2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Learning Pre-Grasp Manipulation of Flat Objects in Cluttered Environments using Sliding Primitives

Jiaxi Wu[1], Haoran Wu[2], Shanlin Zhong[3], Quqin Sun[4], Yinlin Li[3]

***Abstract*— Flat objects with negligible thicknesses like books and disks are challenging to be grasped by the robot because of the width limit of the robot's gripper, especially when they are in cluttered environments. Pre-grasp manipulation is conducive to rearranging objects on the table and moving the flat objects to the table edge, making them graspable. In this paper, we formulate this task as Parameterized Action Markov Decision Process, and a novel method based on deep reinforcement learning is proposed to address this problem by introducing sliding primitives as actions. A weight-sharing policy network is utilized to predict the sliding primitive's parameters for each object, and a Q-network is adopted to select the acted object among all the candidates on the table. Meanwhile, via integrating a curriculum learning scheme, our method can be scaled to cluttered environments with more objects. In both simulation and real-world experiments, our method surpasses the existing methods and achieves pre-grasp manipulation with higher task success rates and fewer action steps. Without fine-tuning, it can be generalized to novel shapes and household objects with more than 85% success rates in the real world. Videos and supplementary materials are available at `https://sites.google.com/view/pre-grasp-sliding`.**

I. Introduction

Grasping is a fundamental research topic in robotics [1][2]. It plays a significance role in manipulation tasks such as sorting [3] and assembly[4]. A common assumption made in previous works about robotic grasping is that the target object is placed in a graspable pose, and the gripper can directly pick the target from at least one orientation. However, this assumption fails when facing flat objects horizontally placed on the table. These flat objects, such as books and disks, are too thin or wide for the robot's gripper to be picked up directly.

Then a grasping strategy is to push or slide the flat objects to the table edge and pick them from their sides. This kind of robotic manipulation before grasping is called "pre-grasp" [5]. There have been some hand-crafted methods and learning-based methods to deal with pre-grasp manipulation of a single object [6] [7]. However, in most cases, the target object intended to pick is in object piles, surrounded by other objects. This means that there is no collision-free path to push the target object to the table edge to make it graspable.

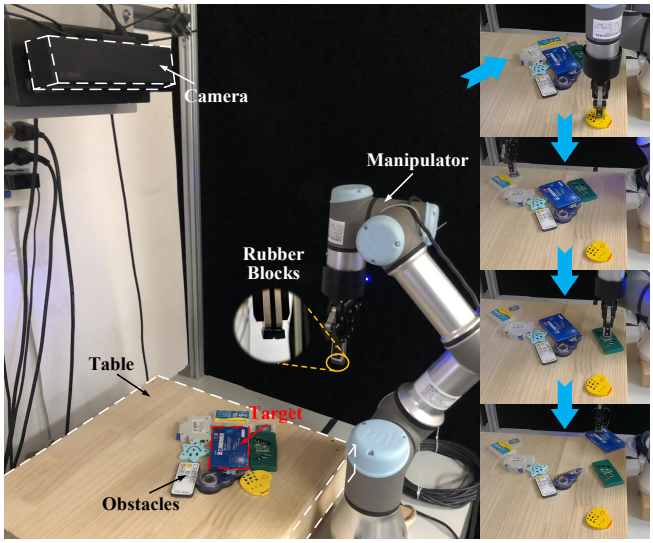


Fig. 1. Pre-grasp sliding manipulation of flat objects in cluttered environments. The robot needs to rearrange objects on the table and move the target object to the table edge, making it graspable.

This results in the failure of existing methods of pre-grasp manipulation for flat objects.

The utilization of non-prehensile actions, such as pushing, sliding, and shifting, can help rearrange the obstacles around the target and make it graspable [8] [9]. Then a prehensile grasping can be performed to pick the target. These non-prehensile actions can also be seen as pre-grasp manipulation which make the target object graspable via a series of action primitives. Nevertheless, existing methods to deal with cluttered environments mainly focus on small objects which can be picked vertically [10] [11]. These methods are more concerned with the partial or total singulation of the target objects from other obstacles.

For flat objects, those manipulations are not sufficient to make them graspable [7]. The target object needs to be pushed to the table edge and exposed enough distance for the gripper to pick it, not just be separated from other obstacles. Meanwhile, all the objects cannot be dropped under the table, avoiding damage to these objects or unnecessary downstream tasks. Therefore, pre-grasp manipulation for flat objects in cluttered environments is a more challenging problem to the existing methods, including hand-crafted methods and learning-based methods. Especially when the object number on the table increases, it is more difficult to rearrange the objects to make the target graspable without dropping any object under the table [12].

In this paper, we propose a novel method based on deep

[1] J. Wu is with the State Key Laboratory of Turbulence and Complex Systems, Intelligent Biomimetic Design Lab, College of Engineering, Peking University, Beijing 100871, China.
[2] H. Wu is with the Department of Automation, University of Science and Technology of China, Heifei 230026, China.
[3]S. Zhong and Y. Li are with the State Key Laboratory of Management and Control for Complex System, Institute of Automation, Chinese Academy of Science, Beijing 100190, China, `yinlin.li@ia.ac.cn`
[4] Q. Sun is with the Science and Technology on Thermal Energy and Power Laboratory, Wuhan 430205, China.

reinforcement learning to make flat objects graspable in cluttered environments. Via introducing sliding primitives as actions, the pre-grasp task is formulized as Parameterized Action Markov Decision Process (PAMDP) [13] with hybrid actions. A weight-sharing policy network is designed to predict the sliding primitive's parameters (continuous action) for each object on the table, and a Q-network is utilized to evaluate these actions and select the acted object (discrete action) among all the candidates. Through a sequence of non-prehensile sliding primitives, object piles can be rearranged, making a collision-free path for the target, and then the flat target can be slid to the table edge for next-step grasping.

Furthermore, curriculum learning (CL) is utilized to extend our method into cluttered environments with more objects, overcoming the exploration problem in reinforcement learning [14]. We define a schedule in which the mean object number increases gradually in the environment. The object number in each episode is sampled from a truncated normal distribution to help the policy trained in scenes with fewer objects generalize to more objects.

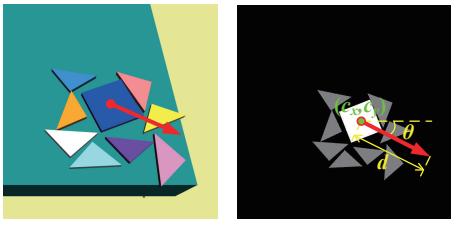
The effectiveness of our method is verified in simulation and reality, both surpassing the existing methods. Moreover, without training in the real world, the pre-grasp policy learned in simulation can be generalized to novel shapes and household objects with high task success rates, demonstrating excellent robustness and generalization. In summary, the main contributions of this paper are as follows:

- A pre-grasp sliding policy with discrete-continuous action space is learned to rearrange the object piles and make the flat object graspable in cluttered environments.
- Without being restrained by object numbers, our method integrates curriculum learning scheme to scale the pre-grasp policy to complex scenarios with more objects.
- The pre-grasp policy is not only evaluated in simulated cluttered scenes, but also in real-world scenarios without extra fine-tuning, of which the results validate the effectiveness and generalization.

II. Method

A. Sliding Primitive

It is challenging for the agent to learn an appropriate pre-grasp policy in cluttered environments because of the overall environmental complexity. Especially, as the number of objects increases, it becomes increasingly difficult to plan a proper path to move the target to the table edge without letting other objects fall off the table. Therefore, the sliding primitive (c_x, c_y, d, θ) is introduced to limit the action space in this paper, where (c_x, c_y) is the acted object's center, d is the sliding distance, and θ is the sliding orientation, as shown in Fig. 2. The robot performs a sliding action (d, θ) of object k to make it move to a specified position. Since the object index k is one-to-one with its center (c_x, c_y), the sliding primitive can also be expressed as (k, d, θ).



(a) Simulated scene (b) Sliding primitive

Fig. 2. The sliding primitive (c_x, c_y, d, θ) is utilized as the basic actions of the robot in this paper to limit the action space.

B. Hybrid Discrete-Continuous Action via Weight Sharing

After introducing the sliding primitive, the agent's action is the acted object index k (discrete variable) and the corresponding action parameters $a^k : (d, \theta)$ (continuous variable), i.e., its action space is reduced to (k, a^k). This hybrid action space problem can be abstracted as PAMDP [13], which consists of finite actions, each parameterized by m continuous action parameters.

An intuitive method to solve the above PAMDP problem is to use a separate policy network $\pi_\varphi(s)$ for each object to choose the action a^k, followed by another Q-network $Q_\omega(s, a^1, ..., a^n)$ to select the acted object index k. However, this method needs $n + 1$ networks for n objects, which will cause difficulties in network convergence as the object number increases.

In our work, the action a^k for different objects has similarities, which can all be represented as (d, θ). Therefore, the network parameters of n policies $\pi_\varphi(s)$ can be shared to improve the data efficiency and final performance. To distinguish various objects, the object's observation o^k is input into the network as state variables, i.e., using $\pi_\varphi(s, o^k)$ to generate the action a^k acting on object k. In addition, another Q-network $Q_\omega(s, o^k, a^k)$ is used to select the object index k^* with the largest Q-value as the acted object.

$$k^* \leftarrow \arg\max Q(s, o^k, \pi(s, o^k)). \tag{1}$$

Then the Bellman equation can be rewritten as

$$Q(s, o^k, a^k) = \mathop{\mathbb{E}}_{r,s'}\left[r+\gamma \max_{k'} Q(s', o^{k'}, \pi(s', o^{k'}))\middle| s, o^k, a^k\right] \tag{2}$$

where s' is the next state, k' is the candidate object index, and $o^{k'}$ is the observation of the candidate object. Compared with the classical Bellman equation, the observation o^k of object k is taken as an input variable to help weight sharing among different objects.

For the policy network $\pi_\varphi(s, o^k)$, we aim to find the parameters φ to maximize the Q-value, when the parameters ω of the Q-network $Q_\omega(s, o^k, a^k)$ are fixed, i.e.,

$$\varphi \leftarrow \arg\max_{\varphi} Q(s, o^k, \pi_\varphi(s, o^k)|\omega). \tag{3}$$

Similar to DQN [15], the Q-network's parameters ω can be estimated by minimizing the mean-squared Bellman error via gradient descent.

$$\omega \leftarrow \arg\min_{\omega}(y_t - Q_\omega(s, o^k, a^k))^2 \tag{4}$$

where $y_t = r+\gamma \max_{k'} Q_{\omega'}(s', o^{k'}, \pi_{\varphi'}(s', o^{k'}))$. The Q-target y_t can also be extended to multi-step estimation to improve the performance further [16].

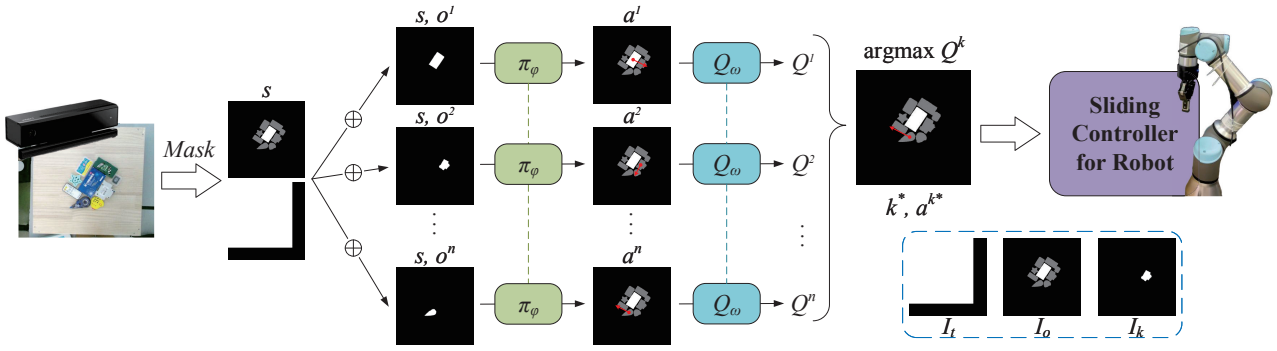


Fig. 3. Overview of our proposed method. A policy network π_φ is utilized to predict the sliding primitive's parameters for each object, and a Q-network Q_ω is adopted to select the acted object among all the objects on the table via Q-values. The policy network π_φ shares weights as denoted by the dashed lines, as does Q-network Q_ω.

Consequently, the loss function of policy network $\pi_\varphi(s, o^k)$ and Q-network $Q_\omega(s, o^k, a^k)$ can be expressed as

$$L_\pi = -Q_\omega(s, o^k, \pi_\varphi(s, o^k)) \tag{5}$$

$$L_Q = (y_t - Q_\omega(s, o^k, a^k))^2. \tag{6}$$

The above update formula is similar to DDPG [17], while its input state is changed to (s, o^k), and its output action is (k, a^k). However, as shown in Fig. 3, unlike DDPG, our method requires n forward operations for all objects to predict the corresponding actions a^k and Q-values Q^k and then selects the action a^{k^*} with the largest Q-value Q^{k^*} for execution.

C. Learning Pre-Grasp Sliding Policy

We aim to learn a suitable pre-grasp policy to make the target graspable in cluttered environments. It is necessary for the agent to know the shapes and distributions of objects on the table. DNN like DenseNet-121 are required to extract features from raw images in [9][11], whereas similar shapes and distributions of flat objects can be extracted from their masks with only several convolutional layers. Therefore, instead of raw images, we adopt the masks of the objects and table as the pre-grasp policy's state. Moreover, the mask states reduce the difficulty of sim-to-real transfer [7].

An example of the masks is shown in the bottom right of Fig. 3. The table mask I_t shows only the lower right part in the camera's view, which is helpful for the agent to determine the graspable position and orientation for the target. And the pixel values for the table are 1, otherwise 0. The object mask I_o is represented by the target and obstacles, where the pixel values for the target are 1, the pixel values for the obstacles are 0.5, and the others are 0. In addition, we take the table position, graspable (whether the target is graspable), and on-table (whether all the objects are on the table) as states, and the last two are 1/-1 binary values.

In our work, the object mask I_o is calculated through the objects' pre-collected masks and their current poses. Since our paper focuses on pre-grasp manipulation to make the object graspable, we obtain the masks as the policy's state using the above easy-to-deploy method. They can also be obtained by instance segmentation to extend our method to other scenarios.

Moreover, to distinguish the specific object, we input the candidate object mask I_k, position, and Euler angle into the networks (denoted by o_k in Fig. 3). It is worth noting that we do not simply adopt the object index k as its observation because an index k is not enough to locate a specific object since all the pixel values for obstacles are 0.5, when they are randomly distributed on the table.

The policy network's output π_φ is the action a^k: (d, θ) for the object k, and the Q-network's output Q_ω is the corresponding action value Q^k. As described in Eq. (1), the acted object k^* is selected as the object with the maximal Q-value. To prevent overestimation of Q-values, the Q-network Q_ω outputs two Q-values simultaneously, and the smallest one is selected as the final action value Q^k [18].

The reward function is essential for reinforcement learning. For complex reward functions, we often obtain undesired behaviors after optimization [19]. Therefore, we use the binary reward in this paper, whose policy is more robust than hand-designed reward functions. If the target is graspable and all objects are on the table, the reward is 0, otherwise -1, i.e.,

$$r = \begin{cases} 0, & \text{if graspable and on-table} \\ -1, & \text{otherwise} \end{cases} \tag{7}$$

where, graspable means the target needs to meet the following conditions: 1) the target's center of mass (CoM) is on the table; 2) there is enough portion of the target outside the table for the gripper to pick; and 3) the grasping point is reachable for the robot. On-table means that all the CoMs of objects are on the table, including the target and obstacles.

D. Scaling by Curriculum Learning

Compared to using a deterministic policy network $\pi_\varphi(s) : S \rightarrow \{A^1, ..., A^n\}$ that predicts n actions at a time in P-DQN [20] and MP-DQN [21], our method feeds the object observation o^k into the network, predicting a^k only. This means that our method can transfer the policy trained on small object piles to complex scenes with more objects, not constrained by the object number.

Therefore, the method is scaled to cluttered environments with more objects via curriculum learning [22], as illustrated in Algorithm 1. It helps alleviate the exploration problem in

Algorithm 1 Scaling by Curriculum Learning

Initialize networks $\pi_\varphi, \pi_{\varphi'}, Q_\omega, Q_{\omega'}$
Initialize curriculum parameter μ, σ, α
$\mu \leftarrow 2, \sigma \leftarrow \alpha \times (n-1)$
for epi $= 1, M$ **do**
  Sample n_i using truncated normal distribution
  $n_i \leftarrow \lfloor \mathcal{TN}(\mu, \sigma, l, h) \rfloor$
  Reset the environment with n_i objects
  Rollout and store one episode in replay buffer
  Perform one step optimization for several iterations
  Evaluate success rate ζ on scenes with μ objects
  Update μ every m episode using ζ
  $\mu \leftarrow \mu + \mathbb{1}_{(\zeta > \beta_h)} - \mathbb{1}_{(\zeta < \beta_l)}$
  if $\mu \geq n$ **then**
    Decay the exploration parameters δ, ϵ ▷ $\xi < 1$
    $\delta \leftarrow \delta \times \xi, \epsilon \leftarrow \epsilon \times \xi$
  end if
end for

cluttered environments to ensure that the target can still be separated from object piles and picked in complex scenarios.

To make the policy trained on fewer objects better transfer to scenes with more objects, truncated normal distribution $\mathcal{TN}(\mu, \sigma, l, h)$ is adopted for sampling to determine the current object number n_i in the environment, where l and h are the bounds for the truncated range. A super-parameter α is utilized to adjust the standard deviation σ of the truncated normal distribution. Meanwhile, the floor function $\lfloor \cdot \rfloor$ is adopted to convert n_i to an integer.

At the beginning of curriculum learning, the environment is reset to 2 objects with a higher probability, i.e., a target and an obstacle. When the agent learns to achieve the simple scenario with a success rate over β_h, we increase the distribution's mean μ to make the next environment more likely. In contrast, when the success rate ζ in the current environment is lower than a threshold β_l, we decrease μ to reduce the task difficulty. We continue repeating the above process until μ reaches the desired object number n.

To improve exploration, we add noise δ on actions and adopt an ϵ-greedy strategy, preventing the pre-grasp policy from falling into local optimums in the early stage. However, it is detrimental to policy convergence because this may result in task failure via changing the proper action and pushing the objects under the table, especially when facing more objects. Therefore, after the schedule, we gradually decay the exploration parameters δ and ϵ with a decay rate ξ to help the policy converge.

III. Experiments

A. Experimental Setting

A 6-axis UR3 robot with a RobotIQ85 gripper is used to perform pre-grasp manipulation in this paper, and the physical simulator is MuJoCo [23]. The agent's task is to make the target graspable in cluttered environments with all the objects keeping on the table, using sliding primitive defined in II-A. The target's shape is fixed to a square, and the obstacle's shape is fixed to a triangle during training.

At the beginning of each episode, n non-overlapping objects (including the target) are initialized on the table with random distribution. Nevertheless, the pre-grasp policy trained in simulation has high robustness and generalization to unseen shapes in the real world, as shown in III-D.

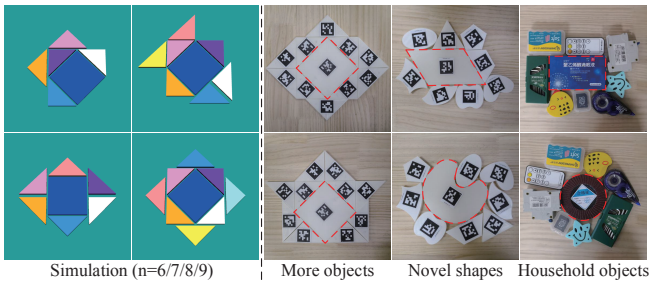


Fig. 4. Examples of challenging arrangements. The blue square in simulation or the object marked with red lines is the target object, and the others are obstacles.

We choose the task success rate (TSR) and mean number of actions (NoA) for 200 trials as the evaluation metrics to compare the performance among different methods. Meanwhile, some challenging arrangements for various object numbers are designed to evaluate the generalization of our method. In these scenarios, the objects are distributed more closely, and some examples are shown in Fig. 4.

B. Comparison to Baselines

We compare our method with the following baselines to prove the effectiveness of our algorithm. If not specifically emphasized, their state and action spaces are consistent with our method for fair comparison.

P-DQN: A method proposed by Xiong et al. to handle the hybrid discrete-continuous action space [20]. Unlike our method, it outputs all actions together and feeds those actions into the Q-network together, preventing it from scaling to complex environments with more objects by curriculum learning.

MP-DQN: A variant based on P-DQN proposed by Bester et al. [21]. It feeds the basis vector $\mathbf{xe_i}$ of actions to the Q-network to avoid false gradients caused by the dependence of Q-values.

Q-TD3: A modified version of our method using an auxiliary Q-network Q_ψ to select the acted object every step. The Q-network Q_ω in TD3 is just used to calculate the loss function of the policy network π_φ, providing the deterministic policy gradient.

CenterTD3: A standard TD3 version, and its action space is (c_x, c_y, d, θ), i.e., it can slide starting from arbitrary point (c_x, c_y) on the table.

TargetTD3: A variant of CenterTD3, and its action space is (d, θ). The start point (c_x, c_y) is limited to the target center, which means it only slides the target at each step and ignores the other objects.

The contrast results are illustrated in Fig. 5. Each method runs 5 random seeds in cluttered environments with 5 objects. Meanwhile, those methods are evaluated in random and challenging scenarios, and their results are shown in Table I. Our method achieves the highest task success rate and fewest number of actions, surpassing all baselines.

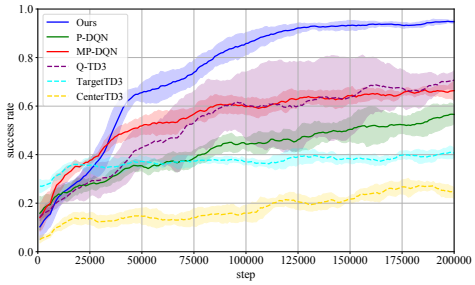


Fig. 5. Learning curves for environments with 5 objects, including a target object and 4 obstacles. The solid lines are the mean success rates of 5 seeds for each method, and the shadow areas are their standard deviation.

TABLE I

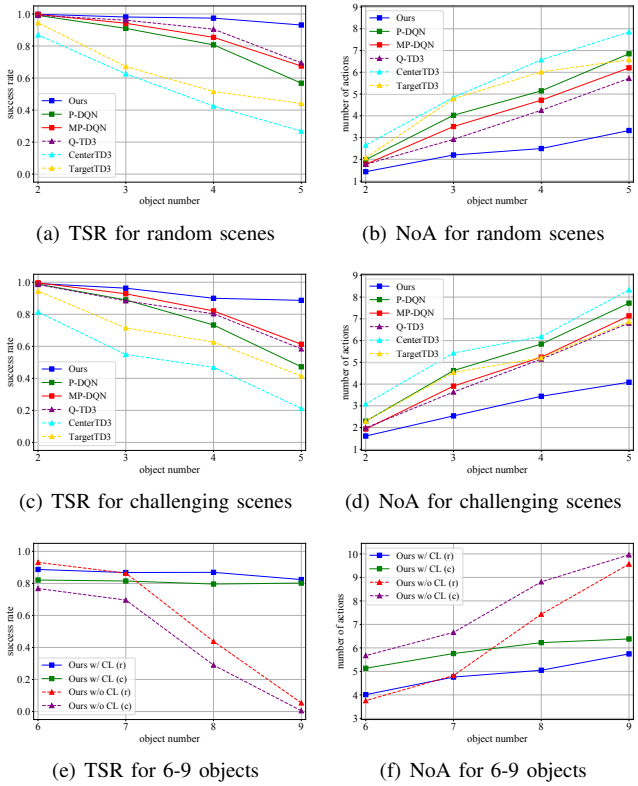
COMPARISON TO BASELINES ON 5 OBJECTS

Method	Random		Challenging	
	TSR	NoA	TSR	NoA
P-DQN	0.568	6.850	0.472	7.722
MP-DQN	0.675	6.204	0.613	7.136
Q-TD3	0.695	5.718	0.584	6.804
CenterTD3	0.269	7.856	0.212	8.330
TargetTD3	0.441	6.582	0.415	6.876
Ours	**0.931**	**3.324**	**0.887**	**4.086**

Compared with P-DQN, MP-DQN alleviates the error estimation of Q-values to a certain extent through its multi-pass method. Nevertheless, its performance is still inferior to our method. Our method provides more precise Q-values for policy network π_φ and Q-network Q_ω by feeding the object observation o^k into the networks and estimating only Q^k instead of all Q-values. According to the results, Q-TD3 has a large standard deviation in 5 seeds, while our method maintains a smaller one. This is because in our method, Q-network Q_ω is not only used to select actions but also provides policy gradient for policy network π_φ through Eq. (5). This ensures a close connection between the object selection and action parameter prediction.

CenterTD3 performs poorly in the pre-grasp task. As its action space is too large, it is difficult to learn an appropriate policy due to exploration challenges. TargetTD3 only slides the target and ignores other objects, making it basically hard to slide the target to the table edge when other objects block its moving path. However, our approach does not just deal with the target but rearranges other obstacles to separate the target from object piles and slide it to the table edge to make it graspable.

We conduct training and testing on environments with 2-5 objects to better analyze the differences between our method and baselines. The evaluation results of random and challenging scenarios are shown in Fig. 6(a)-(d). As the object number increases, the task success rate of each method decreases, and their mean numbers of actions increase. However, our method can maintain a slow descent rate, guaranteeing an 88.7% success rate even in challenging scenes with 5 objects. Although other methods have high success rates on scenes with 2 objects, their problems, including inaccurate Q-value estimation and large/insufficient action space, become apparent as the object number increases. These factors lead to sharp declines in their task success rates.



(a) TSR for random scenes (b) NoA for random scenes

(c) TSR for challenging scenes (d) NoA for challenging scenes

(e) TSR for 6-9 objects (f) NoA for 6-9 objects

Fig. 6. Evaluation results in cluttered environments with 2-9 objects. We run each method for 5 seeds, and test their final model performance. All the points in the curves are the mean results of the 5 final models. The 'r' and 'c' in (e) and (f) are short for random and challenging scenes, respectively.

C. Scaling to More Objects

Our method can be scaled to cluttered environments with more objects via curriculum learning as introduced in Algorithm 1, which is difficult for other methods like P-DQN [20] and MP-DQN [21]. The experimental results for environments with 9 objects are illustrated in Table II. The first row shows the evaluation results of training directly on 9 objects without using curriculum learning. The agent encounters the exploration problem when training on 9 objects' scenes directly. The policy easily pushes some objects under the table, causing task failure and getting a -1 reward. However, our method can achieve $>80\%$ success rates in both random and challenging scenarios via Algorithm 1, as demonstrated in the second row of Table II.

Moreover, we train and test in environments with 6-9 objects to prove the effectiveness of Algorithm 1 better. The experimental results are shown in Fig. 6(e)-(f). When adopting curriculum learning, our method's task success rates always maintain $>80\%$ success rates in environments with different object numbers. However, when removing

TABLE II

SCALING TO ENVIRONMENTS WITH 9 OBJECTS

Method	Random		Challenging	
	TSR	NoA	TSR	NoA
Ours w/o CL	0.054	9.566	0.005	9.960
Ours w/ CL	**0.824**	**5.744**	**0.802**	**6.380**

curriculum learning, the task success rates decrease rapidly as the object number n increases. Meanwhile, the pre-grasp policy trained in random scenes can be generalized to challenging scenes better with the help of Algorithm 1 The method without curriculum learning demonstrates poor generalization, performing much lower success rates under challenging scenes than random ones. In contrast, the policy trained by Algorithm 1 can achieve success rates comparable to random scenarios even in challenging ones.

D. Transfer to the Real World

We further evaluate the performance of the learned policy on the physical robot, without any fine-tuning using real data. A UR3 robot with a RobotIQ85 gripper is adopted to perform the pre-grasp sliding manipulation, and a Kinect 2.0 camera is utilized to locate the objects via Apriltag [24] and generate their masks. Since we adopt sliding actions, 2 rubber blocks are added to the gripper to ensure sufficient contact with the acted object, as shown in Fig. 1. Those can be replaced by force control methods to produce reliable contacts, which are not the focus of this paper. We first compare our method with 3 baselines for 40 trials in challenging scenes with 5 objects, using their respective best model learned in simulation, as shown in Table III. CenterTD3 and TargetTD3 are not evaluated due to their poor performance in simulation.

Our proposed method achieves the pre-grasp task with the highest task success rate and the fewest number of actions, even better than those in simulation. This is because there is no relative movement between the gripper and the object due to the rubber blocks, whereas this relative movement cannot be eliminated in simulation. Moreover, object penetration occurs occasionally in simulation and it results in a large contact force to bound the object off the table, causing task failure. However, this phenomenon does not exist in reality. Furthermore, we test the pre-grasp policy learned via Algorithm 1, and it achieves a 95.0% success rate with about 5.425 steps under the challenging scenarios with 9 objects.

One of the advantages of our method is that it is not limited by the object number n, hence the learned policy can be utilized in scenes with more objects. The model trained at n=9 still achieves a 92.5% success rate on the challenging scenarios with 11 objects. Meanwhile, we perform 40 trials on novel shapes and household objects, respectively. The experimental objects are shown in Fig. 7. Even though the learned policy has not seen these shapes, it still achieves the pre-grasp task with >85% success rates. This proves that our method has robustness and generalization, and it can perform pre-grasp manipulation under random object numbers and unseen objects with various dynamic parameters.

TABLE III

REAL-WORLD RESULTS ON CHALLENGING ARRANGEMENT

	Method	TSR	NoA
n=5	P-DQN	0.725	6.400
	MP-DQN	0.775	5.850
	Q-TD3	0.750	5.475
	Ours	**0.975**	**2.950**
n=9	**Ours w/ CL**	**0.950**	**5.425**
more objects (n=11)	**Ours w/ CL**	**0.925**	**6.125**
novel shapes (n=9)	**Ours w/ CL**	**0.875**	**6.050**
household objects (n=9)	**Ours w/ CL**	**0.850**	**6.300**

(a) Novel shapes (b) Household objects

Fig. 7. Experimental objects in the real world. The objects marked in dotted green are obstacles, and the others are target objects. All selected objects are flat objects that cannot or are not suitable for vertical grasping without pre-grasp manipulation. They are identified and located by Apriltag.

IV. CONCLUSIONS

In this paper, we propose a novel method based on deep reinforcement learning to achieve pre-grasp manipulation of flat objects in cluttered environments. The object masks and sliding primitives are utilized as the states and actions, respectively, improving the data efficiency and learning speed. A weight-sharing policy network is utilized to predict the sliding primitive's parameters for all objects on the table, and a Q-network is trained to estimate the values of these actions and select the acted object. Without being restrained by the object number, curriculum learning is integrated into our method to scale to cluttered environments with more objects.

Moreover, the pre-grasp policy learned in simulation is transferred to reality without fine-tuning using real data, obtaining excellent performance on challenging arrangements. And the learned policy keeps high task success rates on environments with more objects, novel shapes, and household objects in reality, demonstrating great robustness and generalization.

Pre-grasp manipulation is deployed on flat tabletops in this paper, and the actions are limited to sliding primitives. In the future, we will consider pre-grasp manipulation on non-flat tabletops with more types of action primitives, like pushing and rolling. Besides, there is no overlap among objects in this paper, whereas the objects may be stacked together in some scenes, bringing challenges to robotic perception and controlling. Pre-grasp manipulation in that case is another future research target.

ACKNOWLEDGEMENT

This work is partly supported by the National Natural Science Foundation of China (grant no. 61702516, 62203443), the Open Fund of Science and Technology on Thermal Energy and Power Laboratory, Wuhan 2nd Ship Design and Research Institute, Wuhan, P.R. China (grant no. TPL2020C02), and the Science Foundation for Youth of the State Key Laboratory of Management and Control for Complex System (under grant 2022QN09).

REFERENCES

[1] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," *arXiv preprint arXiv:1806.10293*, 2018.

[2] K. Guo, H. Su, and C. Yang, "A small opening workspace control strategy for redundant manipulator based on rcm method," *IEEE Transactions on Control Systems Technology*, vol. 30, no. 6, pp. 2717–2725, 2022.

[3] 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.

[4] H. Qiao, S. Zhong, Z. Chen, and H. Wang, "Improving performance of robots using human-inspired approaches: a survey," *Science China Information Sciences*, vol. 65, no. 12, pp. 1–31, 2022.

[5] L. Y. Chang, S. S. Srinivasa, and N. S. Pollard, "Planning pre-grasp manipulation for transport tasks," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2697–2704.

[6] D. Kappler, L. Y. Chang, N. S. Pollard, T. Asfour, and R. Dillmann, "Templates for pre-grasp sliding interactions," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 411–423, 2012.

[7] J. Wu, S. Zhong, and Y. Li, "Learning pre-grasp pushing manipulation of wide and flat objects using binary masks," in *International Conference on Neural Information Processing*. Springer, 2021.

[8] K. Hang, A. S. Morgan, and A. M. Dollar, "Pre-grasp sliding manipulation of thin objects using soft, compliant, or underactuated hands," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 662–669, 2019.

[9] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018, pp. 4238–4245.

[10] M. Kiatos and S. Malassiotis, "Robust object grasping in clutter via singulation," in *2019 IEEE International Conference on Robotics and Automation*. IEEE, 2019, pp. 1596–1600.

[11] K. Xu, H. Yu, Q. Lai, Y. Wang, and R. Xiong, "Efficient learning of goal-oriented push-grasping synergy in clutter," *arXiv preprint arXiv:2103.05405*, 2021.

[12] M. Moll, L. Kavraki, J. Rosell *et al.*, "Randomized physics-based motion planning for grasping in cluttered and uncertain environments," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 712–719, 2017.

[13] W. Masson, P. Ranchod, and G. Konidaris, "Reinforcement learning with parameterized actions," in *30th AAAI Conference on Artificial Intelligence*, 2016.

[14] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 41–48.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[16] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *32nd AAAI Conference on Artificial Intelligence*, 2018.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[18] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1587–1596.

[19] I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-Maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, and M. Riedmiller, "Data-efficient deep reinforcement learning for dexterous manipulation," *arXiv preprint arXiv:1704.03073*, 2017.

[20] J. Xiong, Q. Wang, Z. Yang, P. Sun, L. Han, Y. Zheng, H. Fu, T. Zhang, J. Liu, and H. Liu, "Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space," *arXiv preprint arXiv:1810.06394*, 2018.

[21] C. J. Bester, S. D. James, and G. D. Konidaris, "Multi-pass q-networks for deep reinforcement learning with parameterised action spaces," *arXiv preprint arXiv:1905.04388*, 2019.

[22] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems*. Springer, 2017, pp. 66–83.

[23] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[24] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3400–3407.