FS6D: Few-Shot 6D Pose Estimation of Novel Objects Supplementary Material

Yisheng He[1] Yao Wang[2] Haoqiang Fan[2] Jian Sun[2] Qifeng Chen[1]
[1]Hong Kong University of Science and Technology [2]Megvii Technology

1. More Results

Model running time. We run our model on a single NVIDIA GeForce RTX 2080Ti GPU. For each support view, it takes an average time of 72 ms for neural network forwarding and 113 ms for pose alignment using the Umeyama algorithm [7] with RANSAC [3].

Pose estimation with ground-truth segmentation. In the main paper, we utilize relaxed ground-truth object bounding boxes to crop out regions of interested objects from the query scene for pose estimation. While LatentFusion [5] utilizes stricter ground-truth segmentation to segment out objects, we report our results on LineMOD dataset following their setting. Specifically, We use our model trained only with ShapeNet6D without fine-tuning on the real LineMOD dataset. As is shown in Table 6, our model without any refinement already surpasses iterative refined LatentFusion. Equipped with post-refinement by ICP, our model obtains further improvement. Moreover, our model (0.34 fps) is 18X faster than LantentFusion (0.018 fps) on a RTX 2080Ti, when both use 16 support views.

Effect of the different number of support views. We ablate the effect of the different number of support views in Table 9. As is shown in the table, our algorithm gets better performances when the number of support views increases. Moreover, it only gains margin performance when we have more than 16 views, which shows that our algorithm does not need too many support views and can get good pose results under the few-shot setting.

Details results on the LineMOD dataset. See Table 7.

Visualization of ShapeNet6D Example images in ShapeNet6D are shown in Fig. 6.

2. Implementation Details

Grouping information of benchmark datasets. We split the LineMOD dataset into three groups. Objects in different groups have no intersection. During network fine-tuning, we select two groups for training and one group as novel objects for testing. The group information of the LineMOD dataset is shown in Table 10. We split the YCB-Video dataset into three groups in a similar way. Group information of the YCB-Video dataset is shown in Table 8.

Support views selection. We select support views from the training set since we do not have the real-world objects in the LineMOD and YCB-Video datasets to capture the support views. We select 16 support views using the farthest rotation sampling for each object to ensure that each part of the object is visible. Specifically, we initialize the set of selected views with a random view from the training set for each object. We then add another object view with the farthest rotation distance from views in the selected set. We repeat this procedure until 16 views of the target object are obtained. We define the distance between two rotations as the Euclidean distance between two unit quaternions following [4, 6]. The formula is as:

$$D(q_1, q_2) = \min\{||q_1 - q_2||, ||q_1 + q_2||\}. \quad (5)$$

where $||\cdot||$ denotes the Euclidean norm and q_1, q_2 the two unit quaternions.

Given the target object's mask labels and pose parameters in the selected support views, we crop out the object region and transform the object point cloud back to the object coordinate system to serve as a reference frame to define the 6D object pose.

3. Fast Registration of Novel Objects

Given a novel object and an RGBD sensor with known intrinsic parameters, we can quickly obtain support views of the novel object in several ways. We provide some examples as follows:

Select from an RGBD video of the novel object. The most simple way is to select support views from an RGBD video of the target object. Specifically, we first place the target object in the center of a clean plane and then capture a video by slowly moving the camera around the object. We use the first frame to define the object coordinate system. Specifically, we mask out the object region by removing the background plane with a plane detection algorithm [2] or least-square-fitting of a plane on the scene point cloud. We define the object coordinate system based on the object point cloud of the first frame. Then, we calculate the pose between the following frame and the first frame. Since the pose difference between adjacent frames of a video is small and the scene background is a clean plane, we can utilize

registration algorithms, i.e., ICP [1], Go-ICP [8] to calculate the relative pose parameters between adjacent frames and obtain the pose parameters between each frame and the first frame. Finally, we can select support views by the farthest rotation sampling algorithm as in Section 2.

To further improve the accuracy of relative pose parameters, we can put the object on a marker board (a plane with several markers on it) and utilize markers to obtain more accurate relative poses.

Collect with the robot arm. For robotic manipulation, we have a robot arm with a camera in hand. We first calibrate the robot arm and the camera between an observed region with a marker board. We define several viewing points with known pose parameters. We then place the novel target object to the observed region and utilize the robot arm to move the camera to those predefined viewing points to capture support views of the novel objects. The pose parameters of support views will be more accurate due to the robustness of the robotic manipulation system.

References

[1] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 2

[2] Chen Feng, Yuichi Taguchi, and Vineet R Kamat. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6218–6225. IEEE, 2014. 1

[3] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 1

[4] Du Q Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, 2009. 1

[5] Keunhong Park, Arsalan Mousavian, Yu Xiang, and Dieter Fox. Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10710–10719, 2020. 1, 3

[6] B Ravani and B Roth. Motion synthesis using kinematic mappings. 1983. 1

[7] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Computer Architecture Letters*, 13(04):376–380, 1991. 1

[8] Jiaolong Yang, Hongdong Li, Dylan Campbell, and Yunde Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE transactions on pattern analysis and machine intelligence*, 38(11):2241–2254, 2015. 2

Figure 6. Example scene images in our ShapeNet6D dataset.

		ape	bench.	camera	can	cat	driller	duck	**eggbox**	**glue**	holep.	iron	lamp	phone	Mean
w/o ref.	Ours	74.0	86.0	88.5	86.0	**98.5**	81.0	68.5	**100.0**	**99.5**	**97.0**	**92.5**	85.0	**99.0**	88.9
w/ ref.	LatentFusion [5]	**88.0**	**92.4**	74.4	88.8	94.5	91.7	68.1	96.3	94.9	82.1	74.6	94.7	91.5	87.1
	Ours+ICP	78.0	88.5	**91.0**	**89.5**	97.5	**92.0**	**75.5**	99.5	**99.5**	96.0	87.5	**97.0**	97.5	**91.5**

Table 6. Quantitative evaluation of different few-shot (16 shots) 6D pose estimation on the LineMOD dataset with ground truth segmentation. w/o ref.: without iterative refinement; w/ ref.: with iterative refinement. Symmetry objects are in bold.

Group 0						Group 1					Group 2				
ape	benchvise	camera	can	cat	mean	driller	duck	**eggbox**	**glue**	mean	holepuncher	iron	lamp	phone	mean
70.5	82.5	72.5	46.5	78.0	70.0	87.0	60.5	100.0	99.5	86.8	94.0	88.0	94.5	97.0	83.4

Table 7. Detailed results of our method on the LineMOD dataset. Symmetry objects are in bold.

Group	Objects
0	002 master chef can, 003 cracker box, 004 sugar box, 005 tomato soup can, 006 mustard bottle, 007 tuna fish can, 008 pudding box
1	009 gelatin box, 010 potted meat can, 011 banana, 019 pitcher base, 021 bleach cleanser, 024 bowl, 025 mug
2	035 power drill, 036 wood block, 037 scissors, 040 large marker, 051 large clamp, 052 extra large clamp, 061 foam brick

Table 8. Group information of the YCB-Video dataset.

# Views	1	4	8	16	32
ADDS AUC↑	79.6	87.3	87.9	88.4	88.6

Table 9. Effect of number of support views on the YCB-Video. The mean ADD-S AUC results are reported.

Group	Objects
0	ape, benchvise, camera, can, cat
1	driller, duck, eggbox, glue
2	holepuncher, iron, lamp, phone

Table 10. Group information of the LineMOD dataset.