VI. SUPPLEMENTARY

A. Experiments with Motion Capture System

1) Experiment Setup: In addition to the original 175 sequences in our dataset, we collected 10 extra validation sequences shown in fig. 4 with precise object pose ground truth using a motion capture system. We made every effort to ensure these 10 additional sequences closely mirror the experimental conditions of the original 175 sequences. These 10 sequences feature 10 different objects: 6 objects come from 6 categories in COPE-119, and 4 objects are from UOPE-56. To provide ground truth poses using the motion capture system, each object was affixed with at least three markers, forming a rigid body frame within the motion capture system coordinate system. We refer to the rigid body frames formed by markers attached to the object and the camera as the object body frame and camera body frame, respectively. The position and orientation of object body in the motion capture system is denoted as $\mathbf{T}_{OB,M}$. It is important to note that the orientation and position of this object body frame differ from those of the object's own coordinate system; there is a fixed but unknown transformation $\mathbf{T}_{O,OB}$ between them. We determined this unknown transformation based on the geometric relationship

$$\mathbf{T}_{O,OB} = (\mathbf{T}_{OB,M})^{-1}\mathbf{T}_{CB,M}\mathbf{T}_{C,CB}\mathbf{T}_{O,C} \tag{10}$$

, where $\mathbf{T}_{C,CB}$ is the handeye calibration result representing the transformation from camera to camera body frame, $\mathbf{T}_{CB,M}, \mathbf{T}_{OB,M}$ are motion capture reading of camera body and object body position and orentation in motion capture system world coordinate, and $\mathbf{T}_C^O$ is the object pose which can be estimated by algorithms like FoundationPose [1]. Note that, to ensure accuracy, in our setup, $\mathbf{T}_{O,C}$ is estimated by [1] first then refined by manual alignment of the object point cloud and its CAD model. Finally, the ground truth object pose is given by

$$\mathbf{T}_{O,C} = (\mathbf{T}_{C,CB})^{-1}(\mathbf{T}_{CB,M})^{-1}\mathbf{T}_{OB,M}\mathbf{T}_{O,OB} \tag{11}$$

. Please note that after the markers are attached to these 10 objects, we reconstructed object CAD models using an RGB-D scanner. However, because the markers themselves reflect infrared light, they interfered with the RGB-D scanner's depth readings, leading to imperfections in certain areas of the object mesh. Despite this, the overall quality remains usable.

2) Implementation Details: We benchmark several key outputs against the ground truth object pose: the raw, unprocessed FoundationPose [1] (referred to as **Raw Abs-Pose**), the refined and smoothed [1] by global Kalman Filter (**Refined Abs-Pose**), as well as the outputs of our relative pose estimator (**Rel-Pose**) and the pose graph optimization results (**PGO-Pose**). In pose graph optimization, the information matrix of the absolute pose edge is set heuristically. First, for each frame, we reproject the object bounding box and axis onto the 2D image based on the **Refined Abs-Pose** estimation. Then, we stitch these images into a video. By observing whether the projected bounding box in the video exhibits jitter, we can assess the quality of the absolute pose estimation. Typically, 2D projection jitter indicates an inconsistency between the object poses in consecutive frames. This means that the object poses across the few frames exhibiting this jitter will inevitably vary in quality—some may be good, some bad, or all of them might be poor. Consequently, these poses are considered unreliable.

In general, the quality of absolute pose estimation is better than that of relative pose estimation. Therefore, we adopt a conservative parameter strategy, setting the diagonal elements of the absolute pose information matrix to 1e5 to reduce the negative impact of the relative pose edge.

For those visually unreliable absolute edges, the information matrix elements are set to 1e2-1e3. Note that the elements of the relative pose edge information matrix are generally around 1e2. This allows the relative pose to effectively compensate for poor absolute pose estimations. Finnally, for our comparative analysis, we utilize ATE (Absolute Trajectory Error) and RPE (Relative Pose Error). The unit of translation error is millimeters and the unit of rotation error is degrees.

3) 3. Results: Here are our experimental comparison results, where we report the mean, max, and median for ATE shown in table VII and RPE shown in table VIII and table IX. Additionally, we randomly select several frames to present qualitative visualization results. The results show that our final ATE and RPE are very close to the ground truth, with an average translation error of less than 3 mm and an average rotation error of less than 0.2 degree, indicating quite good quality. Furthermore, while the overall results after pose graph optimization didn't change significantly, the maximum error decreases. This aligns with the design goal of this module: effectively compensating for poor estimations in **Refined Abs-Pose** and preserving the good ones. From the perspective of relative pose estimation itself, its drift is still noticeable, and its overall accuracy cannot match that of **Refined Abs-Pose**. Therefore, it can only serve to compensate for low-quality absolute poses.

B. Analysis of Unseen Object Pose Estimation Methods

Illustrated in fig. 7, [2]* heavily relies on features extracted by DINOv2 [3] to establish 2D-3D correspondences, which makes it susceptible to pose estimation errors for low-texture or highly symmetric objects. fig. 7 shows that the object (a frying pan) is textureless, and during the detection process, the right-side handle was mistakenly ignored. As a result, the object was erroneously treated as symmetric, leading to incorrect pose estimates.

As shown in fig. 8, we directly employ the model-based [1]. It primarily relies on the input RGB-D data. When the camera undergoes rapid motion, causing motion blur, the pose estimation results becomes error-prone as well. This issue is especially prominent for flat, textureless, and symmetric objects, where the network may incorrectly estimate the orientation of the coordinate axes.

The performance of GigaPose [4] benefits from its time-intensive pre-processing step, where it renders the input CAD model extensively before performing pose estimation. This process allows it to establish better correspondences during object pose estimation. However, since the method relies solely

Fig. 4: The 10 objects in Mocap experiments we used.

TABLE VII: ATE (mm) results.

Sequence ID	Rel-Pose			Raw Abs-Pose			Refined Abs-Pose			PGO Pose		
	ATE. max	ATE. mean	ATE. median	ATE. max	ATE. mean	ATE. median	ATE. max	ATE. mean	ATE. median	ATE. max	ATE. mean	ATE. median
Seq0	484.70	159.37	146.14	24.17	6.15	5.60	**16.85**	**5.98**	**5.56**	**16.85**	5.99	**5.56**
Seq1	364.25	176.07	166.79	**17.01**	4.90	**4.84**	**17.01**	4.89	4.86	**17.01**	**4.88**	4.86
Seq2	534.79	274.89	280.95	**17.16**	5.15	4.92	**17.16**	**5.06**	**4.78**	**17.16**	5.07	**4.78**
Seq3	228.37	85.35	83.48	**16.79**	6.35	**5.37**	**16.79**	6.35	5.38	**16.79**	6.35	5.38
Seq4	357.11	165.51	160.60	**16.87**	**5.91**	**5.84**	16.88	5.92	**5.84**	16.88	5.93	**5.84**
Seq5	257.58	111.89	90.20	**7.59**	**3.14**	**3.00**	**7.59**	3.15	3.01	**7.59**	3.15	3.01
Seq6	209.48	105.43	99.38	16.35	**5.70**	**5.38**	**16.34**	5.71	5.39	16.35	5.72	5.41
Seq7	250.53	117.48	121.19	19.91	4.86	3.93	**18.48**	**4.71**	**3.87**	18.50	**4.71**	**3.87**
Seq8	292.44	92.27	83.14	12.74	3.72	3.47	**12.73**	**3.71**	**3.46**	**12.73**	**3.71**	**3.46**
Seq9	296.40	153.10	165.70	**13.95**	3.90	3.33	**13.95**	**3.85**	**3.25**	**13.95**	**3.85**	**3.25**
Avg.	327.37	144.62	139.76	16.65	4.98	**4.53**	**15.38**	**4.93**	4.54	**15.38**	4.94	4.54

on RGB information and performs feature matching across different viewpoints of the CAD model, it faces significant challenges when dealing with occlusions or textureless objects. As shown in fig. 9, such limitations result in a failure case where the person's hand holding the mouse causes the estimation to break down.

As shown in fig. 10, here we evaluate both the RGB-based and RGB-D-based variants of [5]. This method takes a cropped image (RGB or RGB-D) as input along with the corresponding CAD model. Multiple poses are rendered from the CAD model and then passed to the coarse estimator to obtain the initial poses, which are subsequently refined by

TABLE VIII: RPE results for the rot. (degs).

Sequence ID	Rel-Pose			Raw Abs-Pose			Refined Abs-Pose			PGO Pose		
	RPE. rot.max	RPE. rot.mean	RPE. rot.median	RPE. rot.max	RPE. rot.mean	RPE. rot.median	RPE. rot.max	RPE. rot.mean	RPE. rot.median	RPE. rot.max	RPE. rot.mean	RPE. rot.median
Seq0	5.31	1.15	0.80	5.54	1.12	0.89	3.24	**0.21**	**0.14**	**3.21**	**0.21**	**0.14**
Seq1	7.54	1.05	0.78	9.80	0.95	0.75	5.36	0.15	**0.10**	**3.69**	**0.14**	**0.10**
Seq2	139.58	1.90	0.86	2.95	0.97	0.87	**0.45**	**0.13**	**0.12**	0.71	**0.13**	**0.12**
Seq3	8.71	0.96	0.80	5.64	1.31	1.11	**1.60**	**0.43**	**0.33**	2.88	**0.43**	**0.33**
Seq4	3.59	0.63	0.52	3.00	0.76	0.63	**2.86**	**0.36**	**0.27**	**2.86**	**0.36**	**0.27**
Seq5	3.21	0.77	0.64	3.29	0.83	0.72	**0.26**	**0.10**	**0.09**	**0.26**	**0.10**	**0.09**
Seq6	2.91	0.72	0.59	5.69	0.81	0.58	2.37	**0.14**	**0.09**	**1.44**	**0.14**	**0.09**
Seq7	3.86	0.78	0.64	4.94	0.80	0.61	0.57	**0.13**	**0.11**	**0.55**	**0.13**	**0.11**
Seq8	3.37	0.74	0.68	2.50	0.73	0.62	0.52	**0.11**	**0.09**	**0.49**	**0.11**	**0.09**
Seq9	5.44	0.85	0.58	3.97	0.79	0.66	2.78	**0.16**	**0.13**	**1.77**	**0.16**	**0.13**
Avg.	18.05	0.95	0.69	4.93	0.91	0.74	2.00	**0.19**	**0.15**	**1.79**	**0.19**	**0.15**

TABLE IX: RPE results for the trans. (mm).

Sequence ID	Rel-Pose			Raw Abs-Pose			Refined Abs-Pose			PGO Pose		
	RPE. trans.max	RPE. trans.mean	RPE. trans.median	RPE. trans.max	RPE. trans.mean	RPE. trans.median	RPE. trans.max	RPE. trans.mean	RPE. trans.median	RPE. trans.max	RPE. trans.mean	RPE. trans.median
Seq0	70.27	12.72	9.16	18.49	3.80	3.25	18.45	**3.72**	**3.17**	**13.98**	3.73	3.18
Seq1	56.04	10.16	7.97	18.60	2.17	1.66	18.39	**2.11**	**1.64**	**17.18**	2.12	**1.64**
Seq2	656.47	16.70	9.84	11.71	2.89	2.38	**11.67**	**2.87**	**2.33**	**11.67**	2.88	**2.33**
Seq3	37.90	8.66	7.70	**18.67**	3.31	2.10	18.74	**3.28**	**2.05**	18.74	**3.28**	**2.05**
Seq4	32.24	9.92	8.79	13.85	2.68	2.13	13.83	**2.65**	**2.07**	**12.63**	**2.65**	**2.07**
Seq5	41.07	6.98	6.09	**6.49**	1.28	1.03	6.51	**1.27**	**1.00**	6.51	**1.27**	**1.00**
Seq6	34.65	8.84	7.37	15.16	3.50	3.14	**15.12**	**3.47**	**3.06**	**15.12**	3.49	3.11
Seq7	35.43	8.35	6.72	18.15	2.37	1.51	18.16	**2.27**	**1.46**	**17.85**	**2.27**	**1.46**
Seq8	95.26	7.78	6.16	**11.75**	1.53	1.14	11.76	**1.49**	**1.13**	11.76	**1.49**	**1.13**
Seq9	27.24	6.78	6.05	10.58	1.96	1.72	**10.23**	**1.94**	**1.69**	**10.23**	**1.94**	**1.69**
Avg.	108.66	9.69	7.59	14.35	2.55	2.01	13.35	**2.51**	**1.96**	**12.63**	**2.51**	1.97

the refinement network to produce the final outputs. Note that the final results heavily depend on the coarse estimator network. If the initial pose estimation is inaccurate, then the refinement stage is generally unable to recover the correct poses. Therefore, both RGB and RGB-D approaches are highly dependent on the quality of the multiview input poses. For geometrically symmetric or regular-shaped objects such as the juice box shown in fig. 10, the use of multiple poses from the CAD model can easily lead to incorrect initial guesses. Consequently, failure cases are observed in both the RGB-based and RGB-D-based versions of the method, as illustrated in the figure fig. 10.

SAM6D [6] is a multi-stage method. It first segments any objects using SAM[7], and then associate the segmentation mask to the target object. And through the [8] to estimate the object pose. Incorrect registration may happen in symmetric and textureless object cases, resulting in failure cases such as the one shown in fig. 11.

C. Analysis of Object Pose Tracking Methods

For [9], to ensure a fair benchmark, we replaced its original mask results with masks generated by the state-of-the-art method SAM2 [10]. However, when intensive motion occurs, the object tracking algoritm tends to reinitialize its state. Consequently, this leads to noticeable drift. An example failure case is shown in fig. 12

[11] takes as input an RGB-D sequence along with object segmentation results. It then uses traditional keypoint matching methods to estimate the relative poses between consecutive frames, which are subsequently optimized through an object pose graph. However, as fig. 13 shows, its main drawback is that for objects with sparse texture, the relative pose estimations often become inaccurate, leading to accumulated drift in subsequent frames.

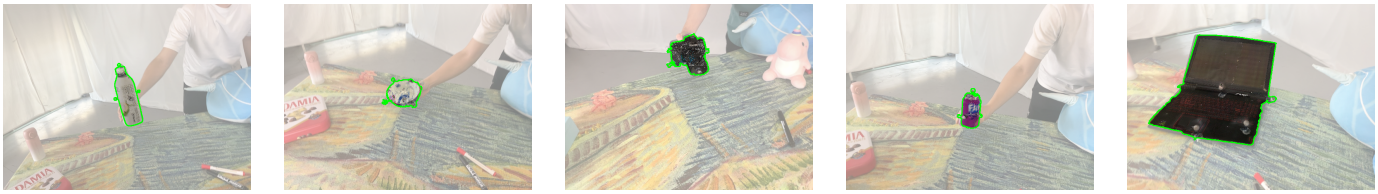
fig. 14) demonstrates overall robust performance without obvious drift or failure cases. However, it still faces considerable challenges when dealing with textureless and symmetric objects. Moreover, as [12] is a Neural Object Field-based approach, it simultaneously estimates the object's 6DoF pose and reconstructs the object's mesh. Consequently, the runtime for processing an entire sequence is considerably limited.

[13], as a popular SLAM framework, takes RGB-D inputs and was originally proposed for camera pose tracking and scene reconstruction. When provided with segmented images and fed into its DBA layer, it operates in an object-centric setting, allowing us to obtain the object pose for each frame. However, since the masked objects provide limited input information, any error in camera pose estimation can lead to significant drift(Fig.15).

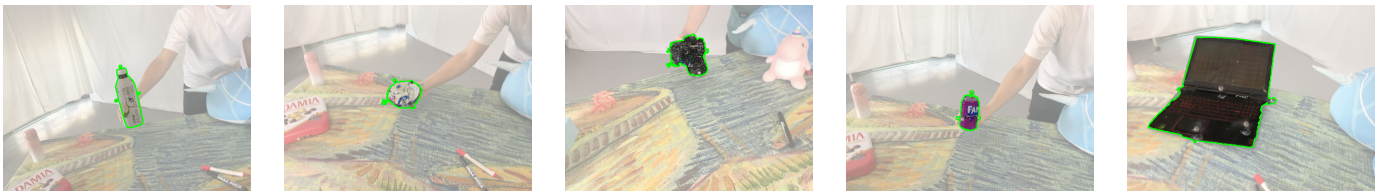
[1] supports both object pose estimation and object pose tracking. We benchmark its tracking mode. As shown in fig. 16, the tracking results in drift failure and tracking loss in rapid motion.

D. Analysis of Category-level Object Estimation Methods

1) Fine-tuning: Moreover, under our *Dynamic Object with Moving Camera scenarios*, the dataset presents different challenges compared to static real-world datasets such as

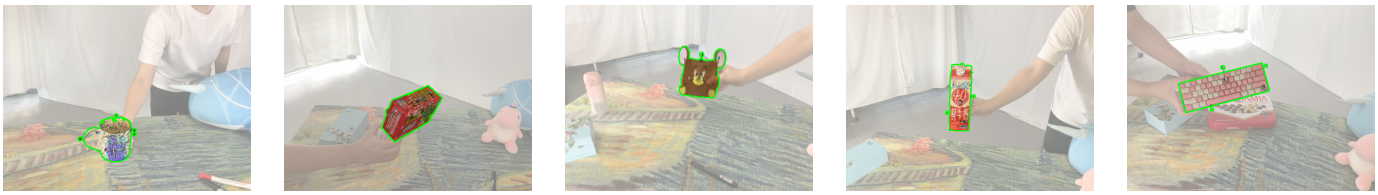


(a) Seq0 - Mocap (b) Seq1 - Mocap (c) Seq2 - Mocap (d) Seq3 - Mocap (e) Seq4 - Mocap

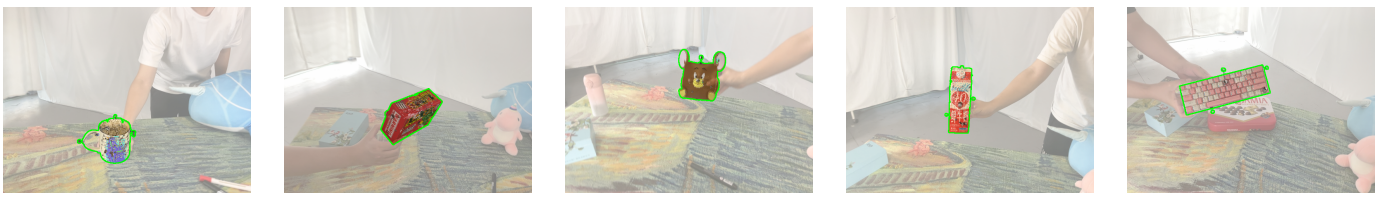


(f) Seq0 - PGO (g) Seq1 - PGO (h) Seq2 - PGO (i) Seq3 - PGO (j) Seq4 - PGO

Fig. 5: Comparison of algorithm performance across Seq0 - Seq4 sequences. Top row shows Mocap results, bottom row shows PGO results for the same sequences.



(a) Seq5 - Mocap (b) Seq6 - Mocap (c) Seq7 - Mocap (d) Seq8 - Mocap (e) Seq9 - Mocap



(f) Seq5 - PGO (g) Seq6 - PGO (h) Seq7 - PGO (i) Seq8 - PGO (j) Seq9 - PGO

Fig. 6: Comparison of algorithm performance across Seqs5 - Seqs9 sequences. Top row shows Mocap results, bottom row shows PGO results for the same sequences.

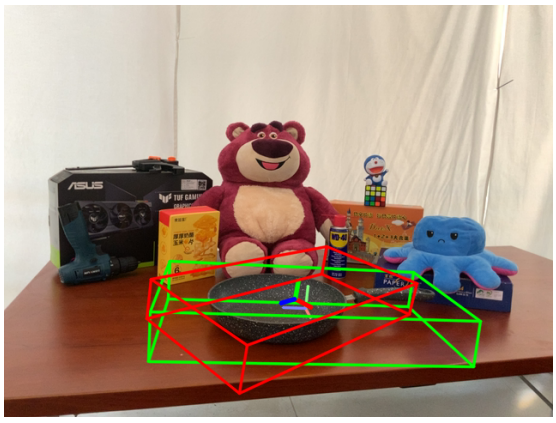


Fig. 7: FoundPose[2]* failure case.

Fig. 8: FoundationPose[1] failure case.

REAL275 [16] or HouseCat6D [17]. Our dataset captures novel viewpoints induced by object rotations along different axes while being held in hand—viewpoints that are difficult to observe in static scenes. These challenges manifest in pose

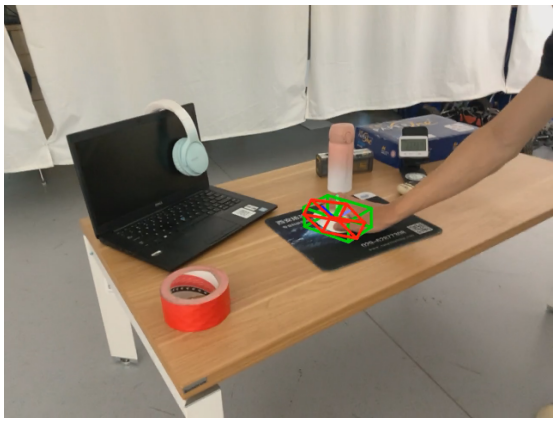
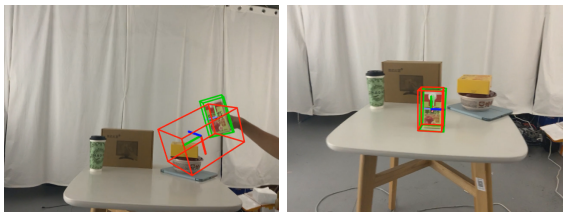


Fig. 9: GigaPose [4] failure case.



(a) MegaPose (RGB) (b) MegaPose (RGB-D)

Fig. 10: MegaPose[5] failure case.

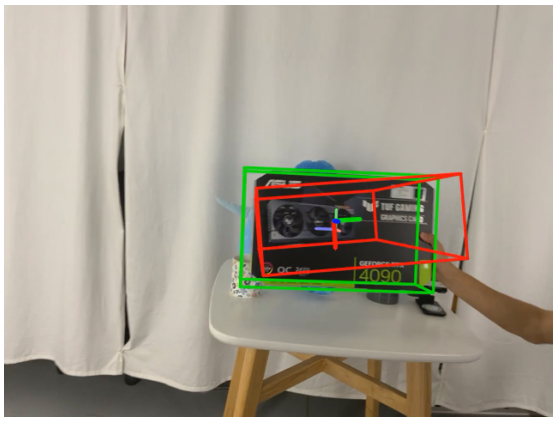


Fig. 11: SAM6D[6] failure case.

Fig. 12: MaskFusion [9] failure case.

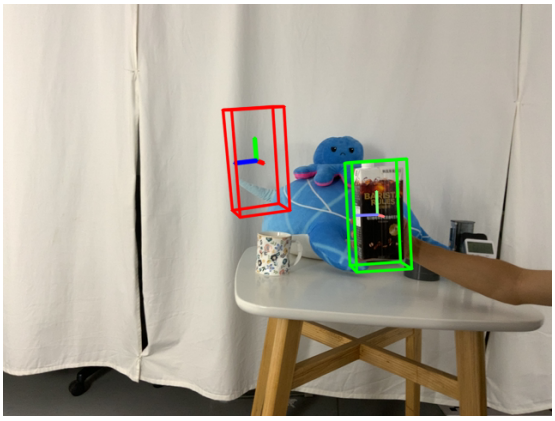


Fig. 13: BundleTrack [11] failure case.

Fig. 14: BundleSDF [12] failure case.

estimation failures for various methods, such as GCASP [18] when a bowl is lifted upward, AG-Pose [14] and GenPose [19] when the camera is moved upward, and DiffusionNOCS [20] when a can is lifted vertically. Furthermore, in our fine-tuning experiments (see Section III), it is evident that fine-tuning on our dataset leads to substantial improvements. For example, in Fig.17 and Fig.18, when comparing SecondPose [15] and AG-Pose [14] before and after fine-tuning, the previously mentioned pose estimation failures caused by vertical lifting

Fig. 15: DROID-SLAM [13] failure case.

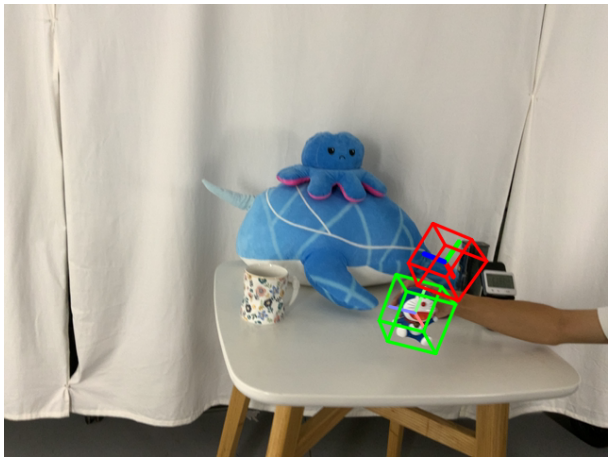
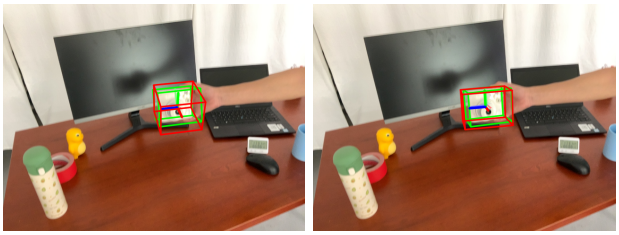


Fig. 16: FoundationPose [1] (tracking) failure case.



(a) Original AG-Pose [14]. (b) Fine-tuned AG-Pose [14].

Fig. 17: AG-Pose [14].

of objects are significantly mitigated.

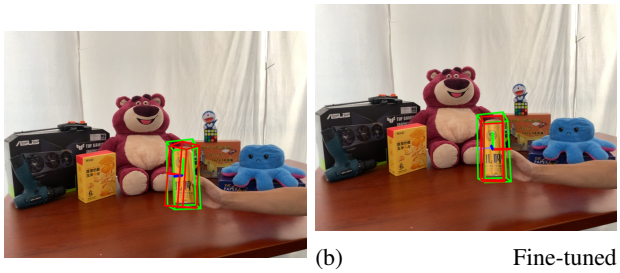
E. COPE methods failure cases.

[16] predicts normalized object points and then apply the Umeyama algorithm [25] to recover the object pose and size. However [25] requires high-quality NOCS map and depth map to recover accurate object pose and size, otherwise, its accuracy is limited. Note that in the original NOCS implementation, its object detection and NOCS map prediction module is coupled. For fairness in benchmarking, we adopt the settings from MV-ROPE [22], where consistent segmentation results similar to other benchmarked methods are provided as input.

Overall, the methods that rely on predicting a NOCS map, such as [16], [20], [22] tend to exhibit significantly higher translation errors compared to methods that directly predict translation, such as [14], [15], [18], [19], [23], [24]. This performance gap can be attributed to the fact that most of the latter methods, including [14], [15], [23], [24], perform translation prediction using PointNet-based architectures[26] and decouple it from rotation prediction. Compared to rotation estimation, translation prediction techniques are relatively more mature and stable.

Regarding COPE algorithms, we found that challenging cases tend to concentrate in two categories: the camera category and the mug category, so we focused our analysis there.

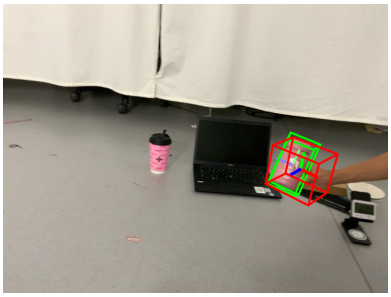
1) The camera category has long been a known challenge in COPE tasks due to the large intra-class variation among commonly seen camera models. Cameras in real-world scenarios exhibit a wide range of appearances, posing significant challenges for the generalization ability of pose estimation networks.
2) The mug category, by contrast, tends to have more consistent geometric shapes. However, failure cases often occur when a mug is being held by a human hand. Such occlusion frequently leads to errors in rotation estimation. Another possible cause of failure is when the mug's opening is not visible in the input image, under such conditions, it becomes difficult for the network to distinguish the upright orientation of the mug.



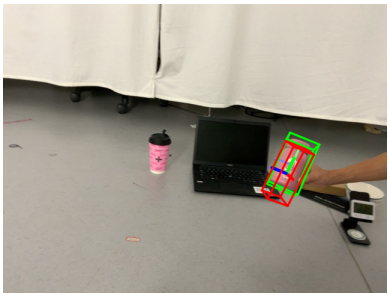
(a) Original SecondPose [15] (b) Fine-tuned SecondPose [15].

Fig. 18: SecondPose [15].

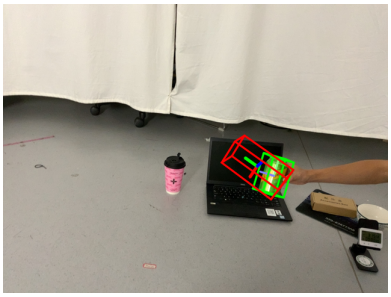
Another issue is that most benchmarked COPE methods are trained in NOCS [16] dataset, whose pose distribution is different from ours, which can be another reason for the failure cases in our test split.



(a) NOCS [21].

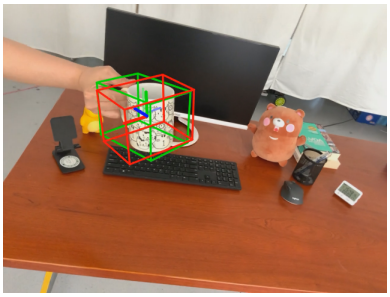


(b) NOCS (MV-ROPE) [22].

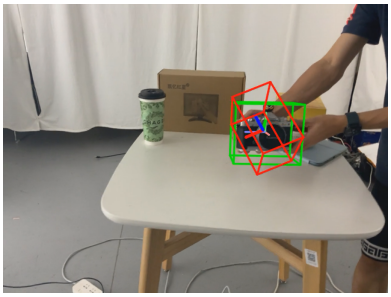


(c) DiffusionNOCS [20].

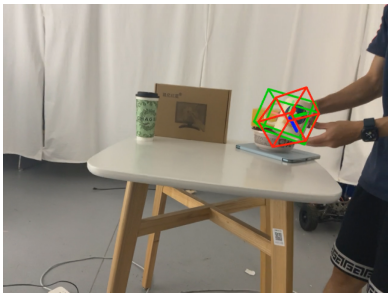
(d) AG-Pose [14].



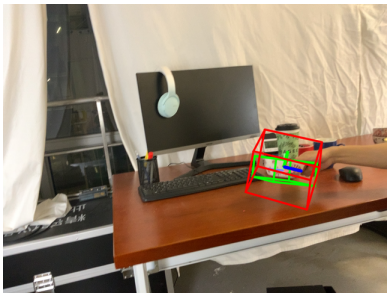
(e) SecondPose [15].



(f) IST-Net [23]



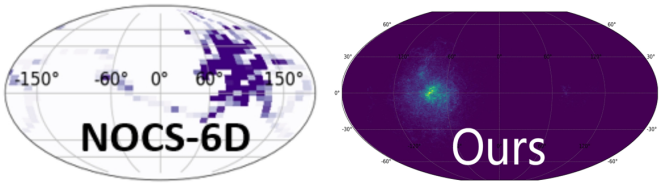
(g) VI-Net [24].



(h) GCASP [18].

(i) GenPose [19].

Fig. 19: COPE methods failure cases.



(a) (b)

Fig. 20: Viewpoint coverage map. Most of our benchmarked COPE algorithm is trained on NOCS-6D [21] dataset. Differ from NOCS-6D [16], our viewpoint coverage is more balanced between top-down and bottom-up views. Note that the left figure is referred to [17].

REFERENCES

[1] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.

[2] E. P. Örnek, Y. Labbé, B. Tekin, *et al.*, "Foundpose: Unseen object pose estimation with foundation features," in *European Conference on Computer Vision*, Springer, 2024, pp. 163–182.

[3] M. Oquab, T. Darcet, T. Moutakanni, *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[4] V. N. Nguyen, T. Groueix, M. Salzmann, and V. Lepetit, "Gigapose: Fast and robust novel object pose estimation via one correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 9903–9913.

[5] Y. Labbé, L. Manuelli, A. Mousavian, *et al.*, "Megapose: 6d pose estimation of novel objects via render & compare," in *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[6] J. Lin, L. Liu, D. Lu, and K. Jia, "Sam-6d: Segment anything model meets zero-shot 6d object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 27 906–27 916.

[7] A. Kirillov, E. Mintun, N. Ravi, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4015–4026.

[8] Z. Qin, H. Yu, C. Wang, *et al.*, "Geotransformer: Fast and robust point cloud registration with geometric transformer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 9806–9821, 2023.

[9] M. Runz, M. Buffier, and L. Agapito, "Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *2018 IEEE international symposium on mixed and augmented reality (ISMAR)*, IEEE, 2018, pp. 10–20.

[10] N. Ravi, V. Gabeur, Y.-T. Hu, *et al.*, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[11] B. Wen and K. Bekris, "Bundletrack: 6d pose tracking for novel objects without instance or category-level 3d models," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2021, pp. 8067–8074.

[12] B. Wen, J. Tremblay, V. Blukis, *et al.*, "Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 606–617.

[13] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in neural information processing systems*, vol. 34, pp. 16 558–16 569, 2021.

[14] X. Lin, W. Yang, Y. Gao, and T. Zhang, "Instance-adaptive and geometric-aware keypoint learning for category-level 6d object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 040–21 049.

[15] Y. Chen, Y. Di, G. Zhai, *et al.*, "Secondpose: Se (3)-consistent dual-stream feature fusion for category-level pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 9959–9969.

[16] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2642–2651.

[17] H. Jung, S.-C. Wu, P. Ruhkamp, *et al.*, "Housecat6d-a large-scale multi-modal category level 6d object perception dataset with household objects in realistic scenarios," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 22 498–22 508.

[18] G. Li, Y. Li, Z. Ye, *et al.*, "Generative category-level shape and pose estimation with semantic primitives," in *Conference on Robot Learning*, PMLR, 2023, pp. 1390–1400.

[19] J. Zhang, M. Wu, and H. Dong, "Generative category-level object pose estimation via diffusion models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 54 627–54 644, 2023.

[20] T. Ikeda, S. Zakharov, T. Ko, *et al.*, "Diffusionnocs: Managing symmetry and uncertainty in sim2real multi-modal category-level pose estimation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2024, pp. 7406–7413.

[21] C. Wang, D. Xu, Y. Zhu, *et al.*, "Densefusion: 6d object pose estimation by iterative dense fusion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3343–3352.

[22] J. Yang, Y. Chen, X. Meng, *et al.*, "Mv-rope: Multi-view constraints for robust category-level object pose and size estimation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2024, pp. 7588–7595.

[23] J. Liu, Y. Chen, X. Ye, and X. Qi, "Ist-net: Prior-free category-level pose estimation with implicit space transformation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 13 978–13 988.

[24] J. Lin, Z. Wei, Y. Zhang, and K. Jia, "Vi-net: Boosting category-level 6d object pose estimation via learning decoupled rotations on the spherical representations," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 14 001–14 011.

[25] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 13, no. 04, pp. 376–380, 1991.

[26] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.