Supplementary Material: Unsupervised Pose-aware Part Decomposition for Man-made Articulated Objects

Yuki Kawana[1], Yusuke Mukuta[1,2], and Tatsuya Harada[1,2]

[1]The University of Tokyo [2]RIKEN

A Illustration of $\mathcal{L}_{\text{pivot}}$

Figure 1 illustrates $\mathcal{L}_{\text{pivot}}$ introduced in Section 3.3 of the main paper in 2D.

B Network architecture

The network architectures of the neural networks employed in the proposed method are depicted in Figure 2. The squircle diagram represents tensors, where the first and the second numbers inside the parentheses indicate the channel and the number of points, respectively. The squircle without the parentheses indicates the scalar value. For the split operation, $N[X, Y]$ denotes the split operation of the input tensor to N sliced tensors with X channels with Y points. For the square diagrams with square brackets, the first and the second numbers in the square brackets indicate the input and output channels, respectively. The green square diagrams indicate multiple identical subnetwork architectures. P denotes the number of points in the input point cloud to encoder E. C denotes the number of coordinate points in the input to the shape decoders $\{G_i^z\}$ and $\{G_i^c\}$, and the discriminator D.

We use the simple PointNet architecture in the author-provided code of [7] as the encoder E. We use two separate MLP networks, F_r^c and F_r^q, for predicting $\{\mathbf{r}_i^c \mid i \in \mathbb{A}^p\}$ and $\{\mathbf{q}_i^c \mid i \in \mathbb{A}^r\}$, respectively. For the normalization layer in F_r^c and F_q^c, we have experimentally found that using instance normalization [14] for F_r^c and layer normalization [1] for F_q^c achieves the best performance. For the joint state s_i, we multiply π to $\{s_i \,|\, y_i = \text{revolute}\}$. For the discriminator D, we use the architecture based on the PointNet [10] implementation in the author-provided code of [11]. The weight of each linear layer in our discriminator is normalized using spectral normalization [8] for stable training.

C Training details

We train our models per category with the same hyperparameter configuration described in Section 3.4 in the main paper for all categories. For the input, we use the point cloud with 4096 points sampled from the surface of the target

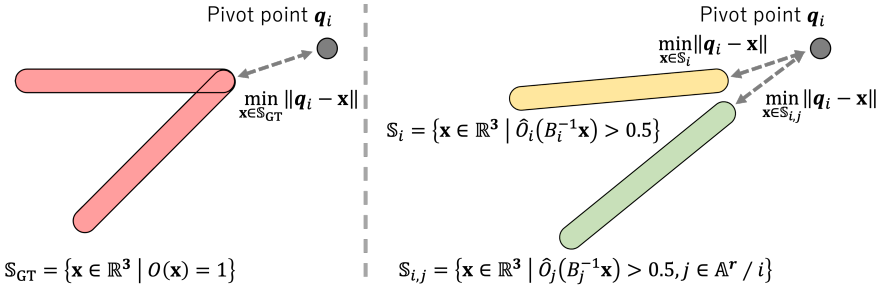

Fig. 1. Illustration of $\mathcal{L}_{\mathrm{pivot}}$ in 2D.

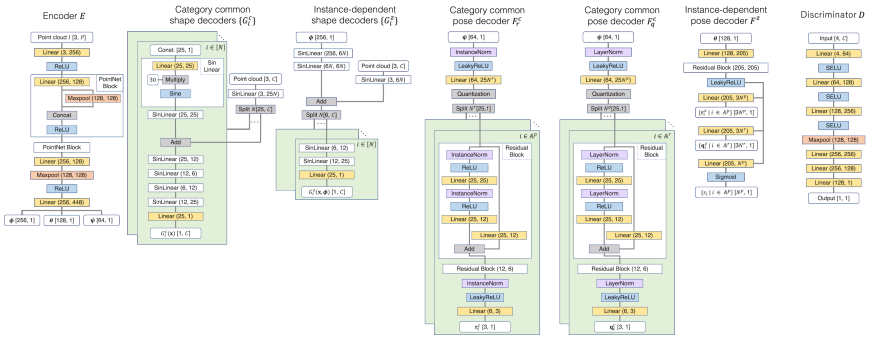

Fig. 2. The architectures of our networks.

shape during the training. Unless otherwise noted, we use the complete shape point cloud. We use a batch size of 18. We train our network in two stages following [2]: first, we train it on an implicit field of 16^3 grids and then on 32^3 grids. For the ground-truth implicit field, for each sample in a batch, we use 4096 3D coordinate points and their corresponding indicator values sampled from either 16^3 or 32^3 grids, depending on the training stage. This multi-stage training strategy on grids with different resolutions is inspired by [2]. We train our network on 16^3 grids in the first training stage. In addition, we set $\mathbf{r}_i = \mathbf{r}_i^c$ in the first stage. Then, we set $\mathbf{r}_i = \mathbf{r}_i^c + \mathbf{r}_i^s$ in the second stage. We determine the number of iterations for each stage according to the reconstruction loss and to the visualization of the reconstructed shapes on the validation data. It takes 2 to 3 days to train one model on a single NVIDIA V100 graphics card with 16 GB of GPU memory.

Model parameter initialization. We use a sine function as a nonlinear activation function and the weight initialization strategy proposed in [12] in our shape decoders, as follows:

$$w \sim \mathrm{U}\left(-\sqrt{\frac{6}{\mathrm{IN}}}, \sqrt{\frac{6}{\mathrm{IN}}}\right) \frac{1}{30} \tag{1}$$

	Drawer	Eye-glasses	Oven	Laptop	Washing machine
Training	24	35	30	73	39
Test	6	7	7	13	6
# of parts	(1 3 0)	(1 0 2)	(1 0 1)	(1 0 1)	(1 0 1)

Table 1. Number of samples per category in each data split. Each sample is augmented by transforming its part pose to generate 100 instances. Numbers in a parethnesis in the last row indicates the ground-truth number of fixed, prismatic and revolute type parts.

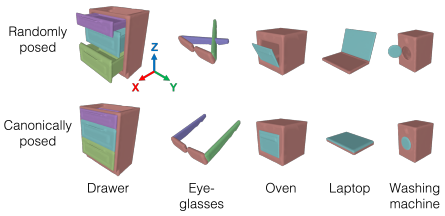

Fig. 3. Visualization of the canonically posed and randomly posed ground-truth meshes of each category. The colors correspond to the different ground-truth part labels. The colored arrows show the axis directions of the coordinate system used in this paper.

where IN is an input channel to a linear layer, U is a uniform distribution, and w is an element of the weight of a linear layer. For a linear layer that takes 3D coordinates as an input, we do not scale the weight w by $\frac{1}{30}$.

As described in the main paper, we set the number of parts used in our model as $N = 8$, which consists of one "fixed" part, three "revolute" parts, and four "prismatic" parts. Each initial joint direction $\mathbf{e}_i$ for the "revolute" and "prismatic" parts are set as follows, with z-axis as up, x-axis as forward, and y-axis as right: $+z$, $-z$ and $+y$ directions for "revolute" parts and $+x$ direction for "prismatic" parts. See Figure 3 for visual correspondence between shapes and axes.

C.1 Training of the baseline models

We use the author-provided implementations for all the baselines. We explain the additional detail of the training for the baselines below.

BSP-Net [2]. Because the models in the author-provided codes of the other part decomposition baselines (BAE-Net [3] and NSD [5]) are trained on 32^3 grids, we also trained BSP-Net on up to 32^3 grids, compared to the 64^3 grids in the original implementation. For training on the eyeglasses category, we could not successfully train the model even with different random seeds with the provided training script. After several trials, we experimentally found that scaling ground-truth indicator values by four for the first 20,000 iterations produced good initialization of the model. On the basis of this finding, we first pre-trained the model using the scaled ground-truth indicator values for 20,000 iterations for the eyeglasses category; then, we trained the model with the provided training script.

NSD [5] and Neural Parts [9]. The model defined in the author-provided code takes an RGB image as an input, which is a more challenging setting for

3D shape reasoning than 3D shape input. We replace the image encoder of the original implementation with the same PointNet-based encoder used in our approach for a fair comparison.

NPCS [6]. In the experiment described in Section 4.2 in the main paper, we modified the original implementation of NPCS to use complete shape point clouds instead of partial point clouds of the depth map as an input with training from scratch, to remove the unnecessary performance degradation caused by pose ambiguity arising from the barely visible articulated part.

D Data preparation

In this section, we describe our data preparation procedure.

D.1 Data split

We split our training and test data according to the per-category data split approach introduced in [6]. We ensure that the test split contains at least six samples per category, except for the laptop category; therefore, the average split ratio is approximately 8:2. For the laptop category, we use 11 samples in the test split to make the split ratio comparable with those of the other categories. The number of samples in each split per category is presented in Table 1.

D.2 Ground-truth implicit field generation

Following [7], we generate the ground-truth implicit field by the volumetric fusion of 100 depth images of a mesh object. For the mesh object, we sample 100 instances with randomly sampled part poses for each sample. For the pose sampling, we uniformly sample the rotation amount for each joint for the revolute joints. For the revolute joints of all categories except the eyeglasses category, we sample the rotation amount between 0° and 135°. For the eyeglasses category, we sample between 0° and 90°. For the prismatic joints of the drawer category, we sample the translation amount between 0 and the maximum amounts of the joints written in the URDF files of each sample in the SAPIEN dataset [16]. After we sample a part pose for each instance, we articulate the sample in its canonical pose (the rotation amount and translation amount were set to 0° and 0, respectively) using the sampled motion amount and ground-truth joint configuration. The canonically posed shape and the randomly posed shape of the same sample are shown in Figure 3. Finally, we normalize the size and location of the instances following [7]. Specifically, we normalize the instances with the maximum extent collected from the instances generated from the same sample.

E Part labeling procedure for evaluation and part segmentation visualization

E.1 Part labeling procedure

In this section, we explain the labeling procedure using the ground-truth part labels of the training samples to evaluate the part segmentation performance, following the same procedure used in [5, 4]. First, for each surface point sampled from the ground-truth part mesh of the instance of the training set, we determine the nearest reconstructed part and vote for the ground-truth part label of that point. Next, we assign each reconstructed part to the part label that has the highest number of votes. Finally, for each surface point sampled from the instance in the test split, we determine the nearest reconstructed part surface and assign the part label of the reconstructed part.

E.2 Part segmentation visualization

To visualize part segmentation, similar to [13], We first measure the distance between a barycentric point of a ground-truth mesh face to the surface of each part. Then we assign a mesh face the label of the part with the shortest distance to the barycentric point. Lastly, we color each face according to the obtained label.

F Additional semantic capability evaluation

F.1 Additional visualization of the part segmentation

We visualize the additional part segmentation results of the proposed approach in Figure 4. Also, we visualize the part segmentation results given various part poses in Figure 9.

F.2 Part segmentation using all the training samples

In Section 4.1 in the main paper, we show that our method works most efficiently by requiring instances with only a limited variety of poses for the initial annotations. We use canonically posed shapes, visualized in Figure 3, in the training set for the initial annotations. This section reports the evaluation setting where annotations of all training instances are available for the initial annotation, which is a favorable setting for the baselines. However, the annotation cost can be much higher in reality than in the previous setting.

The results are shown in Table 2. Even under this setting favorable for the previous works, our method performs comparably with the state-of-the-art part decomposition method BSP-Net [2] using 256 primitives. It is not surprising that using many primitives achieves fewer part segmentation errors because, even when one primitive is inconsistently assigned to the ground-truth part, the

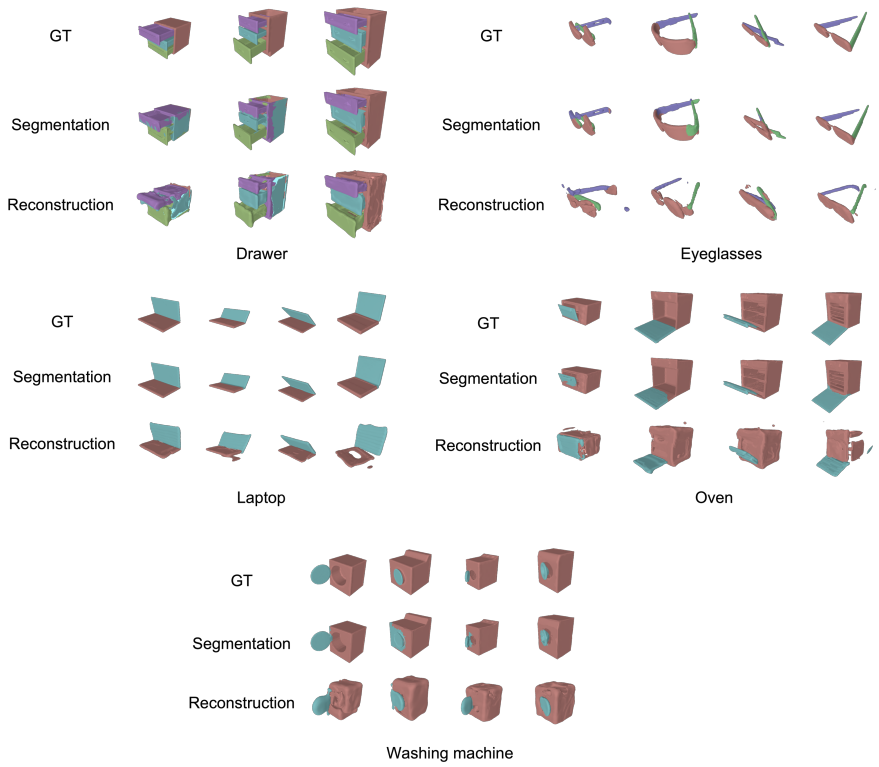

Fig. 4. Visualization of the additional part segmentation results of the proposed approach with various samples. For drawer category, the different between some GT shapes are subtle (e.g., difference in handle shapes), we pick the three samples with distinct shape difference to avoid confusion.

impact on the label IoU is smaller. This is because a smaller portion of the evaluation points becomes erroneous compared with the model using fewer parts or primitives. Note that our research focuses on representing ground-truth articulated parts with consistently the same reconstructed parts by considering the part kinematics, unlike BSP-Net and the other baselines, which can assign different sets of primitives to the same articulated parts without considering the underlying part pose. To show the effectiveness of considering the part kinematics, we show the performance drop from using all training instances to using only the canonically posed instances in the table under the heading "Difference." We can see that our approach has the second best drop with the comparable number with Neural Parts [9], yet higher part parsing performance. This shows that considering the part kinematics contributes to label efficiency by reducing the necessary initial annotation to perform well on the unsupervised part segmentation of articulated objects.

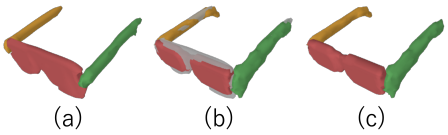

Fig. 5. Relationship between G_i^c, G_i^z and $\hat{O}$. (a) Category-common shape $\hat{O}^c = \max_i\{\sigma(G_i^c(\cdot))\}$. (b) $\max_i\{\sigma(G_i^z(\cdot))\}$ overlaid with $\hat{O}^c$. (c) Predicted shape $\hat{O}$.

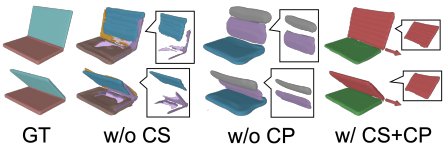

Fig. 6. Qualitative ablation of CS and CP. The arrow indicates the predicted revolute direction.

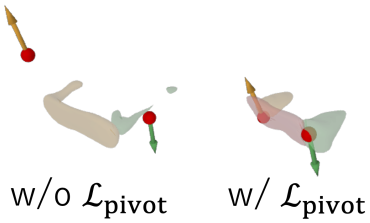

Fig. 7. Qualitative ablation on $\mathcal{L}_{\text{pivot}}$ at training step 1.3k. The red sphere shows the pivot point, and the arrow indicates the joint direction.

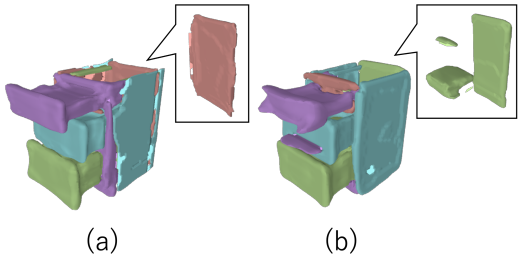

Fig. 8. A failure case of the drawer category. (a) The best-performing model. (b) The failure case. The part which reconstructs the back side is misclassified compared to (a).

F.3 Part segmentation with the aligned number of parts

In Table 3 of the main paper, we have only reported the average of the label IoU performance where we align the predefined maximum number of the parts of all the baselines to the same as ours ($N = 8$). We show the full result in In Table 3.

G Additional part pose evaluation

Because we train our model in an unsupervised fashion, through the labeling process described in Appendix E.1, the part kinematic types of the ground-truth and the assigned reconstructed part do not necessarily match. Moreover, multiple reconstructed parts may be assigned to one ground-truth part. Therefore, we choose EPE as the evaluation metric for part pose estimation due to its kinematic type agnostic property and calculation based on point correspondence between prediction and ground-truth, rather than part-level correspondence. In this section, as an additional part pose evaluation, we evaluate the accuracy of joint parameter estimation for "revolute" and "prismatic" parts.

To avoid the problem of part pose evaluation in unsupervised learning described above, we evaluate the accuracy of joint parameter estimation by considering the prediction is correct when the following three conditions are all satisfied. (1) One reconstructed part is assigned to one ground-truth dynamic

	Drawer	Eye-glasses	Oven	Laptop	Washing machine	mean (All)	mean (Canonical)	Difference (All - Canonical)	# of parts
BAE [3]	6.25	11.11	73.01	25.11	80.32	39.16	39.17	-0.01	1.42/8
BSP[2]	70.29	**74.96**	**89.40**	86.21	**95.28**	**83.23**	76.65	6.58	27.50/256
NSD [5]	38.56	44.06	74.63	74.40	89.01	64.13	63.75	0.39	10
NP [9]	60.56	64.75	85.33	86.22	74.72	74.32	74.31	**0.01**	5
Ours	**74.83**	66.25	82.06	**86.80**	95.18	81.02	**80.99**	0.04	4.16/8

Table 2. Part segmentation performance. We use all the instances in the training set to assign a label to each part as well as to the primitives. "Canonical" denotes the mean label IoU only using the canonically posed instances of the training for the label assignment. "Difference" shows the performance drop from the setting that uses all the instances in the training set to the setting that uses only the canonically posed instances. The average and the predefined maximum numbers of recovered parts or primitives are shown before and after the slash, in the last column. Our method achieves the same level of the label efficiency with Neural Parts with higher part segmentation performance.

	Drawer	Eye-glasses	Oven	Laptop	Washing machine	mean
BAE [3]	6.25*	11.11*	73.06	25.11*	80.30	39.17
BSP [2]	26.62	**71.14**	**85.19**	64.41	86.60	66.79
NSD [5]	34.07	60.06	70.09	70.12	62.97	59.46
NP [9]	61.10	65.47	77.57	62.73	86.69	70.71
Ours	**74.73**	66.18	82.07	**86.81**	**95.15**	**80.99**

Table 3. Part segmentation performance in label IoU with the aligned number of parts for all methods ($N = 8$). The starred numbers indicate the failure of part decomposition and that only one recovered part represents the entire shape.

	Drawer	Eye-glasses	Oven	Laptop	Washing machine	mean
# of assigned parts	1.0	1.0	1.0	1.0	1.0	1.0
Part type accuracy	89.50	83.25	100.0	92.14	100.0	91.46

Table 4. Part assignment evaluation. The first row shows the number of reconstructed parts assigned to the ground-truth parts, and the second row shows the accuracy of part kinematic type matches between the ground-truth and the assigned reconstructed parts for dynamic part types.

part. (2) The part kinematic type is the same between the ground-truth and the assigned reconstructed part. (3) The error of the joint parameters against the ground-truth is less a threshold. This evaluation method is more challenging than EPE because of the influence of (1) and (2) above, besides the prediction error of the joint parameters. We evaluate joint state accuracy and joint direction accuracy. Only for the revolute part, we also evaluate joint axis distance accuracy, defined as the line to line distance between the ground-truth and the predicted line segments consisting of the pivot point and the joint direction.

Figure 10 shows the evaluation results with varying error thresholds. We show the results of NPCS only as a reference; NPCS is a supervised model and assumes that the part segmentation is available during training, and the part kinematic types are also known. In contrast, our method learns both part segmentation and part kinematic type in an unsupervised fashion. Since NPCS does not estimate the pivot point, we only show the results of our method for joint axis distance accuracy. As for the joint state, we see reasonable accuracy of 70.80% for revolute parts on average when the threshold is less than 10 degrees and 79.43% when the threshold is 15 degrees. For the "prismatic" part of the

drawer, our method outperforms the NPCS when the threshold is less than 0.1. For joint direction estimation, in three out of five categories (eyeglasses, laptop, and oven), our method is comparable or outperforming NPCS. In Table 4, we also show the number of reconstructed parts assigned to the ground-truth parts and the accuracy of part kinematic type of dynamic parts matches between the ground-truth and the assigned reconstructed parts. In all categories, the model correctly assigns one part. Moreover, even without part type supervision, our model successfully predicts correct part types with high accuracy of 91.46%. Improving the unsupervised learning of joint parameters under shape supervision is an interesting research direction.

H Formulation details

The intuition behind multiplying G_i^z and G_i^c. Since a shape can exist at positions only where both G_i^z and G_i^c are large through multiplication, for each part, G_i^c defines a category-common shape, and G_i^z provides a shape that reflects input-dependent details around the category-common shape, visualized in Figure 5.

Effectiveness of the second term in $\mathcal{L}_{\text{reconstruction}}$. Optimizing the second term of $\mathcal{L}_{\text{reconstruction}}$ enables directly optimizing G_i^c rather than optimizing it through the first term. It allows to quickly learn a category-common shape, resulting in better G_i^z by G_i^c during the training. Removing the second term results in a significant drop in mean label IoU ($80.99 \rightarrow 51.78$).

The motivation of vector quantization for F^c. Directly regressing the pose parameter value is known to be difficult. Prior work [15] classifies the value into the discretized value ranges and regresses the residual within the classified range. Motivated by this, vector quantization for F^c models discrete modality of the joint parameter, and F^z regresses the residual.

Implementation details of $\mathcal{L}_{\text{pivot}}$ To further regularize the location of the pivot point during the training, we make $\mathbb{S}_{\text{GT}}$ in the first term of Eq. 5 conditional in our implementation as $\tilde{\mathbb{S}}_{\text{GT}} = \mathbb{S}_{\text{GT}} \cap \mathbb{S}_{i,\text{fixed}}$ if $\emptyset \neq \mathbb{S}_{i,\text{fixed}}$ otherwise $\tilde{\mathbb{S}}_{\text{GT}} = \mathbb{S}_{\text{GT}}$, where $\mathbb{S}_{i,\text{fixed}} = \mathbb{S}_i \cap \{\mathbf{x} \mid \hat{O}_j(\mathbf{x}) > 0.5, y_j = \text{fixed}\}$. This formulation considers the predicted fixed part in regularization. After the training proceeds, the fixed part will likely have higher occupancy values around the fixed "base" part of the target shape to which the dynamic parts attach. The above conditioning further constrains the pivot point's location to the ground truth shape around the intersection between the predicted fixed and revolute parts.

I Qualitative ablation study

In this section, we visualize some of the qualitatie results as an additional ablation study. When we remove caninical shape decoder (CS), we frequently find

unsuccessful decomposition; a single part spans two GT parts (purple), and the shape deformation accounts for the shape variation. We visualize an example in Figure 6. The boxes show the parts with their joint states set to 0. Colors indicate the part IDs. With CS, the variation is expressed by part poses with successful decomposition. Without canonical pose decoder (CP), similarily with CS, we find that the model degenerates to express shape variation by different part poses by shape deformation. During the training, we find turning off $\mathcal{L}_{\text{pivot}}$ for correct pivot point localization makes the traning difficult to converge to preferable decomposition, especially for eyeglasses category. As visualized in Figure 7, with $\mathcal{L}_{\text{pivot}}$, the pivot point locates the proper position between parts even at the early stage of the training. However, without $\mathcal{L}_{\text{pivot}}$, the pivot points are off from the reconstructed shape.

J Failure cases and limitation

Because pose-aware part decomposition without explicit supervision is a highly ill-posed task, as a limitation, our method requires manual initialization of part types and joint directions for each part to stabilize the training process, as described in Section 3.1. Although the manual initialization, part decomposition induction by pose constraints with joint parameters, and the proposed losses contribute to stabilizing the training process, different model initialization and stochastic training may result in different part decomposition results due to the unsupervised nature of the approach and the ill-posed target problem. In our experiments, we tried a few random seeds when part decomposition failed in the early stages of the traning for the models reported in Table 2 of the main paper.

We also found our drawer category is more challenging to converge well than the other categories, resulting in degenerated quantitative performance. As visualized in Figure 9, we attribute this cause to its more considerable intra-category size variation. Our canonical shape decoder learns category-specific mean part shapes. The decoder also models canonical part locations for the prismatic part, unlike a revolute part location modeled by its pivot point. Thus, deviating largely from the mean part shapes and large part location difference caused by the size difference weakens the effectiveness of the canonical shape decoder, leading to the semantically less consistent part decomposition. As shown in Figure 9 for the drawer category, the side of the object shapes is misclassified for the first and second objects, compared to the third objects. We also visualize another failure case in Figure 8. The model successfully learns the reasonable part shapes, yet the decomposition is less consistent, as emphasized in the boxes. Due to the small number of parts used by our model, misclassifying a single part could drop the segmentation performance significantly ($74.83 \rightarrow 60.97$). Note that even such a model performs comparably with the leading primitive-based part decomposition method [9] when the number of parts is aligned, as shown in Table 3. One possible extension to tackle this problem can be learning to model size in addition to the shape and pose for each part and letting the canonical shape decoder learn part shape in normalized space in terms of part pose and size.

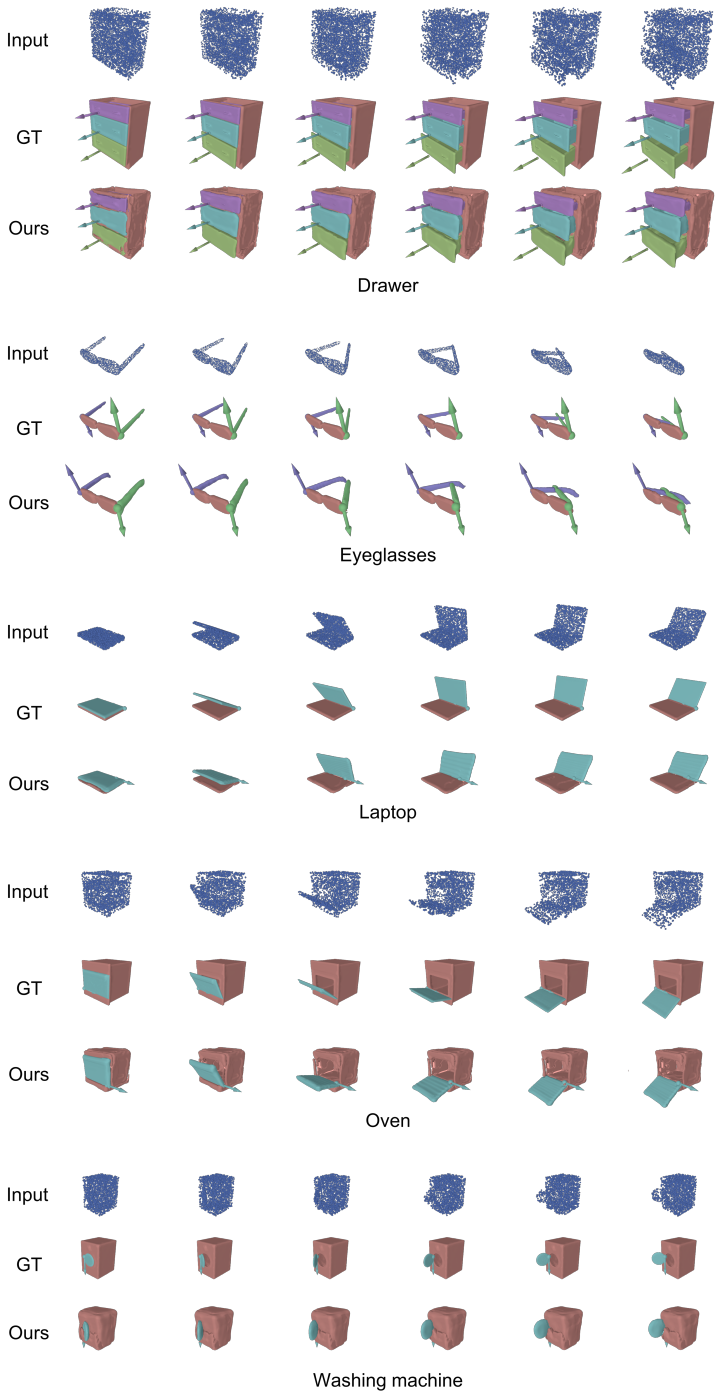

Fig. 9. Visualization of the part segmentation results given input shapes with various part poses. Arrows in the figure indicate the ground-truth or predicted joint directions.

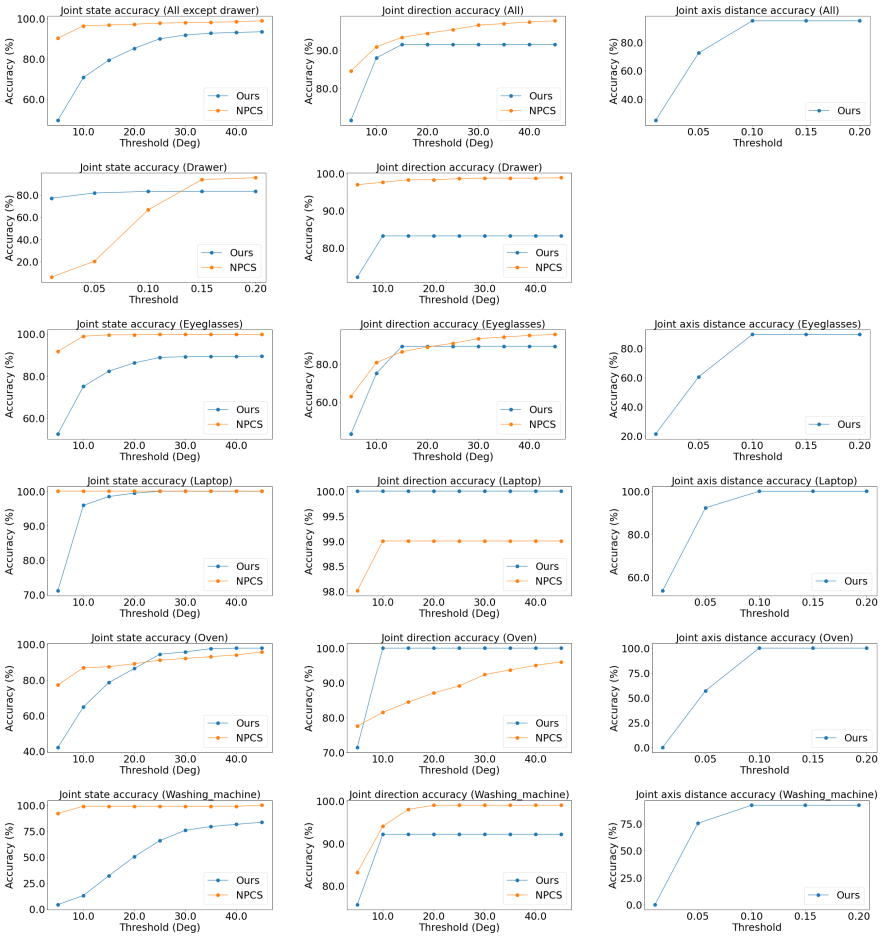

Fig. 10. Joint parameter estimation performance.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
2. Chen, Z., Tagliasacchi, A., Zhang, H.: Bsp-net: Generating compact meshes via binary space partitioning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 45–54 (2020)
3. Chen, Z., Yin, K., Fisher, M., Chaudhuri, S., Zhang, H.: Bae-net: branched autoencoder for shape co-segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR). pp. 8490–8499 (2019)
4. Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., Tagliasacchi, A.: Cvxnet: Learnable convex decomposition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 31–44 (2020)
5. Kawana, Y., Mukuta, Y., Harada, T.: Neural star domain as primitive representation. In: Advances in Neural Information Processing Systems (NeurIPS). pp. 7875–7886 (2020)
6. Li, X., Wang, H., Yi, L., Guibas, L.J., Abbott, A.L., Song, S.: Category-level articulated object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3706–3715 (2020)
7. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4460–4470 (2019)
8. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: Proceedings of the International Conference on Learning Representations (ICLR) (2018)
9. Paschalidou, D., Katharopoulos, A., Geiger, A., Fidler, S.: Neural parts: Learning expressive 3d shape abstractions with invertible neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3204–3215 (2021)
10. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 652–660 (2017)
11. Shu, D.W., Park, S.W., Kwon, J.: 3d point cloud generative adversarial network based on tree structured graph convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019)
12. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. In: Advances in Neural Information Processing Systems (NeurIPS). pp. 7462–7473 (2020)
13. Tulsiani, S., Su, H., Guibas, L.J., Efros, A.A., Malik, J.: Learning shape abstractions by assembling volumetric primitives. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2635–2643 (2017)
14. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
15. 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8876–8884 (2019)
16. Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., et al.: Sapien: A simulated part-based interactive environment. In:

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11097–11107 (2020)