AE-Reorient: Active Exploration based Reorientation for Robotic Pick-and-Place

Hao Luo[1], Zhenyu Wu[1], and Haibin Yan[1]

Beijing University of Posts and Telecommunications, Beijing, China
{haroll_luo,wuzhenyu,eyanhaibin}@bupt.edu.cn

Abstract. Finding objects in dense clutter and placing them in specific poses play an important role in robot manipulation in fields like warehousing and logistics, and have a significant influence on the automation of these fields. However, most methods that perform well in simple clutter do not hold up well in dense clutter because of severe stacking and occlusion between objects, which prevents the target objects from grasping successfully. In this paper, we propose an interactive exploration framework called AE-Reorient based on reinforcement learning to let robots learn the interactive exploration autonomously. AE-Reorient enables robots to actively find the target objects and accomplish the pick-and-place task. Our AE-Reorient generates iterative and interactive actions to improve the visibility of the target objects which is an important factor affecting the success of grasping. Specifically, we first collect the scene information through an RGB-D camera. Then, using the point cloud reconstruction and pose estimation methods to obtain the height map of the clutter, through which we generate the grasp pose or predict the position. We apply interaction to change the state of the scene by pushing if the grasp is unsuccessful. The generated push action is applied to disrupt and recreate a new state of the clutter with the target objects more exposed, which greatly improves the probability of grasp success. Robots can successfully perform pick-and-place tasks in dense clutter by applying our proposed framework. Extensive experiments demonstrate that our AE-Reorient can significantly improve the visibility of target objects in dense clutter and thus improve the grasp success rate.

Keywords: Active Exploration · Interaction · Reorientation · Robotic · Pick-and-place Tasks

1 Introduction

In recent years, vision-based robotic manipulation has become mainstream due to the gradual maturity of computer vision technologies [5,14,17], which enhances the robot's ability to adapt autonomously in complex environments. However, in fields such as warehousing and logistics, finding target objects from dense clutter and placing them in specific poses remains a complex task for robots. Compared

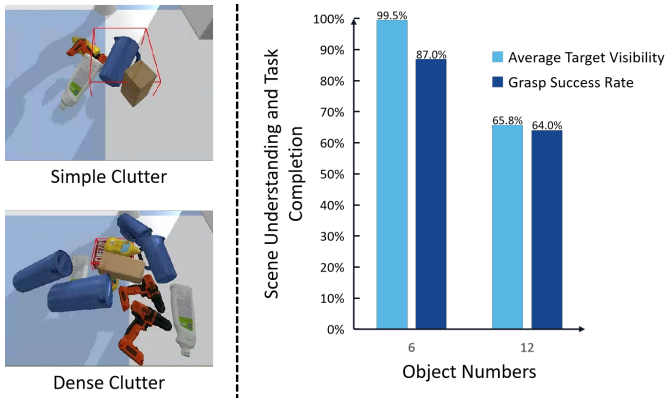

Fig. 1. Relationship between visibility and grasp success. Two scenes with an average of 6 objects and 12 objects. In the dense clutter with more occlusion, the decreased visibility of the target object led to a decrease in the success rate of the grasp.

with traditional robotic pick-and-place tasks, the robot needs to perform higher-level scene understanding to locate the target object and consider its placement pose.

Because of the limitations of the structure or workspace, robots are not always capable of placing objects successfully in one attempt. Motivated by that a supermarket staff turns the product label outward before putting it on the shelf, reorientation methods have been introduced to robot pick-and-place tasks by using intermediate poses to solve the problem of the inability to reach the final pose in one step. The reorientation method has been studied as one of the basic operations in lots of robotic tasks. Traditional reorientation methods have applied pre-designed poses of reorientation for continuous attempts, which requires multiple steps to complete the entire reorientation process. Due to the complexity of robotic tasks, multi-step reorientation methods are not efficient enough and the intermediate state is difficult to understand, which leads to the randomness in task completion. Some researchers have optimized multi-step reorientation by employing models to predict or sample abstract geometric features to find stable reorientation poses. The reorientation poses obtained from these methods have more interpretability, but sacrifice efficiency to a certain extent. Other studies have selected unstable reorientation poses as intermediate states and the released object tends to stabilize on its own, which could improve the efficiency. For the single-step reorientation methods, they are limited by the complexity of the scene and perform poorly in complex scenes due to severe stacking and occlusion. Fig. 1 illustrates that as the complexity of the scene increases, the performance of robots in completing the pick-and-place task decreases significantly.

In this paper, we propose a framework named AE-Reorient to actively explore the scene, which allows the robot to autonomously learn scene interaction strategies through reinforcement learning to search for target objects in clutter. We apply a single-step reorientation method to regrasp the object in an unstable

state and finally place it in a specific pose. More specifically, we collect scene information through an RGB-D camera. Point cloud reconstruction and pose estimation methods are applied in scene understanding to form a height map. We detect the target object in the scene and attempt to grasp it. Active exploration around the target object will be initiated if the grasp fails. AE-Reorient executes the push action to explore the scene and aims at finding the target object. Meanwhile, we propose a reinforcement learning framework. The height map mentioned above is used as the state of the framework. We apply a fully convolutional network to predict the origin and direction of the push in the height map. After the push, we feed back the change of the scene, including the value evaluation of the push action, the visibility of the target object, and the success rate of grasp, as the reward to AE-Reorient. Our contributions are as follows:

1) We propose AE-Reorient for a robotic pick-and-place system that is capable of autonomously exploring complex scenes, searching for and grasping target objects, and placing them in specific poses.
2) We also propose a reinforcement learning framework for robotic scene interaction and active exploration. Our proposed AE-Reorient employs RGB-D information to predict the position for pushing interaction.
3) We train and test our AE-Reorient extensively in simulated complex scenes to verify the significant improvement in the robot's ability to complete target searching and specific pose placement tasks.

2 Related Works

In this section, we briefly introduce the previous work from two aspects of reorientation and regrasping and active interaction exploration.

Reorientation and Regrasping: Object reorientation and regrasping is an important link for the robot to find an intermediate state to transport objects from the initial state to the final state during the pick-and-place task. The existing reorientation and regrasping methods are mainly based on planning. Cheng *et al.* [2] proposed a robot system, which takes part of the objects and point clouds of the supporting environment as inputs and outputs a series of picking and placing operations. They solved the robot manipulation that needs to grasp again due to the diversity of geometric structure in daily objects and the high dimension of state and motion space. Xu *et al.* [16] applied deep neural networks to generate diverse placements of objects on the plane, and proposed a pipeline containing orientation generation, position refinement, and placement discrimination to get stable reorientation poses. Raessa *et al.* [8] proposed a hierarchical motion planning method considering external support surfaces, which constructs a graph taking the grasping attitude as the node and the object attitude as the edge to describe robot manipulation planning. As for the application of object reorientation, Wermelinger *et al.* [13] proposed an approach for grasp planning and object manipulation to reorient highly irregular objects using point clouds and mesh surface reconstruction of objects. Wada *et al.* [9] applied a learned

model to predict the probability of a collision-free path between a pair of waypoints, which improves the efficiency of object reorientation planning. In addition, some research has been conducted on the basis of in-hand manipulation. Yuan *et al.* [19] designed a robot grasper to operate grasp manipulation in a non-anthropomorphic method, which improved the efficiency and success rate of grasping and reorientation. Cruciani *et al.* [4] defined planning for in-hand manipulation and regrasping on a manipulation graph to describe the regrasping problem as a graph search problem, and they proposed a dual-arm system to perform the manipulation through dual-arm coordination.

Active and interactive exploration: Active exploration as part of active learning is already widely applied in a number of other fields [10–12]. Recently, there has been a lot of research on active vision, especially in the field of robotics, which aims to make robots explore scene information more actively. Wu *et al.* [15] proposed an active exploration framework named Smart Explorer to accurately recognize objects in dense clutter scenes. Liu *et al.* [6] proposed GE-Grasp framework to improve the success rate of object grasping in dense scenes with active interaction. Ye *et al.* [18] applied an intrinsic-extrinsic reward setting to train a model for object searching. Novkovic *et al.* [7] proposed an interactive perception system based on reinforcement learning to perform physical interaction. In this work, we use active physical exploration to interact with the scene to help the robot to find the target object in dense clutter.

3 Approach

In this section, we first briefly introduce the pipeline overview of our AE-Reorient, and then introduce the mathematical description of the active interaction problem. After that, we detail the process of push action generation and provide details of training.

3.1 Pipeline Overview

Our basic task is to find the target object in dense clutter and arrange it in a specific position. However, it is usually difficult to find the appropriate grasp position in dense clutter because of the severe stacking and occlusion. We use physical interaction to disturb the clutter to make the target object more exposed, which is confirmed to have a strong correlation with the success of grasp.

Fig. 2 shows the overall pipeline of our AE-Reorient. The RGB-D image is collected to form a depth map and a color map to help the robot to understand the scene. Through the depth map, we use a fully convolutional network(FCN) to predict the position and direction of the push. After the interaction, we put the change of target visibility, push success, and whether the grasp is successful as rewards into the FCN. When the grasp is successful, the reorient and place planner plans collision-free paths for reorientation and placement and then the robot executes. We apply push as the action of our active scene interaction, which is accomplished by moving the end-effector perpendicular to the desktop

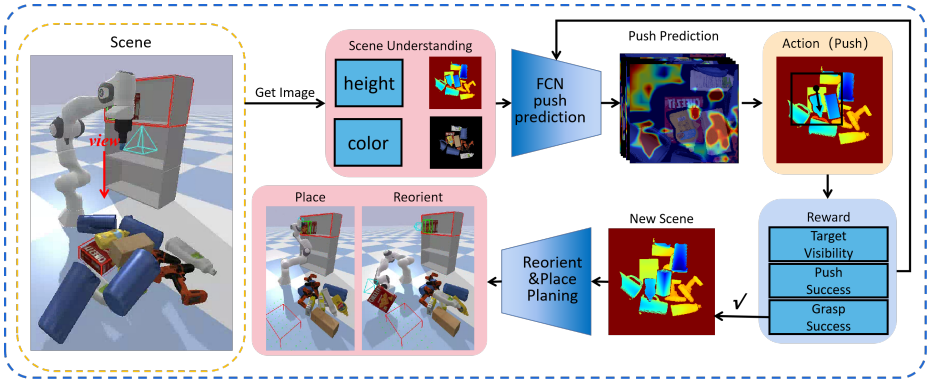

Fig. 2. Pipeline Overview of our AE-Reorient. The scene point cloud reconstructed by the RGB-D camera of the top-down view will be sent to the agent to generate iterative and interactive actions. When the visibility of the object is large enough that the object can be grasped successfully, the robot performs subsequent manipulations.

in a direction parallel to the desktop. Through the method of reinforcement learning, the robot can gradually learn the best pushing position and direction in the process of each execution, to maximize the exposure of the target object, which can improve the possibility of grasping success.

3.2 Problem Formulation

We formulate the task of active interaction as a Markov decision process: at a certain state s_t at a given moment t, the robot takes certain actions a_t according to the strategy π, gets the new state s_{t+1} at the next time $t+1$, and obtains the current reward $R_{a_t}(s_t, s_{t+1})$. The expected reward is as follows:

$$R(s_t, a_t) = \Sigma_{t=0}^{\infty} \gamma^t R_{a_t}(s_t, s_{t+1}) \tag{1}$$

where γ represents the attenuation factor, indicating that with the continuous progress of the decision, the reward will continue to decline.

When different decision functions are defined, different rewards will be obtained. The relationship between decision and reward is described as a cumulative reward by the following functions:

$$V^{\pi}(s) = R(s_0) + \gamma \sum_{s_1} V^{\pi}(s) \tag{2}$$

In this equation, $R(s_0)$is the current reward of the decision taken in the current state, and the second part is the expected reward in the future state.

Once the mathematical relationship between the decision and the reward is confirmed, the goal is to maximize the cumulative reward function $V^{\pi}(s)$ from any state:

$$V^*(s) = \max_{\pi} V^{\pi}(s) = R(s) + \max_{\pi} V^{\pi}(s_1) \tag{3}$$

Our actions are generated by the strategy function $a = \pi(s)$. In the cumulative reward functions, the strategy function is the variation we need to optimize. It can be seen from the above function that the optimal reward corresponds to the optimal decision. The decision function corresponding to the maximized cumulative reward is the optimal decision, and the cumulative reward corresponding to the optimal decision is also the largest, so the optimal decision is as follows:

$$\pi^*(s) = \arg\max_{a \in A} \sum_{s_i \in S} V^*(s, a) \tag{4}$$

where A is the set of all actions and S is all available states.

In this work, our state is the height map generated by the input RGB-D image. Since it requires physical interaction to change the scene, we collect image information again after each interaction, and then updates the state information. As shown in Fig.2, our action is the starting point and direction of the push action generated by the FCN, so the adjustment of our policy function is to adjust the network parameters of the FCN. Our goal is to help the robot to successfully find and grasp the target object in the clutter. Therefore, the goal of strategy selection is to make the interaction more effectively. Based on experimental results, we have found that the visibility of target objects has a significant impact on the success of the grasping task, so we use a series of rewards focusing on visibility to train the robot to find the most effective interaction strategy that can best improve the visibility of the target objects.

A reinforcement learning reward scheme is designed to restrict the robot from finding the most effective push action to improve the visibility of target objects. We use the mask map before and after each interaction to calculate the visibility of the target object in $Vis_{imp} = targetvisibility_{(new)} - targetvisibility_{(pre)}$ and set a series of thresholds to judge the visibility improvement and assign a reward value, which is calculated as follows:

$$V_{(s_t, s_{t+1})} = \begin{cases} -0.1, Vis_{imp} < 0 \\ 0.1, \quad 0 < Vis_{imp} < 0.1 \\ 0.3, \quad 0.1 < Vis_{imp} < 0.3 \\ 0.5, \quad Vis_{imp} > 0.3 \end{cases} \tag{5}$$

In addition, the robot may have some interactions that lead to changes in the scene but do not improve the visibility of the object, so we calculate the difference of the height maps before and after the interaction. When the difference is greater than the threshold, we add 0.2 to the reward. When the grasp is successful, which means the task can be accomplished, we add 1 to the reward.

3.3 Push Generation

Inspired by [20], we use FCN to predict the starting position and direction of interactive actions. Our interaction is a push action, where we control the end of the robot perpendicular to the tabletop and move it in a direction parallel to the tabletop. The used end-effector is a suction gripper, which can directly

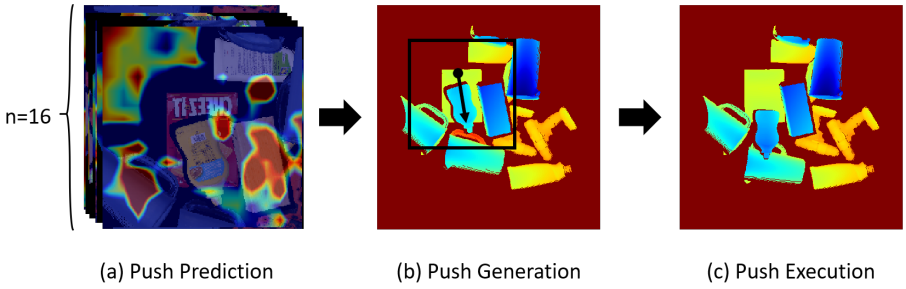

Fig. 3. Process of the push generation and execution. (a) shows the input when we use the FCN to predict the push action, which is 16 height maps in different directions; (b) shows the motion generation and the region of interest (ROI) we set; (c) shows the height map formed after re-obtaining information from the RGB-D camera after the execution of the interactive action.

push the object without initiating the grab. The network will give the three-dimensional coordinates of the starting point and direction of the push, and the end of the actuator will reach the specified point vertically on the tabletop and push a certain distance along the direction, which we set as 20cm. For the choice of pushing direction, we discretize a circle (360°) into 16 directions and evaluate pushing motions for points in the height map in each direction.

Fig. 3 shows the process of generating and executing push actions. First, the height map is fed into the FCN to predict the start point and direction of the push. Note that the height map is rotated 16 times to predict the value evaluation of push actions in each discrete direction. Then, we select the one with the highest score from all the predicted results to generate the push action, which is transformed into the robot's push action after a coordinate transformation, and the scene after the interaction is obtained.

As shown in Fig. 3b, to reduce searching cost, we limit the prediction of push point to the area around the target object and intercept a 224 × 224 pixel window as the region of interest(ROI) on the original 448 × 448 pixel image with the target as the center. The specific method is as follows: first, we find all the pixels occupied by the target object in the mask map, calculate the central coordinates of the target object, and then move 112 pixels from this point along the negative direction of the x and y axes. Depending on this point, we intercept 224 pixels forward along the x and y axes to form the ROI window. When the captured window is out of the original image, we limit the ROI window to the edge of the original image. Note that the coordinate transformation is required when giving the position to the robot for push action.

The interactive action is designed to better complete the pick-and-place task. Only when the grasp fails, the robot will start active interaction to expose and find the target objects. In the training process, we set the exit conditions of interactive actions: (1)Exit the interaction when the grasp is successful, (2)Exit the interaction when the visibility of the target object is greater than 90% but still cannot be grasped successfully or the number of interactions is greater than 10. Since in this case, interaction cannot help to complete the pick-and-place

Fig. 4. The objects selected in our experiments in the simulation environment.

task. (3)Other situations, such as when the robot pushes the target object out of sight, so that the grasping task is no longer possible.

4 Experiment

In this section, we conduct extensive experiments in a PyBullet [3] simulation environment to validate our proposed AE-Reorient. The main goal of the experiments is to verify that (1) our active interaction framework can effectively improve the grasping success rate in heavily occluded scenes, and (2) the reinforcement learning-based interaction generation method outperforms random generation.

4.1 Implementation Details

In the simulation scene, the workspace is a rectangular region with $0.896m \times 0.896m$ to ensure that the robot arm can work properly. The whole scene is perceived by a top-down view RGB-D camera with the resolution set to 640×480 pixels. In the world coordinate system, we set the height map voxelization resolution to $2mm$ and discretize the robot arm XY plane workspace into a grid of 448 pixels. To further improve the effectiveness of the interaction and reduce the computational cost, we crop the 224×224 pixels region in the height map centered on the target and feed it into the agent to generate the push action. In our experiments, we set an upper limit of 10 for the number of interactions in order to avoid numerous ineffective interactions.

The objects of the dense clutter scenes constructed in the experiments are from the YCB [1] dataset, which contains 79 3D models of everyday objects with high-resolution textured RGB images. To ensure the validity of the constructed scenes, we apply the following two rules to filter the objects: (1) the selected objects require good visibility in the depth map in order to reconstruct the scene point cloud, and (2) the selected objects could be grasped by the robot arm employed. We finally selecte 7 objects for the experiments, as shown in Fig. 4. All experiments are conducted on one NVIDIA GTX 2080Ti GPU.

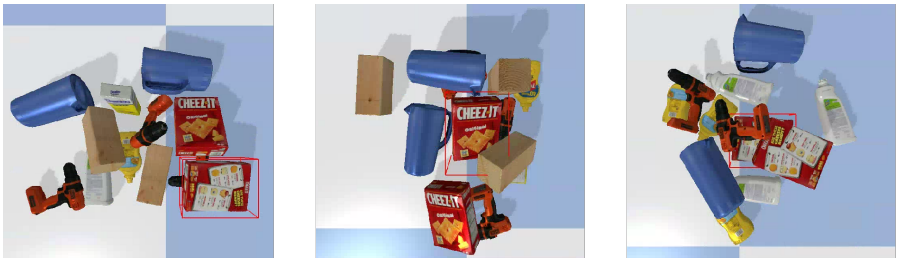

Fig. 5. Visualization of randomly generated scenes. Red boxes mark the target objects for each scene.

Fig. 6. Challenging scene visualization. The objects in each scene are manually adjusted to expect a higher degree of occlusion. The visibility of the target objects in each scene is less than 20%.

4.2 Evaluation Metrics

We evaluate our proposed AE-Reorient in terms of overall execution success and efficiency of executive actions. In the first regard, we leverage the overall success rate of pick and place (including reorient and place) to demonstrate the performance improvement of the proposed framework on dense clutter scenes. Second, we verify the effectiveness of the interaction actions with the visibility of the final grasp. Finally, the efficiency of action execution is measured by the robot arm execution time, which is employed to illustrate the balance between the accuracy and time cost of AE-Reorient.

4.3 Results and Discussions

We perform interactive actions on a Franka Emika Panda robot equipped with a suction gripper, where the solution of the kinematic equations for each joint

Table 1. Comparison on randomly generated and challenging scenes. We report the average of the three measures of overall execution success, final grasp visibility, and execution time, where the success rate and time are only calculated for the scene with successful manipulation.

Setting	Method	Success(%)	Visibility(%)	Time(s)
Random	ReorientBot	23.10	84.60	19.20
	ReorientBot+Random	27.70	85.90	24.88
	AE-Reorient	30.70	88.70	30.52
Challenging	ReorientBot	0.00	15.58	-
	ReorientBot+Random	0.00	37.92	-
	AE-Reorient	16.67	49.68	32.61

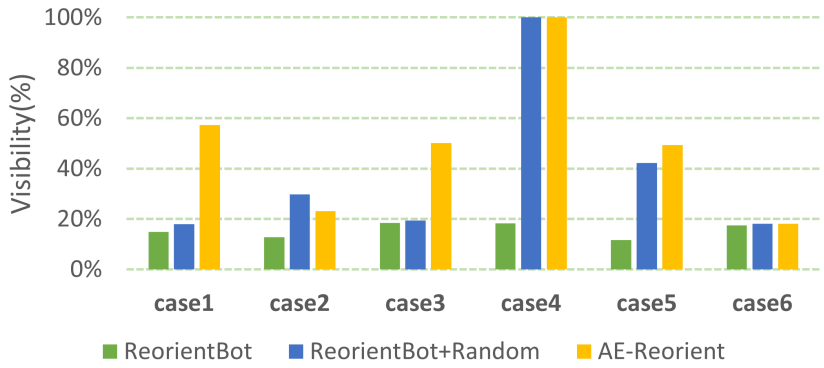

Fig. 7. Visibility comparison results with challenging scenes, AE-Reorient has significant improvement compared to other methods.

of the robot is implemented by a Pybullet built-in module. To verify the effectiveness of AE-Reorient interaction generation, we compare our approach with ReorientBot and the random interaction ReorientBot framework. The average execution time of each scene is also reported to demonstrate the efficiency-success trade-off. In the test scenes, only cases with unsuccessful grasp or target visibility less than 90% will be explored interactively. To facilitate comparison with the selected baseline, we only count the robot-specific operation time, excluding the inference time.

Random clutters: Objects fall into a set workspace in a random order in free fall, where the initial pose and position of each object is random. Since the difficulty of grasping is positively correlated with the object clutter density, we increase the ReorientBot experiment setting of 6 objects to 12, which significantly reduces the visibility of the target objects as shown in Fig. 5. The experimental results of random clutters are shown in Table 1 in the row with the setting as *Random*, where our proposed AE-Reorient significantly enhances the success rate and visibility by 7.60%(30.70% vs. 23.10%) and 4.10%(88.70% vs. 84.60%), compared with the baseline method without active interaction by only increasing the 11.32s execution time. It demonstrates the effectiveness of our proposed AE-Reorient, which further indicates the effectiveness of the reward mechanism based on the degree of visibility and push interaction.

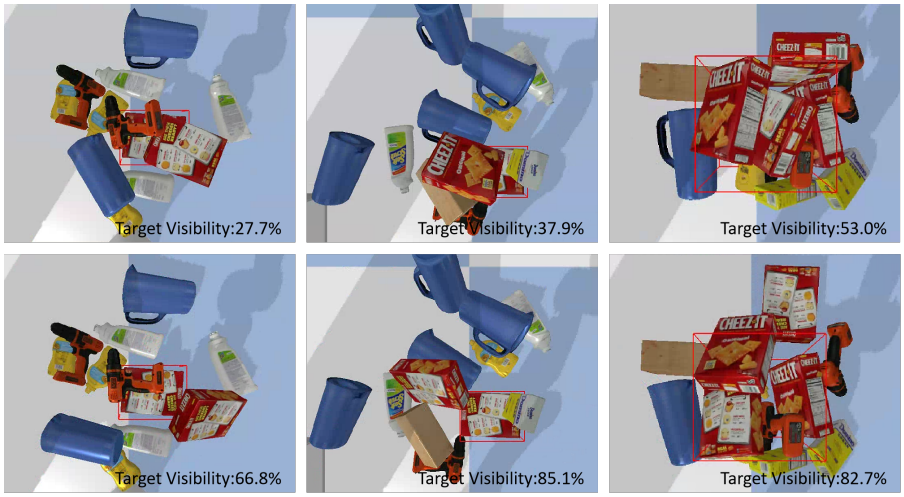

Fig. 8. Visualization of the results of the interaction manipulations, with each row representing the execution sequence of AE-Reorient. As the interaction continues, the visibility of the target object increases significantly.

Challenging clutters: To further validate the effectiveness of AE-Reorient in dense clutter scenes, we evaluated the proposed framework in 6 challenging scenes with less than 20% visibility of the target object as shown in Fig. 6. The experimental results of challenging clutter are shown in Table 1 in the row with the setting as *Challenging*, where our AE-Reorient enhances success rate 16.67%(16.67% vs. 0.00%) compared with the baseline and increases the average target visibility by 34.10%. Since ReorientBot failed to grasp the challenging scene, we cannot count the execution time. Meanwhile, due to the increased complexity of clutter, random interactions increase negligible success rates. Fig. 7 demonstrates the comparative results of visibility for each challenging scene. Due to the overly complex nature of challenging scenes, randomly generated interactions have the potential to enhance the visibility of target objects and achieve higher boosts in a collision-intense manner. In contrast, our AE-Reorient is constrained by the spatial geometry information during interaction and still outperforms the randomly generated approach in general and shows a significant improvement in success rate and visibility, further illustrating the effectiveness of the proposed framework. Fig. 8 illustrates the visualization results of our AE-Reorient interaction, where the visibility of the heavily occluded targets is enhanced to improve the success rate of subsequent manipulations.

5 Conclusion

In this paper, we have proposed AE-Reorient for the pick and place task in dense and cluttered scenes, which decreases the occlusion of target objects and improves the success rate of grasping and placing by active pushing. We reconstruct the scene point cloud with RGB-D sensors and feed the discretized height

map to a reinforcement learning-based agent to generate specific interaction actions to enhance the visibility of the target objects. The exploration of active interaction provides an effective success rate versus execution time tradeoff for grasping target objects in cluttered scenes. Extensive experiments demonstrate the effectiveness and efficiency of the proposed AE-Reorient.

References

1. Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. *arXiv preprint arXiv:1502.03143*, 2015.
2. Shuo Cheng, Kaichun Mo, and Lin Shao. Learning to regrasp by learning to place. *arXiv preprint arXiv:2109.08817*, 2021.
3. Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016.
4. Silvia Cruciani, Kaiyu Hang, Christian Smith, and Danica Kragic. Dual-arm in-hand manipulation and regrasping using dexterous manipulation graphs. *arXiv preprint arXiv:1904.11382*, 2019.
5. Sichao Huang, Ziwei Wang, Jie Zhou, and Jiwen Lu. Planning irregular object packing via hierarchical reinforcement learning. *RAL*, 8(1):81–88, 2022.
6. Zhan Liu, Ziwei Wang, Sichao Huang, Jie Zhou, and Jiwen Lu. Ge-grasp: Efficient target-oriented grasping in dense clutter. In *IROS*, pages 1388–1395, 2022.
7. Tonci Novkovic, Remi Pautrat, Fadri Furrer, Michel Breyer, Roland Siegwart, and Juan Nieto. Object finding in cluttered scenes using interactive perception. In *ICRA*, pages 8338–8344, 2020.
8. Mohamed Raessa, Weiwei Wan, and Kensuke Harada. Planning to repose long and heavy objects considering a combination of regrasp and constrained drooping. *Assembly Automation*, 41(3):324–332, 2021.
9. Kentaro Wada, Stephen James, and Andrew J Davison. Reorientbot: Learning object reorientation for specific-posed placement. In *ICRA*, pages 8252–8258, 2022.
10. Ziwei Wang, Jiwen Lu, Ziyi Wu, and Jie Zhou. Learning efficient binarized object detectors with information compression. *PAMI*, 44(6):3082–3095, 2021.
11. Ziwei Wang, Jiwen Lu, and Jie Zhou. Learning channel-wise interactions for binary convolutional neural networks. *PAMI*, 43(10):3432–3445, 2020.
12. Ziwei Wang, Han Xiao, Yueqi Duan, Jie Zhou, and Jiwen Lu. Learning deep binary descriptors via bitwise interaction mining. *PAMI*, 2022.
13. Martin Wermelinger, Ryan Johns, Fabio Gramazio, Matthias Kohler, and Marco Hutter. Grasping and object reorientation for autonomous construction of stone structures. *RAL*, 6(3):5105–5112, 2021.
14. Zhenyu Wu, Ziwei Wang, Jiwen Lu, and Haibin Yan. Category-level shape estimation for densely cluttered objects. *arXiv preprint arXiv:2302.11983*, 2023.
15. Zhenyu Wu, Ziwei Wang, Zibu Wei, Yi Wei, and Haibin Yan. Smart explorer: Recognizing objects in dense clutter via interactive exploration. In *IROS*, pages 6600–6607, 2022.
16. Peng Xu, Zhiyuan Chen, Jiankun Wang, and Max Q-H Meng. Planar manipulation via learning regrasping. *arXiv preprint arXiv:2210.05349*, 2022.
17. Xiuwei Xu, Ziwei Wang, Jie Zhou, and Jiwen Lu. Binarizing sparse convolutional networks for efficient point cloud analysis. *arXiv preprint arXiv:2303.15493*, 2023.

18. Xin Ye and Yezhou Yang. Efficient robotic object search via hiem: Hierarchical policy learning with intrinsic-extrinsic modeling. *RAL*, 6(3):4425–4432, 2021.
19. Shenli Yuan, Lin Shao, Connor L Yako, Alex Gruebele, and J Kenneth Salisbury. Design and control of roller grasper v2 for in-hand manipulation. In *IROS*, pages 9151–9158, 2020.
20. Andy Zeng, Shuran Song, Stefan Welker, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. In *IROS*, pages 4238–4245, 2018.