Supplementary Materials of HouseCat6D

HyunJun Jung[1,*], Shun-Cheng Wu[1,*], Patrick Ruhkamp[1,*], Guangyao Zhai[1,2,*,†], Hannah Schieber[1,3,*], Giulia Rizzoli[4], Pengyuan Wang[1], Hongcheng Zhao[1], Lorenzo Garattoni[5], Sven Meier[5], Daniel Roth[1], Nassir Navab[1], Benjamin Busam[1,2,6]

[1] Technical University of Munich [2] Munich Center for Machine Learning
[3] FAU Erlangen-Nürnberg [4] University of Padova [5] Toyota Motor Europe [6] 3dwe.ai

Figure 1. **Object Meshes from Symmetric and Partially Symmetric Shape Categories.** Glass (a), bottle (b), can (c), and tube (d) categories are the categories with distinctive symmetry axes. We align the *y* axis to the axis of symmetry. If one surface is larger in area than another side, the *x*-axis is aligned in the perpendicular direction to it. All the objects are rendered in the same scale to highlight the size variance among the same category.

1. Object Meshes and Orientation

The HouseCat6D dataset features 194 highly diverse objects from 10 household object categories with different textures, sizes, and shapes. In this section, we show the meshes of the objects in each category and the descriptions of their orientation.

Glass HouseCat6D aligns the symmetry axis with the *y* axis for the (partially) symmetric objects. Glass objects in our dataset are fully symmetric around *y* axis in accordance with [12] who also align *y* axis and symmetry axis. The *x* and *z* axes serve as any orthogonal axes around the *y* axis as exemplified in Fig. 1 (a).

Bottle Unlike the glass objects, bottle objects in our dataset are sometimes not fully symmetric (i.e. frontal surface is wider than the side) as in Fig. 1 (b). In this case, we define the *x* axis perpendicular to the surface of larger area.

Can Similar to the bottle objects, can objects in our dataset sometimes are not fully symmetric (i.e. some cans are more square and one side is wider than the other side) as shown in Fig. 1 (c). Like the bottle objects, we define the *x* axis perpendicular to the wider side.

Tube Tube objects in our dataset are partially symmetric in shape, such that one side is round at the end while flat on the other side as shown in Fig. 1 (d). As in the can and bottle category, we define the *x* axis perpendicular to the wider side.

Teapot In general, teapots have the shape of one (partially) symmetric body with a handle and tip where the liq-

* Equal contributions
† Corresponding Author (e-mail: guangyao.zhai@tum.de).

Figure 2. **Object Meshes from (Partially) Symmetric Objects With a Handle.** Teapot (a) and cup (b) are the categories with objects that include a (partially) symmetric body with handle. We align the y axis with the symmetry axis of the body and the x axis with the direction from handle to the the other side of the body. All the objects are rendered in the same scale to highlight the size variance among the same category.

Figure 3. **Object Meshes from Flat Shape Categories**. Shoe (a), remote (b) and cutlery (c) are the categories with long, flat and non-symmetric shape. We oriented such shapes in a way that the y axis points in the direction of the upper side and x in the direction of the front side. All the objects are rendered in the same scale to highlight the size variance among the same category.

Figure 4. **Object Meshes for Box category.** Unlike the other categories, the sides of the box are rather defined by their texture. To allow networks to generalize in this category, we orient the meshes by their side length. We set y, x, z as direction of first, second and third longest side. All the objects are rendered in the same scale to highlight the size variance among the same category.

uid comes out. In our dataset, we use the y axis for the direction of the symmetric body and x axis for the direction from the handle to the tip as shown in Fig. 2 (a).

Cup For the cup category, we only use cups with handles that have the shape of one symmetric body with a handle. Thus, similar to the Teapot category, we align the y axis to the direction of the symmetric body and x with the direction from the handle to the other side of the body as shown in Fig. 2 (b).

Shoe Shoes, in general, have a long, flat and non-symmetric shape. For this category, we use only the right side of the slipper as illustrated in Fig. 3 (a). We oriented shoes such that their upper side points in the direction of the y axis and the front side points in the direction of the x axis.

Remote Remotes have relatively flat bodies with long and non-symmetric shapes, as shown in Fig. 3 (b). Similar to the shoe category, remotes are oriented such that their upper side points in the direction of the y axis, and the front side is is oriented in the direction of the x axis.

Cutlery Although the texture of the reflective surface makes a clear distinction between the cutlery category to any other category, the shape itself shares similarity with shoe and remote category. It is flat, long, and non-symmetric (Fig. 3 (c)). Thus, it shares the same orientation scheme, the upper side is aligned with the y axis and the front side points in x direction.

Box Unlike other categories, the sides of the box are defined by their texture. Even a human observer has to inspect the textures on multiple sides of a box to judge which side

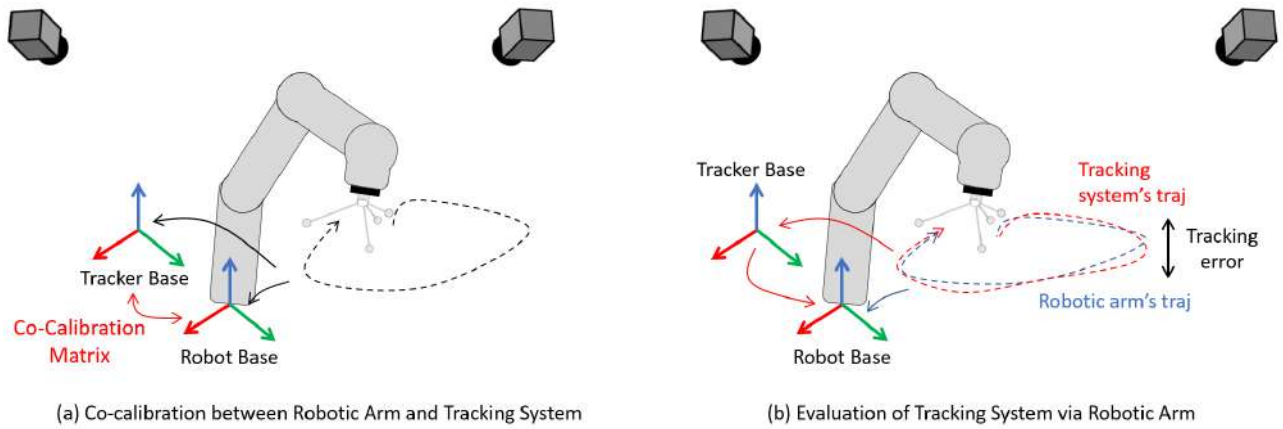


Figure 5. **Tracking System Evaluation.** We use a robotic arm to evaluate the quality of the tracking system. We first (a) co-calibrate the robot and the tracking system such that they share a common reference frame and then (b) run an example trajectory to calculate the difference between the trajectory obtained from the robot and the tracking system for error evaluation.

is the front or upper side *etc*. To make it easier for networks to generalize the orientation of boxes, we orient them by the length of the sides independent of their textures. We use *y*, *x*, *z* for the direction of the first, second, and third longest side as shown in Fig. 4.

2. Hardware Details

In this section, we provide detailed information about the hardware we used for the dataset acquisition.

3D Scanning As shown in Sec.1, our dataset comprises of 10 household categories such as bottle, box, can, cup, cutlery, glass, remote, shoe, teapot, tube. To ensure the high quality meshes we use 3D scanner equipped with turn table and structured light stereo system (EinScan-SP 3D Scanner, SHINING 3D Tech. Co., Ltd., Hangzhou, China), which produces single shot accuracy of ≤ 0.05 mm in a scanning volume of $1200 \times 1200 \times 1200$ mm^3. For photometrically challenging categories like cutlery and glass, self-vanishing 3D scanning spray (AESUB Blue, Aesub, Recklinghausen, Germany) is applied prior to the scanning.

External Tracking System. To ensure broad viewpoint coverage with high-quality annotation without using a checkerboard, we utilize an external tracker system composed of 4 (2x Stereo) ARTTRACK2 cameras (Advanced Realtime Tracking GmbH & Co, Germany) with built-in infrared flash (NIR, 880 nm) and maximum tracking distance of 4.5 m for both object pose and camera pose annotation.

Cameras. Our multi-modal dataset comprises two main modalities: Polarimetric RGB image and active stereo depth. A Phoenix 5.0 MP Polarization camera with Sony IMX264MYR CMOS Polarsens (PHX050S1-QC, LUCID Vision Labs, Inc., Canada) sensor is used to produce the RGB+P images, and Intel RealSense D435 (RealSense D435i, Intel, USA) acquires the depth maps. We specifically choose D435 as the depth sensor over Time-of-Flight sensors as active stereo depth provides, in general, more robust depth on photometrically challenging material [8]. To ensure the best synchronization between the two cameras, we use an external tracking signal provided by a Raspberry Pi (Raspberry Pi Foundation, United Kingdom) with GPIO output and later use the trigger signal as the timestamp of images for post-ex synchronization correction with the tracking system.

3. External Tracking System Evaluation

As mentioned in Sec. 3.2 in the main paper, we evaluate our IR-based external tracking system ARTTRACK2 via a robotic arm. We use a KUKA LBR iiwa 7 R800 (KUKA Roboter GmbH, Augsburg, Germany), a 7 DoF robotic arm certified for industrial use to provide ± 0.1 mm positional reproducibility, as the device to produce the ground truth pose for the comparison. In this section, we describe the detailed steps for the evaluation.

3.1. Robot-Tracker Co-Calibration

The first step to evaluate the tracking system with a robot is to co-calibrate the base of the robot and the tracking system. For this, we attach the calibrated IR tracking body on the robotic End-Effector (EE) as shown in Fig. 5 (a). We then acquire one trajectory from two different coordinate bases, one from the Robot base and the other one from the Tracker base. Similar to hand-eye calibration, we extract the static transformation between the two trajectories using

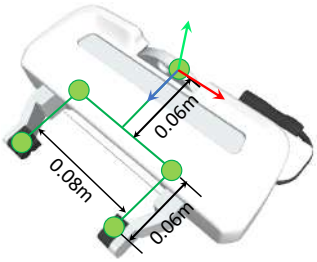


Figure 6. **The model of the parallel-jaw gripper**, whose finger depth is $0.04m$, maximum grasping width is $0.08m$, and the distance between the gripper base and the center of two fingers' base is $0.04m$.

the method of Horn [6]. In this case, the static transformation matrix is the transformation between Tracker Base and Robot Base (marked red in Fig. 5 (a)).

3.2. Trajectory Error Evaluation

After co-calibration, we keep the tracking body on the robotic EE and make an evaluation trajectory that replicates the trajectory in one of the scenes. We repeat the trajectory twice, once with the robot stopping at every capturing position and once with the robot not stopping during the pose capture. The first trajectory serves as an evaluation for the tracking system accuracy in the static case, and the later trajectory serves as an evaluation in the dynamic case. As it is possible to obtain the pose of the tracking body from both, robot and tracking system, in the same coordinate frame using the co-calibration matrix, the error of the tracking system is calculated as the pose difference between the pose from the robotic arm and the pose from the tracking system (Fig. 5 (b)). We measure an error of 0.67 mm / $0.12°$ in the static case and 0.92 mm / $0.16°$ in the dynamic case.

4. Grasping Annotation Pipeline

In this section, we detail the grasping annotation process. The pipeline is illustrated in Fig. 7. For each scene, we first obtain the scene by reconstructing the background (e.g. table) with multiview depth and displacing the object meshes on the top of the background mesh according to their pose. After successfully reconstructing the scene, the meshes are sent to the antipodal sampling module to generate grasp candidates (Fig. 7.a). Then Isaac Gym [11] sorts out the good grasps among all candidates for each object by checking if grasping an object failed. Successful grasps are in green, while failed grasps are in red (Fig. 7.b). Then objects are projected to the tracker base along with their associated grasps to check the collisions and collided grasps are removed from the original ones. Finally, we project these checked grasps to each image base to obtain the ultimate dataset. (Fig. 7.c).

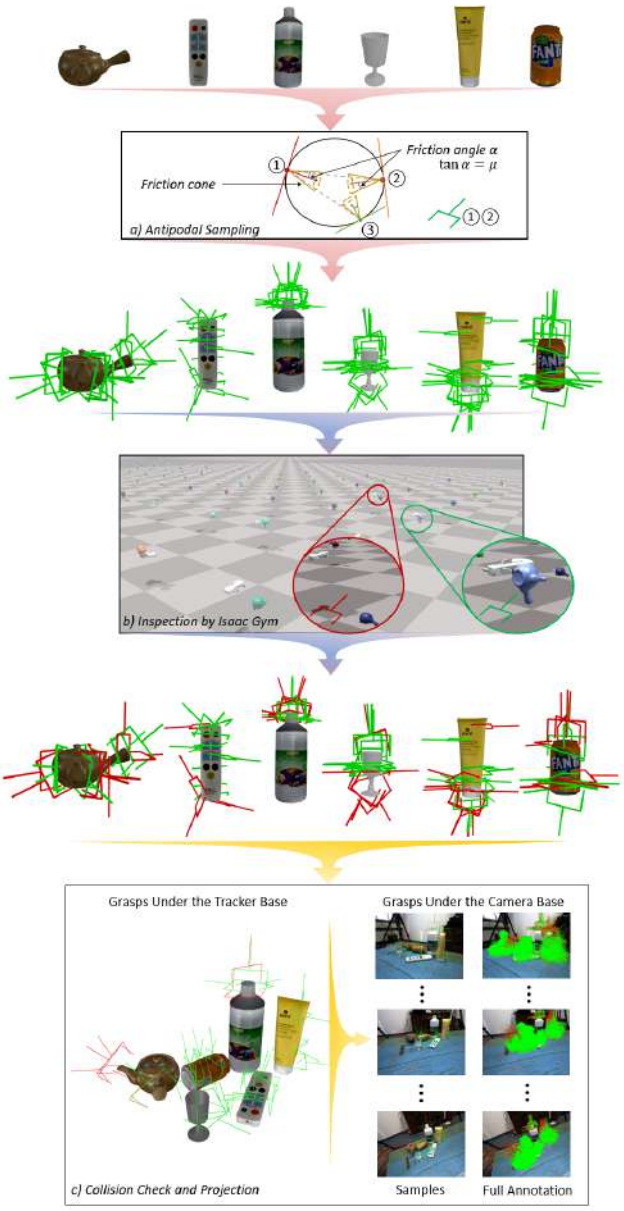


Figure 7. **The pipeline of the grasp annotation process.** We show downsampled grasps for better visualization and show the full annotation at the end for the final performance.

4.1. Scene Mesh Acquisition

To annotate the correct grasping position with collision inspection, it is important to have a full mesh of the scene, which contains objects as well as their platform where the object are placed, such that physical simulation can filter out the grasping points which leads collision of gripper on the other objects and the background. For the objects, we displaced their meshes in the scene with the annotated poses. On the other hand, for the platform, it is not possi-

(a) Annotated 3D Meshes of the scene22 (b) Example of RGB frame of scene22

Figure 8. **Example of Mesh Annotation and Its Corresponding RGB Image.** We annotate the scene by reconstructing the platform and displacing the objects' meshes with their pose. Note that we do not reconstruct the other parts of the background such as the wall as they are not necessary for grasping simulation.

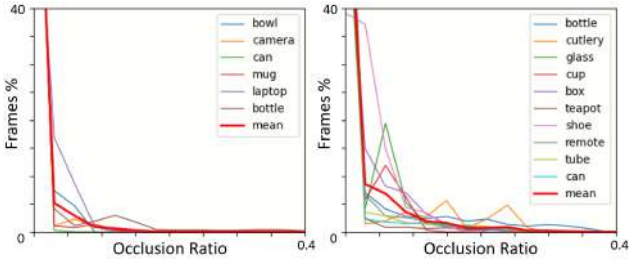


Figure 9. **Occlusion Comparison between NOCS [12] and HouseCat6D.** HouseCat6D covers more occlusions as well as more frequency on the occlusion, which makes the dataset more challenging as well as closer to the real-life scenario.

ble to do the same way as the background is not scanned prior. Instead, we reconstruct the scene with the depth images with the corresponding camera poses using truncated signed distance fusion and hole followed by manual hole filling with Artec Studio 17 Professional (Artec3D, Senningerberg, Luxembourg). An example of the 3D mesh of objects with the reconstruction of the platform is shown in Fig. 8 with an example of an RGB frame from the corresponding scene.

4.2. Antipodal Sampling

Antipodal sampling is a wide-used technique for grasp pose generation, which has been investigated in several previous works [4, 10, 14]. Given an object mesh, this scheme first samples an arbitrary point on the mesh surface as the initial contact point (①) together with a line within a range around the surface normal. The sampling threshold $\mu = \tan(\alpha)$ with the friction angle α restricts the range at which rays can be emitted. A second point (② / ③) is found as the intersection of both mesh and line. Then reject sampling is used to prune the point whose line is not inside the friction cone (③) or whose distance from the initial point is beyond the max width of the gripper model. A successfully sampled grasp G_{obj} is then derived by taking the center point between two contact points (①②) and a randomly sampled rotation around the line. Here, in this work, we set μ as 0.4. The end-effector model we use is a Franka Emika parallel-jaw gripper, as shown in Fig. 6.

4.3. Simulation Inspectation

After obtaining grasp samples, we use a physical engine, namely Isaac Gym, to inspect grasps which are successful. For each object, we parallelly create the same number of simulation environments as of grasps belonging to the object. We inspect whether these grasps are successful by calculating the distance between the gripper and the centroid of the object model 15 seconds after the finger closure defined by individual grasping width. If the distance is less than $0.1m$, we label this grasp as a successful one and vice versa.

4.4. Grasp Projection

This is a two-stage procedure. We retrieve objects in each scene and replicate the first-stage projection for all objects in the scene, where we transform the grasps belonging to an object to the tracker base according to the object pose and check their collisions with the surrounding meshes, including other objects and the background. The collision checking module is from the public library Trimesh. Then we project all grasps to every image frame to obtain the final dataset, utilizing the camera trajectory recorded under the tracker base.

5. Occlusion Analysis

When it comes to detecting the objects and estimating the pose, occlusion and visibility take important roles. In our dataset, we provide the visibility ratio of each category in the scene per frame. The visibility is calculated as follows. Firstly, we render the mask of an object with a given pose, one object per time to prevent occlusion between categories, and count number of pixels in the mask M^{cat}_{full}. Then masks of categories are rendered again but all together so that occlusion is accounted, followed by counting the number of pixels on each object $M^{cat}_{occluded}$, which now has fewer pixels due to occlusion from other objects. Occlusion ratio is calculated as $M^{cat}_{occluded}$ / M^{cat}_{full}, which then averaged over all frames and scenes. We show the ratio on our dataset and as well as on NOCS dataset [12] in Fig. 9 to emphasize the difference in terms of the occlusion in the dataset.

6. Evaluation on Rotation Translation Metric

In Tab. 1, we show the evaluation of baseline on rotation and translation error metric with a set of thresholds: 10°5cm. Similar to 3D IOU, NOCS [7] performs significantly worse

https://trimsh.org/trimesh.collision.html

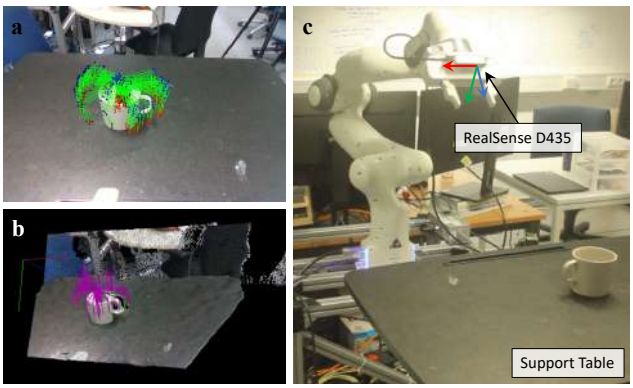


Figure 10. **An example of a real-world grasping trial.** (a) 2D grasp keypoints in the camera view. (b) 3D grasps visualization, with the best in green and the rest in purple. (c) Hardware setup in the third-person view.

Table 1. **Quantitative Evaluation on Rotation and Translation Metric.** For rotation and translation metric, we show average accuracy over all categories.

Threshold	NOCS [12]	GPV-Pose [3]	FS-Net [1]	VI-Net [9]
10°5cm	4.8	22.7	21.6	29.1

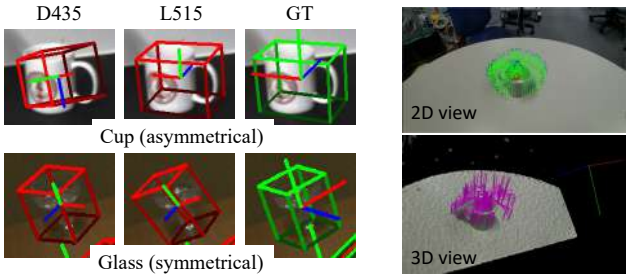
in our dataset compared to other baselines. As mentioned in the main paper, we suspect this is due to issue with inaccurate depth being used for lifting NOCS [7] prediction in 2D into 3D. On the other hand, geometric guided approach such as GPV-Pose [3], FS-Net [1] and VI-Net [9] has better performance with ground truth detection mask. Especially, with better parameterization on rotation, VI-Net [9] performs significantly better compared to other geometric approach, GPV-Pose [3] FS-Net [1].

7. Cross-Dataset Evaluation

HouseCat6D provides depth data from D435 solely. Thus, it is important to show that methods trained on HouseCat6D can generalize with other depth information, as there is a domain gap between an active stereo (D435) and a D-ToF/LiDAR (L515) sensor, as the principle of depth measurement is different. **(i)** To quantitatively evaluate the category-level pose estimation gap, we test HouseCat6D-trained VI-Net on 11 sequences of the HAMMER dataset [8], which is designed for depth estimation and provides depth data for both cameras. We managed to obtain ground truth object poses from MonoGraspNet [15] and report the same metrics as the above ones, with 5° 2 cm being the average.

Depth	Bottle	Can	Cup	Cutlery	Glass	Avg.	5° 2 cm
D435	28.8	**49.9**	72.1	35.1	**83.6**	**53.9**	3.8
L515	**56.6**	17.6	**92.7**	**38.7**	33.6	47.8	**6.6**

We show that, for objects such as bottles and cups, the L515 provides better results than the D435, even though the network is trained with D435. This is because depth quality is better when measuring these categories with L515 [8]. However, unlike active stereo, ToF cameras, including the L515, produce depth maps with strong artifacts on photometrically challenging material, such as transparent and reflective objects [8], resulting in severely degraded performance on the glass, cutlery, and cans. We show some visualization examples when VI-Net is equipped with different depth data in Figure 11.a.



(a) Pose estimation on HAMMER (b) Grasp inference on L515

Figure 11. **Zero-shot pose and grasp inference.** (Zoom for details)

8. Polarization-Based Pose Estimation

There is no such obvious method for category-level pose estimation in the community. Although PPPNet [5] proves that polarization indeed helps in the instance-level task, the lack of a large-scale category-level dataset with polarization modality has so far prohibited research in this direction. To promote this research path, we train a baseline upon VI-Net [9] with additional physical priors calculated from four polarization images, such as DOLP ($H \times W \times 1$) and AOLP ($H \times W \times 1$), that are concatenated on the RGB image along the last dimension to have multi-modality input ($H \times W \times 5$). Then, we feed it to the VI-Net architecture to regress rotations, translations, and sizes. The baseline can be found on the Github . We report 5 categories with better results on IoU_{50}. We show that including polarization significantly improves the performance on short objects, such as cans and remotes, while marginal performance gains can be seen for challenging categories, such as cutlery and glass. Inspecting the transformation metric 5° 2 cm provides a similar conclusion. Again, it is important to note that this area is under-explored, and this baseline implementation is intuitive but basic. However, the improvement we show in this experiment proves the usefulness of the polarization modality and promotes more research in this direction.

Method	Metric	Bottle	Can	Remote	Cutlery	Glass	Avg.
VI-Net	IoU_{50}	79.6	67.0	17.1	76.4	93.7	66.8
VI-Net *w/* pol		**80.2**	**79.9**	**43.4**	**76.8**	**94.3**	**74.9**
VI-Net	5° 2 cm	**38.9**	9.6	1.3	0.2	28.9	15.8
VI-Net *w/* pol		32.5	**13.1**	**7.5**	**0.7**	**30.0**	**16.8**

https://github.com/Junggy/HouseCat6D

Table 2. **Ablation Study on Different Input** Class-wise evaluation of 3D IoU (at 25%/ at 50%) for VI-Net [9] with different training setup.

Approach	Train Set	$3D_{25}$ / $3D_{50}$	Bottle	Box	Can	Cup	Remote	Teapot	Cutlery	Glass	Tube	Shoe
VI-Net [9]	Full	80.7 / 56.4	90.6 / 79.6	44.8 / 12.7	99.0 / 67.0	96.7 / 72.1	54.9 / 17.1	52.6 / 47.3	89.2 / 76.4	**99.1 / 93.7**	**94.9 / 36.0**	85.2 / 62.4
	RV	74.2 / 46.8	91.0 / 76.6	59.1 / 23.5	98.9 / 67.2	76.0 / 36.6	59.4 / 34.3	22.7 / 18.8	79.4 / 57.3	97.7 / 85.3	66.3 / 47.8	91.4 / 20.4
	RS	67.7 / 35.8	90.1 / 68.7	49.0 / 9.8	96.9 / 53.6	87.2 / 48.5	40.2 / 16.3	28.8 / 15.8	67.4 / 49.0	98.5 / 73.6	86.6 / 7.9	32.4 / 14.9

9. Real-World Grasping

Unlike the experiments in simulation which are conducted on the available test set, real-world grasping is more challenging with respect to two facts. First, the objects are more random and are not included in the dataset, with some of them even in the unseen categories, which tests the generalization ability of the network. Second, the appearance of the backgrounds are more complex and the imaging style is also different since the imagery sensor is different from the one collecting the dataset, which tests the robustness of the network.

Hardware Setup We test the trained KGN [2] in real-world scenarios using a 7-DoF Franka Panda robot with a parallel-jaw gripper as the end-effector. The sensor mounted on the gripper base is a RealSense D435 RGB-D camera. The framework is run on an NVIDIA A4000 GPU.

Implementation Details We randomly select support tables with unseen backgrounds as the grasping environment. Then we fix a certain sequence of joint positions for the robot as the home position where the camera observes the table from the side, as shown in Fig. 10.c. We select three types of objects for the test–1) normal objects in the seen categories, 2) normal objects in the unseen categories, and 3) photometrically challenging objects in the seen categories, whose grasp success rates are reported in the main paper. For example, in the first type, we let the robot grasp a cup, shown in Fig. 10. KGN [2] starts to infer 2D grasp keypoints on the image (Fig. 10.a), then it utilizes PnP and 3D keypoints shown in Fig. 6 with camera intrinsics to solve 3D grasp poses (Fig. 10.b).

Cross-Sensor Experiment We also test HouseCat6D-trained KGN [2] with a Franka robot mounting an L515 to conduct a grasping task. Following the procedure in the manuscript, we grasp each category in 15 trials and report the success rate as follows:

Depth	Box	Cup	Glass	Remote	Unknown
D435	12/15	10/15	11/15	5/15	9/15
L515	10/15	10/15	6/15	7/15	7/15

The results indicate the usefulness of both sensors for the grasping task with material and depth quality-dependent performance. In Figure 11.b, we show a glimpse when KGN infers grasps for a cup. More examples have been involved in the upcoming supp. video. Among all attempts, we feel interested in some successful cases of glass grasping, even if the depth vanishes under L515. This success might be attributed to KGN's design, which basically approaches grasp estimation as multiple instance-level pose estimation of the gripper. In situations with unreliable depth data, KGN functions primarily as an RGB-based pose estimation method.

10. Ablation Study

We trained VI-Net [9] on our dataset with different setups, such as reduced viewpoint coverage of camera (RV), reduced number of scenes (fewer objects per category) (RS) to study the impact of different aspect of the dataset on category level 6d pose estimation task. The results are summarized in Tab. 2. For RV and RS setup, we specifically mimic the coverage of PhoCal [13] by using less number of scenes (RS) and selecting the subset of camera trajectory as continuous 250 frames of translation-dominated motion (RV).

Impact of View VS Scenes Compared to having reduced viewpoints (RV) during training, reducing the scene (RS) has a more negative impact on the test evaluation. As the main task of category-level pose estimation is about generalizing on the unseen objects of known categories, we find it beneficial to see more objects and backgrounds even if the viewpoint is limited. This further highlights the advantage of our dataset over NOCS dataset [12] and PhoCal dataset [13] for both the number of scenes and the number of objects. Furthermore, when both RS and RV are combined, there is a significant drop in the performance, which gives an advantage of our dataset over PhoCal [13], where the robotic arm annotations have a clear limitation on the viewpoint coverage as well as the number of scenes.

11. Dataset Sample

Fig. 12 shows example images of our dataset from all 41 scenes. In Fig. 12, we augment rendered object masks together with bounding boxes to highlight the quality of our dataset annotation. Training scenes are augmented with green, test scenes are augmented with yellow, validation scenes are augmented with orange color.

Figure 12. **Dataset Sample.** Our dataset is composed of 41 scenes with high-quality annotations structured in 34 training scenes (green), 5 test scenes (yellow), and 2 validation scenes (orange). We overlay rendered object masks as well as bounding boxes to highlight the quality of our dataset annotation.

References

[1] 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*, pages 1581–1590, 2021. 6

[2] Yiye Chen, Yunzhi Lin, Ruinian Xu, and Patricio A Vela. Keypoint-graspnet: Keypoint-based 6-dof grasp generation from the monocular rgb-d input. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7988–7995. IEEE, 2023. 7

[3] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6781–6791, 2022. 6

[4] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. Acronym: A large-scale grasp dataset based on simulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6222–6227. IEEE, 2021. 5

[5] Daoyi Gao, Yitong Li, Patrick Ruhkamp, Iuliia Skobleva, Magdalena Wysock, HyunJun Jung, Pengyuan Wang, Arturo Guridi, and Benjamin Busam. Polarimetric pose prediction, 2021. 6

[6] Berthold Horn, Hugh Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A*, 5:1127–1135, 1988. 4

[7] Muhammad Zubair Irshad, Thomas Kollar, Michael Laskey, Kevin Stone, and Zsolt Kira. Centersnap: Single-shot multi-object 3d shape reconstruction and categorical 6d pose and size estimation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10632–10640, 2022. 5, 6

[8] HyunJun Jung, Patrick Ruhkamp, Guangyao Zhai, Nikolas Brasch, Yitong Li, Yannick Verdie, Jifei Song, Yiren Zhou, Anil Armagan, Slobodan Ilic, et al. On the importance of accurate geometry data for dense 3d vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 780–791, 2023. 3, 6

[9] Jiehong Lin, Zewei Wei, Yabin Zhang, and Kui 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*, pages 14001–14011, 2023. 6, 7

[10] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Robotics: Science and Systems (RSS)*, 2017. 5

[11] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021. 4

[12] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019. 1, 5, 6, 7

[13] Pengyuan Wang, HyunJun Jung, Yitong Li, Siyuan Shen, Rahul Parthasarathy Srikanth, Lorenzo Garattoni, Sven Meier, Nassir Navab, and Benjamin Busam. Phocal: A multi-modal dataset for category-level object pose estimation with photometrically challenging objects. In *CVPR*, 2022. 7

[14] Guangyao Zhai, Yu Zheng, Ziwei Xu, Xin Kong, Yong Liu, Benjamin Busam, Yi Ren, Nassir Navab, and Zhengyou Zhang. Da^2 dataset: Toward dexterity-aware dual-arm grasping. *IEEE Robotics and Automation Letters*, 7(4):8941–8948, 2022. 5

[15] Guangyao Zhai, Dianye Huang, Shun-Cheng Wu, HyunJun Jung, Yan Di, Fabian Manhardt, Federico Tombari, Nassir Navab, and Benjamin Busam. Monograspnet: 6-dof grasping with a single rgb image. In *IEEE International Conference on Robotics and Automation*. IEEE, 2023. 6