OPD: Single-view 3D Openable Part Detection

Hanxiao Jiang, Yongsen Mao, Manolis Savva, and Angel Xuan Chang

Simon Fraser University
3dlg-hcvc.github.io/OPD/

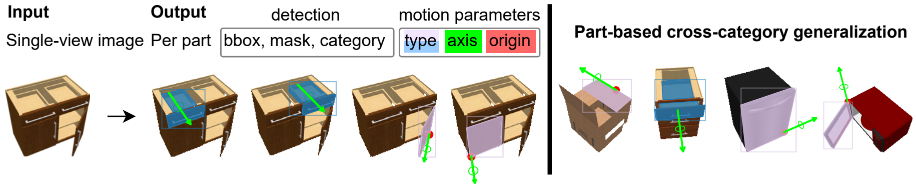

Fig. 1: In the openable part detection (OPD) task the input is a single view image, and the outputs are detected openable parts as well as their motion parameters. We design OPDRCNN: a neural architecture for this task that operates at the part level, allowing generalization across diverse object categories.

Abstract. We address the task of predicting what parts of an object can open and how they move when they do so. The input is a single image of an object, and as output we detect what parts of the object can open, and the motion parameters describing the articulation of each openable part. To tackle this task, we create two datasets of 3D objects: OPDSynth based on synthetic objects, and OPDReal based on RGBD reconstructions of real objects. We then design OPDRCNN, a neural architecture that detects openable parts and predicts their motion parameters. Our experiments show that this is a challenging task especially when considering generalization across object categories, and the limited amount of information in a single image. Our architecture outperforms baselines and prior work especially for RGB image inputs.

Keywords: Articulated parts, single-view motion parameter estimation

1 Introduction

There is increasing interest in training embodied AI agents that interact with the world based on visual perception. Recently, Batra et al. [2] introduced rearrangement as a challenge bringing together researchers in machine learning, vision, and robotics. Common household tasks such as "put dishes in the cupboard" or "get cup from the cabinet" can be viewed as object rearrangement. A key challenge in such tasks is identifying which parts can be opened, and how they move to open and close.

To address this problem, we introduce the *openable part detection* (OPD) task, where the goal is to identify openable parts, and their motion type and parameters (see Figure 1). We focus on predicting the openable parts ("door", "drawer", and "lid") of an object, and their motion parameters from a single RGB image input. More specifically, we focus on container objects (e.g. cabinets, ovens, etc). Containers need to be opened to look for hidden objects to reach, or to place away objects. Methods that can identify what can be opened and how offer a useful abstraction that can be used in more complex tasks [18, 21–23, 31].

Prior work [7, 11, 36] has looked at predicting motion types and parameters from 3D point clouds. Point-cloud based methods rely on depth information, often sampled from a reconstructed mesh that aggregates information from multiple views. In addition, Li et al. [11] assumes that the kinematic chain and the part structure of the objects are given. To handle different kinematic structures, they train a separate model for each structure. This approach is not scalable to the wide variety of structures found in the real world, even for objects in the same category. For example, cabinets may have two, three or four drawers.

We study the task of identifying openable parts and motion parameters from a single-view image, and investigate whether a structure-agnostic image-based approach can be effective at this task. Our approach OpdRcnn extends instance segmentation networks (specifically MaskRCNN) to identify openable parts and predict their motion parameters. This structure-agnostic approach allows us to more easily achieve cross-category generality. That is, a single model can tackle instances from a variety of object categories (e.g. cabinets, washing machines). In addition, we investigate whether depth information is helpful for this task.

In summary, we make the following contributions: i) we propose the OPD task for predicting openable parts and their motion parameters from single-view images; ii) we contribute two datasets of objects annotated with their openable parts and motion parameters: OPDSynth and OPDReal; and iii) we design OpdRcnn: a simple image-based, structure-agnostic neural architecture for openable part detection and motion estimation across diverse object categories. We evaluate our approach with different input modalities and compare against baselines to show which aspects of the task are challenging. We show that depth is not necessary for accurate part detection and that we can predict motion parameters from RGB images. Our approach significantly outperforms baselines especially when requiring both accurate part detection and part motion parameter estimation.

2 Related Work

Part segmentation and analysis. Part segmentation has been widely studied in the vision and graphics communities for both 2D images and 3D representations. Much of the prior work relies on part annotations in 2D images [4, 40], or 3D CAD models with semantically labelled part segmentations [19, 29, 30, 36, 38]. Annotated 3D part datasets have been used to study part segmentation by

Table 1: Summary of prior work on motion parameter estimation (motion type, axis, and rotation origin). We indicate the input modality, whether the method has cross-category generalization (CC), whether part segmentations are predicted (Seg) as opposed to using ground-truth parts, whether object pose (OP) and part state (PS) are predicted. Most prior work takes point cloud (PC) inputs. In contrast, our input is single-view images (RGB, D, or RGB-D).

Method	Input	CC	Seg	OP	PS	#cat	#obj	#part
Snapshot [6]	3D mesh						368	368
Shape2Motion [30]	PC	✓	✓			45	2440	6762
RPMNet [36]	PC	✓	✓				969	1420
DeepPartInduction [37]	Pair of PCs		✓			16	16881	
MultiBodySync [8]	Multiple PCs		✓			16		
ScrewNet [9]	Depth video	✓				9	4496	4496
Liu et al. [14]	RGB video	✓	✓			3		
ANCSH [11]	Single-view PC		✓	✓	✓	5	237	343
Abbatematteo et al. [1]	RGB-D		✓			5	6	8
VIAOP [39]	RGB-D	✓				6	149	618
OpdRcnn (ours)	Single-view RGB(-D)	✓	✓	✓	✓	11	683	1343

rendering images [24] or directly in 3D (e.g. meshes or point clouds) [19, 38]. These datasets have fostered the development of different methods to address part segmentation on images [27, 28, 33]. Many of these focus on human/animal part segmentation, and use hierarchical methods to parse both objects and the parts. Beyond part segmentation, [15] further enhanced the PASCAL VOC Part dataset [4] with state information. In contrast, we directly detect and segment the parts of interest using standard object instance segmentation methods [3, 5].

Unlike prior work, we focus on a small set of openable part categories. Part datasets differ in what parts they focus on (e.g., human body parts, or fine-grained vs coarse-grained object parts). Determining the set of parts of interest can be tricky, as the set of possible parts can be large and ill-defined, with what constitutes a part varying across object categories. Therefore, we focus on a small set of openable parts in common household objects. We believe this set of parts is a small, practical set that is important for object interaction.

Articulated object motion prediction. Part mobility analysis is a long-standing problem in 3D computer graphics. Early work [17] has focused on learning the part mobility in mechanical assemblies. Xu et al. [35] used a mobility tree formalism to further explore object and part mobility in indoor 3D scenes. More recent work[20] proposed a joint-aware deformation framework based on shape analysis and optimization to predict motion joint parameters. Part mobility analysis has also been performed on sequences of RGBD scans [10]. More recently, there has been increasing interest in data-driven methods for studying articulated objects and estimating motion parameters[11, 30]. To support these data-driven

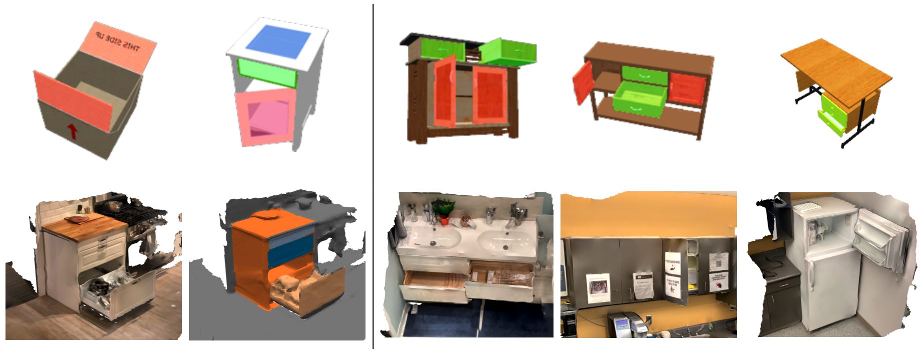

Fig. 2: Example articulated objects from the OPDSynth dataset and the OPDReal dataset. The first row is from OPDSynth. Left: different openable part categories (`lid`, in orange, `drawer` in green, `door` in red). Right: Cabinet objects with different kinematic structures and varying numbers of openable parts. The second row is from our OPDReal dataset. Left: reconstructed cabinet and its semantic part segmentation. Right: example reconstructed objects from different categories.

approaches, there has been concurrent development of datasets of annotated part articulations for synthetic [30, 34, 36] and reconstructed [13, 16] 3D objects.

Table 1 summarizes the part mobility prediction tasks defined by this and other recent work. Hu et al. [6] predict joint parameters given pre-segmented 3D objects. Other work predicts segmentation together with motion parameters, for 2.5D inputs [36, 37], 3D point clouds [30] or for sequences of RGBD scans [10]. Abbatematteo et al. [1], Li et al. [11] have the most similar task setting with our work, predicting the part segmentation, part pose, and joint parameters from a single view image. However, both require depth as input and knowledge of the kinematic chain of each object. They require training a separate model for each object category, where the object category is defined as having the same structure (i.e. same kinematic chain). This means that different models need to be trained for cabinets with 3 drawers and cabinets with 4 drawers. In contrast, we identify all openable parts of an object in an input RGB image, without assuming a specific structure with given number of parts. This allows us to train a single model that generalizes across categories. Note that Abbatematteo et al. [1] also use MaskRCNN for segmentation, but they do not analyze or report the part segmentation and detection performance of their model.

More recent work has started to explore training of single models for motion prediction across categories and structures [8, 9, 9, 14]. Zeng et al. [39] proposed an optical flow-based approach on RGB-D images given segmentation masks of the moving part and fixed part. They evaluate only on ground truth segmentation and do not investigate how part segmentation and detection influences the accuracy of motion prediction. Others have proposed to predict articulated part pose from depth sequences [9], image video sequences [14], or synchronizing multiple point

Table 2: OPD dataset statistics. We create two datasets of objects with openable parts: OPDSynth and OPDReal. The datasets contain various object categories with potentially multiple openable parts. We annotate the semantic part segmentation and articulation parameters on 3D polygonal meshes, allowing us to generate many views of each object with ground truth.

		Category										
		Storage	Table	Bin	Fridge	Microwave	Washer	Dishwasher	Oven	Safe	Box	Suitcase
OPDSynth	Objects	366	76	69	42	13	17	41	24	30	28	7
	Parts	809	187	75	68	13	17	42	36	30	59	7
	Images	89300	20600	9225	7850	1625	2125	5225	4200	3750	6600	875
OPDReal	Objects	231	16	7	12	7	3	3	5	-	-	-
	Parts	787	35	7	27	7	3	3	6	-	-	-
	Images	27394	1175	474	1321	570	159	186	253	-	-	-

clouds [8]. In contrast, we focus on single-view image input and show that even without depth information, we can accurately predict motion parameters.

3 Problem Statement

Our input is a single RGB image I of a single articulated object and the output is the set of openable parts $P = \{p_1 \dots p_k\}$ (i.e. drawers, doors and lids) with their motion parameters $\varPhi = \{\phi_1 \dots \phi_k\}$. Figure 1 illustrates the input and output for our task. For each part $p_i = \{b_i, m_i, l_i\}$, we predict the 2D bounding box b_i, the segmentation mask m_i, and the semantic label $l_i \in \{\texttt{drawer}, \texttt{door}, \texttt{lid}\}$. The motion parameters ϕ_i of each openable part specify the motion type $c_i \in \{\texttt{prismatic}, \texttt{revolute}\}$, motion axis direction $a_i \in R^3$ and motion origin $o_i \in R^3$. Specifically, we have $\phi_i = [c_i, a_i, o_i]$ for revolute joints (e.g., door rotating around a hinge), and $\phi_i = [c_i, a_i]$ for prismatic joints (e.g., drawer sliding out). For simplicity, we assume that each part has only one motion.

4 OPD Dataset

To study our task, we collate two datasets of objects with openable parts, a dataset of synthetic 3D models (OPDSynth) and a dataset of real objects scanned with RGB-D sensors (OPDReal). For OPDSynth, we select objects with openable parts from an existing dataset of articulated 3D models PartNet-Mobility [34]. For OPDReal, we reconstruct 3D polygonal meshes for articulated objects in real indoor environments and annotate their parts and articulation information. Table 2 shows summary statistics of the number of objects and openable parts for each object category (see supplement for more detailed statistics).

OPDSynth. The PartNet-Mobility dataset contains 14K movable parts across 2,346 3D objects from 46 common indoor object categories [34]. We canonicalize the part names to 'drawer', 'door', or 'lid'. We select object categories with openable parts that can serve as containers. For each category, we identify the openable parts and label them as `drawer`, `door`, or `lid` (see supplement). Overall, we collated 683 objects with 1343 parts over 11 categories (see Table 2). We then render several views of each object to produce RGB, depth, and semantic part mask frames. Specifically, for each articulated object we render different motions for each part. In one motion state all parts are set to the minimum value in their motion range. Then for each part, we pick 3 random motion states and the maximum of the motion range, while the other parts remain at the minimum value. We render five images for each motion state from different camera viewpoints, and each image is composited on four different randomly selected background images (see supplement for rendering details).

OPDReal. To construct a dataset of real objects, we take 863 RGB-D video scans of indoor environments with articulated objects (residences, campus lounges, furniture showrooms) using iPad Pro 2021 devices. We obtain polygonal mesh reconstructions from these scans using the Open3D [41] implementation of RGB-D integration and Waechter et al. [25]'s implementation of texturing. We follow an annotate-validate strategy to filter and annotate this scanned data. Specifically, we: 1) Annotate the model quality and filter the scans with bad quality (insufficient geometry to annotate articulations); 2) Annotate the semantic part segmentation; 3) Validate the semantic segmentation; 4) Annotate the articulation parameters for articulated parts; 5) Validate the articulations through animating the moving parts; 6) Annotate a 'semantic OBB' (center at origin, semantic 'up' and 'front' axis direction) for each object; 7) Calculate consistent object pose (i.e. transformation between camera coordinates and object coordinates) based on semantic OBB. After filtering and annotation, we have a total of 763 polygonal meshes for 284 different objects across 8 object categories. Table 2 shows the distribution over different object categories.

We then project the 3D annotations back to the original RGB and depth frames from the scan videos. We select around 100 frames from each video and project the segmentation mask in 3D back to 2D. The articulation parameters in world coordinate are also projected to the camera coordinates of each frame. When selecting frames, we sample one frame every second ensuring that at least 1% of pixels belong to an openable part and at least 20% of parts are visible.

5 Approach

To address openable part detection, we leverage a instance segmentation network to identify openable parts by treating each part as an 'object instance'. Specifically, we augment Mask-RCNN [5] with heads for predicting the motion parameters. Mask-RCNN uses a backbone network for feature extraction and a region proposal network (RPN) to propose regions of interest (ROI) which are fed into branches that refine the bounding box and predict the mask and category label. By training Mask-RCNN on our OPD dataset, we can detect and segment openable parts. We attach additional branches to the output of the `RoiAlign` module to predict motion parameters. We consider two variants: i) OpdRcnn-C directly predicts the motion parameters in camera coordinates; and ii) OpdRcnn-O predicts camera

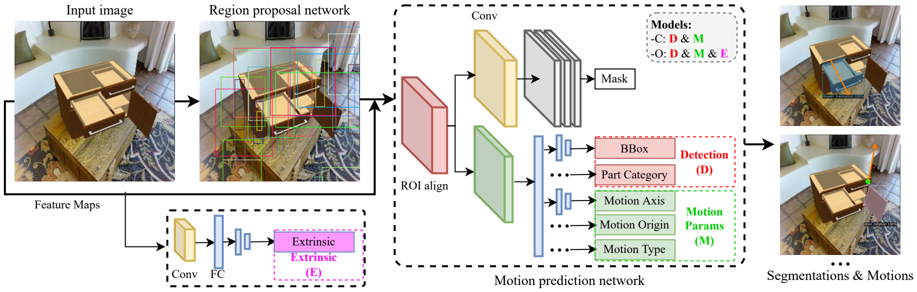

Fig. 3: Illustration of the network structure for our OpdRcnn-C and OpdRcnn-O architectures. We leverage a MaskRCNN backbone to detect openable parts. Additional heads are trained to predict motion parameters for each part.

extrinsics (object pose in our single-object setting) and motion parameters in world coordinates (canonical object coordinates). Figure 3 shows the overall architecture. All models use a cross-entropy loss for categorical prediction and a smooth L1 loss with $\beta = 1$ when regressing real values (see supplement).

OpdRcnn-C. For OpdRcnn-C, we add separate fully-connected layers to the `RoiAlign` module to directly predict the motion parameters. The original MaskRCNN branches predict the part label l_i and part bounding box delta δ_i for each part p_i in the box head. The box delta δ_i is combined with the box proposal from the RPN module to calculate the final output bounding box b_i. We add additional branches to predict the motion parameters $\phi_i = [c_i, a_i, o_i]$ (motion category, motion axis, motion origin) in the camera coordinate frame. We use the smooth L1 loss for the motion origin and motion axis, and the cross-entropy loss for the part joint type. Note that we only apply the motion origin loss for revolute joints, since the motion origin is not meaningful for prismatic joints.

OpdRcnn-O. For OpdRcnn-O, additional layers predict the 6 DoF object pose parameters to establish an object coordinate frame within which to predict motion axes and origins for openable parts. Following prior work [26], the object coordinate frame has consistent up and front orientations. The motivation is that motion origins and axes in object coordinates are more consistent than in camera coordinates. We are only dealing with a single object per image and consistent poses are available for each annotated object, so predicting the object pose is equivalent to predicting the extrinsic parameters of the camera pose. We regress the object pose parameters from image features using convolution and fully-connected layers (see Figure 3). OpdRcnn-O is trained with the same motion parameter loss as OpdRcnn-C, but with motion axes and origins expressed in object coordinates. We treat the extrinsics matrix as a vector of length 12 (9 for rotation, 3 for translation) and use the smooth L1 loss.

Implementation details. We implement our architecture with Detectron2 [32]. We initialize weights from a ResNet50 pretrained on ImageNet, and optimize with SGD. Unless otherwise specified, we use a learning rate of 0.001 with linear warmup for 500 steps and decay by 0.1 at 0.6 and 0.8 of the total steps.

We first train only on the detection and segmentation task with a learning rate of 0.0005 for 30000 steps. Then we pick the best weights for RGB, depth, and RGBD independently. Because our OpdRcnn-C and OpdRcnn-O have the same structure for detection and segmentation, we load the weights from the best detection and segmentation models and fully train with all losses. In both OpdRcnn-C and OpdRcnn-O we use the ratios $[1, 8, 8]$ to weigh the motion category loss, motion axis loss and motion origin loss respectively. In OpdRcnn-O we use 15 as the weight for the object pose loss.

During training we employ image space data augmentation to avoid overfitting (random flip, random brightness, and random contrast). During inference, we use a greedy non-maximum suppression (NMS) with IoU threshold 0.5 and choose the predicted bounding box with highest score. We use a confidence threshold of 0.05, and allow for 100 maximum part detections per image.

6 Experiments

6.1 Metrics

Part detection. To evaluate part detection and segmentation we use standard object detection and segmentation metrics over the part category, as implemented for MSCOCO [12]. In the main paper, we report mAP@IoU=0.5 for the predicted part label and 2D bounding box (**PDet**).

Part motion. To evaluate motion parameter estimation and measure the influence of the motion type we compute detection metrics over the motion type (motion-averaged mAP). This is in contrast with part-averaged mAP which is over the part category. A 'match' for motion-averaged mAP considers the predicted motion type and the 2D bounding box (**MDet**), and error thresholds for the motion axis and motion origin. We set the thresholds to 10° for axis error and 0.25 of the object's diagonal length for the origin distance (predicted origin to GT axis line). Motion parameters are evaluated in the camera coordinate frame. Motion origins for translation joints are all considered to match. We report several variants of part-averaged mAP: **PDet**, **PDet** with motion type matched (+**M**), **PDet** with motion type and motion axis matched (+**MA**), and **PDet** with motion type, motion axis and motion origin matched (+**MAO**). Motion-averaged mAP has the same variants with **MDet** instead of **PDet**. When evaluating motion parameters, prior work [11, 30] has focused on the average error only for correctly detected parts. In contrast, our metrics incorporate whether the part was successfully detected or not (in addition to error thresholds).

6.2 Baselines

As noted by Li et al. [11], knowing the canonical object coordinates can assist with predicting the motion axes and origin. This is because the motion axes are often one of the $\{x, y, z\}$ axes in object coordinates. Similarly, the motion origin for revolute joints is often at the edge of the object. So we design baselines that select (randomly or using the most frequent heuristic) from the three axes and the edges and corners of the normalized object bounding box. These baselines rely on known canonical object coordinates, so we take the part and object pose predictions from OpdRcnn-O. For origin prediction we use ground truth object sizes to convert from normalized to canonical coordinates, so these are strong baselines with access to additional information not available to OpdRcnn.

RandMot. A lower bound on performance that randomly predicts the part type, motion type, motion axis and motion origin. The motion origin is picked randomly from 19 points in the corner, edge center, face center and center of the object bounding box. The motion axis is picked randomly from three axis-aligned directions in the canonical object coordinates.

MostFreq. Selects the most frequent motion type, axis, and origin in object coordinates conditioned on the predicted part category (statistics from train set).

ANCSH. Li et al. [11]'s ANCSH predicts motion parameters (motion axis, motion origin) and part segmentation for single-view point clouds. This work assumes a fixed kinematic chain for all objects in a category. In other words, the number of parts, part labels, and part motion types are given as input. We re-implement ANCSH in PyTorch matching the results reported by the authors. Since ANCSH requires a fixed kinematic chain, we choose the most frequent kinematic chain: objects with one rotating door part (243 out of 683 total objects in OPDSynth). We train ANCSH on this 'one-door dataset'. The main evaluation is on the complete validation and test set (same as other baselines and approaches). The supplement provides additional comparisons on only one-door objects.

OpdPN. Baseline using a PointNet++ backbone to predict instance segmentation and motion parameters directly from an input point cloud. This baseline predicts the part category, part instance id, and motion parameters for each point. This architecture operates on a fixed number of parts so we train on objects with less than 5 parts. See the supplement for details on the architecture.

6.3 Results

Model comparisons. Table 3 shows the performance of the baselines and the two variants of OpdRcnn. We can make a number of observations. The RandMot baseline performs quite poorly for both part detection and part motion estimation metrics on all input scenarios, indicating the challenge of detecting openable parts and estimating their motion parameters. The MostFreq baseline is quite competitive if we only look at part detection and motion type detection (**PDet** and **MDet** metrics). This is not surprising as the MostFreq baseline leverages detections from OpdRcnn-O and the most frequent motion heuristic is relatively strong. However, when we look at precision of motion axis and motion origin predictions (+**MAO** and related metrics) we see that OpdRcnn-O significantly outperforms simpler baselines, especially for the motion-averaged metrics (higher than 60% mAP). It also outperforms the camera-centric OpdRcnn-C,

Table 3: Evaluation of openable part detection and part motion parameter estimation on the OPDSynth test set. The RandMot and MostFreq baselines use detections and extrinsic parameters from OpdRcnn-O. Both variants of OpdRcnn outperform baselines and prior work especially for RGB-only inputs and on metrics accounting for part motion parameter estimation.

		Part-averaged mAP % ↑				Motion-averaged mAP % ↑		
Input	Model	**PDet**	**+M**	**+MA**	**+MAO**	**MDet**	**+MA**	**+MAO**
RGBD	RandMot	5.0	1.3	0.2	0.1	6.2	0.7	0.3
	MostFreq	69.4	66.1	49.2	27.8	73.6	61.6	38.8
	OpdRcnn-C	**69.2**±0.26	**67.3**±0.25	42.6±0.25	40.9±0.31	**75.3**±0.09	55.3±0.23	53.9±0.20
	OpdRcnn-O	68.5±0.31	66.6±0.38	**52.6**±0.26	**49.0**±0.21	74.9±0.13	**65.3**±0.27	**62.9**±0.24
D (PC)	ANCSH [11]	2.7	2.7	2.3	2.1	3.9	3.1	2.8
	OpdPN	20.4	19.3	14.0	13.6	22.0	18.1	17.6
D	OpdRcnn-C	**68.2**±0.39	**66.5**±0.42	41.0±0.21	39.2±0.20	**73.0**±0.09	53.0±0.18	51.5±0.19
	OpdRcnn-O	67.3±0.43	65.6±0.28	**51.2**±0.44	**47.7**±0.24	72.2±0.14	**62.3**±0.25	**60.0**±0.23
RGB	OpdRcnn-C	**67.4**±0.26	**66.2**±0.18	40.9±0.21	38.0±0.19	**75.0**±0.14	53.4±0.20	51.4±0.18
	OpdRcnn-O	66.6±0.28	65.5±0.27	**50.7**±0.23	**46.9**±0.26	74.5±0.26	**63.8**±0.32	**61.5**±0.26

showing the benefit of object-centric representation relative to camera-centric representation when accurate estimation of motion axes and origins is important.

Effect of input modalities. Comparing different input modalities in Table 3 we see that overall, methods do best with RGBD, while D (depth, or point cloud) input, and RGB-only input are more challenging. For the depth input modality we compare against ANCSH [11] and the OpdPN baseline. Note that ANCSH requires separate models for different kinematic chains so it is severely disadvantaged when evaluating on the OPDSynth dataset that includes objects with varying number of parts and differing motion types. We observe that ANCSH is outperformed by OpdPN and both variations of OpdRcnn. One of the limitations of ANCSH is that it requires a prespecified kinematic chain (with fixed number of parts, part categories and motion types). Therefore, we also evaluated ANCSH in a setting constrained to 'single door' objects, where it performs more competitively (see supplement). These observations demonstrate the increased generality of our approach over baselines in terms of handling arbitrary object categories with changing numbers of moving parts and motion types. Moreover, both OpdRcnn variants perform well in the RGB-only setting.

Error metrics. We also compute error metrics as in Li et al. [11]. Prior work computes these metrics only for detected parts that match a ground truth part, thus including different number of instances for different methods. ANCSH and OpdPN have average motion axis error of 10.36° (for 6975 instances) and 6.25° (for 9862 instances) respectively. Both have average motion origin error of 0.09. Our proposed methods have comparable errors but over more detected parts. OpdRcnn-C and OpdRcnn-O have axis error of 9.06° and 6.67° for $\sim$ 22000 instances, and origin error of 0.11. These error metrics are restricted only to matched predictions and fail to capture detection performance differences.

Table 4: Results on the OPDReal test set. Overall, the task is more challenging on real objects. OpdRcnn-O has the highest performance across most metrics, and in particular for motion-averaged metrics including motion estimation (+**MAO**).

		Part-averaged mAP % ↑				Motion-averaged mAP % ↑		
Input	Model	**PDet**	+**M**	+**MA**	+**MAO**	**MDet**	+**MA**	+**MAO**
RGBD	RandMot	5.3	1.4	0.1	0.1	7.1	0.5	0.3
	MostFreq	56.6	54.6	34.0	21.6	71.6	50.4	32.2
	OpdRcnn-C	54.7	53.3	21.8	21.3	**73.4**	32.3	31.5
	OpdRcnn-O	**56.6**	**54.3**	**33.8**	**32.4**	73.3	**50.0**	**48.1**
D	OpdPN	15.4	15.3	12.1	11.5	22.8	17.9	17.0
	OpdRcnn-C	49.4	46.6	12.2	11.6	**61.1**	17.7	17.0
	OpdRcnn-O	**49.2**	**47.3**	**18.1**	**16.1**	60.3	**26.7**	**23.9**
RGB	OpdRcnn-C	**58.0**	**57.0**	22.2	21.3	73.6	32.6	31.4
	OpdRcnn-O	57.8	56.4	**33.0**	**30.8**	**74.0**	**48.7**	**45.7**

Real-world images. We demonstrate that we can apply our method on real-world images to detect openable parts and their motion parameters. We finetune the RGB and RGBD models trained on OPDSynth on the OPDReal training set. All hyperparameters are the same except we set the learning rates to 0.001 for RGB and RGBD, and 0.0001 for depth-only (D) models. Table 4 summarizes performance on our OPDReal dataset. Overall, the task is much more challenging with real data as seen by lower performance across all metrics for all methods. We see that OpdRcnn-O has the best performance overall across most metrics, and in particular for metrics that measure motion parameter estimation.

Qualitative visualizations. Figure 4 shows qualitative results from both the OPDSynth and OPDReal datasets. We see the overall trend that OpdRcnn-O outperforms OpdRcnn-C in terms of motion parameter estimation. We also see that both datasets have hard cases, with OPDReal being particularly challenging due to real-world appearance variability and the limited field of view of single-view images resulting in many partially observed parts.

6.4 Analysis

Are some parts more challenging than others? Table 6 shows the performance of OpdRcnn-O broken down by openable part category. The `drawer` parts exhibit translational motion, `lid` parts have rotational motions, and `door` parts exhibit both. We see that `lid` is more challenging than `drawer` and `door`, with considerably lower part detection mAP and significantly lower part motion estimation performance (+**MA** and +**MAO** metrics). This may be caused by fewer `lid` in the training data (89 `lid` vs 508 `door` and 363 `drawer` parts).

How does ground truth part and pose affect motion prediction? To understand how part detection influences motion prediction, we use ground truth

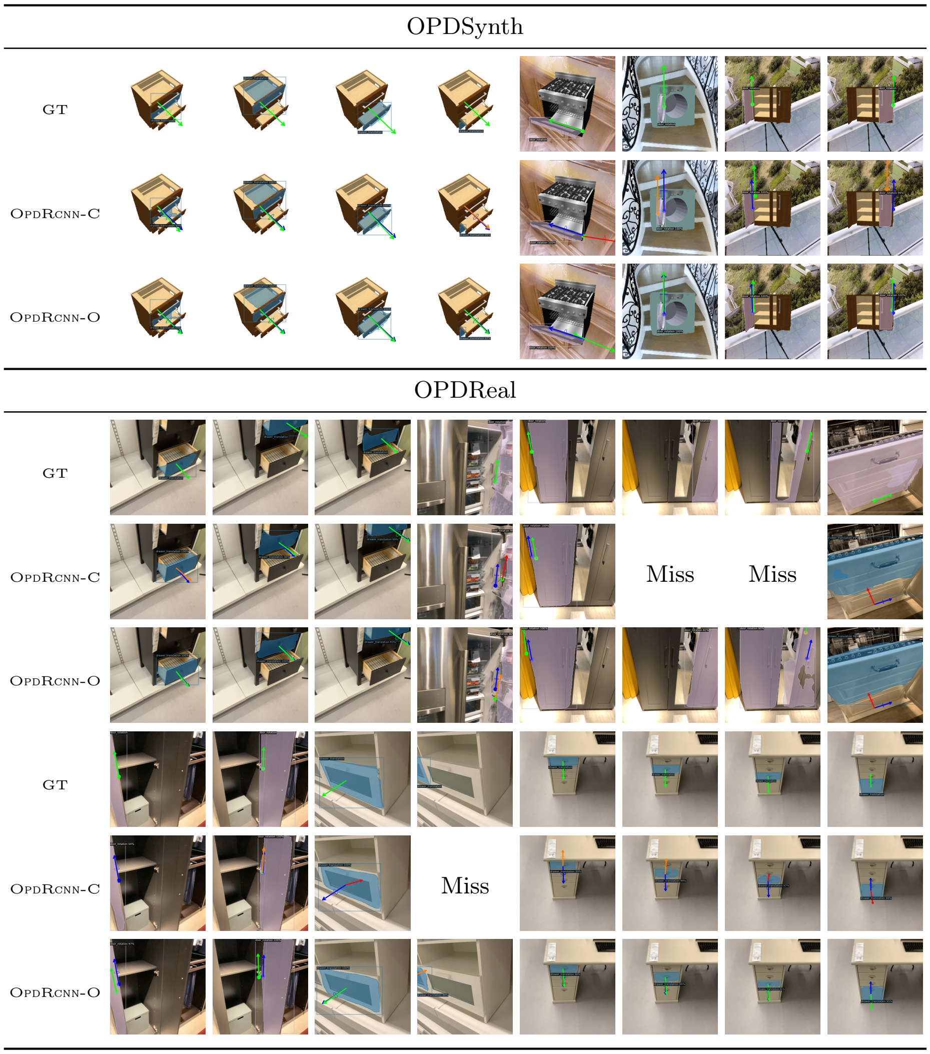

Fig. 4: Qualitative results from the OPDSynth and OPDReal val sets. The first row in each triplet shows the ground truth (GT) with each openable part mask, and the translational or rotational axis indicated in green. Other rows are predictions using OpdRcnn-C and OpdRcnn-O on RGBD inputs. The predicted axis is green if within 5° of the ground truth (also shown, in blue), in orange if within 10°, and in red if more than 10°. The first few OPDSynth examples show that translational `drawer` openable parts are relatively easy. Rotating `door` parts are more challenging (see high error predictions by OpdRcnn-C in the second row from the top). The OPDReal data is more challenging with many high axis error cases and entirely undetected parts (Miss). Particularly hard examples include unusual rotating doors that look like translating drawers (last column in top set), and openable parts only partly visible in the image frame (fourth column).

Table 5: Analysis of performance given ground truth part category, 2D bounding box, and object pose. Results for OpdRcnn-O are on the OPDSynth val set.

	Part-averaged mAP % ↑				Motion-averaged mAP % ↑		
	PDet	+**M**	+**MA**	+**MAO**	**MDet**	+**MA**	+**MAO**
RGBD OpdRcnn-O	72.5±0.34	70.6±0.29	51.7±0.62	47.1±0.59	75.4±0.07	61.6±0.32	59.0±0.32
GT Box2DPart	**99.0**±0.00	**90.9**±0.16	50.6±0.36	45.4±0.27	**89.7**±0.15	58.1±0.32	54.7±0.28
GT Pose	73.1±0.10	71.0±0.05	60.5±0.06	59.4±0.05	75.2±0.08	67.0±0.14	66.2±0.09
GT Box2DPartPose	**99.0**±0.00	90.6±0.37	**65.5**±0.24	**63.8**±0.17	89.5±0.19	**73.3**±0.26	**72.0**±0.30

Table 6: Per-category evaluation ofOpdRcnn-O model on the OPDSynth val set. All metrics use part-averaged mAP. The `drawer` openable parts are easiest overall and do not benefit much from depth information. In contrast `door` and in particular `lid` parts are more challenging and do benefit from depth in the input.

	`drawer`				`door`				`lid`			
Input	**PDet**	+**M**	+**MA**	+**MAO**	**PDet**	+**M**	+**MA**	+**MAO**	**PDet**	+**M**	+**MA**	+**MAO**
RGB	**81.4**	**80.9**	**71.5**	**71.5**	**86.0**	**81.1**	61.6	57.0	58.7	58.4	27.7	17.8
D	70.3	70.1	63.4	63.4	83.3	78.6	61.7	57.4	57.8	57.5	30.0	18.3
RGBD	71.9	71.4	65.9	65.9	85.9	79.9	**63.7**	**59.8**	**62.2**	**61.8**	**31.4**	**19.7**

(GT) for the part label, part 2D bounding box, and object pose with OpdRcnn-O on the OPDSynth validation set (see Table 5. As expected, when the GT part label and box are provided **PDet** is close to 100%. Surprisingly, having just the GT part label and box does not improve motion prediction. The ground truth object pose is more important for predicting the motion axis and origin correctly. Even with GT pose and 2D part information, the motion prediction is still imperfect. See the supplement for additional analysis.

Does depth information help? Table 3 shows that depth only (D) models are outperformed by RGB and RGBD models. Depth information is helpful in conjuction with RGB information as seen by the small performance boost between RGB and RGBD across all metrics. Depth-only models perform worse than RGB-only models across all motion-averaged metrics. We suspect that this is because most openable parts have minimal difference in depth values along their edges, and thus color is more helpful than depth for predicting openable part segmentation masks. Table 6 provides some insight. We see that for the `drawer` and `lid` category where detection is overall more challenging, depth information does not help and the motion prediction results are also bad. For the `door` category where detection results are higher depth offers additional information that improves motion parameter estimation (higher +**MA** and +**MAO** metrics for depth-only (D) and RGBD models). See the supplement for an additional analysis based on breaking down performance across detection mAP ranges.

Can we infer part motion states? A discrete notion of 'motion state' is often useful (e.g., "the fridge door is open" vs "the fridge door is closed"). We manually annotate a binary open vs closed motion state and continuous distance or angle offset from the closed state for all objects. We then add an MLP to the box head to predict binary motion state under a smooth L1 loss. We finetune the OpdRcnn-O model to predict the binary motion state along with part detection and motion parameter estimation. Finetuning is done for 5000 mini-batches. To evaluate motion state prediction we compute mAP values for 'match' or 'no match' of the binary motion state. Here, we define the mAP over motion type to evaluate the motion state prediction. Part-averaged mAP of the OpdRcnn-O model on the OPDSynth validation set is 62.8% for RGBD, 62.1% for RGB and 60.7% for D inputs. Motion-averaged mAP values are 62.0%, 64.2% and 59.9% respectively. These values should be contrasted with **PDet** and **MDet** in Table 3. We see that overall we can predict binary part motion states fairly well, though this too is a non-trivial task. We hypothesize that learning a 'threshold' for a binary notion of open vs closed is challenging (e.g., "fridge door cracked open").

6.5 Limitations

We focused on single objects with openable parts. The objects are fairly simple household objects without complex kinematic chains or complex motions (e.g., we cannot handle bifold doors). We also did not consider image inputs with multiple objects, each potentially possessing openable parts. A simple strategy to address this limitation would be to first detect distinct objects and then apply our approach on each object. Lastly, we focused on estimating the translation or rotation axis and rotation origin parameters but we do not estimate the range of motion for each part. This would be required to estimate a full part pose and to track the motion of an openable part from RGB video data.

7 Conclusion

We proposed the task of openable part detection and motion parameter estimation for single-view RGB images. We created a dataset of images from synthetic 3D articulated objects (OPDSynth) and of real objects reconstructed using RGBD sensors (OPDReal). We used these datasets to systematically study the performance of approaches for the openable part detection and motion estimation task, and investigate what aspects of the task are challenging. We found that the openable part detection task from RGB images is challenging especially when generalization across object and part categories is important. Our work is a first step, and there is much potential for future work in better understanding of articulated objects from real-world RGB images and RGB videos.

Acknowledgements: This work was funded in part by a Canada CIFAR AI Chair, a Canada Research Chair and NSERC Discovery Grant, and enabled in part by support from WestGrid and Compute Canada. We thank Sanjay Haresh for help with scanning and video narration, Yue Ruan for scanning and data annotation, and Supriya Pandhre, Xiaohao Sun, and Qirui Wu for help with data annotation.

References

1. Abbatematteo, B., Tellex, S., Konidaris, G.: Learning to generalize kinematic models to novel objects. In: Proceedings of the 3rd Conference on Robot Learning (2019)
2. Batra, D., Chang, A.X., Chernova, S., Davison, A.J., Deng, J., Koltun, V., Levine, S., Malik, J., Mordatch, I., Mottaghi, R., Savva, M., Su, H.: Rearrangement: A challenge for embodied AI. arXiv preprint arXiv:2011.01975 (2020)
3. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
4. Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., Yuille, A.: Detect what you can: Detecting and representing objects using holistic models and body parts. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1971–1978 (2014)
5. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2961–2969 (2017)
6. Hu, R., Li, W., Van Kaick, O., Shamir, A., Zhang, H., Huang, H.: Learning to predict part mobility from a single static snapshot. ACM Transactions on Graphics (TOG) **36**(6), 227 (2017)
7. Hu, R., Savva, M., van Kaick, O.: Functionality representations and applications for shape analysis. Computer Graphics Forum **37**(2), 603–624 (2018)
8. Huang, J., Wang, H., Birdal, T., Sung, M., Arrigoni, F., Hu, S.M., Guibas, L.: MultiBodySync: Multi-body segmentation and motion estimation via 3D scan synchronization. arXiv preprint arXiv:2101.06605 (2021)
9. Jain, A., Lioutikov, R., Niekum, S.: ScrewNet: Category-independent articulation model estimation from depth images using screw theory. arXiv preprint arXiv:2008.10518 (2020)
10. Li, H., Wan, G., Li, H., Sharf, A., Xu, K., Chen, B.: Mobility fitting using 4D RANSAC. Computer Graphics Forum **35**(5), 79–88 (2016)
11. Li, X., Wang, H., Yi, L., Guibas, L., Abbott, A.L., Song, S.: Category-level articulated object pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
12. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: European conference on computer vision, pp. 740–755, Springer (2014)
13. Liu, L., Xu, W., Fu, H., Qian, S., Han, Y., Lu, C.: AKB-48: A real-world articulated object knowledge base. arXiv preprint arXiv:2202.08432 (2022)
14. Liu, Q., Qiu, W., Wang, W., Hager, G.D., Yuille, A.L.: Nothing but geometric constraints: A model-free method for articulated object pose estimation. arXiv preprint arXiv:2012.00088 (2020)
15. Lu, C., Su, H., Li, Y., Lu, Y., Yi, L., Tang, C.K., Guibas, L.J.: Beyond holistic object recognition: Enriching image understanding with part states. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6955–6963 (2018)
16. Martín-Martín, R., Eppner, C., Brock, O.: The RBO dataset of articulated objects and interactions. The International Journal of Robotics Research **38**(9), 1013–1019 (2019)
17. Mitra, N.J., Yang, Y.L., Yan, D.M., Li, W., Agrawala, M.: Illustrating how mechanical assemblies work. ACM Transactions on Graphics-TOG **29**(4), 58 (2010)

18. Mittal, M., Hoeller, D., Farshidian, F., Hutter, M., Garg, A.: Articulated object interaction in unknown scenes with whole-body mobile manipulation. arXiv preprint arXiv:2103.10534 (2021)
19. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 909–918 (2019)
20. Sharf, A., Huang, H., Liang, C., Zhang, J., Chen, B., Gong, M.: Mobility-trees for indoor scenes manipulation. Computer Graphics Forum **33**(1), 2–14 (2014)
21. Shridhar, M., Thomason, J., Gordon, D., Bisk, Y., Han, W., Mottaghi, R., Zettlemoyer, L., Fox, D.: Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10740–10749 (2020)
22. Srivastava, S., Li, C., Lingelbach, M., Martín-Martín, R., Xia, F., Vainio, K.E., Lian, Z., Gokmen, C., Buch, S., Liu, K., et al.: BEHAVIOR: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In: Conference on Robot Learning, pp. 477–490, PMLR (2022)
23. Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., Savva, M., Batra, D.: Habitat 2.0: Training home assistants to rearrange their habitat. Advances in Neural Information Processing Systems (2021)
24. Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M.J., Laptev, I., Schmid, C.: Learning from synthetic humans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 109–117 (2017)
25. Waechter, M., Moehrle, N., Goesele, M.: Let there be color! — Large-scale texturing of 3D reconstructions. In: Proceedings of the European Conference on Computer Vision, Springer (2014)
26. Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S., Guibas, L.J.: Normalized object coordinate space for category-level 6D object pose and size estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2642–2651 (2019)
27. Wang, J., Yuille, A.L.: Semantic part segmentation using compositional model combining shape and appearance. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1788–1797 (2015)
28. Wang, P., Shen, X., Lin, Z., Cohen, S., Price, B., Yuille, A.L.: Joint object and part segmentation using deep learned potentials. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1573–1581 (2015)
29. Wang, X., Zhou, B., Fang, H., Chen, X., Zhao, Q., Xu, K.: Learning to group and label fine-grained shape components. ACM Transactions on Graphics (TOG) **37**(6), 1–14 (2018)
30. Wang, X., Zhou, B., Shi, Y., Chen, X., Zhao, Q., Xu, K.: Shape2Motion: Joint analysis of motion parts and attributes from 3D shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8876–8884 (2019)
31. Weihs, L., Deitke, M., Kembhavi, A., Mottaghi, R.: Visual room rearrangement. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
32. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)

33. Xia, F., Wang, P., Chen, X., Yuille, A.L.: Joint multi-person pose estimation and semantic part segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6769–6778 (2017)
34. Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., Yi, L., Chang, A.X., Guibas, L., Su, H.: SAPIEN: A simulated part-based interactive environment. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11097–11107 (2020)
35. Xu, W., Wang, J., Yin, K., Zhou, K., Van De Panne, M., Chen, F., Guo, B.: Joint-aware manipulation of deformable models. ACM Transactions on Graphics (TOG) **28**(3), 1–9 (2009)
36. Yan, Z., Hu, R., Yan, X., Chen, L., Van Kaick, O., Zhang, H., Huang, H.: RPM-Net: recurrent prediction of motion and parts from point cloud. ACM Transactions on Graphics (TOG) **38**(6), 240 (2019)
37. Yi, L., Huang, H., Liu, D., Kalogerakis, E., Su, H., Guibas, L.: Deep part induction from articulated object pairs. ACM Transactions on Graphics (TOG) **37**(6), 209 (2019)
38. Yi, L., Kim, V.G., Ceylan, D., Shen, I.C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A scalable active framework for region annotation in 3D shape collections. ACM Transactions on Graphics (ToG) **35**(6), 1–12 (2016)
39. Zeng, V., Lee, T.E., Liang, J., Kroemer, O.: Visual identification of articulated object parts. arXiv preprint arXiv:2012.00284 (2020)
40. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ADE20K dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
41. Zhou, Q.Y., Park, J., Koltun, V.: Open3D: A modern library for 3D data processing. arXiv:1801.09847 (2018)