Supplementary Material of HS-Pose: Hybrid Scope Feature Extraction for Category-level Object Pose Estimation

Linfang Zheng[1,4] Chen Wang[1,2] Yinghan Sun[1] Esha Dasgupta[4] Hua Chen[1]
Aleš Leonardis[4] Wei Zhang*[1,3] Hyung Jin Chang[4]

[1]Department of Mechanical and Energy Engineering, Southern University of Science and Technology
[2]Department of Computer Science, the University of Hong Kong
[3]Peng Cheng Laboratory, Shenzhen, China
[4]School of Computer Science, University of Birmingham
{lxz948,exd949}@student.bham.ac.uk, cwang5@cs.hku.hk, sunyh2021@mail.sustech.edu.cn
{chenh6,zhangw3}@sustech.edu.cn,{a.leonadis,h.j.chang}@bham.ac.uk

1. Choice of the Baseline Method.

We choose GPV-Pose [5] as the baseline for the following reasons:

- **To demonstrate the effectiveness of the components in the HS-layer regarding pose estimation.** GPV-Pose is one of the state-of-the-art **3D-GC [10] based** category-level object pose estimation methods. It is suitable for us to show how each component of the HS-layer incrementally added onto the 3D-GC layer influences the performance of pose estimation.
- **To compare the HS-layer with the strategies proposed by other methods.** For example, SSP-Pose [16] and RBP-Pose [15] are also developed based upon GPV-Pose. The former leverages the prior-shape information and uses a shape deformation module to improve performance. The latter enhances GPV-Pose by a *residual bounding box projection* (SPRV) module and a shape deformation module. We compare with SSP-Pose to demonstrate the effectiveness of STE. We also show the influence of the RF-F approach by comparing it with RBP-Pose. In the experiments, our simple STE and RF-F method outperform their counterparts in strict metrics (*e.g.*, IoU_{75}, $5°2$cm, and $5°5$cm metrics) and achieve competitive results in other metrics.

2. About the Object Detector

For a fair comparison, as when compared against other methods [5, 15, 16], we also utilize the MaskRCNN [6] to detect the objects in our experiments. It is worth noting that our method is not limited to MaskRCNN [6]. Other object detectors such as SD-MaskRCNN [4] and PointNet [11] can also be used.

3. About the Speed

Since the speed can be different when performed on different machines, we only use the results of the speed to demonstrate that our method can achieve real-time performance and do not emphasise a speed comparison with other methods.

3.1. The Speed of GPV-Pose

For a fair speed comparison with the baseline, GPV-Pose [5], we report the speed of GPV-Pose on our machine with the same evaluation code as ours. The speed of GPV-Pose achieved on our machine (69 FPS) is faster than the original paper (20 FPS) due to the following reasons:

- **The difference between the machines.** The original paper of GPV-Pose reports the speed test on a single TITAN X GPU, while we test GPV-Pose on a single RTX 3090 GPU with an Intel(R) Core(TM) i9-10900K CPU, 32 GB RAM. The speed is 33 FPS on our machine.
- **The difference in the evaluation code.** Our evaluation code is a refactored version of GPV-Pose's code. We change some for-loop operations to batch operations and remove unnecessary calculations (*e.g.* the bounding box voting and symmetric point cloud reconstruction) during inference. These changes significantly boost the speed from 33 FPS to 69 FPS. All the changes have passed unit tests to ensure they get the same results as the original code.

*The corresponding author.

Table 1. **Comparison with the state-of-the-art methods on REAL275 dataset.** Overall best results are in bold, and the second-best results are underlined. *Type* lists the type of input data for pose estimation. *Syn.* denotes whether the synthetic data is used during training.

Method	Type	Syn.	IoU_{25}	IoU_{50}	IoU_{75}	5°2cm	5°5cm	10°2cm	10°5cm	10°10cm	Speed(FPS)
NOCS [13]	RGB-D	✓	**84.9**	80.5	30.1	-	9.5	13.8	26.7	26.7	5
CASS [1]	RGB-D	✓	84.2	77.7	15.3	19.5	23.5	50.8	58.0	58.3	-
SPD [12]	RGB-D	✓	83.4	77.3	53.2	19.3	21.4	43.2	54.1	-	4
DualPoseNet [9]	RGB-D	✓	-	79.8	62.2	29.3	35.9	50.0	66.8	-	2
SGPA [2]	RGB-D	✓	-	80.1	61.9	35.9	39.6	61.3	70.7	-	-
CR-Net [14]	RGB-D	✓	-	79.3	55.9	27.8	34.3	47.2	60.8	-	-
Self-DPDN [8]	RGB-D	✓	-	**83.4**	**76.0**	46.0	50.7	**70.4**	78.4	-	-
SPD [12]	RGB	✓	-	75.2	46.5	15.7	18.8	33.7	47.4	-	4
SAR-Net [7]	D		-	79.3	62.4	31.6	42.3	50.4	68.3	-	10
FS-Net[1] [3]	D		84.0	81.1	63.5	19.9	33.9	-	69.1	71.0	20
SSP-Pose [16]	D		84.0	82.3	66.3	34.7	44.6	-	77.8	79.7	25
RBP-Pose [15]	D		-	-	67.8	38.2	48.1	63.1	79.2	-	25
GPV-Pose [5]	D		84.1	83.0	64.4	32.0	42.9	55.0	73.3	74.6	**69**
Ours (10 neighbors)	D		84.2	82.1	74.7	**46.5**	55.2	68.6	82.7	83.7	50
Ours (20 neighbors)	D		84.3	82.8	75.3	46.2	**56.1**	68.9	**84.1**	**85.2**	38

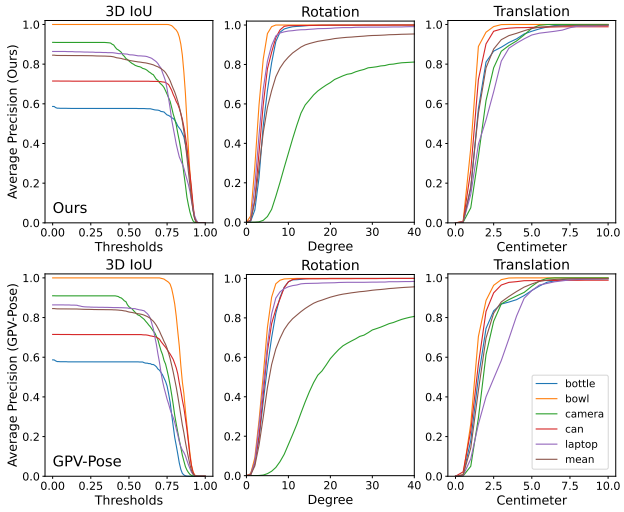

Figure 1. **Per-category comparison between our method and GPV-Pose.** We demonstrate average precision *v.s.* different error thresholds on the REAL275 dataset.

4. Comparison with State-of-the-Arts Methods

We add the comparison between the proposed HS-Pose with methods that use different data modalities (*e.g.* RGB and RGB-D) in this section.

4.1. Results on REAL275 dataset.

The comparison between the proposed method and the state-of-the-art methods on the REAL275 dataset is shown in Table 1. Our method outperforms the depth-only methods in 7 out of 8 pose estimation and size estimation metrics and achieves comparable performance in the remaining metric. Our depth-only method also achieves competitive results with the RGB-D-based approaches and outperforms them in several pose estimation metrics (*e.g.* **56.1%** (ours) vs. 50.7% on 5°5cm metric). It is worth noting that many of them are trained with synthetic data or using CAMERA25 and REAL275 for mixed training, which results in a large number of training images and many more objects (over 1K objects for CAMERA25 and Real275 mixed training) for training. In contrast, our method is trained on 1.6K real images of 18 objects. In Figure 1, we present the average precision of each category under different thresholds and compare it with the GPV-Pose.

Table 2. **Comparison with state-of-the-art methods on CAMERA25 dataset.** Overall best results are in bold, and the second-best results are underlined. *Type* lists the type of input data for pose estimation. *Prior* denotes whether the method uses shape priors.

Method	Type	Prior	IoU_{50}	IoU_{75}	5°2cm	5°5cm	10°2cm	10°5cm
SPD [12]	RGB-D	✓	93.2	83.1	54.3	59.0	73.3	81.5
CR-Net [14]	RGB-D	✓	**93.8**	88.0	72.0	76.4	81.0	87.7
SGPA [2]	RGB-D	✓	93.2	88.1	70.7	74.5	**82.7**	88.4
NOCS [13]	RGB-D		83.9	69.5	32.3	40.9	48.2	64.6
DualPoseNet [9]	RGB-D		92.4	86.4	64.7	70.7	77.2	84.7
SPD [12]	RGB	✓	93.1	84.6	50.2	54.5	70.4	78.6
SAR-Net [7]	D	✓	86.8	79.0	66.7	70.9	75.3	80.3
SSP-Pose [16]	D	✓	-	86.8	64.7	75.5	-	87.4
RBP-Pose [15]	D	✓	93.1	89.0	73.5	79.6	82.1	89.5
GPV-Pose [5]	D		93.4	88.3	72.1	79.1	-	89.0
Ours (10 neighbors)	D		93.3	**89.4**	73.3	80.5	80.4	89.4
Ours (20 neighbors)	D		93.4	89.3	**74.0**	**82.0**	80.3	**90.2**

4.2. Results on CAMERA25 dataset.

We test the proposed method on the CAMERA25 dataset and show the comparison results of the proposed method with other approaches in Table 2. We achieved top and second scores on 5 out of 6 metrics (4 tops and 1 second) with no need for RGB data.

[1]We use the results provided by the GPV-Pose, which uses the GPV-Pose's decoder for a fair comparison and shows higher performance than the originally reported results of the FS-Net.

Table 3. **Per-category results of our method on REAL275 dataset.**

category	IoU_{25}	IoU_{50}	IoU_{75}	5°2cm	5°5cm	10°2cm	10°5cm	10°10cm	5°	10°	2cm	5cm	10cm
bottle	57.7	57.7	54.8	43.0	53.1	80.0	95.4	98.5	66.9	99.2	81.0	96.5	99.5
bowl	100.0	100.0	100.0	92.1	95.6	96.5	100.0	100.0	95.6	100.0	96.5	100.0	100.0
camera	90.9	82.3	65.2	2.3	3.1	28.3	35.7	35.8	3.1	35.8	60.9	98.1	100.0
can	71.4	71.4	70.5	68.6	75.0	90.0	98.5	98.5	77.8	99.7	90.0	98.6	98.8
laptop	86.1	84.9	67.1	49.3	79.5	52.4	94.0	96.8	80.8	96.9	52.4	94.6	99.5
mug	99.2	96.4	90.8	23.8	25.0	64.6	72.6	72.6	25.0	72.6	88.5	100.0	100.0
average	84.2	82.1	74.7	46.5	55.2	68.6	82.7	83.7	58.2	84.0	78.2	98.0	99.6

Table 4. **Per-category results of our method on CAMERA25 dataset.**

category	IoU_{25}	IoU_{50}	IoU_{75}	5°2cm	5°5cm	10°2cm	10°5cm	10°10cm	5°	10°	2cm	5cm	10cm
bottle	93.9	93.8	90.9	80.1	96.7	80.7	97.8	99.4	98.5	99.8	80.7	97.9	99.5
bowl	96.9	96.8	96.8	98.4	98.6	99.4	99.8	99.8	98.7	99.8	99.4	99.8	99.9
camera	94.8	85.4	74.3	51.2	55.1	65.0	70.6	70.9	55.5	71.4	86.9	99.0	99.6
can	92.5	92.4	92.2	99.0	99.4	99.0	99.5	99.5	99.9	100.0	99.0	99.5	99.6
laptop	98.4	97.4	90.6	75.6	85.2	81.1	92.7	97.0	89.0	97.1	83.3	95.6	99.9
mug	94.1	93.8	91.9	35.4	47.9	57.4	76.2	76.2	49.1	76.9	75.9	99.5	99.6
average	95.1	93.3	89.4	73.3	80.5	80.4	89.4	90.5	81.8	90.8	87.5	98.6	99.7

5. Per-category Results

The per-category results trained on the REAL275 and CAMERA25 datasets are shown in Table 3 and Table 4, respectively.

6. Settings of Noise Resistance Experiments

In the ablation study [AS-6], we compared the outlier robustness of the proposed method and the baseline. We define *outliers* as the points that do not belong to the target object. The *outlier ratio* is defined as the ratio of the outliers' number to the total point number of the input point cloud. We use the REAL275 dataset for testing and generate the noisy input data by sampling points from the background and the object region according to the outlier ratio. To ensure a fair comparison, the noisy data used for testing the proposed and baseline methods is the same.

7. Ablation Study on ORL

We demonstrate the effectiveness of the proposed outlier robust feature extraction layer (ORL) in Fig. 2. We test the noise resistance of the proposed HS-Pose with and without ORL using different outlier ratios (from 0.0% to 40.0%) in the input point cloud. The figure shows that the ORL successfully enhances the performance on the size-pose-joint metric (IoU_{75}), translation metric (2cm), and rotation metric (10°) across different noise levels.

8. Ablation Study on the Neighbor Numbers

To investigate the influences of the neighbor numbers, we test the performance of the proposed method using different neighbor numbers (from 3 to 40) in the RF-F and ORL[2]. The experiments are separated into three groups to evaluate the impact: 1) change the RF-F's neighbor number with the ORL's neighbor number fixed, 2) change the ORL's neighbor number with the RF-F's neighbor number fixed, and 3) change the neighbor numbers of the RF-F and ORL simultaneously.

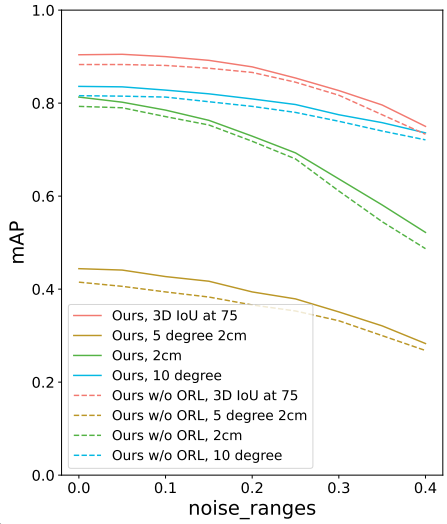

Figure 2. **The comparison of noise resistance of the proposed HS-Pose with (Ours) and without outlier robust feature extraction layer (ORL) (Ours w/o ORL).** We show the influences of ORL in differences metrics (*e.g.* 2cm, IoU_{75}, 5°2cm, and 10°). After adding ORL, the performance is enhanced across different noise levels (from 0.0% to 40.0% outliers).

[2]Due to the limit of the GPU memory size, we set the batch size to 16, 16, and 8 for 20, 30, and 40 neighbors, respectively.

Table 5. **Performance of the proposed method when changing the neighbor number of RF-F.** The neighbor number of the ORL is fixed to 10 in this experiment. Overall best results are in bold, and the second-best results are underlined.

Neighbor Number	3	5	10	20	30	40
5°2cm	39.8	41.5	**46.5**	<u>46.1</u>	44.3	41.6
5°5cm	49.2	51.4	<u>55.2</u>	**56.7**	54.7	54.4
IoU_{75}	72.9	72.8	**74.7**	<u>73.4</u>	**74.7**	71.9
Speed (FPS)	**64**	60	50	41	34	30

Table 6. **Performance of the proposed method when changing the neighbor number of ORL.** The neighbor number of RF-F is fixed to 10 in this experiment. Overall best results are in bold, and the second-best results are underlined.

Neighbor Number	3	5	10	20	30	40
5°2cm	43.3	<u>43.6</u>	**46.5**	42.7	43.1	39.4
5°5cm	53.1	53.0	<u>55.2</u>	**55.3**	54.4	53.7
IoU_{75}	74.6	72.8	**74.7**	72.7	73.9	71.1
Speed (FPS)	**52**	51	50	48	48	49

8.1. Change RF-F's Neighbor Number Only

Table 5 shows the performance of the proposed method using different neighbor numbers in the RF-F with the ORL's neighbor number fixed to 10. As seen from the table, finding more neighboring 3D points using feature distance requires a longer time, while a certain range of neighbor numbers (around 10-20 neighbors) produces better precision than other numbers. Specifically, the speed decreased from 64 FPS to 30 FPS when increasing the neighbor number from 3 to 40. In the meantime, the performance on 5°2cm, which starts at 39.8%, reaches its best at 46.5% when using 10 neighbors, after which it begins to decline and ultimately reaches a score of 41.6% at 40 neighbors. Generally, using 10 neighbors for RF-F achieves the overall best performance while maintaining fast speed. The reason why insufficient and excessive neighbor numbers adversely affect the precision might be that fewer neighbors cannot fully characterize the global geometric feature, whereas an excessive number of neighbors may obscure the geometric structural information in the formed receptive field.

8.2. Change ORL's Neighbor Number Only

Table 6 shows the performance of the proposed method using different neighbor numbers in the ORL with the RF-F's neighbor number fixed to 10. According to the table, the speed for finding neighboring points in 3D space is relatively stable, which only dropped by 4 FPS when the neighbor number increased from 3 to 40. Compared to the RF-F, the neighbor number impacts the speed less. The reason is that in RF-F, the nearest neighbors are found in higher dimensional feature space. In terms of precision, an appropriate range of neighbor numbers is beneficial for ORL to balance finding the reliable points and outliers. Similar to RF-R, using 10 neighbors performs better than other values in our experiments.

Table 7. **Performance of the proposed method when changing the neighbor number of the ORL and RF-F together.** The neighbor number of ORL and RF-F are set to the same in this experiment. Overall best results are in bold, and the second-best results are underlined.

Neighbor Number	3	5	10	20	30	40
5°2cm	39.0	41.7	**46.5**	46.2	<u>46.4</u>	39.6
5°5cm	47.6	52.8	55.2	<u>56.1</u>	**56.6**	55.8
IoU_{75}	73.1	72.7	74.7	**75.3**	<u>75.2</u>	70.3
Speed (FPS)	64	59	50	38	30	26

8.3. Change the Neighbor Number Simultaneously

Table 7 shows the performance of the proposed method with the neighbor numbers of the ORL and RF-F changing simultaneously. As shown in the table, when the neighbor numbers are around 10-30, the performance of our method is best. Moreover, in this range, using the same number of neighbors leads to better precision compared to fixing one of the neighbor numbers to 10. The reason might be that with increasing neighbor number in RF-F, more global geometric structure information can be found, and the possibility to include uninformed points is also increased. Therefore, with the neighbor number in ORL also increased, the effect brought by these uninformed points can be compensated, thus resulting a better performance. However, with too many neighbors, the performance still deteriorates because the balance between identifying reliable points and rejecting outliers is hurt.

9. Qualitative Results

More qualitative results comparing our method with the GPV-pose are shown in Fig.3.

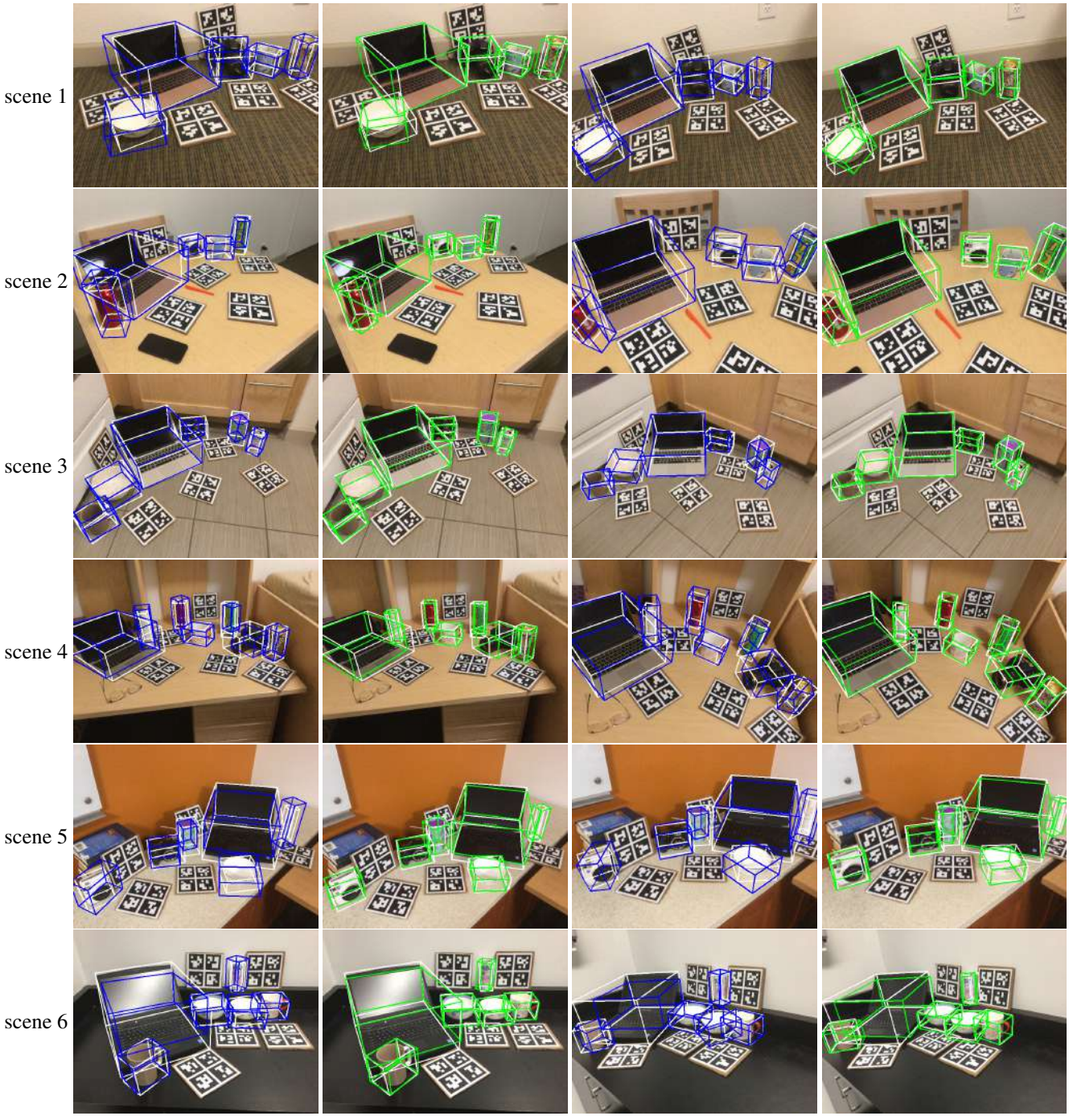

Figure 3. **More qualitative results of our method (green line) and the GPV-Pose (blue line) on the REAL275 dataset.** We choose two instances from each scene. The ground truth results are shown with white lines. The estimated rotations of symmetric objects (*e.g.* bowl, bottle, and can) are considered correct if the symmetry axis is aligned.

References

[1] Dengsheng Chen, Jun Li, Zheng Wang, and Kai Xu. Learning canonical shape space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[2] Kai Chen and Qi Dou. Sgpa: Structure-guided prior adapta-

tion for category-level 6d object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2773–2782, October 2021. 2

[3] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, Linlin Shen, and Ales Leonardis. Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1581–1590, June 2021. 2

[4] Michael Danielczuk, Matthew Matl, Saurabh Gupta, Andrew Li, Andrew Lee, Jeffrey Mahler, and Ken Goldberg. Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019. 1

[5] Yan Di, Ruida Zhang, Zhiqiang Lou, Fabian Manhardt, Xiangyang Ji, Nassir Navab, and Federico Tombari. Gpv-pose: Category-level object pose estimation via geometry-guided point-wise voting, 2022. 1, 2

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. 1

[7] Haitao Lin, Zichang Liu, Chilam Cheang, Yanwei Fu, Guodong Guo, and Xiangyang Xue. Sar-net: Shape alignment and recovery network for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6707–6717, June 2022. 2

[8] Jiehong Lin, Zewei Wei, Changxing Ding, and Kui Jia. Category-level 6d object pose and size estimation using self-supervised deep prior deformation networks, 2022. 2

[9] Jiehong Lin, Zewei Wei, Zhihao Li, Songcen Xu, Kui Jia, and Yuanqing Li. Dualposenet: Category-level 6d object pose and size estimation using dual pose network with refined learning of pose consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3560–3569, October 2021. 2

[10] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1797–1806, 2020. 1

[11] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1

[12] Meng Tian, Marcelo H Ang Jr, and Gim Hee Lee. Shape prior deformation for categorical 6d object pose and size estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 2

[13] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J. Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[14] Jiaze Wang, Kai Chen, and Qi Dou. Category-level 6d object pose estimation via cascaded relation and recurrent reconstruction networks. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4807–4814. IEEE, 2021. 2

[15] Ruida Zhang, Yan Di, Zhiqiang Lou, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Rbp-pose: Residual bounding box projection for category-level pose estimation, 2022. 1, 2

[16] Ruida Zhang, Yan Di, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Ssp-pose: Symmetry-aware shape prior deformation for direct category-level object pose estimation, 2022. 1, 2