Revisiting Articulated Parts Perception in Robot Manipulation

Xiaoqian Wu, Yejie Guo, Xiaoyang Chen, Lixin Yang, Cewu Lu*, Yong-Lu Li*
Shanghai Jiao Tong University
{enlighten, gyj123, cxy_computer, siriusyang, lucewu, yonglu_li}@sjtu.edu.cn

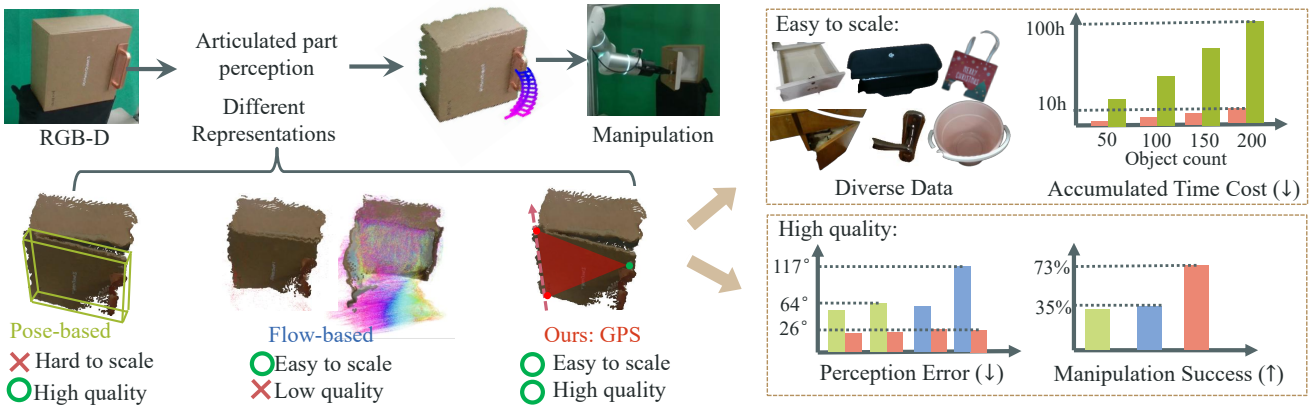

Figure 1. **Overview.** We aim to enhance robotic manipulation by improving articulated part perception from a single RGB-D image. The core of our approach is a novel affordance representation, GPS, which is easy to scale with high-quality data. Our model outperforms existing pose-based and flow-based methods in part perception accuracy and manipulation success rate.

Abstract

We are surrounded by various objects with movable, articulated parts, e.g.*, box, handle, door. An accurate and generalizable perception of articulated parts is essential to enhance robotic manipulation capabilities. Building on this need, recent efforts in articulated parts perception have followed two main directions: One line of work uses pose-based representation, which requires high manual cost; in parallel, affordance-based methods extract future object motion from point tracking without additional manual efforts, but suffer from low-quality data. In this paper, we propose a new representation of articulated parts,* ***G****eometric* ***P****rimary* ***S****tructure* ***(GPS)****, an abstraction of the part geometry structure to balance scalability and quality. For efficient and scalable data collection, GPS is integrated with a portable Virtual Reality (VR) device and requires only one minute to annotate one object sequence. This direct human annotation provides higher quality than the estimated affordance. With this efficient VR-GPS system, we collect* ***41K*** *frames for* ***234*** *objects across six part classes, and train a generalizable GPS model with a single RGB-D object image as input. For object manipulation, we deploy a heuristic policy based on GPS prediction. Without any in-domain fine-tuning, our method achieves an* ***73%*** *success rate, covering 270 initial states for 9 objects. Our code, data and reusable tool are available at https://enlighten0707.github.io/gps.*

*Corresponding author.

1. Introduction

Accurately estimating object state is crucial for robots to perform diverse manipulation tasks. While recent advances have improved state estimation for rigid objects [10, 36], non-rigid objects such as articulated objects remain challenging due to their complex kinematic structures. In this paper, we focus on enhancing the perception and estimation of articulated parts from a single RGB-D image to improve robotic manipulation capabilities, as illustrated in Fig. 1.

Recent efforts in articulated parts perception have followed two main directions: pose-based and affordance-

Figure 2. **VR hardware and interfaces.** The headset tracks the fingers and renders the tracking points as red points.

based representations. *Pose-based* methods represent parts as segmentation and pose estimation, defining canonical positions and orientations for each part class [8, 18]. However, obtaining such data requires significant manual effort: synthetic CAD models are created by professional artists [20, 27, 39], and real-world objects require elaborate scanning, modeling, and frame-wise pose annotation [22, 23, 42]. Although post-processing methods have emerged to reconstruct articulated objects from visual inputs [13, 16, 19, 24], they still face limitations including category restriction [13], long processing time [16], and sensitivity to errors in real-world scenarios [19, 24].

In parallel, *affordance-based* methods [1, 37, 41] model object motion by predicting future point trajectories, often referred to as object flow. The ground truth flow is typically extracted automatically from human-object interaction videos via point tracking followed by temporal downsampling. While this minimizes manual annotation, it suffers from two key issues: **1) Tracking Error.** Point tracking [21, 40] is prone to inaccuracies caused by self-occlusion or camera movement. In Fig. 1, when opening the door, the trajectories near the edge deviate significantly from the true motion, and those near the axis should not have been truncated so prematurely; **2) Flow Ambiguity.** The mapping from point trajectories to articulated properties is inherently ambiguous. The scale of the extracted flow varies for objects of similar type but different sizes, and sub-flows are temporally inconsistent for manipulations with the same range of motion but non-uniform execution speed. This makes the predicted flow highly sensitive to both object scale and temporal variations.

To address these limitations, we need an abstraction of the part geometry that is both informative and scalable. Our solution comprises two key components: **1) Explicit Axis Annotation.** We propose to explicitly annotate and predict the motion axis, *e.g.*, a revolute axis for a laptop lid, a prismatic axis for a drawer. This axis captures an inherent invariance during part motion, which helps to mitigate noise from point tracking and the ambiguity in flow. To enable accurate and efficient annotation, we employ a Virtual Reality (VR) device, Meta Quest 3, with SLAM capabilities. Before interacting with the object, virtual points are placed on the axis position. These points remain stationary in 3D space and are unaffected by headset movement, and are rendered in the headset for annotators to verify and adjust. **2) Hand as a Motion Proxy.** Instead of tracking the object part directly, which can be unreliable, we use the human hand firmly grasping the part as a stable motion proxy. The hand has a consistent structure and is typically closer to the camera, making it more suitable for robust tracking than the object part. We track the midpoint between the thumb and index finger to represent the part's motion, and this hand point is also visually rendered in the headset. Example interfaces are shown in Fig. 2. Leveraging this efficient data collection method, we introduce the **G**eometric **P**rimary **S**tructure **(GPS)** as a new affordance representation. As is shown in Fig. 1, GPS is defined by three keypoints: two (red) determine the axis, and a third (green) is attached to the hand. The three keypoints form a plane (red), constraining the part rotation. Part translation can be formulated similarly, constrained by a prismatic axis and a plane perpendicular to it.

Our VR-GPS system requires only one minute to annotate a hand-object interaction video, without any manual post-processing, and yields higher-quality data than estimated flow. Using this system, we have collected a dataset including **41K** RGB-D frames for **234** objects across six part classes, providing rich object knowledge. Based on this data, we propose a generalized GPS prediction model that transfers well to other datasets and outperforms both pose-based and flow-based methods in articulated parts understanding. As the first step, we verify the effectiveness of our data and hope the community will join us to enlarge our dataset continuously with the reusable VR-GPS system.

For robot manipulation, we develop a heuristic policy, using the predicted GPS to select initial grasp proposals from AnyGrasp [6] and generate subsequent waypoints. Real-robot experiments cover 270 initial states (*i.e.*, different part poses and camera views) for 9 objects with diverse appearances. Without any in-domain fine-tuning, our method achieves an impressive 73% success rate.

Our main contributions are: 1) The novel GPS representation for articulated parts, which balances between scalability and quality in data collection. 2) A VR-collected GPS dataset rich in object geometry knowledge. 3) A generalizable GPS prediction model is proposed that demonstrates superior performance in facilitating real-world robotic manipulation of daily objects.

2. Related Work

To enhance robot manipulation, a system should understand both interaction semantics [9, 31, 38] and geometry of manipulated objects. In this paper, we focus on understanding articulated object geometry. Prior work on articulated object representation can be broadly grouped into two main categories: pose-based and affordance-based methods.

Pose-based methods aim to estimate part segmentation and 6-DoF pose, often defined in a Normalized Part Co-

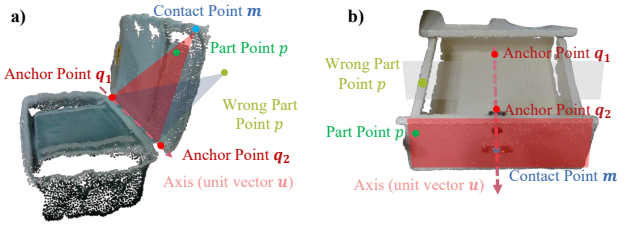

Figure 3. **Geometric structure formulation.** (a) Part rotation along revolute axis; (b) part translation along prismatic axis.

ordinate Space (NPCS) for each object category [18, 34]. GAPartNet [8] extends this idea by introducing cross-category part classes based on functional similarity. Data for pose-based methods primarily come from three sources: 1) Synthetic datasets, which provide high-fidelity 3D assets created by artists in simulation platforms [8, 25, 27, 32, 39]; 2) Real-world scans, involving professionally captured object models [2, 11, 20, 26] with frame-wise pose and camera annotations from hand-object interactions [22, 23, 42]; 3) Post-processing methods that recover articulation from visual inputs. For example, PARIS [19] and ArtGS [24] leverages neural radiance fields and 3D Gaussians to reconstruct objects and estimate joints, and RSRD [16] uses a 4D differentiable part model to recover object motions from an object scan and single monocular video.

Affordance-based methods focus on identifying *where* to manipulate (*i.e.*, contact point) an object and *how* to manipulate (*i.e.*, future trajectory). Contact point prediction are often predicted from annotated point clouds [5] or semantic features from diffusion models [14]. Beyond contact points, predicting the future trajectory of an object is also crucial for robot manipulation. VRB [1] derive affordance cues from hand-object contact and hand motion, employing 2D representations as guidance for robot learning. However, 2D affordance offers only coarse supervision. GFlow [41] extracts future object motion from point tracking on the HOI4D dataset [22], assuming known camera parameters for each frame. Yet, in practice, most real-world data lacks precisely annotated camera parameters and consists only of visual inputs. While recent methods improve dynamic scene reconstruction from monocular views [21, 35, 43], they still struggle with articulated objects and remain error-prone.

3. Definition

3.1. Part Rotation

Many real-world objects have rotatable parts, *e.g.*, box lid, kettle, book, lamp. Given an input point cloud $\mathcal{P} \in \mathbb{R}^{N\times 3}$ of an object, its structure is constrained as $\{\mathbf{u}, \mathbf{q}, \mathbf{m}\}$. Here $\mathbf{u} \in \mathbb{R}^3$ is a unit vector of the revolute axis direction, $\mathbf{q}$ is an anchor point to determine the position of the evolute axis, and $\mathbf{m}$ is a contact point where a movable part contacts with a human hand or a robot end-effector. The trajectory of $\mathbf{m}$ concerning a rotation angle θ is

$$\mathbf{m}(\theta) = \cos(\theta)I\cdot\mathbf{m}+(1-\cos(\theta))\mathbf{u}\mathbf{u}^T\cdot\mathbf{m}+\sin(\theta)\mathbf{R}\cdot\mathbf{m}+\mathbf{q}, \tag{1}$$

where I denotes an identity matrix and R denotes the skew symmetric matrix of $\mathbf{u}$.

Our GPS is defined as $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{p}\}$ (Fig. 3(a)). The axis $\{\mathbf{u}, \mathbf{q}\}$ is determined by two anchor points $\{\mathbf{q}_1, \mathbf{q}_2\}$, where $\mathbf{q}_1 = \mathbf{q}$, $\mathbf{q}_2 = \mathbf{q} + c\mathbf{u}$, c is an arbitrary constant. The contact point $\mathbf{m}$ is not unique. For example, when opening the bag in Fig. 3(a), we can touch different positions on three edges. Thus, learning the exact position of a contact point will increase the difficulty of GPS learning. Also, the predicted $\mathbf{m}$ cannot be benchmarked accurately. Therefore, we define a part point $\mathbf{p}$ as:

$$\mathbf{p}\cdot((\mathbf{q}_1-\mathbf{m})\times(\mathbf{q}_2-\mathbf{m})) = 0, \tag{2}$$

$\mathbf{p}$ should be on the plane defined by $\mathbf{q}_1, \mathbf{q}_2, \mathbf{m}$, constraining the principal structure. Additionally, in robot manipulation experiments (Sec. 6), we conduct an ablation study and find that using loose constraints $\mathbf{p}$ is more generalized than $\mathbf{m}$ to select better grasp proposals.

3.2. Part Translation

Other objects have parts that translate along a prismatic axis. Its structure is constrained by $\{\mathbf{u}, \mathbf{m}\}$, where $\mathbf{u}$ is a unit vector of the axis direction, $\mathbf{m}$ is the contact point (Fig. 3(b)). The trajectory of $\mathbf{m}$ with respect to an offset δ is:

$$\mathbf{m}(\theta) = \mathbf{m} + \delta\mathbf{u}, \tag{3}$$

Its GPS is $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{p}\}$, where $\{\mathbf{q}_1, \mathbf{q}_2\}$ defines axis direction: $\mathbf{u}\cdot(\mathbf{q}_1-\mathbf{q}_2) = 0$. For translational parts, the axis defines the direction of motion but not its absolute position. To simplify model learning, we fix this axis to pass through the part's geometric center. As is demonstrated in Sec. 3.1, the contact point $\mathbf{m}$ is loosed to part point $\mathbf{p}$:

$$(\mathbf{p}-\mathbf{m})\cdot(\mathbf{q}_1-\mathbf{q}_2) = 0, \tag{4}$$

$\mathbf{p}$ should be on the plane defined by normal vector $\mathbf{q}_1-\mathbf{q}_2$ and contact point $\mathbf{m}$.

4. Data Collection

4.1. System Design

Our system is built around the Meta Quest 3 VR device. The system consists of the following components: **1) VR headset** serving as a display and providing spatial computation. Based on its Augmented Reality (AR) functionality, we can virtually place a point in the real 3D world, track its movement, and record its coordinates in the scene point cloud. Hand tracking is performed using the built-in

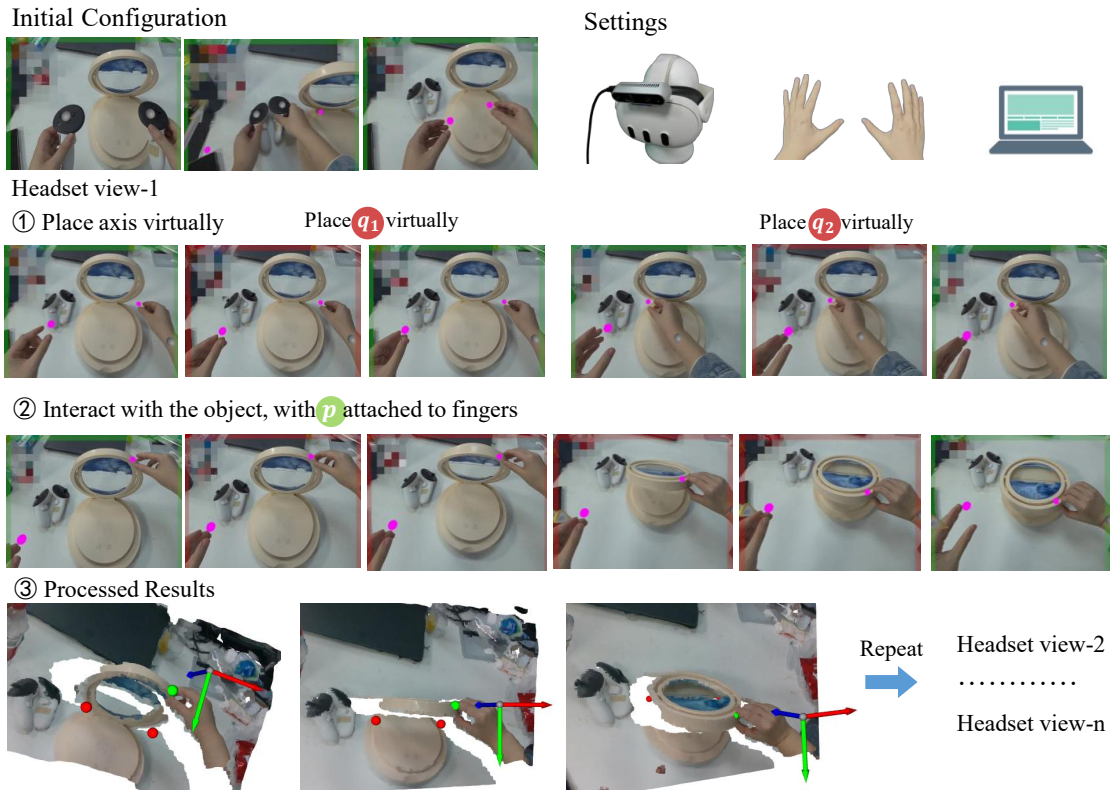

Figure 4. **Hardware settings and annotation pipeline.** Before interacting with an object, the annotator places axis $\{\mathbf{q_1}, \mathbf{q_2}\}$ virtually. During interaction, the part point $\mathbf{p}$ is attached with fingers. For each object, multiple RGB-D videos with different headset views are recorded. The annotator begins and ends the recording by performing a pinch gesture with their non-interacting hand.

VR function, with the midpoint between the thumb and index finger being tracked and rendered in real time within the headset interface. **2) Intel RealSense D435 camera** is mounted on the headset via a 3D-printed bracket to capture RGB-D data and reconstruct the scene point cloud [3]. The virtual point is defined in the world frame of the VR device, while the scene point cloud is defined in the RealSense camera frame. Therefore, after recording, the coordinate of the virtual point is transformed into the camera frame, using calibration parameters between RealSense and the headset. **3) Laptop** receiving and storing data streams from both the VR device and the RealSense camera.

We design the annotation process to record GPS in real time without requiring complex post-processing. To annotate the axis points $\{\mathbf{q_1}, \mathbf{q_2}\}$, the annotator places virtual points along the axis *before* interaction. Then, the points stay *fixed* in the current 3D space despite headset move and camera view move, with the help of spatial memory in VR computing. To annotate the part point $\mathbf{p}$, the annotator interacts with the object while recording RGB-D video, with the virtual points *moving* with the annotator's fingers.

The overall annotation pipeline is shown in Fig 4. After initial configuration, the annotator annotates each object sequentially. For each object, the annotator places axis points $\{\mathbf{q_1}, \mathbf{q_2}\}$ virtually records RGB-D videos, then changes to another headset view and repeats the process. After recording, we apply coordinate transformation and object segmentation [30]. Finally, we obtain the object RGB-D data with GPS annotation across different object part poses and camera views.

One unique advantage of AR-based annotation is that one point can be placed *anywhere* in the 3d space. To annotate points, a direct way is to click on pixels after recording RGB-D videos and map pixel coordinates to 3D points based on the camera intrinsic. However, when the correct points are not on the surface facing the camera, *e.g.*, the revolute axis of a thin lamp, it is difficult to accurately annotate them. Moreover, the real-time visibility of virtual points allows annotators to adjust and correct placements during interaction, which is an advantage over offline methods such as hand reconstruction or 3D GUI-based annotation. Furthermore, built on a widely available commercial VR device, our VR-GPS is portable, not limited to lab settings.

4.2. Data Analysis

Low Cost. The device cost of our VR-GPS is relatively

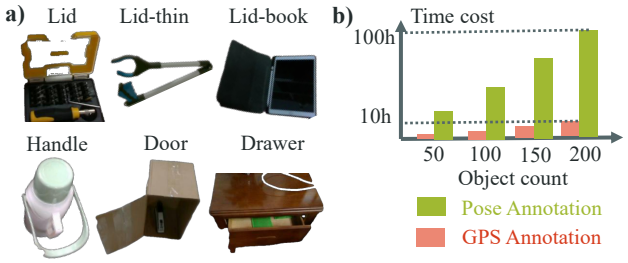

Figure 5. **Dataset overview.** Our VR-GPS is diverse and efficient.

low (*800 dollars*), without an expensive MoCap system or 3D scanning devices. For each object, three videos with different camera views are recorded. The average time to annotate one video is one minute, which is efficient.

Dataset Statistics. Using our portable and efficient VR-GPS, we collect 41K frames for 234 objects. As shown in Fig. 5(a), the objects belong to six part classes: Lid (*e.g.*, Box, Laptop), Lid-thin (*e.g.*, Pole, Stapler), Lid-book (*e.g.*, Ipad, Booklet), Handle (*e.g.*, Kettle, Bucket), Door (*e.g.*, Safe, Cabinet), Drawer. The drawer has a prismatic axis, while the others have a revolute axis. Fig. 5(b) compares the time cost of GPS annotation against pose-based annotation, highlighting the efficiency of our approach.

Data Quality. The error of the virtual point coordinate mainly comes from poor lighting conditions, or too small a distance between the annotator and the VR-defined boundary. We double-check the collected data to ensure quality. After checking, 3% of the data is filtered.

5. Geometric Structure Learning

5.1. Model Design

Feature Extraction. Given RGB image $\mathcal{I} \in \mathbb{R}^{H\times W\times 3}$, the depth map $\mathcal{D} \in \mathbb{R}^{H\times W}$, our goal is to predict GPS parameters $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{p}\}$. Following CAPNet [12], we use SAM2 [30] and FeatUp [7] to extract RGB feature maps, where each pixel corresponds to a vector with dimension $d_s = 480$ representing the semantic information of the RGB image at the corresponding location. Subsequently, we concatenate each RGB feature vector with its corresponding 3D point cloud $\mathcal{P} \in \mathbb{R}^3$ in a point-wise manner. To aggregate part category semantics, we use CLIP [29] text encoder to convert category descriptions into semantic features. Then their dimensions are reduced via MLPs to $d_t = 6$, and concatenated with $\mathcal{P}$. Finally, the merged features $\mathcal{P}_t \in \mathbb{R}^{N\times(6+d_s+d_t)}$ are processed via PointNet++ [28] to extract the densely fused RGBD features f.

Part Rotation Loss Design. For part rotation along revolute axis, the geometric features f is processed via three separate MLPs to predict axis direction $\hat{\mathbf{u}}$, the axis anchor point $\hat{\mathbf{q}_1}$, and the part point $\hat{\mathbf{p}}$. The training loss is

Category	Method	AADE (↓)	AAOE(↓)	APDE(↓)
Laptop	CAPNet [12]	37.33°	0.33	62.78°
	Ours-Flow	23.98°	0.13	16.78°
	Ours	**17.17°**	**0.13**	**9.26°**
Trashcan	CAPNet [12]	34.54°	0.28	46.88°
	Ours-Flow	29.57°	0.25	28.93°
	Ours	**24.46°**	**0.21**	**18.27°**
Safe	CAPNet [12]	56.49°	0.48	85.10°
	Ours-Flow	27.51°	0.38	51.36°
	Ours	**15.52°**	**0.26**	**33.75°**
Bucket	CAPNet [12]	48.21°	0.41	61.70°
	Ours-Flow	39.05°	0.31	28.28°
	Ours	**32.05°**	**0.27**	**19.60°**
Category	**Method**	**AADE (↓)**	**AAOE(↓)**	**APOE(↓)**
Drawer	CAPNet [12]	37.82°	0.13	0.12
	Ours-Flow	17.30°	0.26	0.16
	Ours-GPS	**14.32°**	**0.25**	**0.14**

Table 1. GPS learning performance on HOI4D [22] .

$\mathcal{L} = \mathcal{L}_{ad} + \mathcal{L}_{ao} + \mathcal{L}_{pd}$, referring to the axis direction loss, axis offset loss, and part direction loss:

$$\mathcal{L}_{ad} = 1 - \left| \frac{\hat{\mathbf{u}}}{\|\hat{\mathbf{u}}\|} \cdot \frac{\mathbf{q}_1 - \mathbf{q}_2}{\|\mathbf{q}_1 - \mathbf{q}_2\|} \right|, \tag{5}$$

$$\mathcal{L}_{ao} = \left| \frac{\hat{\mathbf{q}_1} - \mathbf{q}_1}{\|\hat{\mathbf{q}_1} - \mathbf{q}_1\|} \cdot \frac{\mathbf{q}_2 - \mathbf{q}_1}{\|\mathbf{q}_2 - \mathbf{q}_1\|} \right|, , \tag{6}$$

$$\mathcal{L}_{pd} = 1 - \frac{(\mathbf{q}_2 - \mathbf{q}_1) \times \mathbf{p}}{\|(\mathbf{q}_2 - \mathbf{q}_1) \times \mathbf{p}\|} \cdot \frac{(\mathbf{q}_2 - \mathbf{q}_1) \times \hat{\mathbf{p}}}{\|(\mathbf{q}_2 - \mathbf{q}_1) \times \hat{\mathbf{p}}\|}, \tag{7}$$

Part Translation Loss Design. For part translation along the prismatic axis, the training loss is $\mathcal{L} = \mathcal{L}_{ad} + \mathcal{L}_{ao} + \mathcal{L}_{po}$, referring to the axis direction loss, the axis offset loss and the part offset loss $\mathcal{L}_{po}$:

$$\mathcal{L}_{po} = \frac{|((\mathbf{q}_2 - \mathbf{q}_1) \times (\mathbf{p} - \mathbf{q}_1)) \cdot (\hat{\mathbf{p}} - \mathbf{q}_1)|}{\|(\mathbf{q}_2 - \mathbf{q}_1) \times (\mathbf{p} - \mathbf{q}_1)\|}. \tag{8}$$

5.2. Evaluation

5.2.1. Benchmark

Metrics. To evaluate GPS prediction, we design metrics that quantify the direction and offset errors of both the axis and the part. For the axis, we adopt metrics Average Axis Direction Error (AADE) and Average Axis Offset Error (AAOE). For the parts, we adopt Average Part Direction Error (APDE) for part rotation, and Average Part Offset Error (APOE) for part translation. The maximum offset error is 2 with the point cloud normalized into a unit cube.

Test Datasets. The GPS model is trained on our VR-GPS dataset. To assess its generalization capability, we evaluate the model on two external datasets: HOI4D [22] and RGBD-Art [12]. **HOI4D** contains egocentric RGB-D videos capturing human-object hand interactions. **RGBD-Art** contains synthetic articulated objects annotations built

Category	Method	AADE (↓)	AAOE(↓)	APDE(↓)
Laptop	GFlow [41]	51.58°	0.33	116.05°
	Ours-Flow	33.51°	0.29	49.27°
	Ours	**22.76°**	**0.19**	**22.81°**
Trashcan	GFlow [41]	52.40°	0.82	120.00°
	Ours-Flow	38.90°	0.57	42.07°
	Ours	**27.65°**	**0.27**	**20.79°**
Safe	GFlow [41]	65.78°	0.68	102.77°
	Ours-Flow	28.92°	0.29	45.81°
	Ours	**19.29°**	**0.23**	**22.96°**
Bucket	GFlow [41]	59.76°	0.74	128.74°
	Ours-Flow	53.98°	0.46	69.03°
	Ours	**36.25°**	**0.18**	**38.85°**
Category	**Method**	**AADE (↓)**	**AAOE(↓)**	**APOE(↓)**
Drawer	GFlow [41]	61.83°	0.81	0.98
	Ours-Flow	28.41°	0.31	0.38
	Ours-GPS	**23.78°**	**0.20**	**0.21**

Table 2. GPS learning performance on RGBD-Art [12].

upon GAPartNet [8] dataset, featuring photorealistic RGB images and depth noise simulated like real sensors. They both provide ground-truth part segmentation and pose estimation, and we convert it into GPS by deriving $\mathbf{q}_1$, $\mathbf{q}_2$, and $\mathbf{p}$ from the bounding boxes.

Test Object Categories. We evaluate on five object categories: Laptop, Trashcan, Door, Bucket, and Drawer, the overlapping categories among VR-GPS, the test datasets, and the baselines. These categories correspond to the following parts: Laptop and Trashcan have a Lid, Safe has a Door, and Drawer has a Drawer. The part classes Lid-thin and Lid-book are excluded due to a lack of suitable test data in HOI4D and RGBD-Art, where corresponding object categories are absent or interactions are largely pick-and-place with minimal articulation. These categories will instead be evaluated in the robot experiments in Sec. 6.

5.2.2. Performance Comparison

We first evaluate our method against the pose-based method, **CAPNet** [12], predicting articulated part segmentation and pose, and trained on synthetic dataset RGBD-Art. We convert its output into our GPS representation and evaluate on HOI4D, which serves as an out-of-domain benchmark for both methods. We do not compare performance on RGBD-Art because it overlaps with CAPNet's training domain. As shown in Tab. 1, GPS outperforms CAPNet across all five categories. Although CAPNet employs depth noise augmentation, it still struggles with the sim-to-real gap, whereas our real-world data strategy alleviates this issue. We observe a relatively higher GPS prediction error for the Bucket category, possibly because of the large depth sensor noise on its thin handle.

We next compare with the flow-based method **GFlow** [41], a state-of-the-art 3D scene flow predictor. We use its public checkpoint ScaleFlow-L trained on HOI4D and evaluate on the out-of-domain RGBD-Art benchmark. The predicted flow is converted into GPS for direct comparison. In Tab. 2, GFlow exhibits large errors. We attribute the results to the inherent sensitivity of flow and the limited diversity of its training data, which hinders its ability to generalize to novel object instances.

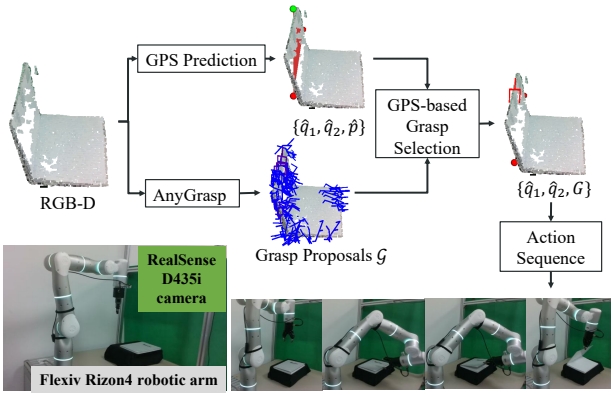

Figure 6. Heuristic manipulation policy based on GPS prediction.

We further fine-tune GFlow with our data (**Ours-Flow**). To adapt our data for flow-based training, we use TraceAnything [21] to extract globally aligned point trajectories under a moving camera. Following the setup in GFlow, each interaction sequence is divided into 4 timesteps. For a fair comparison with GPS, we modify the original GFlow by using object-level point clouds with annotated masks and performing per-point flow prediction. Tab. 1 and 2 show that Ours-GPS outperforms Ours-Flow because Ours-Flow suffers from inaccurate tracking results and inherent ambiguity of flow. Additionally, the performance of Ours-GPS degrades on RGBD-Art compared to HOI4D, primarily due to the simulated depth data in RGBD-Art lacking real-world fidelity. Nevertheless, we selected RGBD-Art for evaluation as it contains diverse articulated objects.

6. Real Robot Experiments

6.1. Heuristic Policy

For a robot to manipulate an object in the real physical world, we plan feasible robot trajectories based on predicted GPS $\{\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2, \hat{\mathbf{p}}\}$. The process is detailed in Fig. 6. This involved two modules: **1) Initial grasp selection** to decide where to grasp the objects. We use AnyGrasp [6] to generate grasp proposals $\mathcal{G} = \{\mathbf{G}_k\}_{k=1}^{K}$. Then, GPS predictions are used to select the best initial grasp $\mathbf{G}$, which corresponds to an end-effector pose $\mathbf{T}_1$. To select $\mathbf{G}$, GPS predictions are used for a scoring function of grasp proposals. A good grasp should be close to the plane defined by $\{\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2, \hat{\mathbf{p}}\}$. The additional constraints are combined with the original grasp confidence scores to form the final grasp

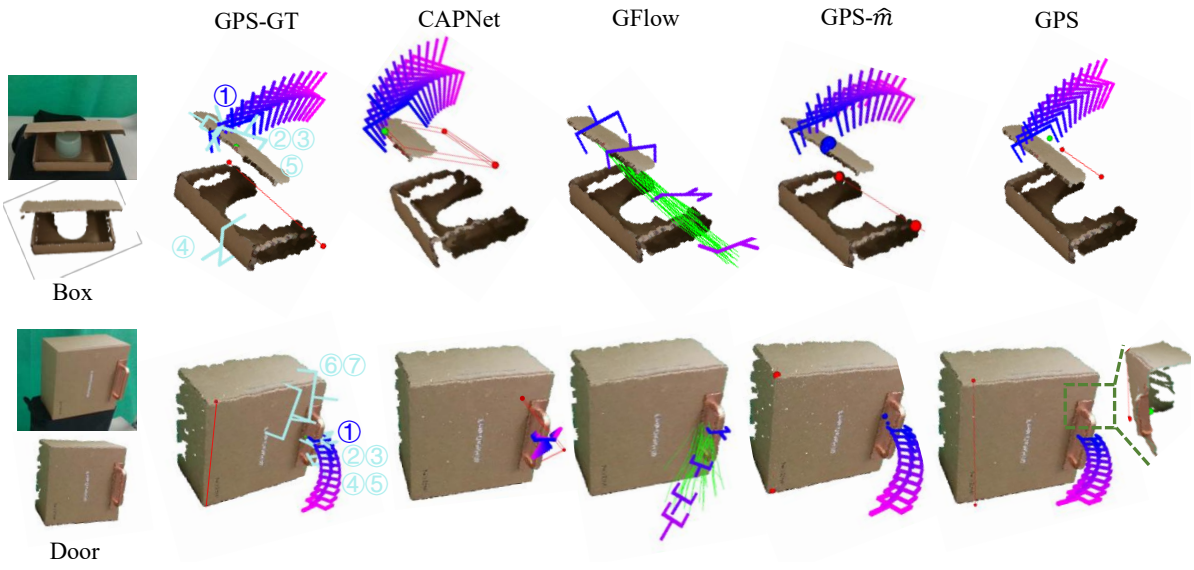

Figure 7. **Visualization results of robot experiments.** Waypoints $\mathcal{T}_\mathbf{G} = \{\mathbf{T}_t\}_{t=1}^{\mathbf{t}}$ are depicted in gradient color from blue to purple. In GPS-GT and GPS, red points denote $\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2$, a green point denotes $\hat{\mathbf{p}}$, and cyan grasps marks different initial grasp poses. For Door task in GPS, an additional view is provided to clearly display the otherwise occluded $\hat{\mathbf{p}}$. In CAPNet, the predicted part bounding boxes are shown in red. In GFlow, the predicted flow is visualized as a green line, spanning a total of four steps. In GPS-$\hat{\mathbf{m}}$, the blue points indicate $\hat{\mathbf{m}}$.

scores. For objects with a prismatic joint, the criterion is the distance to the $\{\hat{\mathbf{q}_1}, \hat{\mathbf{q}_2}, \hat{\mathbf{p}}\}$ plane ; **2) Waypoints generation** to decide how to move objects after grasping. After grasping the target part, the manipulation policy explicitly utilizes $\{\hat{\mathbf{q}_1}, \hat{\mathbf{q}_2}\}$ and the robot's current state to generate action sequences $\mathcal{T}_\mathbf{G} = \{\mathbf{T}_t\}_{t=1}^{\mathbf{t}}$ based on Eq. 1, 3.

6.2. Settings

For real-robot experiments, we set up one Flexiv Rizon4 arm equipped with a gripper and an Intel RealSense D435 RGB-D camera. As shown in Fig. 6, the camera is mounted on the wrist of the robotic arm, which was calibrated in an eye-in-hand configuration. GPS and AnyGrasp [6] prediction results are in the camera frame; thus, we transform them from the camera frame into the robot base frame using the calibrated hand-eye matrix.

We test on 9 objects with diverse appearances. Their categories and part classes are: Box (Lid), Document-Box (Lid), Bucket (Handle), Door (Door), Drawer (Drawer), Notebook (Lid-book), Folder (Lid-book), Lamp (Lid-thin), Clapperboard (Lid-thin). An object is successfully manipulated if its part is rotated by 50°(revolute axis) or moved 5cm (prismatic axis). Each waypoint rotates a part by 5°(revolute axis) or move it by 0.5cm (prismatic axis). The camera is moved to different views to obtain the object point cloud before manipulation. Each object is tested with 30 trials, each trial different state, *i.e.*, a combination of 6 different camera views and 5 different initial part poses.

After generating the waypoints $\mathcal{T}_\mathbf{G} = \{\mathbf{T}_t\}_{t=1}^{\mathbf{t}}$, we check it with planning algorithm RRT*[15] to avoid kinematically infeasible trajectory. If verified, the trajectory is executed. Otherwise, we directly mark this trial as a failure.

6.3. Results

We use results from different perception methods to generate initial grasp and axis-guided waypoints, and use the success rate to measure perception model performance. The results and visualization are shown in Tab 3 and Fig. 7. We mainly answer three questions: **1)** Can we find a good initial grasp from the loose plane constraint from GPS? **2)** How does GPS perform as a perception module in robot manipulation? **3)** What are the typical failure cases?

6.3.1. Initial Grasp Evaluation

To verify GPS's ability to find a good initial grasp with a loose plane constraint, we use human-annotated GPS-GT to generate initial grasp and waypoints, execute, and report the success rate. As GPS-GT is not predicted by a GPS model, we can exclude GPS prediction error and verify the GPS representation itself. Two visualization results for the box and door are illustrated in Fig. 7. The box example demonstrates how the $\mathbf{p}$-constraint effectively modulates grasp selection. While the cyan grasp G_4 has the highest original confidence, it drops to the 4th rank after the GPS-based geometric re-scoring, leading to the selection of the top-ranked blue grasp G_1 instead. The door example shows the importance of the original grasp of confidence. Grasps G_6 and G_7, though geometrically close to $\mathbf{p}$, are assigned low final scores due to their low grasp confidence. Tab. 3 reports

Method	Box	Document-Box	Bucket	Door	Drawer	Notebook	Folder	Lamp	Clapperboard	Avg-overlap	Avg-all
GPS-GT	93%	93%	87%	93%	87%	90%	90%	93%	97%	91%	91%
CAPNet [12]	60%	30%	63%	40%	43%	13%	27%	13%	7%	47%	33%
GFlow [41]	60%	37%	60%	17%	53%	17%	27%	23%	20%	45%	35%
GPS-$\hat{\mathbf{m}}$	77%	57%	53%	53%	50%	63%	70%	33%	67%	58%	58%
GPS	**93%**	**90%**	**67%**	**67%**	**60%**	**73%**	**70%**	**67%**	**73%**	**75%**	**73%**

Table 3. **Robot manipulation successful rate.** We test on 9 objects, each with 30 trials. The first 5 objects are overlapped categories with baselines [12, 41]. We average the success rate on them as "Avg-overlap". The successful rate for all the 9 objects is "Avg-all".

an overall 91% success rate over 270 trials, verifying the effectiveness of GPS even in a geometrically loose form.

6.3.2. Performance Comparison

We next evaluate our method by replacing GPS-GT with predictions from a learned **GPS** model. As is shown in Tab. 3, our approach achieves an average success rate of 73% without any in-domain fine-tuning. Fig. 7 shows that the predicted GPS closely aligns with the ground-truth annotation, leading to similar successful trajectories. Below we compare against three baselines: CAPNet [12], GFlow [41] and GPS-$\hat{\mathbf{m}}$. They are all related to select grasp from a given or inferred contact point $\mathbf{m}$. We calculate the distance between a grasp and the contact point, and combine it with the original grasp score to select $\mathbf{G}$.

CAPNet [12] predicts the part bounding box, from which we derive the articulation axis and contact point. For the door example Fig. 7, CAPNet fails to correctly recognize the closed door structure, leading to an invalid prediction. For the box example, the estimated axis is inaccurate, causing the robot to bend the lid rather than open it properly. Overall, CAPNet attains only a 33% success rate across the tested objects, primarily due to the sim-to-real gap.

For **GFlow** [41], we use the heuristic policy in its original paper: a contact point $\mathbf{m}$ is manually given, query points near $\mathbf{m}$ are selected to predict scene flow, and Singular Value Decomposition is used to align the end-effector's motion with the predicted flow, which has 4 execution steps. Despite the given contact point (actually unfair for comparison), GFlow [41] is still prone to errors, with a 35% success rate. This is largely due to its training on limited data diversity and the inherent difficulty in learning reliable flow fields. As is shown in Fig. 7, the predicted flow for the box deviate significantly, and the door flow deviates downward.

For **GPS-$\hat{\mathbf{m}}$**, we conduct ablation study to predict $\mathbf{m}$ instead of predicting $\mathbf{p}$, as mentioned in Sec. 3.1. To acquire ground-truth $\mathbf{m}$, we fit a Gaussian mixture model (GMM) [1] to the closest 200 points around $\mathbf{p}$, parameterized by $\{\mu_i, \Sigma_i\}_{i=1}^5$. The model predicts $\{\mu_i\}_{i=1}^5$ coordinates instead of $\mathbf{p}$. The learned model performs poorer with a 58% success rate. The folder example in Fig. 8 is a typical failure case, where the prediction $\hat{\mathbf{m}}$ is wrong and far from easily graspable edges. Thus, using loose geometric constraints $\mathbf{p}$ is more generalized for manipulation.

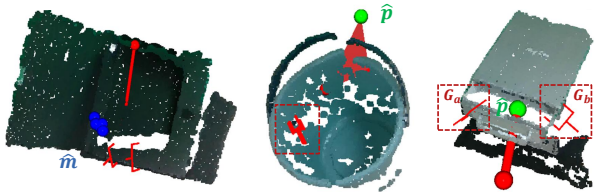

Figure 8. **Failure cases.** The folder is a failure example for GPS-$\hat{\mathbf{m}}$, and the bucket and drawer are failure examples for GPS. Red points are $\{\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2\}$, green points are $\hat{\mathbf{p}}$, and blue points are $\hat{\mathbf{m}}$.

6.3.3. Failure Case Analysis

We show failure cases in Fig. 8. GPS is not predicted well under a few part poses and camera views, mainly influenced by point cloud noise, *e.g.*, the bucket example. In other cases, GPS fails to select a proper grasp despite a correct prediction. For example, $\hat{\mathbf{p}}$-error is small for the drawer, but a wrong grasp G_b near the body is selected, instead of a more reasonable grasp G_a. This is because the scoring function is not flexible enough to balance between grasp score and geometric constraints, which can be improved by fine-tuning the grasping model with GPS input, or integrating GPS with more advanced methods (*e.g.*, diffusion policy [4, 33], VLA model [17]) in future work.

7. Conclusion

This paper proposes a novel affordance representation GPS for articulated part estimation. It balances data scalability with annotation quality. With a data-efficient system integrated with VR device, we collect the VR-GPS dataset with rich object knowledge. The learned GPS model has better perception performance and can facilitate a robot to manipulate daily objects via a heuristic policy.

8. Acknowledgments

This work was supported in part by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China, the National Natural Science Foundation of China under Grant No. U25A20442, 62306175, Shanghai Municipal Science and Technology Major Project No. 2025SHZDZX025G14.

References

[1] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023. 2, 3, 8

[2] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015. 3

[3] Sirui Chen, Chen Wang, Kaden Nguyen, Li Fei-Fei, and C Karen Liu. Arcap: Collecting high-quality human demonstrations for robot learning with augmented reality feedback. *arXiv preprint arXiv:2410.08464*, 2024. 4

[4] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023. 8

[5] Shengheng Deng, Xun Xu, Chaozheng Wu, Ke Chen, and Kui Jia. 3d affordancenet: A benchmark for visual object affordance understanding. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1778–1787, 2021. 3

[6] Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 39(5):3929–3945, 2023. 2, 6, 7

[7] Stephanie Fu, Mark Hamilton, Laura Brandt, Axel Feldman, Zhoutong Zhang, and William T Freeman. Featup: A model-agnostic framework for features at any resolution. *arXiv preprint arXiv:2403.10516*, 2024. 5

[8] Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7081–7091, 2023. 2, 3, 6

[9] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022. 2

[10] Xingyi He, Jiaming Sun, Yuang Wang, Di Huang, Hujun Bao, and Xiaowei Zhou. Onepose++: Keypoint-free one-shot object pose estimation without cad models. *Advances in Neural Information Processing Systems*, 35:35103–35115, 2022. 1

[11] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 international conference on computer vision*, pages 858–865. IEEE, 2011. 3

[12] Jingshun Huang, Haitao Lin, Tianyu Wang, Yanwei Fu, Xiangyang Xue, and Yi Zhu. Cap-net: A unified network for 6d pose and size estimation of categorical articulated parts from a single rgb-d image. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 11654–11664, 2025. 5, 6, 8

[13] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022. 2

[14] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrun Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2024. 3

[15] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011. 7

[16] Justin Kerr, Chung Min Kim, Mingxuan Wu, Brent Yi, Qianqian Wang, Ken Goldberg, and Angjoo Kanazawa. Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction. *arXiv preprint arXiv:2409.18121*, 2024. 2, 3

[17] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 8

[18] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3706–3715, 2020. 2, 3

[19] Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. Paris: Part-level reconstruction and motion analysis for articulated objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 352–363, 2023. 2, 3

[20] Liu Liu, Wenqiang Xu, Haoyuan Fu, Sucheng Qian, Qiaojun Yu, Yang Han, and Cewu Lu. Akb-48: A real-world articulated object knowledge base. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14809–14818, 2022. 2, 3

[21] Xinhang Liu, Yuxi Xiao, Donny Y Chen, Jiashi Feng, Yu-Wing Tai, Chi-Keung Tang, and Bingyi Kang. Trace anything: Representing any video in 4d via trajectory fields. *arXiv preprint arXiv:2510.13802*, 2025. 2, 3, 6

[22] Yunze Liu, Yun Liu, Che Jiang, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, Zhoujie Fu, He Wang, and Li Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21013–21022, 2022. 2, 3, 5

[23] Yun Liu, Haolin Yang, Xu Si, Ling Liu, Zipeng Li, Yuxiang Zhang, Yebin Liu, and Li Yi. Taco: Benchmarking generalizable bimanual tool-action-object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21740–21751, 2024. 2, 3

[24] Yu Liu, Baoxiong Jia, Ruijie Lu, Junfeng Ni, Song-Chun Zhu, and Siyuan Huang. Artgs: Building interactable replicas of complex articulated objects via gaussian splatting. *arXiv preprint arXiv:2502.19459*, 2025. 2, 3

[25] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 3

[26] Roberto Martín-Martín, Clemens Eppner, and Oliver Brock. The rbo dataset of articulated objects and interactions. *The International Journal of Robotics Research*, 38(9):1013–1019, 2019. 3

[27] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao 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*, pages 909–918, 2019. 2, 3

[28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 5

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 5

[30] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 4, 5

[31] Dandan Shan, Jiaqi Geng, Michelle Shu, and David F Fouhey. Understanding human hands in contact at internet scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9869–9878, 2020. 2

[32] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. igibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7520–7527. IEEE, 2021. 3

[33] Chenxi Wang, Hongjie Fang, Hao-Shu Fang, and Cewu Lu. Rise: 3d perception makes real-world robot imitation simple and effective. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2870–2877. IEEE, 2024. 8

[34] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2642–2651, 2019. 3

[35] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. 3

[36] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024. 1

[37] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 2

[38] Xiaoqian Wu, Yong-Lu Li, Jianhua Sun, and Cewu Lu. Symbol-llm: leverage language models for symbolic system in visual human activity reasoning. *Advances in neural information processing systems*, 36:29680–29691, 2023. 2

[39] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11097–11107, 2020. 2, 3

[40] Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. Spatialtrackerv2: 3d point tracking made easy. *arXiv preprint arXiv:2507.12462*, 2025. 2

[41] Chengbo Yuan, Chuan Wen, Tong Zhang, and Yang Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024. 2, 3, 6, 8

[42] Xinyu Zhan, Lixin Yang, Yifei Zhao, Kangrui Mao, Hanlin Xu, Zenan Lin, Kailin Li, and Cewu Lu. Oakink2: A dataset of bimanual hands-object manipulation in complex task completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 445–456, 2024. 2, 3

[43] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024. 3