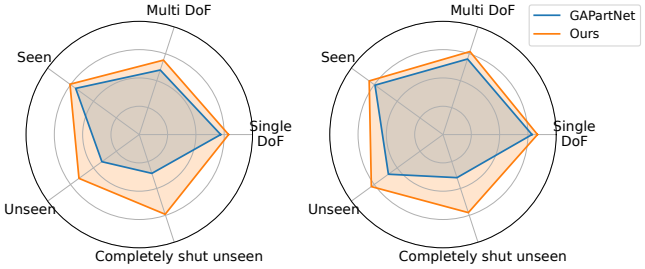


Fig. 4: Comparison of PokeNet and GAPartNet on different object categories. Left: axis direction accuracy; Right: axis displacement accuracy (higher is better). GAPartNet struggles with fully shut parts, while PokeNet uses human demonstrations to generalize robustly.

DoF objects directly from human demonstrations. Our approach surpasses state-of-the-art articulated object modeling methods on simulated dataset by 25% and on real-world dataset by 30%. In addition, we contribute a new annotated real-world dataset of articulated objects that captures human interactions. Unlike prior methods, our framework makes no restrictive assumptions about the number of degrees of freedom or object categories, and requires no prior object knowledge beyond a specified maximum limit. Finally, we demonstrate that robots can leverage the learned representations to successfully manipulate diverse novel objects in real-world settings.

IX. Acknowledgment

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-24-1-0239.

References

[1] B. Abbatematteo, S. Tellex, and G. D. Konidaris, "Learning to generalize kinematic models to novel objects," in *Conference on Robot Learning*, 2019.

[2] H. Zhang, B. Eisner, and D. Held, "Flowbot++: Learning generalized articulated objects manipulation via articulation projection," in *Proceedings of The 7th Conference on Robot Learning*, 2023, pp. 1222–1241.

[3] B. Eisner*, H. Zhang*, and D. Held, "Flowbot3d: Learning 3d articulation flow to manipulate articulated objects," in *Robotics: Science and Systems (RSS)*, 2022.

[4] Z. Chen, A. Walsman, M. Memmel, K. Mo, A. Fang, K. Vemuri, A. Wu, D. Fox, and A. Gupta, "Urdformer: A pipeline for constructing articulated simulation environments from real-world images," *arXiv preprint arXiv:2405.11656*, 2024.

[5] 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," *arXiv preprint arXiv:2211.05272*, 2022.

[6] A. Jain, R. Lioutikov, C. Chuck, and S. Niekum, "Screwnet: Category-independent articulation model estimation from depth images using screw theory," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 670–13 677.

[7] A. Jain, S. Giguere, R. Lioutikov, and S. Niekum, "Distributional depth-based estimation of object articulation models," in *Conference on Robot Learning*, 2022, pp. 1611–1621.

[8] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.

[9] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *arXiv preprint arXiv:2106.10689*, 2021.

[10] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, 2023.

[11] S. Kim, J. Ha, Y. H. Kim, Y. Lee, and F. C. Park, "Screwsplat: An end-to-end method for articulated object recognition," *arXiv preprint arXiv:2508.02146*, 2025.

[12] Q. Yu, X. Yuan, Y. jiang, J. Chen, D. Zheng, C. Hao, Y. You, Y. Chen, Y. Mu, L. Liu, and C. Lu, "Artgs:3d gaussian splatting for interactive visual-physical modeling and manipulation of articulated objects," *arXiv preprint arXiv:2507.02600*, 2025.

[13] Y. Liu, B. Jia, R. Lu, J. Ni, S.-C. Zhu, and S. Huang, "Artgs: Building interactable replicas of complex articulated objects via gaussian splatting," *arXiv preprint arXiv:2502.19459*, 2025.

[14] J. Kerr, C. M. Kim, M. Wu, B. Yi, Q. Wang, K. Goldberg, and A. Kanazawa, "Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction," in *8th Annual Conference on Robot Learning*, 2024.

[15] X. Li, H. Wang, L. Yi, L. Guibas, A. L. Abbott, and S. Song, "Category-level articulated object pose estimation," *arxiv preprint arXiv:1912.11913*, 2020.

[16] L. Yi, H. Huang, D. Liu, E. Kalogerakis, H. Su, and L. Guibas, "Deep part induction from articulated object pairs," *ACM Trans. Graph.*, 2018.

[17] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 272–277.

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

[19] R. Martín-Martín, S. Höfer, and O. Brock, "An integrated approach to visual perception of articulated objects," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 5091–5097.

[20] N. Nie, S. Y. Gadre, K. Ehsani, and S. Song, "Structure from action: Learning interactions for articulated object 3d structure discovery," *arxiv preprint arXiv:2207.08997*, 2022.

[21] J. Liu, A. Mahdavi-Amiri, and M. Savva, "PARIS: Part-level reconstruction and motion analysis for articulated objects," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

[22] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3141–3150.

[23] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "SAPIEN: A simulated part-based interactive environment," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[24] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[26] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, pp. 2280–2292, 2014.

[27] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, 2020, p. 213–229.

[28] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017.

[29] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.