ReAgent: Point Cloud Registration using Imitation and Reinforcement Learning
Supplementary Material

Dominik Bauer, Timothy Patten and Markus Vincze
TU Wien
Vienna, Austria
{bauer,patten,vincze}@acin.tuwien.ac.at

In the following, we provide detailed results for the experiments on ModelNet40 and ScanObjectNN in Section 1. We show the application of our approach to the object pose estimation task on the LINEMOD dataset in Section 2. In addition, a detailed view on the architecture and definitions of the used metrics are given in Sections 3 and 4.

1. Details of Experiments

This section provides a more detailed analysis of the main results. For all methods, we use the same settings and models as in Section 4 of the main text.

1.1. ADI Precision-Recall Curves

Figure 1 shows the underlying precision-recall curves that were used to compute the ADI AUC metric for the held-out ModelNet40 models in Table 1 (left) in the main text.

As shown in the zoom-ins on the threshold range [0, 1], both classical approaches achieve highest recall for very low ADI precision thresholds. The prediction of the learning-based approaches seems to always introduce a slight misalignment. Moreoever, FGR performs best overall on ScanObjectNN, while ICP never achieves 100% recall.

The relative deterioration over datasets of PointNetLK is similar to our approach, presumably stemming from the similar embedding. DCP-v2 generally performs worst of the learning-based approaches, which, we assume, is due to imperfect correspondences in all settings. Overall, we observe that DCP-v2 is able to achieve similar top recall, but is weaker on tight precision thresholds. We hypothesize that this is due to the one-shot nature of DCP-v2, whereas the iterative approaches may further refine their predictions.

Both ReAgent versions outperform the compared learning-based methods in all conditions. Above about $0.6\%d$, ReAgent performs best on synthetic data. ReAgent (IL+RL) outperforms the IL-only variant on held-out models above about $1\%d$. The farther the dataset deviates from the training conditions, the more ReAgent's (IL+RL) performance deteriorates as compared to ReAgent (IL).

1.2. Transformation and Noise Magnitude

The magnitude of the initial transformations and the noise in Section 4 of the main text are in-line with previous work [8, 10]. To show how the compared methods are affected by varying noise and initial conditions, we provide an ablation study in Figure 2. We evaluate using the held-out ModelNet40 models as in Table 1 (left) in the main text.

Noise Magnitude: In Figure 2a, the standard deviation of the distribution from which the noise is sampled is varied. The mean of the distribution remains constant at 0 and clipping remains constant at 0.1; increased from 0.05 as compared to the main experiments.

FGR performs best in the noise-free condition but is heavily affected by increasing noise magnitude. Comparing the trend of the mean values, the other methods are similarly affected with ReAgent retaining overall best performance over all magnitudes.

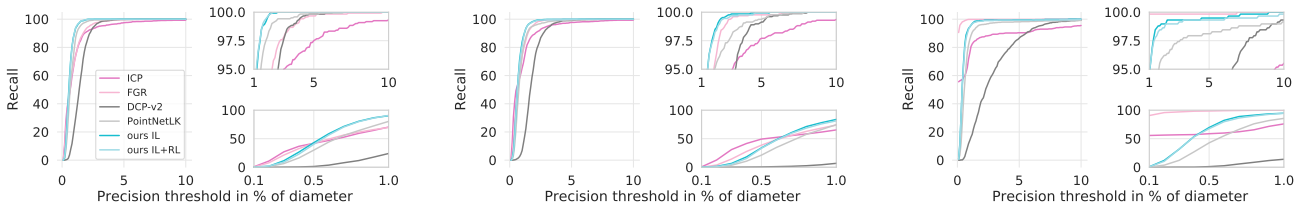
Initial Transformation: In Figures 2b-c, the respective component of the transformation is varied, while the other is kept as in the experiments in the main text. The magnitude of the transformation is varied by increasing the upper-bound parameter of the transformation distributions.

Figure 2b indicates that the PointNet-based methods are able to retain high accuracy within the range of initial rotations seen during training (up to $45\,\mathrm{deg}$ per axis). FGR, using the true surface normals of the models, is barely affected by increasing rotation magnitude.

In Figure 2c, the performance of ICP is heavily affected by the translation magnitude. This is due to a fixed upper-bound for the correspondence distance of 0.5. ReAgent also shows a slight decrease in accuracy with the highest translation magnitude, as its step sizes are limited and thus more iterations need to be spent for initial alignment.

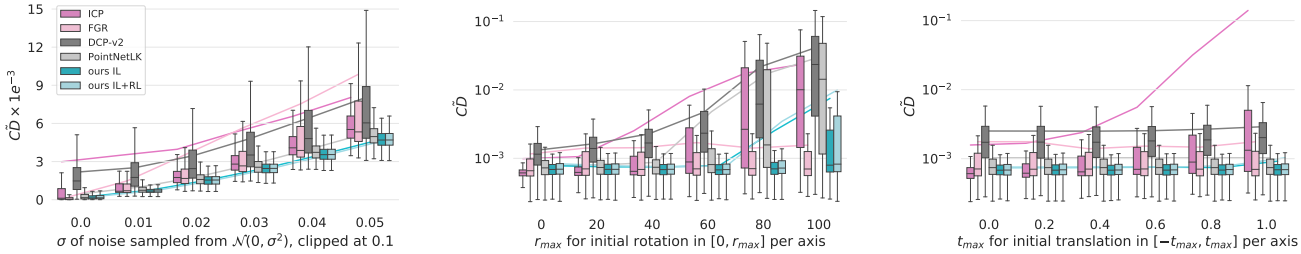
1.3. Step-by-Step Results

Table 1 shows the results of ReAgent (IL+RL) after each iteration. T_c indicates cumulative runtime for inference using a single observation. The conditions are identical to



(a) Held-out ModelNet40 models. (b) Held-out ModelNet40 categories. (c) ScanObjectNN.

Figure 1: Precision-recall curves for the ADI AUC results in Section 4 of the main text. Best viewed digitally.



(a) Varying noise magnitude. (b) Varying initial rotation. (c) Varying initial translation.

Figure 2: Results on held-out ModelNet40 models with varying noise. The boxes show $[q_{.25}, q_{.75}]$ and the median value. The whiskers indicate $[q_{.25} - 1.5IQR, q_{.75} + 1.5IQR]$, with $IQR = q_{.75} - q_{.25}$. The colored lines show the trend of the respective mean values. Note that large differences between mean and median are due to outliers.

iter.	MAE (↓) R	MAE (↓) t	ISO (↓) R	ISO (↓) t	ADI (↑) AUC	$\tilde{CD}$ (↓) $\times 1e^{-3}$	T_c [ms]
init	22.35	0.238	44.49	0.478	3.4	225.6335	0
1	12.74	0.101	25.34	0.204	39.0	45.5217	3
2	6.31	0.050	12.17	0.100	69.9	10.7966	5
ICP	3.59	0.028	7.81	0.063	90.6	3.4882	9
3	3.32	0.025	6.46	0.052	83.7	2.9290	7
4	2.02	0.015	3.98	0.032	89.7	1.3366	9
PNLK	1.64	0.012	3.33	0.026	93.0	1.0305	45
5	1.50	0.011	2.97	0.024	92.8	0.9018	11
6	1.33	0.010	2.66	0.022	94.0	0.7814	13
7	1.33	0.010	2.64	0.022	94.3	0.7592	15
8	1.36	0.010	2.69	0.022	94.4	0.7528	17
9	1.40	0.010	2.76	0.023	94.7	0.7498	19
10	1.47	0.011	2.87	0.023	94.4	0.7499	21

Table 1: Results per iteration for ReAgent (IL+RL) on held-out ModelNet40 models. See Table 1 (left) in the main text.

those in Table 1 (left) in the main text, using held-out ModelNet40 models. The step-by-step results show that the exponential scale allows to pick large step sizes to achieve a rough alignment initially. Within about 3 steps, our approach achieves an accuracy similar to ICP in less time. Smaller step sizes are used subsequently and the performance of, for example, PointNetLK is reached after about 5 steps and 11ms of runtime. We observe that, for the last steps, ReAgent further refines the Chamfer distance by aligning closer to an indistinguishable pose – the error with respect to the true pose (indicated by MAE and ISO) increases slightly, while $\tilde{CD}$ reduces. If accuracy with respect to the true rotation and translation is preferred over the symmetry-aware metrics, the number of iterations and thereby the runtime could be further reduced. For consistency across experiments, however, we report the results after 10 iterations in the main results.

Qualitative step-by-step examples are given in Figure 3. ReAgent quickly achieves a rough alignment within about 3 steps, which is then further refined using smaller step sizes.

2. Application: Object Pose Estimation

In object pose estimation, the task equivalent to point cloud registration is referred to as *object pose refinement*. Refinement is a step to significantly increase accuracy, starting from an initial pose provided by an object pose estimator. By evaluation on the LINEMOD dataset [2], commonly used in this domain, the performance of the presented approach in this real-world scenario is highlighted. LINEMOD consists of 15 objects of which 13 are used for evaluation. Two of those are considered symmetric for evaluation (in italics in Table 2).

The training phase is slightly modified on LINEMOD. Instead of using a pre-training phase, we directly use the combined approach with a learning rate of $1e^{-3}$, halving it every 20 epochs for a total of 100 epochs. The influence of

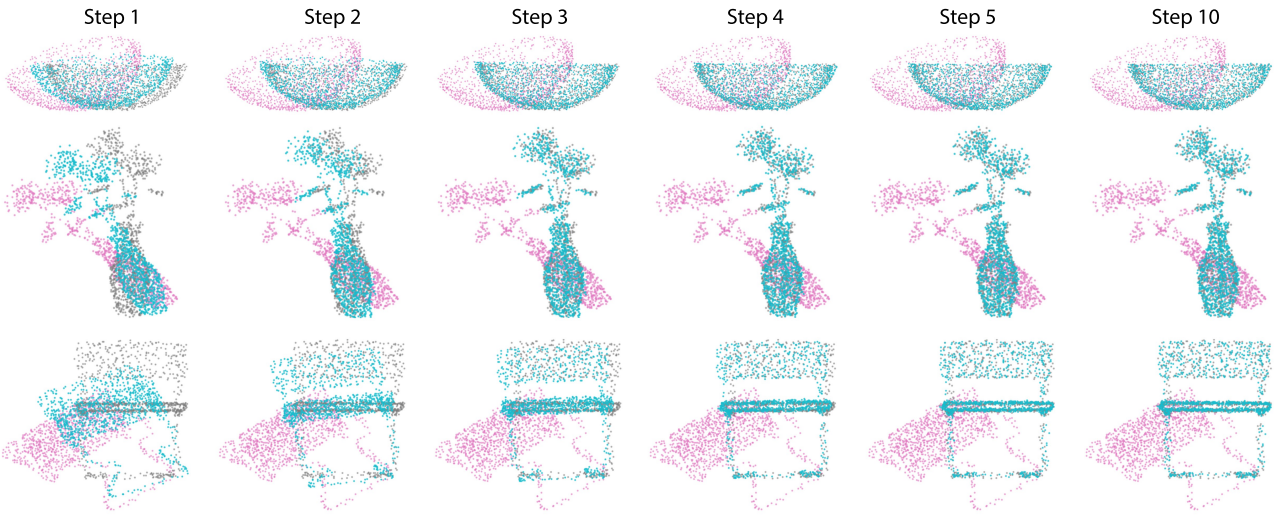


Figure 3: Qualitative examples. Stepwise results of ReAgent (IL+RL) on held-out ModelNet40 models. Target (gray), initial (magenta) and current source point clouds (cyan). Best viewed digitally.

	RGBD-based		RGB-based						depth-based		
class	DenseFusion	DenseFusion ref.	DPOD	DeepIM	DPOD ref.	PoseCNN	PFRL	DeepIM	ICP-based	ours IL	ours IL+RL
ape	79.5	92.3	53.3	78.7	87.7	27.8	60.5	77.0	79.1	**97.2**	96.9
vise	84.2	93.2	95.3	98.4	98.5	68.9	88.9	97.5	97.9	**99.6**	**99.6**
cam	76.5	94.4	90.4	97.8	96.1	47.5	64.6	93.5	93.5	99.0	**99.3**
can	86.6	93.1	94.1	97.6	**99.7**	71.4	91.3	96.5	98.7	99.6	99.5
cat	88.8	96.5	60.4	85.2	94.7	56.7	82.9	82.1	96.0	**99.8**	99.7
driller	77.7	87.0	97.7	91.6	98.8	65.4	92.0	95.0	84.2	98.8	**99.0**
duck	76.3	92.3	66.0	80.2	86.3	42.8	55.2	77.7	84.1	**96.9**	96.6
eggbox	**99.9**	99.8	99.7	99.7	**99.9**	98.3	99.4	97.1	98.4	99.8	**99.9**
glue	99.4	**100.0**	93.8	99.5	96.8	95.6	93.3	99.4	99.1	99.2	99.4
puncher	79.0	92.1	65.8	75.7	86.9	50.9	66.7	52.8	97.2	98.4	**98.6**
iron	92.1	97.0	99.8	99.7	**100.0**	65.6	75.8	98.3	90.6	97.9	97.5
lamp	92.3	95.3	88.1	98.2	96.8	70.3	96.6	97.5	94.0	**99.8**	99.7
phone	88.0	92.8	74.2	91.4	94.7	54.6	69.1	87.7	85.8	97.7	**97.8**
mean	86.2	94.3	83.0	91.8	95.1	62.8	79.7	88.6	92.1	**98.7**	**98.7**

Table 2: Results on LINEMOD for $AD < 0.1d$, initialized by DenseFusion (left), DPOD (mid) and PoseCNN (right).

the PPO loss term is reduced to $\alpha = 0.1$. Separate results for training using only IL are provided for comparison.

Baselines: We compare our method to the reported performances of RGBD (DenseFusion [8]), RGB-only (DPOD [11], DeepIM [3], PFRL [5]) and depth-only (the ICP-based method used in [9]) object pose refinement approaches. In each block in Table 2, the left-most column indicates the results of the method used for initialization (gray background). With our approach, we use the results provided for PoseCNN [9]. As in related work [5, 3], we utilize the segmentation masks provided together with the initial poses.

Metrics: Hinterstoisser et al. [2] propose two widely used evaluation metrics for object pose estimation. The Average Distance of Model Points (ADD) measures the distance between corresponding points under estimated and ground truth pose. To deal with symmetric objects, the ADD with Indistinguishable Views (ADI) metric instead considers the distance between closest points. We indicate the mixed use of ADD and ADI by abbreviating with AD. The reported recall values are computed at precision thresholds of 10, 5 and 2% of the object diameter. Please see Section 4 for a formal definition of the metrics.

Results: For training, we use the split defined in related work [1, 4, 6] and sample point clouds of size 1024 per training image as source. The sampling selects a random point on the object and finds its nearest neighbors in image space. $p\%$ of the points are sampled from the object (based on the ground-truth mask) and $100 - p\%$ are sampled from the surrounding background, where p is uniformly random in $[50, 100\%]$. Thereby, we simulate partial observation of the object and imprecise segmentation. As target, we uniformly random sample 1024 points from the correspond-

Figure 4: Qualitative examples on LINEMOD using ReAgent (IL+RL). As shown in the top row, 1024 points are sampled within the estimated segmentation mask. The black box indicates the zoomed-in view. Outlines for target (gray), initial (magenta) and current source pose (cyan) are shown. The last column shows a failure case. Best viewed digitally.

class	PoseCNN	DeepIM	ICP-based	ours IL	ours IL+RL
ape	5.2	48.6	38.0	70.6	**71.7**
vise	27.3	80.5	81.9	95.3	**96.0**
cam	12.5	74.0	56.1	87.7	**89.6**
can	26.2	84.3	81.2	95.7	**95.8**
cat	22.6	50.4	81.9	95.2	**95.6**
driller	23.7	79.2	59.3	97.1	**97.9**
duck	9.9	48.3	50.0	65.0	**69.4**
eggbox	73.9	77.8	93.1	**99.1**	98.9
glue	66.5	95.4	90.1	**98.7**	98.3
puncher	13.0	27.3	64.7	**91.3**	90.1
iron	23.2	86.3	60.9	**92.3**	91.5
lamp	29.6	86.8	85.9	98.8	**98.9**
phone	16.2	60.6	48.4	**90.9**	**90.9**
mean	26.9	69.2	68.6	90.6	**91.1**

Table 3: Per-class results on LINEMOD for $AD < 0.05d$, initialized using PoseCNN.

class	PoseCNN	DeepIM	ICP-based	ours IL	ours IL+RL
ape	0.0	14.3	2.9	7.5	**9.0**
vise	1.6	37.5	25.8	38.5	**39.9**
cam	0.5	**30.9**	4.2	17.8	24.8
can	1.0	**41.4**	10.6	39.7	41.3
cat	1.0	17.6	18.6	**41.6**	39.5
driller	1.6	35.7	5.8	46.5	**49.7**
duck	0.3	**10.5**	3.5	6.8	6.9
eggbox	17.9	34.7	**73.3**	72.4	73.2
glue	15.4	57.3	41.9	**76.4**	74.1
puncher	0.5	5.3	6.8	**31.2**	29.5
iron	0.7	**47.9**	5.0	34.2	34.9
lamp	1.6	45.3	44.0	**67.8**	66.9
phone	0.8	22.7	4.5	24.6	**25.7**
mean	3.3	30.9	19.0	38.8	**39.6**

Table 4: Per-class results on LINEMOD for $AD < 0.02d$, initialized using PoseCNN.

ing object model. The target point cloud is normalized to be mean centered and the farthest point to be of distance 1. The same translation and scaling is applied to the source under ground-truth pose. As such, the distance from the origin provides an inductive bias on whether an (aligned) point belongs to the model or the background. Finally, we apply an uniformly random initialization error to the source, with the translation magnitude sampled from $[0, 1]$ and the rotation magnitude sampled from $[0, 90]$deg. During testing, we uniformly random sample 1024 points within the estimated segmentation mask and initialize the source using the estimated pose, both provided by the PoseCNN results [9].

As shown in Table 2, our approach outperforms all compared methods with respect to the mean AD and on most per-class results. Note that while it is more difficult for RGB methods to estimate an accurate z-translation from the camera, they more easily recover from bad segmentation masks – and vice-versa for depth-based methods. For those methods that provide results using stricter AD thresholds, we additionally provide a comparison in Tables 3 and 4. Again, our approach increases accuracy by several percent, even achieving accuracy on the stricter $0.05d$ threshold that is competitive to the performance of the compared methods on the permissive $0.1d$ threshold. The results, moreover, indicate that the addition of RL is especially beneficial for these stricter thresholds. Since we train a single model and do not provide the agent with any information on the object class, prioritizing features that support refinement of one class might hinder that of another one. Sacrificing some generality by introducing class labels as input to ReAgent should allow a more uniform increase of performance.

For the step-wise illustrations in Figure 4, we zoom-in on the objects, as indicated by the black box in the top row, to highlight the high pose accuracy achieved by our ReAgent, barely distinguishable from the ground-truth pose. The pose accuracy already increases significantly within the first few ReAgent steps. For consistency, we keep using 10 steps on LINEMOD (as in the ModelNet40 experiments). While this might be reduced in real-world applications to speed-up refinement even more (below the 22ms achieved at the moment with 10 steps), the higher number of steps enables increased robustness to initialization errors.

A runtime comparison is more difficult on LINEMOD as we rely on reported numbers using different hardware. Still, the 22ms required by our approach (GTX 1080) compares favorably to 30ms for DPOD refinement (Titan X), 83ms for DeepIM (GTX 1080 Ti) and PFRL with 240ms (RTX 2080 Ti). Wang et al. [7] report no hardware and only provide runtimes for scenes of 3 to 6 objects, with 20+10ms for DenseFusion refinement (20ms for initial embeddings) and 10.4s for the ICP-based method from [9]. The latter refines multiple hypotheses and uses rendering-based verification.

3. Architecture Details

Table 5 details the architecture with all used layers, their input and output dimensions. Note that the initial embedding is computed for both source and target with shared weights. Also, the action embeddings and policies are computed for both rotation and translation, although the layers are given only once in Table 5.

4. Definition of Used Metrics

The metrics provided for our experiments are commonly used in related work [8, 10, 2]. We provide their definition in condensed form in the following section.

Layer	In	Out
Embeddings $\phi(X'_i)$ and $\phi(Y)$ (shared)		
Conv1d	$N \times 3$	$N \times 64$
ReLU		
Conv1d	$N \times 64$	$N \times 128$
ReLU		
Conv1d	$N \times 128$	$N \times 1024$
max	$N \times 1024$	1×1024
State S via $\phi(X'_i) \oplus \phi(Y)$		
concat	$2(1 \times 1024)$	2048
Embeddings $\phi_R(S)$ and $\phi_t(S)$		
FC	2048	512
ReLU		
FC	512	256
ReLU		
Value $\hat{v}$, using $\phi_R(S) \oplus \phi_t(S)$		
concat	2(256)	512
FC	512	256
ReLU		
FC	256	1
Policies $\pi_R(\phi_R(S))$ and $\pi_t(\phi_t(S))$		
FC	256	33
reshape	33	3×11

Table 5: ReAgent network architecture.

Mean Absolute Error (MAE): The MAE between a vector of estimated v' and true values v is defined as

$$MAE_v = \frac{1}{3} \sum |v' - v|, \tag{1}$$

where v is either the vector of Euler angles in degrees representing the rotation or the translation vector.

Isotropic Error (ISO): While MAE considers axes individually, ISO is computed over the full rotation and full translation. For the rotation error ISO_R, we compute the angle of the residual rotation matrix by

$$ISO_R = \arccos \frac{trace(R'^{-1}, R) - 1}{2} \tag{2}$$

and Euclidean distance between the estimated and true translation

$$ISO_t = ||t' - t||_2. \tag{3}$$

Chamfer Distance ($\tilde{CD}$): The Chamfer distance is used in the reward function of ReAgent. It is defined as

$$CD(X, Y) = \frac{1}{|X|} \sum_{x \in X} \min_{y \in Y} ||x - y||_2^2. \tag{4}$$

Modified Chamfer Distance $\tilde{CD}$: A modified variant is proposed in [10]. Compared to MAE and ISO, it considers the distances between points and not the transformations. It therefore implicitly considers symmetry. Based on the definition of the Chamfer distance, it is defined as

$$\tilde{CD}(X, Y) = CD(X, Y_{clean}) + CD(Y, X_{clean}), \tag{5}$$

where X_{clean} and Y_{clean} are the respective point cloud before applying noise and we compute $\tilde{CD}(X', Y)$.

Average Distance of Model Points (ADD) and Average Distance of Model Points with Indistinguishable

Views (ADI): The ADD is proposed in [2]. Given a model under an estimated pose X' and under the true pose Y, it is defined as the mean distance between corresponding points

$$ADD = \frac{1}{|Y|} \sum_{y \in Y, x' \in X'} ||y - x'||_2. \tag{6}$$

In addition, Hinterstoisser et al. [2] propose to account for symmetrical true poses by considering the closest point pairs, defined as

$$ADI = \frac{1}{|Y|} \sum_{y \in Y} \min_{x' \in X'} ||y - x'||_2. \tag{7}$$

The ADI recall for a specific precision threshold and N test samples is

$$ADI_{th} = \frac{1}{N} \sum_i \begin{cases} 0, & ADI_i > th \\ 1, & ADI_i \leq th, \end{cases} \tag{8}$$

where ADI_i is the ADI of the i^{th} test sample. The ADD recall is computed analogously. Note that, as $\tilde{CD}$, the ADI metric implicitly considers symmetry.

ADI Area-under-Curve (ADI AUC): ADI_{th} is computed at uniformly spaced thresholds up to maximum precision threshold. This results in a monotonically increasing precision-recall curve. ADI AUC is then defined as the area under this curve

$$AUC = \frac{1}{th_{max}} \sum_{th \in [0:\Delta:th_{max}]} ADI_{th} \cdot \Delta, \tag{9}$$

where Δ is the threshold spacing. We use $th_{max} = 0.1d$ and $\Delta = 1e^{-3}d$, where d is the diameter of the point cloud computed as maximal distance between any two points

$$d = \max_{x_1, x_2 \in X} ||x_1 - x_2||_2. \tag{10}$$

References

[1] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3364–3372, 2016. 3

[2] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *ACCV*, pages 548–562, 2012. 2, 3, 5, 6

[3] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Eur. Conf. Comput. Vis.*, pages 683–698, 2018. 3

[4] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Int. Conf. Comput. Vis.*, pages 3828–3836, 2017. 3

[5] Jianzhun Shao, Yuhang Jiang, Gu Wang, Zhigang Li, and Xiangyang Ji. Pfrl: Pose-free reinforcement learning for 6d pose estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11454–11463, 2020. 3

[6] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 292–301, 2018. 3

[7] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3343–3352, 2019. 5

[8] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Int. Conf. Comput. Vis.*, pages 3523–3532, 2019. 1, 3, 5

[9] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Robot.: Sci. Syst.*, 2018. 3, 4, 5

[10] Zi Jian Yew and Gim Hee Lee. Rpm-net: Robust point matching using learned features. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11824–11833, 2020. 1, 5

[11] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *Int. Conf. Comput. Vis.*, pages 1941–1950, 2019. 3