OP-Align: Object-level and Part-level Alignment for Self-supervised Category-level Articulated Object Pose Estimation

Yuchen Che[1], Ryo Furukawa[2], and Asako Kanezaki[1]

[1] Tokyo Institute of Technology, Tokyo, Japan
cheyuchen.titech@gmail.com, kanezaki@c.titech.ac.jp
[2] Accenture Japan Ltd, Tokyo, Japan
rfurukaward@gmail.com

Abstract. Category-level articulated object pose estimation focuses on the pose estimation of unknown articulated objects within known categories. Despite its significance, this task remains challenging due to the varying shapes and poses of objects, expensive dataset annotation costs, and complex real-world environments. In this paper, we propose a novel self-supervised approach that leverages a single-frame point cloud to solve this task. Our model consistently generates reconstruction with a canonical pose and joint state for the entire input object, and it estimates object-level poses that reduce overall pose variance and part-level poses that align each part of the input with its corresponding part of the reconstruction. Experimental results demonstrate that our approach significantly outperforms previous self-supervised methods and is comparable to the state-of-the-art supervised methods. To assess the performance of our model in real-world scenarios, we also introduce a new real-world articulated object benchmark dataset[3].

Keywords: 6DOF object pose estimation · Dataset creation · Unsupervised learning

1 Introduction

Articulated objects, comprising multiple parts connected by revolute or prismatic joints with varying joint states (rotational angle of a revolute joint or translation length of a prismatic joint), commonly exist in the real world. The interactions between humans and these objects give rise to numerous practical applications, such as robot manipulations and automation in industrial processes [5, 25]. Therefore, pose estimation for these objects has become a crucial problem in computer vision. We focus on accomplishing the category-level articulated object pose estimation through a self-supervised approach. Our objective is to use a point cloud of unknown articulated objects within known categories obtained from a single-frame RGB-D image segmented by detection models such

[3] Code and dataset are released at https://github.com/YC-Che/OP-Align.

Method	w/o Pose Supervision	w/o Shape Supervision	Single Frame	Real-time Inference
PartMobility [35]	✓	✓		
UPPD [16]	✓		✓	✓
EAP [24]	✓	✓	✓	
Ours	✓	✓	✓	✓

Table 1: Overview of works on self-supervised category-level articulated object pose estimation.

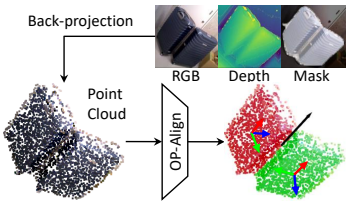

Fig. 1: Illustration of the articulated object pose estimation.

as Mask-RCNN [9] as input. Then, we infer each part's pose and segmentation, each joint's direction and pivot, as illustrated in Fig. 1. We aim to achieve this *without utilizing pose and shape annotations* during training. Due to the varying shapes, poses, and complex real-world environments, this task is ill-posed and remains challenging.

Many works have focused on solving the aforementioned task under simpler problem settings. Unsupervised Pose-aware Part Decomposition (UPPD) [16] utilizes object shape annotations as a substitute for pose annotations. PartMobility [35] utilizes multiple-frame point clouds of the same object under different joint states. However, these methods still face limitations when confronted with scenarios where shape information is unavailable or when dealing with single-frame data. To the best of our knowledge, Equi-Articulated-Pose (EAP) [24] is the only work that has tackled this task with single-frame point cloud as input and without shape or pose annotations on a synthetic dataset. EAP guides the network to learn part-by-part reconstruction of the input shapes by combining disentangled information, such as canonical part shapes, object structure, and part-level poses, in a self-supervised manner. To achieve such disentanglement, EAP extracts part-level SE(3)-equivariant shape feature of a local region, instead of object-level SE(3)-equivariant one, from an input and part-level poses. Since part-level poses are not given in nature, EAP requires iterative updates of such poses. It also uses an inner iterative operation, Slot-Attention [26], for segmenting parts. These iterative operations sacrifice inference speed.

We propose Object-level and Part-level Alignment (OP-Align), a novel self-supervised approach that learns object-level alignment, part-level alignment, and canonical reconstruction of the entire object rather than the part-by-part reconstructions. The core idea is that part segmentation and part-level pose estimation should be done for objects with low object-level pose variance. Based on this idea, we reconsider the order of the process of the part-by-part reconstruction approach (EAP) and propose a new learning strategy. In our approach, the network first generates a reconstruction that maintains the canonical pose and joint state for the entire input and aligns the input with the reconstruction at the object-level to reduce the overall pose variance. Then, the network segments parts followed by aligning each part of the object-level aligned input and the corresponding part of the reconstruction by simulating joint movement. Our ap-

proach does not employ iterative operation, thus achieving real-time inference speed. A comparison with previous works is presented in Tab. 1.

We compare OP-Align with other methods on a synthetic dataset. To further test OP-Align's performance, we generate a real-world RGB-D dataset with multiple categories of articulated objects. Experimental results demonstrate that our approach achieves state-of-the-art performance with other self-supervised methods and comparable performance with other *supervised* methods on the synthetic dataset and the real-world dataset while achieving real-time inference speed.

Our contributions are summarized as follows:

1. We propose a new model designed for category-level articulated object pose estimation in a self-supervised manner, which requires no of pose or shape annotations.
2. We generate a new real-world RGB-D dataset for the category-level articulated object pose estimation.
3. We conduct experiments on a synthetic dataset and our real-world dataset. Our model achieves comparable performance with the state-of-the-art supervised methods and significantly outperforms previous self-supervised methods while achieving real-time inference speed.

2 Related Works

Category-level rigid object pose estimation: This task focuses on predicting an unknown rigid object's pose from images. NOCS [39] predicts the per-pixel coordinates in canonical space from RGB-D images. Several methods [12, 13, 37] further employ CAD models from ShapeNet dataset [2] to generate shape templates and use iterative closest point (ICP) [34] for matching the pose. Commonly used backbone for this task is 3D graph convolution network (3DGCN) [22] and PointNet++ [33]. These methods require expensive large-scale dataset annotation. Some approaches attempt to accomplish this task in a self-supervised manner. With the CAD model available, several methods [36, 38] render the predicted pose with the CAD model as a synthetic image and compare it with the input image. Some methods focus on the multi-view RGB images provided cases [11, 19]. Especially, SE(3)-eSCOPE [21] achieved this task with single-view input and without pose annotations or CAD models. They use the SE(3)-equivariant backbone, Equivariant Point Net (EPN) [3], to simultaneously conduct SE(3)-invariant shape reconstruction as a reference frame, and predict the SE(3)-equivariant pose transformation which sends input to the reconstruction.

Category-level articulated object pose estimation: This task focuses on predicting part-level pose, part-level segmentation, and joint information for unknown objects within known categories. Previous methods [1, 14, 20, 41] try to solve this task with RGB-D image or video input by directly estimating the part-level pose. Some methods [18, 29, 31] transfer the task into a movable shape reconstruction task with neural implicit representation [27, 30] and predict the pose indirectly. Some methods [8, 15] parameterize the joint movement with active interaction with articulated objects. To reduce the segmentation cost, [23]

uses semantic segmentation annotation and transfers it into part segmentation to conduct semi-supervised learning. However, similar to the rigid object pose estimation, the cost of the dataset annotation limits the application of these methods. To solve this task in a self-supervised manner, UPPD [16] utilizes the annotation of object shape instead of the annotation of object pose. Some methods [10, 35] used multi-view observation with the same object in different joint states to predict the joint movement. EAP [24] solved such a task with a single-frame point cloud input and without shape or pose annotation. EAP repeats the process of segmenting each part, reconstructing the per-part SE(3)-invariant shape, and predicting the per-part pose multiple times to gain a refined pose estimation. However, directly segmenting parts for inputs with different poses and shapes is challenging, often resulting in poor accuracy, and the inference speed is unsuitable for real-time applications.

Articulated Object Dataset: Synthetic datasets of articulated objects such as Shape2Motion, SAPIEN, and PartNet [28, 40, 43] are commonly used in the articulated object pose estimation. Compared to RGB-D images captured from the real world, these datasets lack the consideration of complicated real-world environments. HOI4D [25] collects multiple articulated and rigid object mesh data and RGB-D images in human-object interaction. However, due to the mismatch between the depth and RGB channels, a non-negligible amount of noise is present in their ground-truth annotation of part segmentation based solely on the RGB channels.

3 Method

Category-level articulated object pose estimation can be defined as follows. Given a point cloud $\mathbf{X} \in \mathbb{R}^{3\times N}$ of an articulated object consisting of P parts, we assign each point to a part, predict the rotation and translation for each part, and provide the pivot and the direction for each joint. To solve this problem, our model predicts each point's segmentation probability $\mathbf{W} \in \mathbb{R}^{P\times N}$, each joint's pivot and direction $\{\mathbf{c}_{[i]} \in \mathbb{R}^3, \mathbf{d}_{[i]} \in \mathbb{R}^3 \mid i \in \{1, 2, \ldots, J\}\}$, and the rotation and the translation for each part $\{\mathbf{R}_{[i]} \in \mathrm{SO}(3), \mathbf{t}_{[i]} \in \mathbb{R}^3 \mid i \in \{1, 2, \ldots, P\}\}$. During training, we assume that the number of parts P and the type of joints (revolute or prismatic) are given. Specifically, OP-Align assumes that each joint connects two independent parts, resulting in $J = P - 1$ joints, which cover most of the articulated object categories found in daily environments.

The pipeline of OP-Align is shown in Fig. 2. At the object-level phase, OP-Align initially employs Efficient SE(3)-equivariant Point Net (E2PN) [44] for object-level pose selection from a discretization of the SE(3) group, and generate canonical reconstruction. At the part-level phase, two PointNets (PNs) [32] with shared weights perform part segmentation and joint parameters estimation separately for the input aligned with object-level pose and the canonical reconstruction. The obtained joint parameters generate the part-level alignment between the input and the canonical reconstruction, aligning each part of the

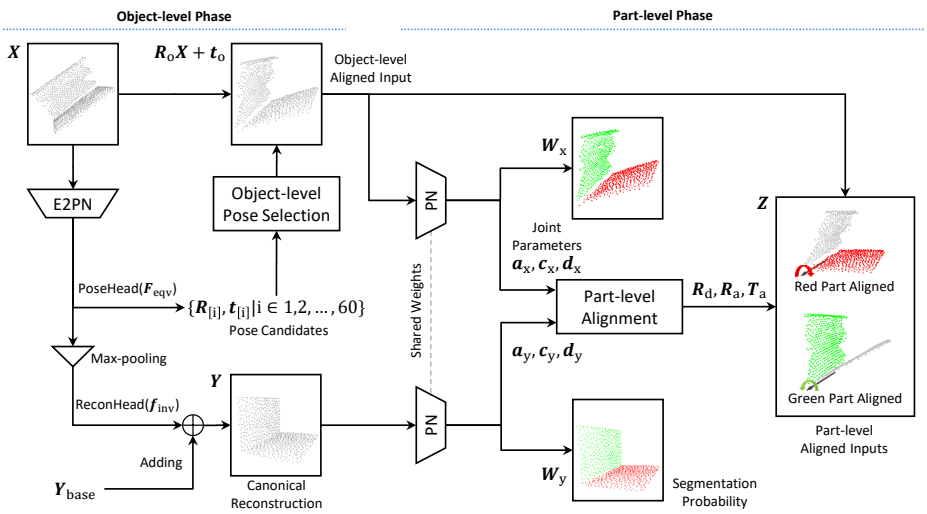

Fig. 2: Pipeline of OP-Align. At the object-level phase, for the input point cloud $\mathbf{X}$, we use the E2PN [44] backbone to predict and select object-level pose $\mathbf{R}_o, \mathbf{t}_o$ from pose candidates, and generate the canonical reconstruction $\mathbf{Y}$ by adding a learnable parameter called category-common base shape $\mathbf{Y}_{base}$. At the part-level phase, two PointNets [32] with shared weights predict the part segmentation probability $\mathbf{W}_x, \mathbf{W}_y$, joint states $\mathbf{a}_x, \mathbf{a}_y$, joint pivots $\mathbf{c}_x, \mathbf{c}_y$, and joint directions $\mathbf{d}_x, \mathbf{d}_y$ for object-level aligned input $\mathbf{R}_o\mathbf{X}+\mathbf{t}_o$ and reconstruction $\mathbf{Y}$, to generate part-level alignment $\mathbf{R}_d, \mathbf{R}_a, \mathbf{T}_a$ that aligns each part of $\mathbf{X}$ to the corresponding part of $\mathbf{Y}$ as part-level aligned inputs $\mathbf{Z}$.

input with its corresponding part of the reconstruction by simulating the joint movement.

In Section 3.1, we will introduce the concept of object-level and part-level alignment and the required weighted point cloud distance for training. Then we will introduce the object-level phase and part-level phase of our model in Section 3.2 and Section 3.3.

Notice in this section, for a rank n tensor A, we denote the $(i_1, i_2, \ldots, i_n)$-element (a rank 0-tensor) as $A_{[i_1,i_2,\ldots,i_n]}$. Moreover, we use NumPy [7] like notation to extract a tensor from A (but each index starts from 1). For example, $A_{[i_1]}$ denote the i_1-th rank $(n-1)$ tensor along the first axis and $A_{[:,i_2]}$ denote the i_2-th rank $(n-1)$ tensor along the second axis.

3.1 Preliminaries

Expansion from rigid objects to articulated objects To solve the rigid object pose estimation in a self-supervised manner, SE(3)-eSCOPE [21] utilizes a SE(3)-equivariant backbone to disentangle shape and pose by generating an SE(3)-invariant shape reconstruction and selecting SE(3)-equivariant pose from candidates in a discretization of the SE(3) group for aligning the reconstruction and input. They observed that poses of SE(3)-invariant reconstructions for ob-

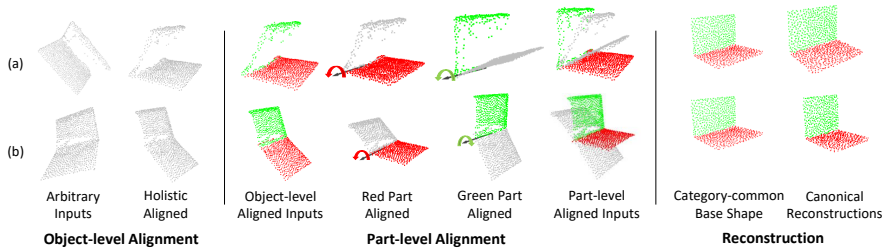

Fig. 3: Illustration of the object-level alignment, part-level alignment, and the reconstruction of two inputs (a) and (b). Object-level alignment aligns the inputs with the canonical reconstructions holistically. Part-level alignment simulates joint movement to align each part. The category-common base shape remains consistent for all inputs, and the canonical reconstruction further fits the shape details of each input.

jects in the same category are often consistently aligned. However, for articulated objects, each part's pose is also influenced by joint movement. This complexity renders the reconstruction generated by the SE(3)-eSCOPE unable to maintain consistent poses for all the parts.

To extend such an approach to articulated objects, as depicted in Fig. 3, we use object-level alignment to reduce the overall pose variance, part-level alignment to simulate joint movement and align each part, and generate reconstruction with canonical pose and joint state for any objects. Specifically, For object-level alignment, we use a similar strategy with SE(3)-eSCOPE, by selecting the pose generating the smallest point cloud distance between the reconstruction and input, among multiple pose candidates, in other words, anchors. For part-level alignment, we collectively align each part of the input to the corresponding part of the reconstruction by aligning joint direction and pivot, then rotate/translate the input along the joint direction, to obtain multiple part-level aligned inputs each of which is aligned only with the corresponding part of the reconstruction. It is essential to note that each part-level aligned input also leaves other parts unaligned. We use this phenomenon and calculate each point's distance between each part-level aligned input and the reconstruction to determine whether a point in each part-level aligned input belongs to the currently aligned part which guides the part segmentation learning. To stabilize the reconstruction, we add a category-common base shape as learnable parameters to represent a common shape of all the objects in the same category.

Weighted Point Cloud Distances We combine part segmentation probability with the point cloud distance between the part-level aligned inputs and the reconstruction to learn part segmentation and part alignment simultaneously. To achieve this, we use weighted point cloud distances, and later, part segmentation probability will sometimes be set as weights. A commonly used point cloud distance is the chamfer distance (CD), and we also employ the Density-awarded Chamfer Distance (DCD) [42]. Given two point clouds $\mathbf{P}$ and $\mathbf{Q}$, the single-

directional weighted CD (L1) and DCD from $\mathbf{P}$ to $\mathbf{Q}$ with the weight $\mathbf{w}$ are defined as

$$\begin{aligned} \mathrm{CD}(\mathbf{P}, \mathbf{Q}, \mathbf{w}) =& \frac{1}{|\mathbf{P}|} \sum_{n=1}^{|\mathbf{P}|} \mathbf{w}_{[n]} \min_{m \in \{1,2,\ldots,|\mathbf{Q}|\}} \left\| \mathbf{P}_{[n]} - \mathbf{Q}_{[m]} \right\|, \\ \mathrm{DCD}(\mathbf{P}, \mathbf{Q}, \mathbf{w}, \alpha) =& \frac{1}{|\mathbf{P}|} \sum_{n=1}^{|\mathbf{P}|} \mathbf{w}_{[n]} \min_{m \in \{1,2,\ldots,|\mathbf{Q}|\}} \left(1 - e^{-\alpha \left\| \mathbf{P}_{[n]} - \mathbf{Q}_{[m]} \right\|_2} \right). \end{aligned} \tag{1}$$

The sensitive distance range of DCD can be adjusted with the hyper-parameter α.

3.2 Object-level phase

In the Object-level phase, OP-Align performs object-level pose selection, following a methodology similar to SE(3)-eSCOPE [21], and generate canonical reconstruction.

By feeding the input $\mathbf{X}$, E2PN [44] backbone initially outputs the SE(3)-equivariant feature $\mathbf{F}_{\mathrm{eqv}} \in \mathbb{R}^{D \times 60}$. This feature is generated by 60 anchors representing different poses of the object. Here, 60 is the number of elements of the icosahedral rotation group, a discretization of the 3D rotation group SO(3). Then, we max pool $\mathbf{F}_{\mathrm{eqv}}$ among anchor dimension to obtain a SE(3)-invariant feature $\mathbf{f}_{\mathrm{inv}} \in \mathbb{R}^D$. We use PoseHead, consisting of multi-layer perceptron (MLP), to output per-anchor rotation and translation $\{(\mathbf{R}_{[i]}, \mathbf{t}_{[i]}) = \mathrm{PoseHead}(\mathbf{F}_{\mathrm{eqv}[:,i]}) \mid i \in \{1, 2, \ldots, 60\}\}$. To obtain the canonical reconstruction $\mathbf{Y} \in \mathbb{R}^{3 \times N}$, we also use an MLP called ReconHead and a learnable parameter $\mathbf{Y}_{\mathrm{base}}$ which represents the category-common base shape and is of the same size as $\mathbf{Y}$. The canonical reconstruction $\mathbf{Y}$ is obtained by adding the output of ReconHead and $\mathbf{Y}_{\mathrm{base}}$; $\mathbf{Y} = \mathrm{ReconHead}(\mathbf{f}_{\mathrm{inv}}) + \mathbf{Y}_{\mathrm{base}}$.

We also need to select the correct object-level pose from per-anchor rotation and translation $\{(\mathbf{R}_{[i]}, \mathbf{t}_{[i]})\}$. We calculate the single-directional CD between the input transformed by the rotation and the translation of each anchor and the reconstruction. Then we select the anchor's rotation and translation that minimize CD as an object-level pose;

$$\mathbf{R}_{\mathrm{o}},\ \mathbf{t}_{\mathrm{o}} = \operatorname*{argmin}_{i \in \{1,2,\ldots,60\}} \mathrm{CD}(\mathbf{R}_{[i]}\mathbf{X} + \mathbf{t}_{[i]}, \mathbf{Y}, \mathbf{1}), \tag{2}$$

where $\mathbf{1}$ represents the vector with all elements equal to 1. Notice that we do not expect the object-level pose $\mathbf{R}_{\mathrm{o}}$, $\mathbf{t}_{\mathrm{o}}$ obtained here to be accurate because we have not considered joint movement in this phase of the model. However, $\mathbf{R}_{\mathrm{o}}$, $\mathbf{t}_{\mathrm{o}}$ can reduce the overall pose variance for subsequent non-SE(3)-equivariant model's inputs by applying $\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}$ as object-level aligned input.

Object-level Losses We employ DCD as the object-level reconstruction loss

$$\mathcal{L}_{\mathrm{o}} = \mathrm{DCD}(\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}, \mathbf{Y}, \mathbf{1}, \alpha_{\mathrm{L}}) + \mathrm{DCD}(\mathbf{Y}, \mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}, \mathbf{1}, \alpha_{\mathrm{R}}), \tag{3}$$

where $\alpha_{\mathrm{L}} = 30$ and $\alpha_{\mathrm{R}} = 120$.

In addition, two regularization losses are introduced to make reconstructions more stable. The first one is for shape variance between category-common base shape $\mathbf{Y}_{\mathrm{base}}$ and canonical reconstruction $\mathbf{Y}$ and is defined by

$$\mathcal{L}_{\mathrm{regS}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{Y}_{[i]} - \mathbf{Y}_{\mathrm{base}[i]} \right\|_2 . \tag{4}$$

The second one is a local density regularization to ensure the reconstruction does not contain outliers and avoids sparse density in certain parts. It is defined by

$$\mathcal{L}_{\mathrm{regD}} = \frac{1}{K-1} \sum_{i=2}^{K} \mathrm{Var}(\|\mathbf{Y} - \mathrm{KNN}(\mathbf{Y}, k)\|), \tag{5}$$

where $\mathrm{KNN}(\mathbf{Y}, k)$ refers to the k-th nearest point from each point in $\mathbf{Y}$, and we set $K = 64$ in this paper.

3.3 Part-level phase

In this phase, we focus on segmenting both the object-level aligned input and the reconstruction into parts and estimating their joint parameters. By comparing the obtained joint pivots, joint directions, and joint states, we determine the relative pose transformations to align each part of the input with the corresponding part of the reconstruction.

OP-Align uses two PNs [32] with shared weights to process the object-level aligned input $\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}$ and the reconstruction $\mathbf{Y}$ separately. These two PNs output the segmentation probabilities $\mathbf{W}_{\mathrm{x}}, \mathbf{W}_{\mathrm{y}} \in \mathbb{R}^{P \times N}$, joint pivots $\mathbf{c}_{\mathrm{x}}, \mathbf{c}_{\mathrm{y}} \in \mathbb{R}^{(P-1)\times 3}$, joint directions $\mathbf{d}_{\mathrm{x}}, \mathbf{d}_{\mathrm{y}} \in \mathbb{R}^{(P-1)\times 3}$ and per-part joint states $\mathbf{a}_{\mathrm{x}}, \mathbf{a}_{\mathrm{y}} \in \mathbb{R}^{(P-1)\times 2}$ from each joint, where subscripts x and y indicate outputs from $\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}$ and $\mathbf{Y}$ respectively. Here, joint state $\mathbf{a}_*$ represents joint angles for revolute joints and translation lengths for prismatic joints, and the dimension of the second axis of $\mathbb{R}^{(P-1)\times 2}$ reflects the assumption that each joint connect two parts. We define the part-level aligned inputs $\mathbf{Z}_{[j,i]}$, $j = 1, 2, \ldots, P-1$, $i = 1, 2$, obtained by a relative transformation that aligns the i-th part connected to the j-th joint of $\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}$ with the corresponding part of $\mathbf{Y}$ by

$$\mathbf{Z}_{[j,i]} = \begin{cases} \mathbf{R}_{\mathrm{a}[j,i]}\mathbf{R}_{\mathrm{d}[j]}((\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}) - \mathbf{c}_{\mathrm{x}[j]}) + \mathbf{c}_{\mathrm{y}[j]} & \text{(revolute joint)}, \\ \mathbf{R}_{\mathrm{d}[j]}((\mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}) - \mathbf{c}_{\mathrm{x}[j]}) + \mathbf{c}_{\mathrm{y}[j]} + \mathbf{T}_{\mathrm{a}[j,i]} & \text{(prismatic joint)}. \end{cases} \tag{6}$$

Here, $\mathbf{R}_{\mathrm{d}[j]}$ is a rotation matrix of the joint direction alignment that sends the joint direction $\mathbf{d}_{\mathrm{x}[j]}$ to $\mathbf{d}_{\mathrm{y}[j]}$; $\mathbf{R}_{\mathrm{d}[j]}\mathbf{d}_{\mathrm{x}[j]} = \mathbf{d}_{\mathrm{y}[j]}$. $\mathbf{R}_{\mathrm{a}[j,i]}$ is the rotation matrix of joint state alignment, the rotation of a revolute joint with rotation angle $\mathbf{a}_{\mathrm{y}[j,i]} - \mathbf{a}_{\mathrm{x}[j,i]}$ around the axis $\mathbf{d}_{\mathrm{y}[j]}$. And $\mathbf{T}_{\mathrm{a}[j,i]}$ is the joint state alignment translation $\mathbf{d}_{\mathrm{y}[j]}(\mathbf{a}_{\mathrm{y}[j,i]} - \mathbf{a}_{\mathrm{x}[j,i]})$ which represents a translation of a prismatic joint. The illustration of such alignments are shown in Fig. 4. By applying the above equation to each part, OP-Align generates a point cloud set, part-level aligned inputs $\mathbf{Z} = \{\mathbf{Z}_{[j,i]} \mid i \in \{1, 2\}, j \in \{1, 2, \ldots, P-1\}\}$ where each part of the input $\mathbf{X}$ is aligned to the corresponding part of the reconstruction $\mathbf{Y}$.

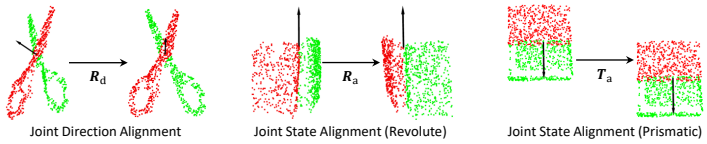

Fig. 4: Illustration of joint direction alignment $\mathbf{R}_{\mathrm{d}}$, joint state alignment $\mathbf{R}_{\mathrm{a}}$ that simulating revolute joint movement, and $\mathbf{t}_{\mathrm{a}}$ that simulating prismatic joint movement.

Corresponding part assignment Objects with more than two parts, such as `eyeglasses` or `basket`, have some shared parts, each of which is connected with multiple joints. These shared parts result in the number of part-level aligned inputs $|\ \mathbf{Z}\ |$ not necessarily being the same as the number of parts P. To correlate $\mathbf{Z}$ with part segmentation probability, we assign one part label $\sigma(j,i)$ to each pair of a joint j and a part i connected to this joint, $j = 1, 2, \ldots, P-1$, $i = 1, 2$. We require the assignment σ to satisfy two conditions: (1) for any $j \in \{1, 2, \ldots, P-1\}$ $\sigma(j,1) \neq \sigma(j,2)$ and (2) for any $p \in \{1, 2, \ldots, P\}$ there exist j and i such that $\sigma(j,i) = p$. Let σ be the assignment that minimizes the (sum of) segmentation-weighted CD calculated by $\sum_j \sum_i \frac{1}{\mathbf{b}_{[j,i]}} \mathrm{CD}(\mathbf{Z}_{[j,i]}, \mathbf{Y}, \mathbf{W}_{\mathrm{x}[\sigma(j,i)]})$ among all possible assignments satisfying (1) and (2). Here $\mathbf{b}_{[j,i]}$ denotes the number of times the part $\sigma(j,i)$ is shared. During the inference phase, we use the mean translation by linear interpretation and the mean rotation by the spherical linear interpolation (SLERP) as the shared part's pose.

Part-level Losses We employ a segmentation-weighted DCD as the part-level reconstruction loss

$$\mathcal{L}_{\mathrm{p}} = \sum_{j=1}^{P-1} \sum_{i=1}^{2} \frac{1}{\mathbf{b}_{[j,i]}} (\mathrm{DCD}(\mathbf{Z}_{[j,i]}, \mathbf{Y}, \mathbf{W}_{\mathrm{x}[\sigma(j,i)]}, \alpha_{\mathrm{L}}) + \mathrm{DCD}(\mathbf{Y}, \mathbf{Z}_{[j,i]}, \mathbf{W}_{\mathrm{y}[\sigma(j,i)]}, \alpha_{\mathrm{R}})). \tag{7}$$

We also add some regularization. We assume that the mean segmentation probability of each part exceeds the threshold β in the reconstruction and apply the segmentation regularization by

$$\mathcal{L}_{\mathrm{regW}} = \frac{1}{P} \sum_{p=1}^{P} \max \left(\beta - \frac{\sum_{i=1}^{N} \mathbf{W}_{\mathrm{y}[p,i]}}{N}, 0 \right), \tag{8}$$

where β is set to 0.05. we consider that the part-level aligned inputs of one shared part should coincide and introduce a regularization loss $\mathcal{L}_{\mathrm{regP}}$;

$$\mathcal{L}_{\mathrm{regP}} = \frac{1}{2(P-1)} \sum_{j=1}^{P-1} \sum_{i=1}^{2} \left\| \mathbf{Z}_{[j,i]} - \overline{\mathbf{Z}_{[j,i]}} \right\|_2, \tag{9}$$

where $\overline{\mathbf{Z}_{[j,i]}}$ indicates the mean shape of $\{\mathbf{Z}_{[a,b]} | \sigma(a,b) = \sigma(j,i)\}$. And since the reconstruction should have a fixed canonical joint state, we define the recon-

Table 2: Overview of the real-world dataset. The real-world dataset contains object categories with different number of parts, number of joints, and joint types.

Category	Training		Testing		Object			Detection
	Image	Instance	Image	Instance	Part	Joint(prismatic)	Joint(revolute)	
`basket`	974	4	449	2	3	0	2	SAM [17]
`drawer`	884	4	452	2	2	1	0	SAM [17]
`laptop`	740	4	412	2	2	0	1	Mask-RCNN [9]
`scissors`	922	4	421	2	2	0	1	Mask-RCNN [9]
`suitcase`	813	4	381	2	2	0	1	Mask-RCNN [9]

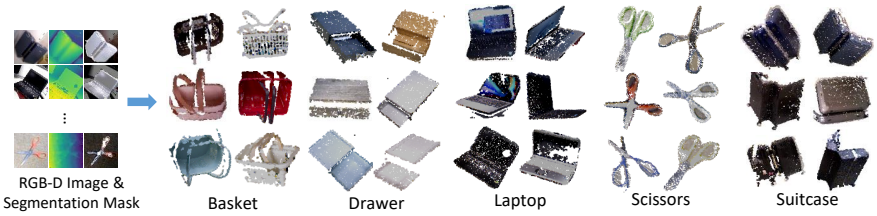

Fig. 5: Example of object point cloud in the real-world dataset. We use RGB-D images and object segmentation masks to back-project object point cloud.

struction $\mathbf{Y}$'s joint state $\mathbf{a}_{\mathrm{y}}$ as zero and apply the joint state regularization by

$$\mathcal{L}_{\mathrm{regA}} = \frac{1}{2(P-1)} \sum_{j=1}^{P-1} \sum_{i=1}^{2} \mathbf{a}_{\mathrm{y}[j,i]}^2. \tag{10}$$

Finally, since both the predicted joint pivots of the input and that of the reconstruction should be close to the object itself, we applied a regularization defined by

$$\mathcal{L}_{\mathrm{regJ}} = \mathrm{DCD}(\mathbf{c}_{\mathrm{y}[j]}, \mathbf{Y}, \mathbf{1}, \alpha_{\mathrm{L}}) + \mathrm{DCD}(\mathbf{c}_{\mathrm{x}[j]}, \mathbf{R}_{\mathrm{o}}\mathbf{X} + \mathbf{t}_{\mathrm{o}}, \mathbf{1}, \alpha_{\mathrm{R}}), \tag{11}$$

as the joint pivot regularization.

4 Real-world Dataset

To evaluate the performance of OP-Align in real-world scenarios, we introduce our novel real-world dataset. The real-world dataset contains 5 categories of articulated objects, `basket`, `laptop`, `suitcase`, `drawer`, and `scissors`, captured by ASUS Xtion RGB-D camera. For each category, we randomly select 4 objects for training and 2 objects for testing. For each object, we set 8 random joint states and captured about 30 frames of RGB-D images for each. We also generated object segmentation masks predicted with detection models such as Mask-RCNN [9] or Segment Anything Model (SAM) [17]. The object point cloud can be generated by combining the depth channel of RGB-D images with a segmentation mask. Tab. 2 and Fig. 5 show an overview of this dataset. The annotation of the real-world dataset includes each part's segmentation, rotation, and translation and each joint's pivot and direction.

5 Experiments

Datasets: We use a synthetic dataset generated by authors of EAP [24] and our real-world dataset for evaluation. The synthetic dataset contains `laptop`, `safe`, `oven`, `washer`, and `eyeglasses` categories, selected from the mesh data in HOI4D [25] and Shape2Motion [40] dataset. We follow EAP [24]'s authors to render these mesh data into the partially observed point cloud, simulating the point cloud observation from a single-view camera.
Baselines: For the synthetic dataset, we choose EAP [24] and 3DGCN [22] as self-supervised and supervised method baselines. We also report the results of a ICP algorithm, and NPCS [20] with EPN [3] backbone, which the authors of EAP [24] implemented. For the real-world dataset, we trained 3DGCN [22] and PointNet++ [33] as supervised method baselines.
Evaluation Metrics: For the synthetic dataset, we follow EAP [24] and report the mean values of segmentation IoU, part rotation error, part translation error, joint direction error, and the distance from a point to a line as joint pivot error. For the real-world dataset, we follow category-level 6D object pose estimation methods [4, 6, 39] and choose the mean average precision (mAP) with multiple thresholds. An instance's part pose is considered correct if the mean translation and rotation error of each part are both below the given thresholds. Specifically, we use thresholds $5, 10, 15$cm for translation, and $5°, 10°, 15°$ for rotation. We also use the same thresholds for joint pivot and direction. For part segmentation, we use the mean value of intersection over union (IoU) of each part and thresholds of $75\%, 50\%$ as metrics.
Evaluation Strategies: Because OP-Align is a self-supervised model, it only predicts the relative poses of the input and the reconstruction instead of the poses defined by humans. Therefore, to evaluate our model's performance, we need to determine the poses of the reconstruction parts. To achieve this, we follow EAP [24] and utilize ground truth labels from the training set. In preparation, for each training data and each part, a relative pose between the reconstruction and the input is obtained through Equation 6 by using a trained model, which, in combination with the ground truth pose, derives an estimated pose of each part of the reconstruction. We use these estimated poses to determine one common pose for each part of the reconstruction via a RANSAC-based method. For evaluation, we use the common pose as the pose of each part of the reconstruction. See supplement material for more details. We also note that for symmetric object categories such as `basket`, `laptop`, `scissors` and `suitcase`, the part segmentation is easily replaced with each other. For each object, among all possible permutations of indices of segmentation labels, we choose the permutation with the largest mean IoU over parts. The poses of parts are also permuted according to the chosen permutation.
Training Settings: We trained a model for each category for 20,000 iterations with a batch size of 24. We used the Adam optimizer with a learning rate of 0.0001 and halved the learning rate every 5,000 iterations. The total loss is defined as $\lambda_{\text{o}}\mathcal{L}_{\text{o}} + \lambda_{\text{p}}\mathcal{L}_{\text{p}} + \lambda_{\text{regS}}\mathcal{L}_{\text{regS}} + \lambda_{\text{regD}}\mathcal{L}_{\text{regD}} + \lambda_{\text{regW}}\mathcal{L}_{\text{regW}} + \lambda_{\text{regP}}\mathcal{L}_{\text{regP}} + \lambda_{\text{regA}}\mathcal{L}_{\text{regA}} + \lambda_{\text{regJ}}\mathcal{L}_{\text{regJ}}$, where $(\lambda_{\text{o}}, \lambda_{\text{p}}, \lambda_{\text{regS}}, \lambda_{\text{regD}}, \lambda_{\text{regW}}, \lambda_{\text{regP}}, \lambda_{\text{regA}}, \lambda_{\text{regJ}}) =$

Table 3: The mean metrics on partially observed point cloud from the synthetic dataset. *Supervision* refers to the annotations used in training.

Method	Supervision			Segmentation	Rotation	Translation	Pivot	Direction	Memory	Speed
	Pose	Segmentation	Joint	IoU↑	(degree)↓	↓	↓	(degree)↓	(GB)↓	(FPS)↑
3DGCN [22]	✓	✓	✓	**94.05**	11.61	0.093	**0.084**	9.78	-	-
NPCS-EPN [20]	✓	✓	✓	-	11.05	**0.080**	0.147	15.20	-	-
ICP		✓		66.45	44.12	0.242	-	-	-	-
EAP [24]				68.46	10.44	0.121	0.162	23.09	9.23	<1
Ours				80.70	**8.10**	0.129	0.110	**6.63**	**2.31**	**41**

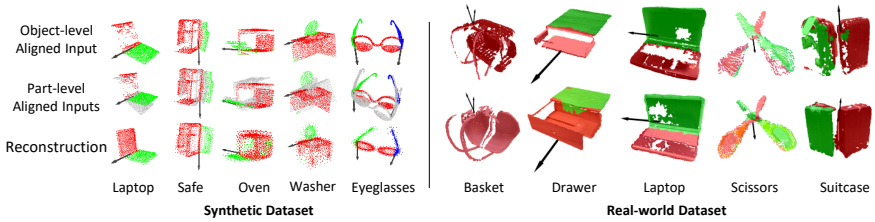

Fig. 6: Visualization of object-level aligned inputs, part-level aligned inputs, and reconstructions of OP-Align on the synthetic dataset (left) and two testing instances on the real-world dataset in each category (right). Segmentation is indicated by color, and joints are indicated by black arrow.

$(10, 10, 100, 10, 10, 10, 10, 10)$. We randomly sample 1024 points without RGB information from each object as input.

5.1 Results on the Synthetic Dataset

We compare the performance of OP-Align on the partially observed point cloud from the synthetic dataset with other methods. As the results in Tab. 3, OP-Align exceeds other self-supervised methods by a large margin on multiple metrics. These results show that OP-Align can provide accurate joint and part pose prediction along with part segmentation. The visualization shown in Fig. 6 (left) demonstrates that object-level alignment can align the input with reconstruction holistically, and part-level alignment can align each part of the input with the corresponding part of the reconstruction. Also, thanks to the object-level alignment for reducing the pose variance, our method achieved higher part segmentation performance when compared with EAP [24]. However, the part segmentation performance still has room for improvement. Our assumption is that supervised 3DGCN [22] can directly learn segmentation with the geometric feature from the point cloud, while OP-Align leverages the difference of point distance between each part-level aligned input and the reconstruction for the indirect learning of segmentation probability with $\mathcal{L}_{\text{p}}$. Especially in the region close to the joint, where points of part-level aligned inputs easily overlap, the point distance had no significant difference between each part-level aligned input, resulting in sub-optimal segmentation performance. We also compared our model in terms of

Table 4: The comparison of mAP metrics on the real-world dataset. *Supervision* refers to the annotations used in training.

Method	Supervision			Segmentation↑		Joint↑			Part↑		
	Pose	Segmentation	Joint	IoU75%	IoU50%	5°5cm	10°10cm	15°15cm	5°5cm	10°10cm	15°15cm
3DGCN [22]	✓	✓	✓	**83.31**	**95.83**	**47.51**	**85.79**	**94.59**	13.07	**46.77**	**68.66**
PointNet++ [33]	✓	✓	✓	19.83	42.20	21.06	57.38	75.56	4.47	23.25	39.82
Ours				23.79	50.42	12.57	63.59	74.04	**14.79**	46.09	59.76

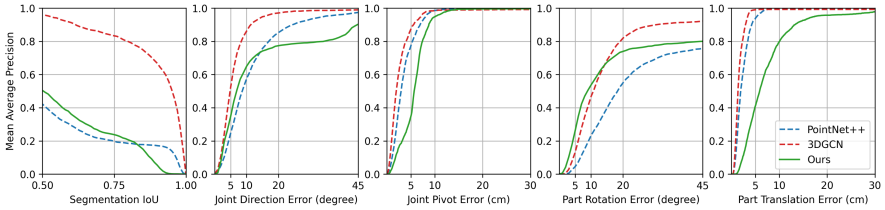

Fig. 7: The comparison of mAP metrics on the real-world dataset.

inference speed and GPU memory with EAP [24]. OP-Align utilizes less GPU memory and achieves faster inference speed.

5.2 Results on the Real-world Dataset

We conduct self-supervised training for OP-Align and compared the result with supervised 3DGCN [22] and PointNet++ [33] on the real-world dataset. The results are shown in Tab. 4 and Fig. 7 and the visualization is shown in Fig. 6 (right). OP-Align achieves results better than or comparable to PointNet++ [33] on all the metrics, and results comparable to 3DGCN [22] on part metrics, even without any annotations. However, similar to the results on the synthetic dataset, part segmentation learning with $\mathcal{L}_{\text{p}}$ requires accurate point distance between part-level aligned inputs and the reconstruction, which is extremely challenging in real-world environments where outliers and missing points commonly exist. We also notice that our model still lags behind supervised methods in terms of the joint pivot and part translation metrics, as shown in `laptop` and `suitcase` visualization in Fig. 6. This phenomenon may be because the predicted joint pivots by our model, while capable of achieving part-level alignment, may not necessarily overlap with the actual joint pivots in reality. This also affects the performance of part translation based on joint movement.

5.3 Ablation Studies

We conduct four different ablation experiments on the real-world dataset, related to the shape variance regularization $\mathcal{L}_{\text{regS}}$, the reconstruction density regularization $\mathcal{L}_{\text{regD}}$, the segmentation regularization $\mathcal{L}_{\text{regW}}$, and the joint pivot regularization $\mathcal{L}_{\text{regJ}}$, as shown in Tab. 5. Examples of the reconstructions of objects in the real-world dataset `laptop` category are shown in Fig. 8. As the

	$\mathcal{L}_{\text{regS}}$	$\mathcal{L}_{\text{regD}}$	$\mathcal{L}_{\text{regW}}$	$\mathcal{L}_{\text{regJ}}$	Segmentation↑ 50%	Joint↑ 15°15cm	Part↑ 15°15cm
(a)		✓	✓	✓	**51.87**	42.79	<u>35.39</u>
(b)	✓		✓	✓	50.36	<u>51.52</u>	32.15
(c)	✓	✓		✓	39.73	32.49	27.51
(d)	✓	✓	✓		47.52	36.27	20.49
Full	✓	✓	✓	✓	<u>50.42</u>	**74.04**	**59.76**

Table 5: Results of ablation studies.

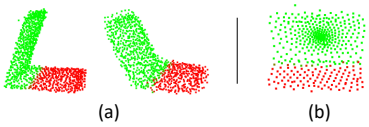

Fig. 8: Reconstruction examples of ablation model (a) and (b).

reconstructions and performance of ablation model (a) show, without $\mathcal{L}_{\text{regS}}$, the reconstructions' joint state is not fixed, which results in a huge performance drop at metrics of joint and part prediction. For ablation model (b), without $\mathcal{L}_{\text{regD}}$, reconstruction's points are concentrated into a small region, which affects the overall performance of our model. For ablation model (c), without $\mathcal{L}_{\text{regW}}$, some objects are regarded as single-part objects, and we fail to generate valid joint parameters. Finally for ablation model (d), without $\mathcal{L}_{\text{regJ}}$, joint pivot may be placed outside of the object, resulting in poor performance on both joint and part pose metrics.

6 Failure Cases and Limitations

Failure Cases: We found that OP-Align fails for objects belonging to categories where some parts comprise only a small fraction of the entire object and their movement does not significantly affect the overall shape. For `basket` category, as shown in Fig. 6, the handle parts account for 16.9% of the entire object (median in the testing set) and the movement of these parts results in small changes to the overall shape. This means that even without part-level alignment, our canonical reconstruction is sufficiently close to the overall object. This also leads to our model's inability to correctly segment parts and predict joint movements.
Limitations: OP-Align requires the number of parts and joint types as known information, which limits its ability to learn from objects in categories with unknown joint types or variable numbers of joints and parts.

7 Conclusion

We proposed a novel approach, OP-Align, and a new real-world dataset for the self-supervised category-level articulated object pose estimation. Our approach achieves state-of-the-art performance among self-supervised methods and comparable performance to previous supervised methods, yet with real-time inference speed. Our future plan is to design a self-supervised universal pose estimation model, which can be trained with inner-category data and automatically detect the number of parts, number of joints, and joint type.

Acknowledgements: We thank Ryutaro Yamauchi and Tatsushi Matsubayashi from ALBERT Inc. (now Accenture Japan Ltd.) for their insightful suggestions and support. This work was supported by JST FOREST Program, Grant Number JPMJFR206H.

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. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
3. Chen, H., Liu, S., Chen, W., Li, H., Hill, R.: Equivariant point network for 3d point cloud analysis. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 14514–14523 (2021)
4. Chen, W., Jia, X., Chang, H.J., Duan, J., Shen, L., Leonardis, A.: Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1581–1590 (2021)
5. Chu, R., Liu, Z., Ye, X., Tan, X., Qi, X., Fu, C.W., Jia, J.: Command-driven articulated object understanding and manipulation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8813–8823 (2023)
6. Di, Y., Zhang, R., Lou, Z., Manhardt, F., Ji, X., Navab, N., Tombari, F.: Gpv-pose: Category-level object pose estimation via geometry-guided point-wise voting. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 6781–6791 (2022)
7. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E.: Array programming with NumPy. Nature **585**(7825), 357–362 (Sep 2020). https://doi.org/10.1038/s41586-020-2649-2, https://doi.org/10.1038/s41586-020-2649-2
8. Hausman, K., Niekum, S., Osentoski, S., Sukhatme, G.S.: Active articulation model estimation through interactive perception. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA). pp. 3305–3312. IEEE (2015)
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Int. Conf. Comput. Vis. pp. 2961–2969 (2017)
10. Huang, J., Wang, H., Birdal, T., Sung, M., Arrigoni, F., Hu, S.M., Guibas, L.J.: Multibodysync: Multi-body segmentation and motion estimation via 3d scan synchronization. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7108–7118 (2021)
11. Insafutdinov, E., Dosovitskiy, A.: Unsupervised learning of shape and pose with differentiable point clouds. Adv. Neural Inform. Process. Syst. **31** (2018)
12. Irshad, M.Z., Kollar, T., Laskey, M., Stone, K., Kira, Z.: Centersnap: Single-shot multi-object 3d shape reconstruction and categorical 6d pose and size estimation. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA). pp. 10632–10640. IEEE (2022)
13. Irshad, M.Z., Zakharov, S., Ambrus, R., Kollar, T., Kira, Z., Gaidon, A.: Shapo: Implicit representations for multi-object shape, appearance, and pose optimization. In: Eur. Conf. Comput. Vis. pp. 275–292. Springer (2022)
14. Jiang, H., Mao, Y., Savva, M., Chang, A.X.: Opd: Single-view 3d openable part detection. In: Eur. Conf. Comput. Vis. pp. 410–426. Springer (2022)
15. Jiang, Z., Hsu, C.C., Zhu, Y.: Ditto: Building digital twins of articulated objects from interaction. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5616–5626 (2022)

16. Kawana, Y., Mukuta, Y., Harada, T.: Unsupervised pose-aware part decomposition for man-made articulated objects. In: Eur. Conf. Comput. Vis. pp. 558–575. Springer (2022)
17. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. arXiv preprint arXiv:2304.02643 (2023)
18. Lei, J., Daniilidis, K.: Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 6624–6634 (2022)
19. Li, C., Bai, J., Hager, G.D.: A unified framework for multi-view multi-class object pose estimation. In: Eur. Conf. Comput. Vis. pp. 254–269 (2018)
20. Li, X., Wang, H., Yi, L., Guibas, L.J., Abbott, A.L., Song, S.: Category-level articulated object pose estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3706–3715 (2020)
21. Li, X., Weng, Y., Yi, L., Guibas, L.J., Abbott, A., Song, S., Wang, H.: Leveraging se(3) equivariance for self-supervised category-level object pose estimation from point clouds. Adv. Neural Inform. Process. Syst. **34**, 15370–15381 (2021)
22. Lin, Z.H., Huang, S.Y., Wang, Y.C.F.: Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1800–1809 (2020)
23. Liu, G., Sun, Q., Huang, H., Ma, C., Guo, Y., Yi, L., Huang, H., Hu, R.: Semi-weakly supervised object kinematic motion prediction. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21726–21735 (2023)
24. Liu, X., Zhang, J., Hu, R., Huang, H., Wang, H., Yi, L.: Self-supervised category-level articulated object pose estimation with part-level se (3) equivariance. In: Int. Conf. Learn. Represent. (2023)
25. Liu, Y., Liu, Y., Jiang, C., Lyu, K., Wan, W., Shen, H., Liang, B., Fu, Z., Wang, H., Yi, L.: Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21013–21022 (2022)
26. Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., Kipf, T.: Object-centric learning with slot attention. Adv. Neural Inform. Process. Syst. **33**, 11525–11538 (2020)
27. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4460–4470 (2019)
28. 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: IEEE Conf. Comput. Vis. Pattern Recog. pp. 909–918 (2019)
29. Mu, J., Qiu, W., Kortylewski, A., Yuille, A., Vasconcelos, N., Wang, X.: A-sdf: Learning disentangled signed distance functions for articulated shape representation. In: Int. Conf. Comput. Vis. pp. 13001–13011 (2021)
30. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 165–174 (2019)
31. Paschalidou, D., Katharopoulos, A., Geiger, A., Fidler, S.: Neural parts: Learning expressive 3d shape abstractions with invertible neural networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3204–3215 (2021)
32. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 652–660 (2017)

33. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems **30** (2017)
34. Segal, A., Haehnel, D., Thrun, S.: Generalized-icp. In: Robotics: science and systems. vol. 2, p. 435. Seattle, WA (2009)
35. Shi, Y., Cao, X., Zhou, B.: Self-supervised learning of part mobility from point cloud sequence. In: Computer Graphics Forum. vol. 40, pp. 104–116. Wiley Online Library (2021)
36. Sundermeyer, M., Marton, Z.C., Durner, M., Triebel, R.: Augmented autoencoders: Implicit 3d orientation learning for 6d object detection. Int. J. Comput. Vis. **128**, 714–729 (2020)
37. Tian, M., Ang, M.H., Lee, G.H.: Shape prior deformation for categorical 6d object pose and size estimation. In: Eur. Conf. Comput. Vis. (2020)
38. Wang, G., Manhardt, F., Shao, J., Ji, X., Navab, N., Tombari, F.: Self6d: Self-supervised monocular 6d object pose estimation. In: Eur. Conf. Comput. Vis. pp. 108–125. Springer (2020)
39. 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: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2642–2651 (2019)
40. Wang, X., Zhou, B., Shi, Y., Chen, X., Zhao, Q., Xu, K.: Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8876–8884 (2019)
41. Weng, Y., Wang, H., Zhou, Q., Qin, Y., Duan, Y., Fan, Q., Chen, B., Su, H., Guibas, L.J.: Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In: Int. Conf. Comput. Vis. pp. 13209–13218 (2021)
42. Wu, T., Pan, L., Zhang, J., Wang, T., Liu, Z., Lin, D.: Density-aware chamfer distance as a comprehensive metric for point cloud completion. arXiv preprint arXiv:2111.12702 (2021)
43. 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.J., Su, H.: SAPIEN: A simulated part-based interactive environment. In: IEEE Conf. Comput. Vis. Pattern Recog. (June 2020)
44. Zhu, M., Ghaffari, M., Clark, W.A., Peng, H.: E2pn: Efficient se (3)-equivariant point network. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1223–1232 (2023)