SceneReplica: Benchmarking Real-World Robot Manipulation by Creating Replicable Scenes Supplementary Material

Ninad Khargonkar*   Sai Haneesh Allu*   Yangxiao Lu   Jishnu Jaykumar P
Balakrishnan Prabhakaran   Yu Xiang
The University of Texas at Dallas
{firstname.lastname}@utdallas.edu   *equal contribution

A Grasp and Motion Planning

We used Graspit to compute the 6D-grasps for each object in the benchmark. The grasps were computed in an offline manner due to the time constraints imposed by Graspit. Attempting to run Graspit for each iteration in a scene would take too long time and hence, instead we decided to compute the grasps offline. We repeatedly run the grasp sampling algorithm until we obtain a desired number of grasps around the object. Finally, on the initially generated grasp set, we apply farthest point sampling on grasp translation component to ensure that the grasps are spread out across an object's surface. Fig. 2 illustrates the generated grasps for all 16 YCB objects. In all our experiment pipelines we try top-down grasping in case motion planning fails for a given grasp. Both model-based top-down grasping and model-free top-down grasping work in a similar fashion by considering a top-down view of object's point cloud.

We compute a top-down grasp for an object by obtaining the two principal components for the point cloud's X-Y dimensions. We use the smaller component for obtaining the gripper width and it also gives the orientation to align the gripper with respect to the object. The only difference between model-based and model-free top-down grasping is that with a model-free method, we are restricted to the partial point cloud of the object. For motion planning obstacles, in model-based grasping we simply use the estimated pose with the object's known 3D mesh. In model-free methods since we don't have models, one strategy is to simply use the bounding box of the partial point cloud. However, this did not work well in practice for cluttered scenes due to large obstacle boxes for motion planning, which then inevitably fails to find a motion plan to grasp. Hence, we resort to a similar strategy as used before for top-down grasp alignment and compute an oriented bounding box using the PCA of object points in X-Y plane.

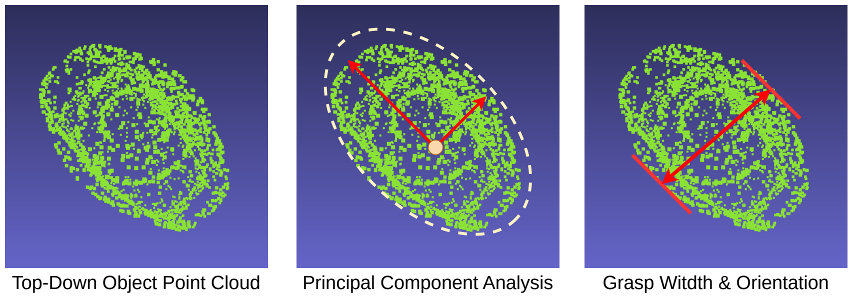

Top-Down Object Point Cloud
Principal Component Analysis
Grasp Witdth & Orientation

Figure 1: Top down grasp from point cloud

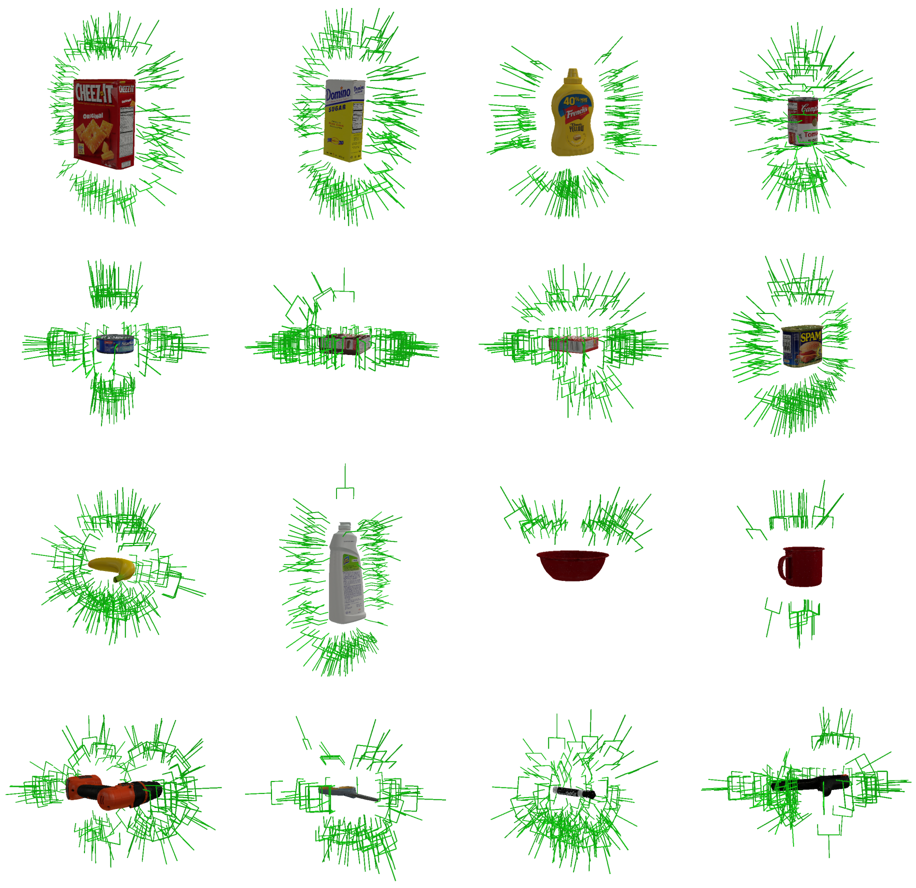


Figure 2: Offline grasps generated using Graspit! for model-based grasping experiments. Pre-grasp poses shown for better visualization of underlying object.

B Scene Generation

A successful and valid manipulation requires reachability of each object with respect to the robot arm. For generating the scenes in SceneReplica, we also ensure that all scenes are placed well within the robot's reachable space on the table. To find such "safe" areas for object placement, we first discretize the table's region into a dense grid and then check for each grid location's reachability. Next, we only spawn the objects at the selected grid locations in randomly sampled stable poses, along with a random rotation along global Z axis. The random rotation along Z axis ensures that even if the same stable pose is selected for an object, the scenes for it look different. The candidate spawn locations for an object are the neighbor grid locations of already placed objects to discourage well-separated scenes, while the object spawning is made collision free by checking for collisions using their 2D bounding boxes in X-Y plane. Once we have a candidate scene with 5 collision-free spawned objects, we test motion planning to each of them using the offline grasp dataset. If any of the objects does not have a feasible motion plan, we reject the scene and move on with a new scene generation.

Such a pipeline thus ensures both diverse and feasible scenes. Finally, a set of 20 scenes is selected from hundreds of candidate scenes using criteria on (1) an object's minimum count in the set and, (2) a scoring on pose-diversity of the set. While picking a scene set from the list of generated scene, we ensure that the count distribution of objects in the set is roughly uniform. Once we have a valid set of 20 scenes, we then compute the pose-based diversity. For a given set of 20 scenes, we first compute the count of unique poses for each object. This then also gives us a count-based probability distribution using which we compute the entropy and use that as the scoring function. The scoring functions ensures that scene-sets with the same repeated poses for objects are penalized and scene-sets with diversity are encouraged.

Type	Phase	Description
Perception failure	Pre-Grasping	Target object is not recognized, for example, no pose estimation or no segmentation mask
Perception failure	Pre-Grasping	Grasp not found, perception error of obstacle > threshold
Perception failure	During-Grasping	Grasp planned, failed to grasp and lift, perception error of target > threshold
Perception failure	During-Grasping	Grasp planned, hit obstacle, perception error of obstacle > threshold
Planning failure	Pre-Grasping	No perception failure, but no plan found to grasp target
Planning failure	During-Grasping	Grasp planned, no perception failure, failed to grasp and lift or hit obstacle
Execution failure	Post-Grasping	Failed to place object after grasping and lifting

Table 1: Different types of failures in our pick-and-place experimental analysis.

Method #	Perception	Grasp Planning	Motion Planning	Control	Ordering	Pick-and-Place Success	Grasping Success
Model-based Grasping							
1	PoseRBPF [1]	GraspIt! [2] + Top-down	OMPL [3]	MoveIt	Near-to-far	58 / 100	64 / 100
1	PoseRBPF [1]	GraspIt! [2] + Top-down	OMPL [3]	MoveIt	Fixed	59 / 100	59 / 100
2	PoseCNN [4]	GraspIt! [2] + Top-down	OMPL [3]	MoveIt	Near-to-far	47 / 100	48 / 100
2	PoseCNN [4]	GraspIt! [2] + Top-down	OMPL [3]	MoveIt	Fixed	40 / 100	45 / 100
3	GDRNPP [5, 6]	GraspIt! [2] + Top-down	OMPL [3]	MoveIt	Near-to-far	**66 / 100**	69 / 100
3	GDRNPP [5, 6]	GraspIt! [2] + Top-down	OMPL [3]	MoveIt	Fixed	62 / 100	64 / 100
Model-free Grasping							
4	UCN [7]	GraspNet [8] + Top-down	OMPL [3]	MoveIt	Near-to-far	43 / 100	46 / 100
4	UCN [7]	GraspNet [8] + Top-down	OMPL [3]	MoveIt	Fixed	37 / 100	40 / 100
5	UCN [7]	Contact-graspnet [9] + Top-down	OMPL [3]	MoveIt	Near-to-far	60 / 100	63 / 100
5	UCN [7]	Contact-graspnet [9] + Top-down	OMPL [3]	MoveIt	Fixed	60 / 100	64 / 100
6	MSMFormer [10]	GraspNet [8] + Top-down	OMPL [3]	MoveIt	Near-to-far	38 / 100	41 / 100
6	MSMFormer [10]	GraspNet [8] + Top-down	OMPL [3]	MoveIt	Fixed	36 / 100	41 / 100
7	MSMFormer [10]	Contact-graspnet [9] + Top-down	OMPL [3]	MoveIt	Near-to-far	57 / 100	65 / 100
7	MSMFormer [10]	Contact-graspnet [9] + Top-down	OMPL [3]	MoveIt	Fixed	61 / 100	**70 / 100**
8	MSMFormer [10]	Top-down	OMPL [3]	MoveIt	Fixed	56 / 100	59 / 100
End-to-end Learning-based Grasping							
9	Dex-Net 2.0 [11] (Top-Down Grasping)		OMPL [3]	MoveIt	Algorithmic	43 /100	51 / 100

Table 2: Different grasping frameworks evaluated on SceneReplica using a Fetch mobile manipulator.

C Additional Results

C.1 Pick-and-place Failure Analysis

For a pick-and-place failure observed during an experiment run, we classify the failure into several different categories: (1) perception failure (P_EF), (2) planning failure (P_LF) and (3) execution failure (EF). Table 1 describes these failures in detail. We include such detailed metrics to highlight the failure points during grasping rather than simply labeling each trial as a complete success or failure in Table 2. The overall trend seen from Tables 3 and 4 is that perception and planning failures dominate the error terms. Tables 3 and 4 also show difficulties in grasping small and thin objects like the scissor, the marker, etc., due to high uncertainty in their depth and small graspable area.

C.2 Pose Estimation Validation

We have also visualized the results for the validation on pose estimation methods where we plot an ADD threshold-accuracy curve and a frequency histogram for pose rotation angle error as shown in Figure 3. In the threshold-accuracy curves, we tweak the tolerance of translation and rotation difference with ground truth for classifying a pose prediction as successful or unsuccessful. As we can see, the latest GDRNPP [5, 6] method outperforms other methods in terms of pose estimation which also translates to a better result in pick-and-place grasping as shown in Table 3.

D Scene Replication for Different Robot Platform

D.1 Robot Reachable Space Verification

We show an example about extending the scene reachability for a different platform with a Franka-Panda arm. The arm is loaded in the simulation environment with just slight changes in height owing to the difference in morphology (we assume researchers have access to height adjustable table as noted in the paper). Then using a similar procedure as that for Fetch robot, we spawn small cubes in dense grid locations on the table. A valid motion plan to each cube is checked and cubes with no plan are removed. This gives us an idea about the reachable space of the new robot as seen in Fig.2 of the paper. The scenes can then be spawned inside this reachable space and reference images are generated given the choice of custom camera parameters as described in Section D.2.

Object	Count	Method 1				Method 2				Method 3				Method 4			
		S	P$_E$F	P$_L$F	EF	S	P$_E$F	P$_L$F	EF	S	P$_E$F	P$_L$F	EF	S	P$_E$F	P$_L$F	EF
Order: Near-to-Far																	
003 cracker box	6	5	-	1	-	1	4	1	-	3	2	1	-	2	2	2	-
004 sugar box	5	5	-	-	-	1	4	-	-	5	-	-	-	2	2	1	-
005 tomato soup can	7	6	1	-	-	6	1	-	-	5	1	-	1	3	-	4	-
006 mustard bottle	7	6	1	-	-	3	2	2	-	7	-	-	-	2	1	4	-
007 tuna fish can	6	1	1	4	-	3	2	1	-	1	5	-	-	6	-	-	-
008 pudding box	5	5	-	-	-	4	1	-	-	5	-	-	-	3	-	2	-
009 gelatin box	7	3	4	-	-	2	4	1	-	6	-	1	-	3	2	1	1
010 potted meat can	7	6	1	-	-	3	2	2	-	7	-	-	-	4	1	2	-
011 banana	7	4	-	2	1	2	4	1	-	6	-	1	-	1	-	6	-
021 bleach cleanser	5	3	-	-	2	1	1	1	2	3	1	-	1	2	1	1	1
024 bowl	7	2	4	1	-	5	2	-	-	2	4	1	-	5	-	2	-
025 mug	5	2	1	-	2	3	2	-	-	4	-	1	-	3	1	1	-
037 scissors	7	1	2	4	-	1	3	3	-	4	3	-	-	1	-	5	1
035 power drill	7	2	1	3	1	3	2	2	-	1	3	2	1	-	7	-	-
040 large marker	6	1	4	1	-	4	1	1	-	2	4	-	-	3	1	2	-
052 extra large clamp	6	6	-	-	-	5	1	-	-	5	1	-	-	3	-	3	-
ALL	100	58	20	16	6	47	36	15	2	66	24	7	3	43	18	36	3
Order: Randomly-Fixed																	
003 cracker box	6	5	1	-	-	2	4	-	-	5	1	-	-	2	3	1	-
004 sugar box	5	4	1	-	-	1	3	1	-	5	-	-	-	3	-	2	-
005 tomato soup can	7	7	-	-	-	6	1	-	-	6	1	-	-	2	1	4	-
006 mustard bottle	7	7	-	-	-	2	4	1	-	5	1	1	-	3	2	2	-
007 tuna fish can	6	2	4	-	-	1	4	-	1	2	3	1	-	4	1	-	1
008 pudding box	5	4	-	1	-	3	1	1	-	3	2	-	-	3	-	2	-
009 gelatin box	7	3	4	-	-	4	-	2	1	5	2	-	-	7	-	-	-
010 potted meat can	7	7	-	-	-	5	1	1	-	7	-	-	-	1	3	2	1
011 banana	7	2	1	4	-	2	4	1	-	3	2	2	-	5	-	2	-
021 bleach cleanser	5	3	2	-	-	1	3	1	-	4	-	-	1	1	1	2	1
024 bowl	7	4	3	-	-	1	4	2	-	2	4	1	-	3	2	2	-
025 mug	5	5	-	-	-	3	2	-	-	3	-	1	1	4	-	1	-
037 scissors	7	-	1	6	-	-	2	5	-	4	3	-	-	-	3	4	-
035 power drill	7	2	1	4	-	2	2	3	-	-	6	1	-	-	7	-	-
040 large marker	6	-	5	1	-	3	3	-	-	2	3	1	-	-	1	5	-
052 extra large clamp	6	6	-	-	-	5	-	1	-	6	-	-	-	2	-	4	-
ALL	100	59	25	16	-	40	42	17	1	62	28	8	2	37	24	35	4

Object	Count	Method 5				Method 6				Method 7				Method 8			
		S	P$_E$F	P$_L$F	EF	S	P$_E$F	P$_L$F	EF	S	P$_E$F	P$_L$F	EF	S	P$_E$F	P$_L$F	EF
Order: Near-to-Far																	
003 cracker box	6	3	1	2	-	2	3	1	-	4	1	-	1	✗	✗	✗	✗
004 sugar box	5	5	-	-	-	3	1	1	-	5	-	-	-	✗	✗	✗	✗
005 tomato soup can	7	6	-	1	-	2	-	5	-	2	2	3	-	✗	✗	✗	✗
006 mustard bottle	7	5	1	1	-	1	-	5	1	6	-	1	-	✗	✗	✗	✗
007 tuna fish can	6	5	1	-	-	5	-	1	-	5	1	-	-	✗	✗	✗	✗
008 pudding box	5	4	-	1	-	4	-	1	-	4	1	-	-	✗	✗	✗	✗
009 gelatin box	7	7	-	-	-	4	-	3	-	6	-	1	-	✗	✗	✗	✗
010 potted meat can	7	3	2	1	1	1	-	6	-	5	2	-	-	✗	✗	✗	✗
011 banana	7	2	-	5	-	5	-	2	-	6	-	-	1	✗	✗	✗	✗
021 bleach cleanser	5	2	-	2	1	-	1	3	1	-	1	2	2	✗	✗	✗	✗
024 bowl	7	7	-	-	-	5	-	1	1	6	-	-	1	✗	✗	✗	✗
025 mug	5	1	1	3	-	2	-	3	-	3	-	2	1	✗	✗	✗	✗
037 scissors	7	3	2	2	-	-	2	5	-	-	2	2	3	✗	✗	✗	✗
035 power drill	7	2	4	-	1	1	3	3	-	3	3	-	1	✗	✗	✗	✗
040 large marker	6	1	2	3	-	3	-	3	-	2	2	2	1	✗	✗	✗	✗
052 extra large clamp	6	4	1	1	-	-	1	5	-	2	1	2	1	✗	✗	✗	✗
ALL	100	60	15	22	3	38	11	48	3	57	16	15	12	✗	✗	✗	✗
Order: Randomly-Fixed																	
003 cracker box	6	4	1	1	-	3	2	1	-	4	1	1	-	2	3	1	-
004 sugar box	5	3	-	2	-	3	1	1	-	5	-	-	-	5	-	-	-
005 tomato soup can	7	6	1	-	-	1	-	5	1	7	-	-	-	4	-	1	2
006 mustard bottle	7	1	1	3	1	-	-	6	1	3	-	1	3	6	-	1	-
007 tuna fish can	6	4	1	1	-	2	1	3	-	4	1	-	1	5	-	1	-
008 pudding box	5	4	1	-	-	4	-	1	-	4	-	-	1	5	-	-	-
009 gelatin box	7	1	6	-	-	4	1	2	-	6	-	1	-	7	-	-	-
010 potted meat can	7	6	1	-	-	4	1	2	-	4	-	2	2	4	1	2	-
011 banana	7	1	-	6	-	3	1	2	1	5	1	1	-	3	1	3	-
021 bleach cleanser	5	1	1	2	1	1	1	3	-	1	-	2	2	3	1	1	-
024 bowl	7	7	-	-	-	4	-	2	1	7	-	-	-	7	-	-	-
025 mug	5	4	-	-	1	2	1	2	-	2	-	3	-	1	1	3	-
037 scissors	7	2	2	3	-	2	2	3	-	2	4	1	-	1	4	2	-
035 power drill	7	3	4	-	-	-	3	3	1	4	1	1	1	1	4	2	-
040 large marker	6	3	2	1	-	2	1	3	-	2	4	-	-	2	2	1	1
052 extra large clamp	6	3	2	1	-	1	-	5	-	2	2	2	-	-	5	1	-
ALL	100	60	17	20	3	36	15	44	5	61	14	15	10	56	22	19	3

Table 3: Statistics of our grasping experiments for each YCB object (Methods 1-8). S: #pick-and-place success, P$_E$F: #perception failure, P$_L$F: #planning failure, EF: #execution failure
.

Object	Count	Method 9			
		S	P$_E$F	P$_L$F	EF
Order: Algorithmic					
003 cracker box	6	-	-	5	1
004 sugar box	5	4	-	1	-
005 tomato soup can	7	2	-	4	1
006 mustard bottle	7	2	-	4	1
007 tuna fish can	6	-	-	6	-
008 pudding box	5	4	-	1	-
009 gelatin box	7	6	-	1	-
010 potted meat can	7	5	-	2	-
011 banana	7	6	-	1	-
021 bleach cleanser	5	-	-	4	1
024 bowl	7	6	-	-	1
025 mug	5	2	-	3	-
037 scissors	7	-	-	7	-
035 power drill	7	-	-	6	1
040 large marker	6	3	-	2	1
052 extra large clamp	6	3	-	2	1
ALL	100	43	-	49	8

Table 4: Statistics of our grasping experiments for each YCB object (Method#9). S: #pick-and-place success, P$_E$F: #perception failure, P$_L$F: #planning failure, EF: #execution failure

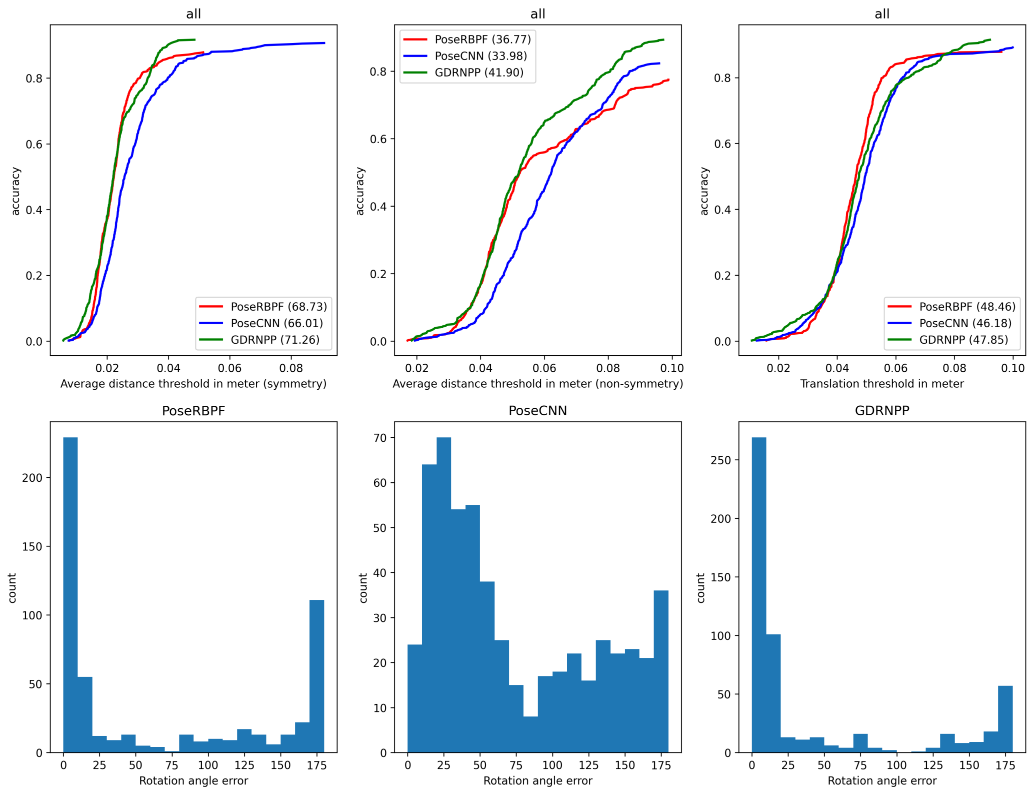


Figure 3: Visualization of pose estimation validation

D.2 Reference Image Generation with Different Camera

Here we demonstrate a replication procedure for the reference scene images under different camera settings. With a different robot platform, the reference scene images can be regenerated in simulation given the new camera's parameters. Using the outlined procedure the reference images can be generated even if the camera is separate from the robot platform.

- Spawn the objects within the reachable space of new robot as shown in Section D.
- Adjust the robot's camera by changing the pose so that all objects are visible.
- Render the images in simulation using the camera and use them as reference while setting up the scene in real-world.

For an example, refer to Table 5 which contains reference images for two scenes generated using the same ground truth poses of the objects, but under two different camera settings. Note that the real-world camera parameters should match the one used in simulation for accurate scene replication.

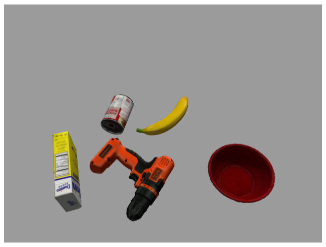
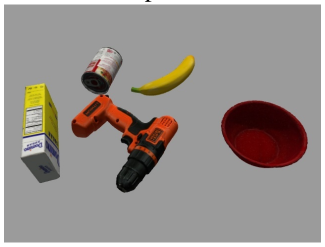
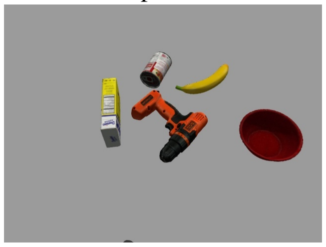
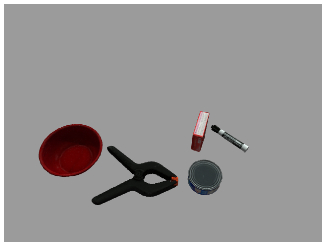
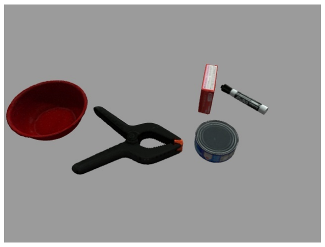
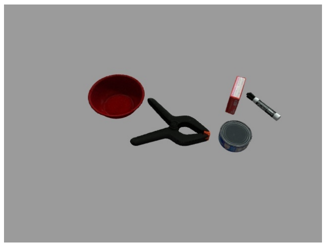
Generated Scenes		
Fetch	Camera placement I	Camera placement II
		
		

Table 5: Reference images with two different camera settings

References

[1] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox. Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking. *IEEE Transactions on Robotics*, 37(5):1328–1342, 2021.

[2] A. T. Miller and P. K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

[3] I. A. Sucan, M. Moll, and L. E. Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[4] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[5] G. Wang, F. Manhardt, F. Tombari, and X. Ji. GDR-Net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16611–16621, June 2021.

[6] X. Liu, R. Zhang, C. Zhang, B. Fu, J. Tang, X. Liang, J. Tang, X. Cheng, Y. Zhang, G. Wang, and X. Ji. Gdrnpp. `https://github.com/shanice-l/gdrnpp_bop2022`, 2022.

[7] Y. Xiang, C. Xie, A. Mousavian, and D. Fox. Learning rgb-d feature embeddings for unseen object instance segmentation. In *Conference on Robot Learning*, pages 461–470. PMLR, 2021.

[8] A. Mousavian, C. Eppner, and D. Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2901–2910, 2019.

[9] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13438–13444. IEEE, 2021.

[10] Y. Lu, Y. Chen, N. Ruozzi, and Y. Xiang. Mean shift mask transformer for unseen object instance segmentation. *arXiv preprint arXiv:2211.11679*, 2022.

[11] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.