Using Dense Object Descriptors for Picking Cluttered General Objects with Reinforcement Learning

Hoang-Giang Cao*[1,3], Weihao Zeng*[1], and I-Chen Wu[1,2,3]

[1]*Department of Computer Science, National Yang Ming Chiao Tung University, Taiwan*
[2]*Pervasive Artificial Intelligence Research (PAIR) Labs, Taiwan*
[3]*Research Center for IT Innovation, Academia Sinica, Taiwan*

Abstract

We propose a reinforcement learning method for picking cluttered general objects using visual descriptors with suction grasp. In this paper, we learn cluttered object descriptors (CODs), which could represent rich object structures, and use the pre-trained CODs network along with its intermediate outputs to train a picking policy. We conduct experiments to evaluate our method. Our CODs could consistently represent known and unknown cluttered general objects, which allowed for the picking policy to robustly pick cluttered general objects. The resulting policy could pick 96.69% of unseen objects that are 2X as cluttered as the training scenarios.

1 Introduction

Grasping, one of the most important manipulation skills, is essential for many real-world robotics applications. Deep reinforcement learning (DRL) is a promising approach for solving the grasping problem. Kalashnikov proposed QT-Opt, a vision based robotic manipulation system on real robots using DRL [9], which could automatically learn appealing behaviors, such as re-grasping and pre-grasp. Suction grasping, despite its reliability and simplicity, attracts far less attention than other types of grasps in the research community [1, 3]. Zeng manually labeled RGB-D images for training suction grasp network in the 2017 Amazon Robotics Challenge [15]. Shao proposed a self-supervised method for suction grasping in clutter environments, but Shao only considered cylinders [13]. A better solution for picking cluttered general objects using suction grasp is lacking.

A good representation is critical to achieve good performance for DRL methods [4]. Florence proposed Dense Object Nets (DONs), which generate dense descriptors with rich object structure information [6]. Recent works extend DONs on manipulation tasks [7, 14, 2]. To better represent cluttered general objects in the grasping problem, we utilize DONs to generate representations.

In this paper, we present a novel method capable of picking general objects from cluttered environments. We obtain the cluttered objects descriptors (CODs) network using DONs, and train a picking policy using the final and intermediate outputs of the CODs network. We employ Actor-Critic [11], a reinforcement learning method, to train the picking policy. For a given RGB-D input, the network outputs a policy represented by probabilities of picking at each pixel location. Our method could effectively generalize to more cluttered unseen objects. Since we use reinforcement learning, we could train our method without supervision.

Contributions. The main contributions of this paper can be summarized as follows: 1) we extend DONs [6] to learning CODs using RGB-D images with domain randomization. 2) we proposed a novel deep reinforcement learning approach that employ intermediates outputs of the trained CODs

*Equal contribution.

NeurIPS 2021 Workshop on Robot Learning: Self-Supervised and Lifelong Learning, Virtual

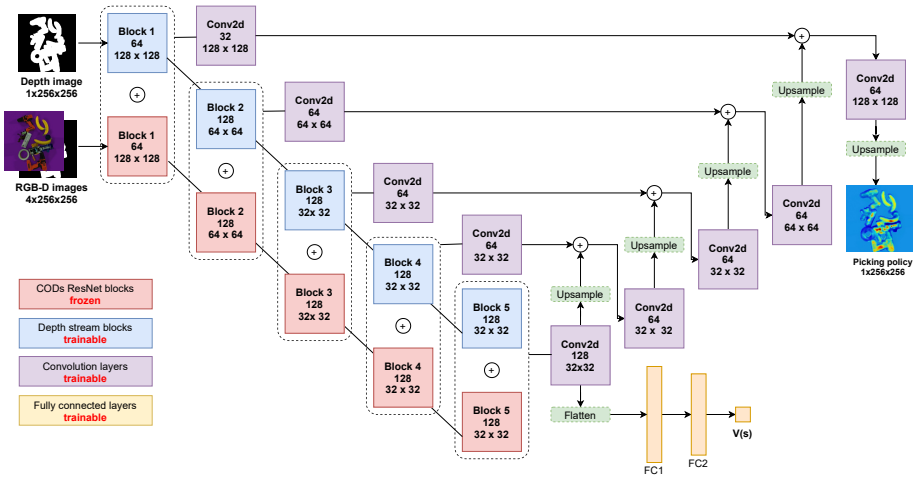


Figure 1: The network structure. The red boxes are the CODs stream; the blue boxes are the depth stream; the purple boxes are 2D convolution layers, which are connected in a U-Net fashion.

network to better pick cluttered general objects. 3) we conduct experiments to demonstrate that our method out-performs previous methods, and can generalize to unseen cluttered objects.

2 Task Definition

We focus on picking object from a basket with cluttered general objects using a single suction pad. The input is $I = (I_{RGB}, I_D)$, where I_{RGB} and I_D are the RGB and depth images of the current basket. The action $a = (x, y)$ represent the pixel coordinate that indicates where to pick using the suction pad. We calculate the 3D point corresponding to the selected pixel, and attempt to pick the object at the point from the surface normal direction. In the beginning of an episode, we randomly drop objects in the basket, and we attempt to pick objects out from the basket.

3 Method

3.1 Cluttered Objects Descriptor

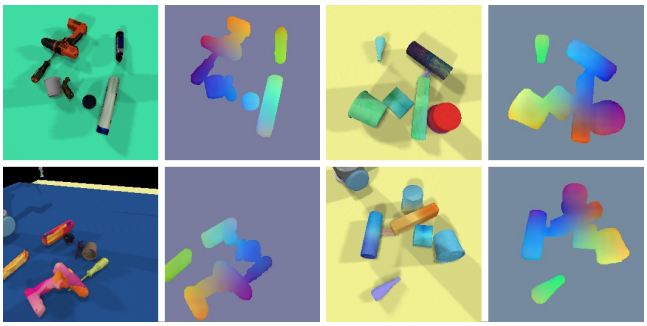
Our method for training the CODs network for cluttered objects is mainly inspired by DONs [6]. We employ the self-supervised contrastive loss from DONs on cluttered objects with randomization. Unlike original DONs, we train the descriptor with cluttered objects directly. The scene configuration is similar to picking environment later. Additional training details are contained in the Appendix.

3.1.1 Data Generation and Randomization

To find the best input configuration to represent cluttered objects, we study the impacts of different input configurations (depth, RGB, RGB-D). We make use of randomization to help CODs learn object geometries rather than the textures, since object geometries are critical for suction grasping. We randomize object textures and the workingspace (a table) texture before capturing each RGB-D image from random different camera poses.

3.2 Reinforcement Learning for Picking Policy

We use Actor-Critic [11] as the reinforcement learning method to train our picking policy network. Inspired by Shao [13], we better utilize the representation power of the CODs network by using its intermediate outputs. Shao fed the outputs of ResNet blocks to a U-Net like network. Similarly to Shao, we fed the intermediate outputs of the CODs network, a ResNet, to a U-Net like structure, as shown in Figure 1. Additionally, we have another stream of ResNet blocks for the depth input. As the result, we have two streams of ResNet blocks, one of which is the pre-trained CODs network. We



(a) (b)

Figure 2: The CODs are consistent under texture randomization. (a) CODs are consistent among seen cluttered objects. (b) CODs are consistent among unseen cluttered objects.

concatenate the corresponding outputs of ResNet blocks and, feed them forward through a U-Net like structure. Unlike Shao, the weights of the CODs network are frozen during RL training. Additionally, we add a multi-layer-perceptron at the bottleneck of the U-Net structure for value in the Actor-Critic method. Thanks to reinforcement learning, our method not only learns how to grasp, but also learns to avoid collisions and suction grasps that would cause invalid robot arm configurations.

4 Experiment

4.1 Cluttered-Object Dense Descriptors

Similar to Sundaresan [14], we evaluate CODs by calculating the matching distance, the L2 pixel distance between the best matching pixel coordinate and the ground-truth matching coordinate. The matching distances are normalized by the diagonal pixel distance of the image.

We compare the effectiveness of using 1) different input configurations: depth, RGB, RGB-D, and 2) texture randomization. As shown in Table 1, RGB-D with randomization out-performs other input and randomization configurations on both the GraspNet test split objects and the novel objects. We achieve 4.56% and 6.38% of matching distance on GraspNet test split objects and novel objects. The resulting CODs performs well on unseen objects, and is robust. It could consistently find matching pixels under texture randomizations and occlusions, as shown in Figure 2. Based on the experiment results, we use the CODs with RGB-D and randomization in our picking network.

4.2 Picking Cluttered General Objects

We have three metrics for evaluating the performance: 1) Completion. The % of completion runs over all runs. A completion run is a run that all objects are picked before the episode terminates. 2) Average picked object. The average number of objects picked in all runs. 3) Success rate. The average % number of successful picks per completion run.

We compare with the following methods: 1) Suction Grasp Region Prediction: Shao [13] proposed a network structure that combines ResNet34-stride8 and U-Net. We use Shao's structure in our training pipeline with the same environment and hyper-parameters. 2) VPG Net: Zeng's [15] fully convolutional network with DenseNet backbone. We added upsample convolutional layers to increase the output size from 8x8 to 128x128 3) Direct CODs: a network structure that directly uses the output from CODs with a U-Net. 4) Depth Only: our method using only the depth input. 5) CODs Only: our method using only the CODs input.

Objects	Matching Distance ($1e^5$ pairs of pixels)				
	RGBD w/ rand	RGB w/ rand	Depth	RGBD	RGB
GraspNet (train split)	**0.0311**	0.0331	0.0332	0.095	0.098
GraspNet (test split)	**0.0456**	0.0506	0.0518	0.120	0.121
Novel	**0.0638**	0.0741	0.0747	0.138	0.147

Table 1: Result of CODs with Randomization

We train each method in the simulation environment with the GraspNet train split on 10 objects. We then test our method in the environment that much more cluttered than the training scenarios. For testing, we run 50 episodes with 20 and 30 objects on 2 sets: 1) the GraspNet test split objects. 2) novel household objects.

As shown in Table 2, Table 3, and Table 4, our method out-performs other methods on most of the metrics. Our method could effectively generalize to more cluttered scenarios with unseen objects. Our method, which uses both the depth stream and CODs stream, performs better than the method that uses only the depth stream and Shao's method, which has the same network structure as our method, by a large margin. In addition, our method, which uses the intermediate outputs of the CODs network, out-matches the method that only uses the final output. Hence, using the intermediate outputs of the CODs network helps improve the picking performance as well.

Dataset	#objects	Shao's method	VPG Net	Direct CODs	Depth Only	CODs Only	CODs + Depth
GraspNet (test split)	20	34.00%	43.33%	46.66%	91.30%	71.42%	**96.66%**
	30	23.90%	23.33%	34.48%	78.57%	92.00%	**93.33%**
Novel objects	20	14.89%	46.66%	46.66%	70.21%	82.10%	**96.69%**
	30	4.65%	46.66%	46.42%	62.5%	36.0%	**68.90%**

Table 2: Result of Picking (Completion)

Dataset	#objects	Shao's method	VPG Net	Direct CODs	Depth Only	CODs Only	CODs + Depth
GraspNet (test split)	20	16.12	15.80	16.45	19.32	17.75	**19.50**
	30	22.69	22.23	23.59	27.50	28.80	**29.10**
Novel objects	20	13.11	17.25	16.90	18.14	18.30	**18.87**
	30	21.32	24.00	24.53	**27.49**	25.96	27.20

Table 3: Result of Picking (Average Picked Object)

Dataset	#objects	Shao's method	VPG Net	Direct CODs	Depth Only	CODs Only	CODs + Depth
GraspNet (test split)	20	49.10%	49.07%	50.95%	**66.20%**	59.64%	64.90%
	30	47.70%	50.05%	48.74%	**71.35%**	57.23%	63.71%
Novel objects	20	31.90%	47.56%	47.56%	62.80%	62.08%	**66.20%**
	30	38.11%	48.66%	45.20%	61.00%	55.56%	**68.20%**

Table 4: Result of Picking (Success Rate)

5 Conclusion

In this paper, we extend the Dense Object Descriptors [6] to better represent cluttered objects, and the result is the CODs. CODs are able to robustly represent cluttered objects under texture and viewing angles changes. Hence, CODs could focus on the object geometries, which is a useful characteristic for suction grasping. As the result, our method could learn to effectively pick cluttered general objects while avoiding collisions and control failures. Our picking policy could pick 96.69% unseen objects that are 2X as cluttered as the training scenarios. For future works, we will extend CODs to better focus on objects, and apply it to sim2real problems.

Acknowledgments and Disclosure of Funding

This research is partially supported by the Ministry of Science and Technology (MOST) of Taiwan under Grant Number 109-2634-F-009-019 and 110-2634-F-009-022 through Pervasive Artificial Intelligence Research (PAIR) Labs, and the resource partially supported by Industrial Technology Research Institute (ITRI) of Taiwan. The authors also thank anonymous reviewers for their valuable comments.

References

[1] Hanwen Cao, Hao-Shu Fang, Wenhai Liu, and Cewu Lu. Suctionnet-1billion: A large-scale benchmark for suction grasping. *arXiv preprint arXiv:2103.12311*, 2021.

[2] Chun-Yu Chai, Keng-Fu Hsu, and Shiao-Li Tsao. Multi-step pick-and-place tasks using object-centric dense correspondences. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4004–4011. IEEE, 2019.

[3] Guoguang Du, Kai Wang, Shiguo Lian, and Kaiyong Zhao. Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review. *Artificial Intelligence Review*, 54(3):1677–1734, 2021.

[4] Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. Is a good representation sufficient for sample efficient reinforcement learning? *arXiv preprint arXiv:1910.03016*, 2019.

[5] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11444–11453, 2020.

[6] Peter Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *Conference on Robot Learning*, 2018.

[7] Aditya Ganapathi, Priya Sundaresan, Brijen Thananjeyan, Ashwin Balakrishna, Daniel Seita, Jennifer Grannen, Minho Hwang, Ryan Hoque, Joseph E Gonzalez, Nawid Jamali, et al. Learning dense visual correspondences in simulation to smooth and fold real fabrics. *arXiv preprint arXiv:2003.12698*, 2020.

[8] Stephen James, Marc Freese, and Andrew J Davison. Pyrep: Bringing v-rep to deep robot learning. *arXiv preprint arXiv:1906.11176*, 2019.

[9] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1928–1937. JMLR.org, 2016.

[12] E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013. www.coppeliarobotics.com.

[13] Quanquan Shao, Jie Hu, Weiming Wang, Yi Fang, Wenhai Liu, Jin Qi, and Jin Ma. Suction grasp region prediction using self-supervised learning for object picking in dense clutter. In *2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)*, pages 7–12. IEEE, 2019.

[14] Priya Sundaresan, Jennifer Grannen, Brijen Thananjeyan, Ashwin Balakrishna, Michael Laskey, Kevin Stone, Joseph E Gonzalez, and Ken Goldberg. Learning rope manipulation policies using dense object descriptors trained on synthetic depth data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9411–9418. IEEE, 2020.

[15] Andy Zeng, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois R Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3750–3757. IEEE, 2018.

[16] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018.

A Appendix A Experiment Setup

A.1 Simulation

We use CoppeliaSim [12], a simulation engine, and PyRep [8], a robotics learning toolkit, to both create the synthetic dataset for training CODs, and model the simulation environment for training and experiments. See Figure 3 for an example of the simulation environment. We generate dataset for training the CODs network by capturing RGB and depth images from different camera poses. We use the UR-5 robot arm and the suction pad provided by CoppeliaSim.

We modify the original suction cup from CoppeliaSim to mimic the real-world suction pad more closely. As shown in Figure 4, in addition to a single ray proximity sensor on the center in the original suction pad, we add 6 evenly spaced ray proximity sensors near the border of the suction pad. An object is considered successfully picked if all 7 proximity sensors detects the object. All ray proximity sensors are 7mm.

Similar to Zeng [15], we consider the surface normals for picking. After selecting an action $a = (x, y)$, a pixel coordinate in the image space for picking. We calculate the point cloud, and estimate surface normals using Open3D [16]. We obtain the 3D point p and the surface normal vector n corresponding to the pixel specified by a. To prevent the robot arm from picking from near horizontal directions, we clip n to be at most 60 degrees from the up-right direction, and obtain the clipped vector n'. The suction pad approaches p from n' direction, and attempt to pick an object.

A.2 Dataset

We evaluated our method on 3 datasets. 1) 28 objects from the GraspNet train split, and 2) 47 objects from the GraspNet test split, and 3) 13 novel household objects [5]. See Figure 6 for examples of objects from each dataset, and different levels of clutteredness. We select 55 objects that are suitable for suction grasping from the GraspNet objects. In this paper, the GraspNet train and test splits refer to the selected objects in the original GraspNet train and test split.

A.3 Direct CODs

Figure 5 shows our Direct CODs network structure which use directly use the representation from CODs be the input of picking network.

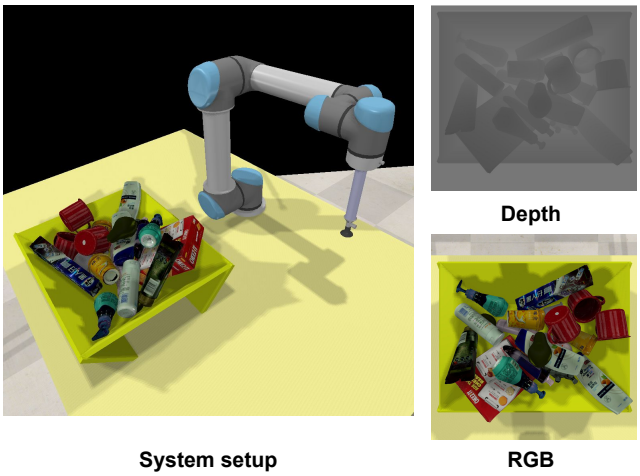


Figure 3: The simulation picking system setup, and sample RGB and depth images. We use the UR-5 robot arm with a suction pad gripper.

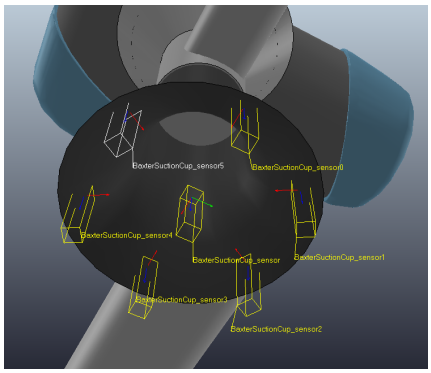


Figure 4: The modified suction pad and proximity sensors. An object is considered successfully grasped if all 7 7mm ray proximity sensors detects the object.

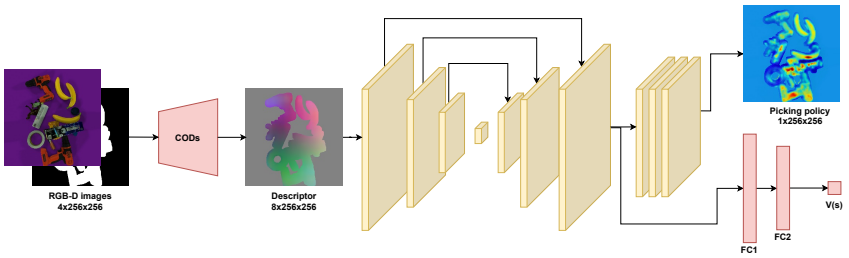


Figure 5: The network structure that directly feeds output of the CODs network to a U-Net.

B Appendix B Training Details

B.1 Self-Supervised Contrastive Loss

We use the contrastive loss from [6]. Given an input image, $I \in \mathbb{R}^{W \times H \times X}$ where X can be either $1, 3, 4$ depending whether the input is depth, RGB, or RGB and depth, we map I to a dense descriptor space $\mathbb{R}^{W \times H \times D}$. For each pixel in the input image, we have a D-dimensional descriptor vector. For a pair of inputs I_a and I_b captured by cameras of different poses on the same fixed objects, we can find pairs of matching pixels, where the pixels in two images corresponds to the same vertex in the 3D reconstruction. The dense descriptor network, f, is trained via a pixel wise contrastive loss to minimize the distances between descriptors of matching pixels, and keep descriptors of non-matching pixels at least M distance apart. The loss function is

$$\mathcal{L}_{\text{m}}(I_a, I_b) = \frac{1}{N_{\text{m}}} \sum_{N_{\text{m}}} \|f(I_a)(u_a) - f(I_b)(u_b)\|_2^2 \tag{1}$$

$$\mathcal{L}_{\text{nm}}(I_a, I_b) = \frac{1}{N_{\text{hard-negatives}} > 0} \sum_{N_{\text{nm}}} max(0, M - \|f(I_a)(u_a) - f(I_b)(u_b)\|_2)^2 \tag{2}$$

$$N_{\text{hard-negatives}} = \sum_{N_{\text{non-matches}}} \mathbb{1}(M - \|f(I_a)(u_a) - f(I_b)(u_b)\|_2) > 0 \tag{3}$$

$$\mathcal{L}(I_a, I_b) = \mathcal{L}_{\text{m}}(I_a, I_b) + \mathcal{L}_{\text{nm}}(I_a, I_b) \tag{4}$$

where "m" and "nm" represents match and non-match; $f(I)(u)$ represents the descriptor of I at pixel coordinate u; $\mathbb{1}$ is the indicator function.

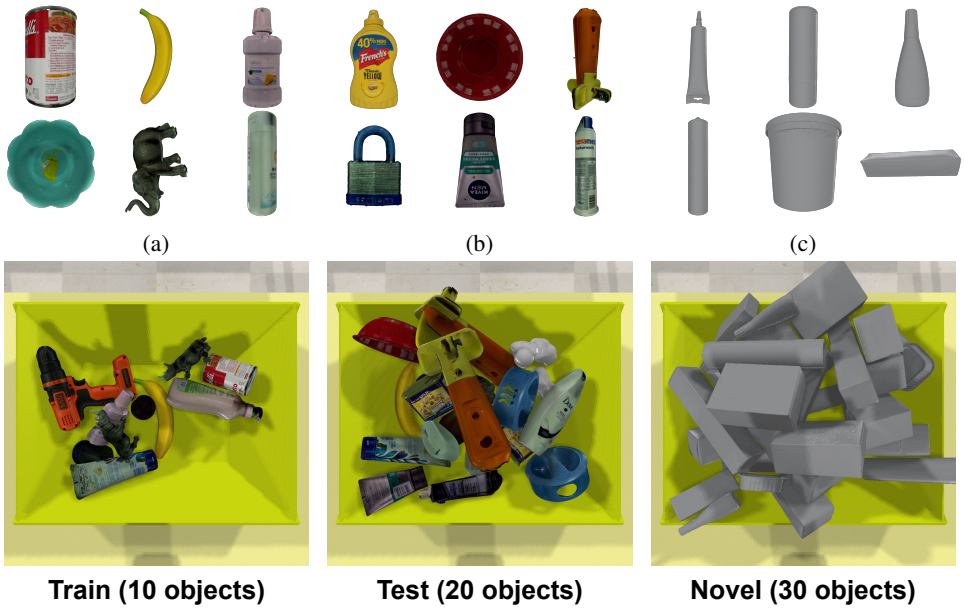


Figure 6: Sample objects. (a) Training dataset from GraspNet train split. (b) Testing dataset from GraspNet test split. (c) Testing dataset of novel household objects. (d) Each dataset with 10, 20, 30 objects.

B.2 Training CODs

We use the ResNet34_8s as the CODs network structure, same as [6]. For each pair of images, we sample 100 pairs of match pixels on the objects, and 1500 pairs of non-match pixels for each of the three type of non-matches: object-to-object, object-to-background, background-to-background. See Figure 7 for examples of pairs of match and non-match pixels. The networks are trained for 120k iterations using the Adam optimizer [10] with a weight decay of $1e^{-4}$ and a batch size of 1 on a single Nvidia GTX-1080 Ti and a Xeon CPU at 2GHz. The learning rate was set to $1e^{-1}$, and it decays by 0.9 every 5k iterations. The descriptor vector dimension is 8, and M is 0.5.

B.3 Training Picking Policy

PPO settings. We use the Adam optimizer [10] with a learning rate of 0.0005 and a momentum of 0.9. The hyper-parameters for Actor-Critic are the following: entropy coefficient is beta is 0.001, the clipping parameter epsilon is 0.2, and the discount factor is 0.3. The rewards are as follows: $+0.1$ for each successful pick, -0.1 for a failed pick, and -1 for terminal steps.

Terminal conditions. An episode terminates upon fulfillment of any of the following conditions. a) **Completion.** All objects have been successfully picked. b) **Action number limit.** Number of actions exceed 2 times of the number of objects at the beginning of the episode. c) **Control failures.** The robot arm is un-controllable or unsafe to operate with. For example, the robot arm collides with the basket, the table, or itself, and the robot arm controller fails.

(a) Match pixels. (b) Object-object non-match pixels.

Figure 7: Data generation. Match and non-match pixels with domain randomization.

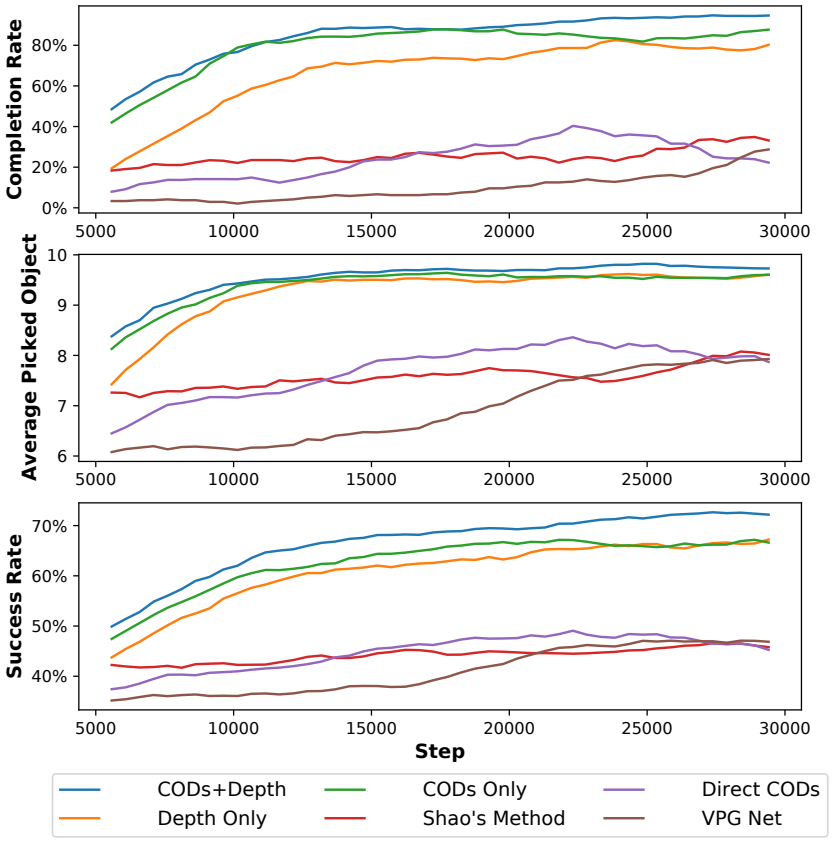


Figure 8: Evaluation metrics for all methods during training.