

# Command-driven Articulated Object Understanding and Manipulation

## Supplementary Material

Ruihang Chu<sup>1</sup> Zhengzhe Liu<sup>1</sup> Xiaoqing Ye<sup>2</sup> Xiao Tan<sup>2</sup>  
Xiaojuan Qi<sup>3\*</sup> Chi-Wing Fu<sup>1,4</sup> Jiaya Jia<sup>1,4</sup>  
<sup>1</sup>CUHK <sup>2</sup>Baidu Inc. <sup>3</sup>HKU <sup>4</sup>SmartMore

In this document, we first present the detailed experiment settings in Sec. A. We then provide additional experimental results in Sec. B to extensively demonstrate the capability of our Cart. More visualization results are shown in Sec. C and discussions are given in Sec. D. We recommend reviewers to watch the attached *video* that exhibits animated manipulation procedures.

### A. Detailed Experiment Setups

**Network architecture.** Following CSC [1], we employ the same Sparse 3D U-Net as the backbone of Seg-Net. As Seg-Net is used for the structure-agnostic part segmentation, which is difficult, we observe that using other backbones will produce inferior performance. For the three prediction heads of Seg-Net, each consists of two MLP layers, the first of which is followed by batch normalization and ReLU. In Art-Net, we employ a PointNet++ [5] backbone (the multi-scale grouping paradigm) to extract per-point features.

**Dataset statistics.** In Table 4, we list the number of object instances for each object category in our benchmark. The *cabinet*, *oven*, and *laptop* categories are from the Shape2Motion [6] dataset, and others come from the PartNet-Mobility [7] dataset. To stabilize network training, we clean up the samples by ignoring the motion of extremely small object parts, which are beyond the capacity of the models. For example, the control button of a microwave is too small to localize.

**Implementation details.** We build our models using PyTorch and train one model per object category. Before feeding point clouds into the network, we follow the common 3D segmentation pipeline [1] to adopt data augmentation strategies for Seg-Net. Yet, no augmentation operation is employed to train Art-Net. For all experiments, the model is trained on four NVIDIA 2080Ti GPUs for 10k steps, with a batch size of  $4 \times 8$ . We use the default AdamW optimizer with an initial learning rate of 0.01. At the Seg-Net inference stage, the sphere radius  $r$  for part clustering is set to 2 mm.

\*Corresponding Author










Num.									
	cabinet	oven	laptop	eyegla	micro	fridge	stapler	table	storage
train	25	29	71	50	10	38	18	86	36
val	2	2	5	5	2	2	2	8	3
test	3	3	8	10	4	4	3	12	6

Table 4. Data statistics. On each object category, we list the number of object instances for training, validation, and testing, respectively.

Method	cab	ove	lap	eye	mic	fri	sta	tab	sto
RPM-Net [8]	93.2	100	100	100	100	100	100	93.7	100
ANCSH [4]	97.4	100	100	100	100	100	100	98.0	100
Ditto [3]	98.9	100	100	100	100	100	100	99.2	100
Cart (Ours)	<b>99.2</b>	100	100	100	100	100	100	<b>99.2</b>	100

Table 5. Comparison on mean accuracy of the predicted joint types (*i.e.*, revolute or prismatic) on nine object categories.

Method	cab	ove	lap	eye	mic	fri	sta	tab	sto
ANCSH [4]	-	0.04	0.10	0.10	0.05	-	0.06	-	0.05
Cart (Ours)	<b>0.04</b>	<b>0.02</b>	<b>0.02</b>	<b>0.05</b>	<b>0.02</b>	<b>0.03</b>	<b>0.03</b>	<b>0.03</b>	<b>0.03</b>

Table 6. Comparison on prediction accuracy of the articulation states. It is measured by the normalized distance (or angle) errors.

When training models on synthetic datasets [6, 7], we randomly select a point from the object’s point cloud and use its coordinate to denote the part to operate. In case of real robotic scenarios equipped with RGB-D cameras, we could specify a 3D point by picking its projection coordinates in the 2D image. Then, the associated 3D position in the camera frustum can be found using the depth information.

### B. Additional Experiments

**Joint type prediction.** Table 5 shows the comparison results of joint type prediction. Since this task can be easily solved by a binary classification, all compared methods can accurately identify the motion type by analyzing object geometry. Our Cart attains the best performance as well.

**State prediction.** A key factor to our test-time state adapta-

$\mathcal{L}_{p2a}$	cab	ove	lap	eye	mic	fri	sta	tab
☒	0.05	0.04	0.02	0.03	0.03	0.05	0.03	0.05
☑	<b>0.04</b>	<b>0.02</b>	<b>0.02</b>	<b>0.03</b>	<b>0.03</b>	<b>0.04</b>	<b>0.02</b>	<b>0.04</b>

Table 7. Effects of the point-axis distance loss, denoted as  $\mathcal{L}_{p2a}$ . It effectively reduces position errors of the predicted revolute axis.

tion is that Seg-Net should accurately predict the articulation state for each object part. Otherwise, we cannot reliably measure the state difference between the manipulated shapes and ground truths and refer to it for motion adjustment. As shown in Table 6, Seg-Net attains satisfactory performance and yields fewer prediction errors compared to ANCSH [4].

**Ablation on loss  $\mathcal{L}_{p2a}$ .** For joint parameter prediction, we apply an additional point-axis distance loss to supervise the position of the revolute axis, as illustrated in Sec. 4.2. This loss jointly considers the original point coordinates and the predicted point-to-axis projections, thus strengthening the constraint on axis position. The results in Table 7 demonstrate that it brings consistent performance improvement.

**Iteration rounds of TTSA.** Fig. 9 shows articulation state errors in the process of Test-time State Adaptation (TTSA). TTSA gradually adjusts the amount of movement to reduce the final state errors. It typically converges in 70-80 rounds.

**Failure cases.** Due to our segment-then-prediction paradigm, we observe that inaccurate part segmentation may easily lead to manipulation mistakes. Fig. 10 shows two typical failure cases of our method. The upper region shows that Cart may fail to separate multiple spatially-close parts of the object with complex structure. Thus, the manipulated shape differs from the ground truth. At the bottom region, Cart cannot discriminate the front and back faces of a microwave, resulting in two lid predictions (denoted by blue and green colors).

The fundamental reason is due to the inadequate visual observation, since Cart takes only one single point cloud without texture information. Future work would enhance the visual input, *e.g.*, by combing 2D RGB images, to improve the part discovery performance. This strategy has also been mentioned in Sec. 6 of the main paper.

## C. More Visualizations

We show quantitative visualization results of Cart on various object categories in Fig. 11. Taking as input a point cloud with an arbitrary articulation state, Cart can infer the inherent articulation structure and manipulate the specified part(s) to the target state, according to the command.

## D. Discussions

**The choice of template-based command.** In this work, we opt for template-based commands over human natural language instructions to facilitate easy implementation and

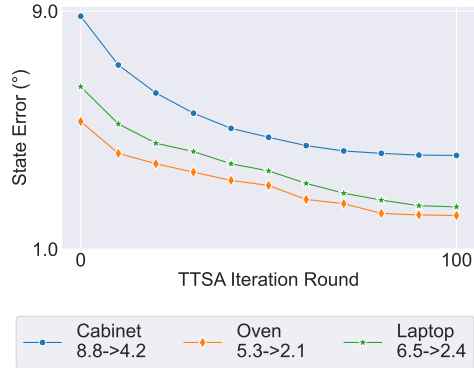


Figure 9. Errors of the articulation states at each test-time adaptation round. We conduct experiments on three object categories.

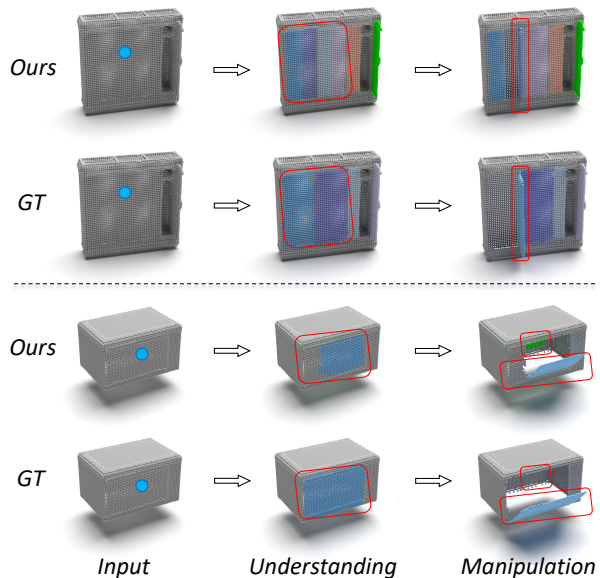


Figure 10. Failure cases. The red boxes denote the main difference between our predictions and ground truths.

precise manipulation control. As the command space is determined by the states nodes, adjusting state node number  $K$  can support fine-grained commands. For instance, with  $K=30$ , our approach can manipulate articulated objects with the minimum precision unit of 3 degrees. It's far beyond the representation accuracy of conventional NLP models.

**Limitations.** Occlusion issues are typical for models with single or partial observations, potentially giving rise to inaccurate segmentation results. As discussed in the earlier failure case, this may compromise the predictions of articulation parameters and ultimately hinder the manipulation. To mitigate this issue, we could leverage the active perception technique to enhance visual observations. Additionally, incorporating image input into our framework is another way, which could leverage the advances in 2D articulation recog-



Figure 11. Visualizing how Cart enables object manipulations. Distinct object parts have different colors. The dots in the leftmost column denote the parts chosen to be operated.

nition technology, *e.g.*, OPD [2], to improve the articulation estimation.

**Future work.** There are two primary directions for work extension. The first direction is to incorporate more convenient interaction ways into our framework to facilitate human-instructed manipulation. Specifically, driven by the transformative impact of large language models like ChatGPT, intergrating their APIs might enable robots to understand complex human instructions and operate the objects accordingly. Secondly, we should take into account the physical action affordance (*e.g.*, joint friction) to better adapt our approach to real-world robot tasks.

## References

- [1] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *CVPR, 2020*. 1
- [2] Hanxiao Jiang, Yongsen Mao, Manolis Savva, and Angel X Chang. Opd: Single-view 3d openable part detection. In *ECCV, 2022*. 4
- [3] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *CVPR, 2022*. 1
- [4] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *CVPR, 2020*. 1, 2
- [5] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS, 2017*. 1
- [6] Xiaogang Wang, Bin Zhou, Yahao Shi, Xiaowu Chen, Qingping Zhao, and Kai Xu. Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In *CVPR, 2019*. 1
- [7] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *CVPR, 2020*. 1
- [8] Zihao Yan, Ruizhen Hu, Xingguang Yan, Luanmin Chen, Oliver van Kaick, Hao Zhang, and Hui Huang. Rpm-net: Recurrent prediction of motion and parts from point cloud. *TOG, 2019*. 1