Efficient learning of goal-oriented push-grasping synergy in clutter

Kechun Xu, Hongxiang Yu, Qianen Lai, Yue Wang, Rong Xiong

***Abstract*— We focus on the task of goal-oriented grasping, in which a robot is supposed to grasp a pre-assigned goal object in clutter and needs some pre-grasp actions such as pushes to enable stable grasps. However, in this task, the robot gets positive rewards from environment only when successfully grasping the goal object. Besides, joint pushing and grasping elongates the action sequence, compounding the problem of reward delay. Thus sample inefficiency remains a main challenge in this task. In this paper, a goal-conditioned hierarchical reinforcement learning formulation with high sample efficiency is proposed to learn a push-grasping policy for grasping a specific object in clutter. In our work, sample efficiency is improved by two means. First, we use a goal-conditioned mechanism by goal relabeling to enrich the replay buffer. Second, the pushing and grasping policies are respectively regarded as a generator and a discriminator and the pushing policy is trained with supervision of the grasping discriminator, thus densifying pushing rewards. To deal with the problem of distribution mismatch caused by different training settings of two policies, an alternating training stage is added to learn pushing and grasping in turn. A series of experiments carried out in simulation and real world indicate that our method can quickly learn effective pushing and grasping policies and outperforms existing methods in task completion rate and goal grasp success rate by less times of motion. Furthermore, we validate that our system can also adapt to goal-agnostic conditions with better performance. Note that our system can be transferred to the real world without any fine-tuning. Our code is available at https://github.com/xukechun/Efficient_goal-oriented_push-grasping_synergy**

I. INTRODUCTION

Grasping plays an essential role in robot manipulation since it is a basic action for lots of complex tasks e.g. table organization [1]. Among these tasks, the robot usually faces a cluttered scene, where the tight packing around the goal objects may seriously degenerate the successful grasping. Inspired by the human behavior, the synergy between pushing and grasping becomes a solution to deal with such scenarios. By pushing the clutter, there can be space around the goal object for robot gripper insertion. To bring the skill to the robot, the vital problem is to develop the manipulation policy for synergy so that grasping with high success rate can be achieved using less steps of pushing.

There are several works [2] [3] [4] [5] [6] establishing the learning strategies to develop the synergies between pushing and grasping. [2] learns pushing and grasping policies equally using two parallel networks for push and grasp. [4] and [5] design a hierarchical framework by evaluating whether the robot is ready to grasp after current push step.

Kechun Xu, Hongxiang Yu, Qianen Lai, Yue Wang, Rong Xiong are with the State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China.

Corresponding author, wangyue@iipc.zju.edu.cn

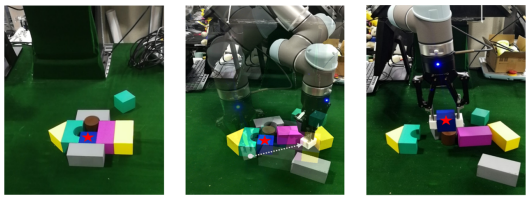


Fig. 1. An example scenario (left) of goal-oriented grasping tasks, in which the robot is supposed to grasp the object labeled with a star in clutter. The tight packing around the goal object degenerates the successful grasping thus synergy between grasping and pre-actions like pushing is needed. Our system is capable of planning push actions (middle) to isolate the goal object and finally executing a stable grasp action (right) to pick it.

One disadvantage of these methods is that the grasping object cannot be designated, thus only applicable to goal-agnostic tasks. Simply adapting these methods to goal-oriented tasks is to apply a mask of goal object on the grasping map, which, however, showing unsatisfactory results [7].

In contrast, goal-oriented grasping task, which is to grasp a pre-assigned goal object from a cluttered scene, is a more challenging problem because of the additional goal information. [8] uses physics-based analysis to compute pushing and grasping with a known 3D model of the goal object. [9] and [10] employ sparse rewards for training. Hence the robot gets positive rewards only when the goal object is singulated. Note that it is more difficult for the robot to acquire a positive reward from the environment in goal-oriented tasks compared with goal-agnostic tasks, causing low sample efficiency for training. To deal with this problem, [7] develops denser rewards by assigning rewards for pushing totally with handcraft to encourage moving objects and removing clutter. But [11] argues that the relationship between pushing and grasping is complex and coupled, leading to the handcrafted design of an optimal pushing reward function very difficult.

In this paper, we propose an efficient learning-based push-grasping synergy policy for goal-oriented grasping tasks (Fig. 1) by addressing the problems of sample inefficiency and sparse rewarding. We state the task as a goal-conditioned hierarchical reinforcement learning problem, and learn the model from scratch. By leveraging a goal-conditioned learning mechanism, denser rewards can be acquired to accelerate the learning. Specifically, if successfully grasping an object other than the designated one, we relabel the goal of corresponding experience in the replay buffer to the grasped one to improve the sample efficiency. Then, we design rewards for pushing by applying adversarial training inversely to build the synergy between pushing and grasping. With grasping as a discriminator, the adversarial training encourages the robot to push the clutter and generate a scene where the goal object is graspable with high probability. Thus the efficiency can

be improved. As a whole, we train the grasp and push nets in an alternating way to relieve the problem of distribution mismatch, finally developing the coordination between the two actions. Experiments conducted in simulation and real-world environments demonstrate that our system can achieve better performances in the goal-oriented grasping task with higher task completion and grasp success rate in less steps (Sec. IV). Note that our system is transferred from simulation to the real world without any additional data collection for fine-tuning. Moreover, our study on goal-oriented grasping is also applicable for goal-agnostic tasks by removing the target information (validated in Sec. IV). To summarize, the main contributions of this paper are:

- We propose a relabeling driven goal-conditioned strategy and an adversarial training based dense reward formulation to improve the sample efficiency for learning goal-oriented synergy between pushing and grasping.
- We propose an additional alternating training stage by learning the grasp network and push network in turn to improve the system performance against the distribution mismatch problem.
- We evaluate the learned system not only on both simulated random and designed cluttered scenes, but also on real world scenarios without extra fine-tuning, of which the results validate the effectiveness.

II. RELATED WORK

A. Grasping

Robotic grasping techniques have been well studied for decades. Such methods can be divided into two categories: analytical and data-driven [12]. Classic analytical methods require 3D models of known objects to find stable force closures for grasping [13] [14] [15]. However, most of the time it is difficult to get accurate models of novel objects. Recent data-driven approaches [16] [17] focus on directly mapping from visual observations to actions. But most of them assume the scene is scattered enough in which objects are well isolated. Learning for robotic grasping in cluttered environments is studied in [18] [19] [20] [21]. Unfortunately, picking objects in a dense clutter with only grasp actions is insufficient [19]. Pre-grasp manipulations such as pushing might be necessary for singulation.

B. Pushing assisted grasping

Synergies between pushing and grasping have been explored in [2] [3] [4] [5] [6], which aim to rearrange cluttered objects by push actions to enable future grasps. Zeng *et al.* [2] proposes a model-free deep reinforcement learning framework to learn joint policies of pushing and grasping with parallel architecture. [4] and [5] evaluate whether the goal object is ready for grasping by analytical metrics or neural networks, otherwise executing push actions to rearrange the clutter. Huang *et al.* [6] predicts push actions as well as states after pushing so that it is more intuitive to judge whether it is time to grasp.

Compared with above goal-agnostic methods, goal-oriented push-grasping in cluttered scene is much less explored, except for [7] [8] [9] [10]. [8] requires the 3D model of the goal object and uses a physics-based analysis of pushing to compute the motion of each object in the scene. [9] learns a pushing policy via Q-learning to singulate the goal object from its surrounding clutter. [10] proposes a split DQN framework which speeds up training process compared with [9]. [7] enables the goal object to be invisible in the initial state and uses a Bayesian-based policy to search it. Besides, other works on learning pushing networks for singulation in the context of mechanical search can also be regarded as push assisted grasping methods in a broad sense. [22] uses a push proposal network to rank pushes sampled upon object segments. The label of a push action is determined by a user who assesses the outcome of the action. [23] proposes an RL approach for active and interactive perception to search a goal object in clutter, in which rewards for actions are handcrafted based on exploration and detection results. [24] applies a teacher-guided strategy to train the pushing policy with pushing rewards based on the change of occlusion.

Analogous to above goal-oriented methods [7] [9] [10] [24], our approach is data-driven. There are two main differences. First, methods in [7] [9] [10] design the pushing reward function totally by handcraft, which might require many iterations of tuning [25] and lack of generalization to new scenarios [26]. In contrast, a data-driven reward will automatically optimize itself through the robot's experience to learn insight beyond human intuition, which is also more time-saving and effective. However, although totally handcrafted rewards are limited and defective, they can serve as effective approximation of the true rewards [26] to effectively guide the agent. Our approach aims to get pushing rewards mainly from a data-driven neural network which predicts the probability of successful grasps, to learn from posterior experience, combined with proxy handcrafted pushing rewards as important prior knowledge. Second, previous goal-oriented grasping or singulation works do not leverage experience of non-goal objects' grasping or singulation, thus requiring more samples to train the policy. [7] needs around 2k training episodes to build its push-grasping coordinator with 80% grasp success rate. [24] converges to high improvements in graspability within 8k training steps. Other goal-agnostic works, like VPG [2], spend more than 2.5k training episodes to reach 80% grasp success rate. Inspired by Hindsight Experience Replay (HER) [27], a goal relabeling algorithm, we use a goal-conditioned learning mechanism which relabels the grasping experience with grasping goals to improve the sample efficiency. Furthermore, we extend the goal of pushing as its corresponding grasping goals.

III. METHOD

We model the goal-oriented push-grasping problem as a goal-conditioned MDP, which is very similar to that in VPG [2], with a new symbol for the goal object representation as g. In this formulation, the symbol definitions of policy, reward and Q function are all conditioned on the goal g,

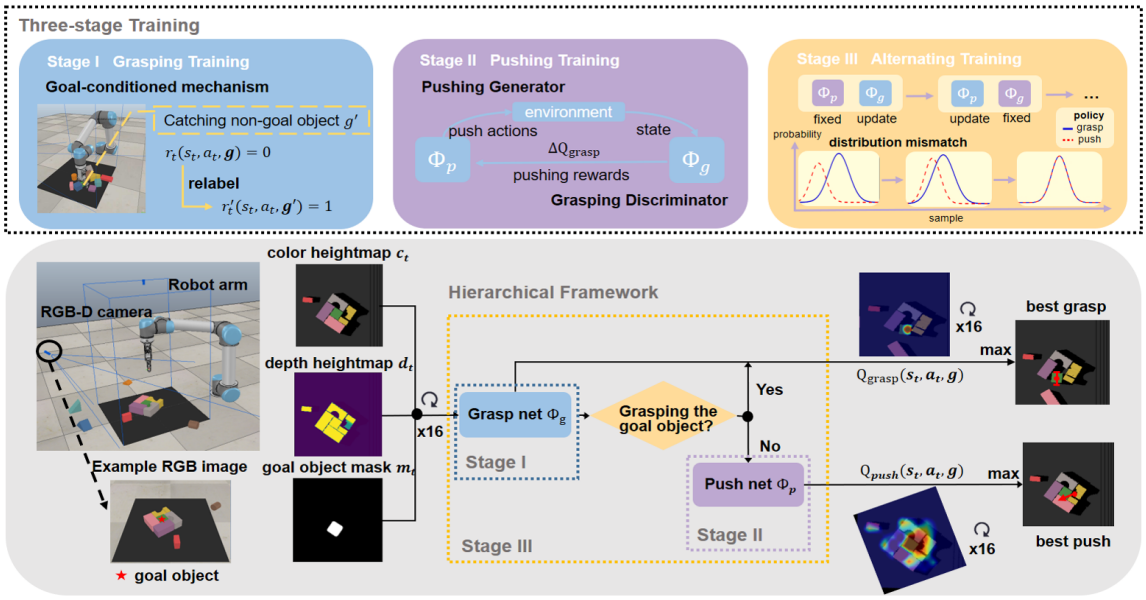


Fig. 2. **Overview** of our hierarchical framework and three-stage training approach. We fix a camera to capture RGB-D images of the workspace and construct visual heightmaps and the goal mask by orthographically projecting images in the gravity direction. The grasp net serves as a discriminator that evaluates whether is ready to grasp the goal object to choose between pushing and grasping. Color heightmaps, depth heightmaps and masks of the goal object are fed into the grasp and push nets, which predict pixel-wise grasp and push Q maps. Example of Q maps and predicted actions are also presented in this figure. Push is parameterized by a start position and a pushing orientation with a fixed pushing length, while grasp is defined by a middle position and an orientation of the parallel jaw of the gripper with a fixed pushing width.

which is defined as $\pi(s_t|g)$, $R(s_t, a_t, g)$ and $Q_\pi(s_t, a_t, g)$ respectively.

As shown in Fig. 2, we fix a camera to capture RGB-D images of the workspace. Then we represent each state s_t with a color heightmap c_t as well as a depth heightmap d_t, obtained by projection of the original RGB-D images in the gravity direction [2]. We represent the goal object as a mask heightmap m_t, which indicates the positions occupied by the object in the workspace. These heightmap representations are rotated by 16 angles with a stationary step size of 22.5° before feeding into networks to account for different grasp rotations w.r.t the z-axis.

We formulate the task of pushing and grasping in cluttered environments as a hierarchical reinforcement learning problem. For high-level control, our grasp net ϕ_g serves as a discriminator that scores current state for goal object grasping. If the score for grasping the goal object exceeds a threshold, the robot will execute the grasp action with maximal Q value. Otherwise low-level control will be activated: the robot executes push actions to transfer the state until it becomes graspable for the discriminator. Both push and grasp nets (*i.e.* ϕ_p, ϕ_g) take as input the heightmap representations of state s_t as well as the masks of goal object, and output dense pixel-wise maps of Q values with the same image size and resolution as that of heightmap representations. Every Q value at a pixel represents the future expected reward of executing primitive at the corresponding 3D location whose depth component is computed from the depth heightmap. As for our action definition, push is parameterized by a start position and a pushing orientation with a fixed pushing length, while grasp is defined by a middle position and an orientation of the parallel jaw of the gripper with a fixed width.

To train this hierarchical framework, a three-stage training approach is proposed as follows.

Stage I: Goal-conditioned Grasping Training. In the first stage, we train a goal-conditioned grasp net. To improve the sample efficiency, a goal-conditioned learning mechanism is introduced to relabel the goal of grasping experience to the grasped one. After enough episodes, expected Q values of successful grasps are stable above a specific value Q_g^* (whose determination method will be discussed in Sec IV-B in detail), which is set as a threshold that distinguishes appropriate states for goal object grasping.

Stage II: Goal-conditioned Pushing Training. In this stage, we focus on training a push net, and parameters of the pre-trained grasp net are fixed. Push is considered effective only when assisting future grasps, thus our pushing reward function is designed based on the grasp net by utilizing adversarial training inversely. Positive rewards will be obtained if pushes improve the graspable probability predicted by the grasp net. Also we use the goal-conditioned mechanism which relabels the goal of grasping and sequential pushing experience with the grasped one to improve the sample efficiency.

Stage III: Alternating Training. In the previous stage, we just consider the grasp net as an expert and fix its network parameters when training the push net. However, the former grasping net is trained in a relatively scattered environment where there is little occlusion around the goal object with only grasp actions. In this stage, we aim to further improve the performance of our pushing and grasping policies in cluttered environments by alternately training to relieve the the problem of distribution mismatch.

A. Goal-conditioned Grasping Training

In this stage the robot only executes grasp actions, and each grasp corresponds to an episode. To reduce the influence of occlusion, we create a relatively scattered scene where the goal object is well isolated by randomly dropping a objects in the workspace. We define the grasping reward function as:

$$R_g = \begin{cases} 1, & \text{if grasping goal object successfully} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

There are essential differences between the two cases where the robot gets no reward: failed grasps are totally harmful to the grasping performance, while the wrong object catches can serve as treasured experiences to improve the sample efficiency. Under this motivation, we use a relabeling driven goal-conditioned strategy. When the goal object is occluded by others, the robot is likely to accidentally grasp a non-goal object g'. If that happens, we relabel the goal g with g' in the transition tuple $(s_t, a_t, r_t, s_{t+1}, g)$ as $(s_t, a_t, r'_t, s_{t+1}, g')$, where $r'_t := R(s_t, a_t, g')$ and save this experience into the replay buffer to train our policy. By leveraging these wrong-object catching samples, we improve the sample efficiency thus speeding up the training.

After enough training episodes, Q values of successful grasps are stable above a specific value Q_g^*. In the following stages, we consider the current state to be ready for goal object grasping when the highest predicted Q value within the goal mask exceeds Q_g^*.

B. Goal-conditioned Pushing Training

For this stage our target is to learn pushes to enable future grasps using as few steps as possible. We limit the push number up to five in an episode with a grasp in the end. Besides, if the highest Q value within the goal mask exceeds the threshold Q_g^*, the agent will immediately execute the terminal grasp action. During one episode b different objects are randomly dropped in the scene to create clutter.

Pushes are favorable when enabling later grasps. However, the interaction between pushing and grasping is complex and coupled, thus difficult to model pushing rewards totally by handcraft [11]. To face this challenge, we train our pushing policy under the guidance of the grasp net by applying adversarial training inversely. Since grasp is a one-step action, the grasp net can evaluate graspable probability of the current state with its predicted grasp q values. We regard the grasp net as a discriminator that distinguishes a good state for goal object grasping with a Q value threshold Q_g^* which infers a high graspable probability. Correspondingly, push net is modeled as a generator that aims to fool the grasping discriminator by predicting push actions which rearrange the clutter to a state with maximal grasp q value of the goal object exceeding Q_g^*. We define the state transition function as $\mathcal{T}$, with $s_{t+1} = \mathcal{T}(s_t, a_t)$. Then our training objective is formulated as:

$$\min_{\phi_p} V(D_g, G_p) = E\left[\log\left(1 - D_g\left(G_p\left(s_t, g\right)\right)\right)\right] \tag{2}$$

$$s_{t+1} = G_p(s_t, g) = \mathcal{T}\left(s_t, \text{argmax}_{a_t}\, \phi_p\left(s_t, a_t, g\right)\right) \tag{3}$$

$$D_g(s_t) = \begin{cases} 1, & \delta > 0 \\ -\delta, & \delta < 0 \end{cases} \tag{4}$$

where $\delta = \max_{a_t} \phi_g\left(s_t, a_t, g\right) - Q_g^*$. In other words, the target of each push action is to continually increase the maximal grasp Q value of the goal object until exceeding Q_g^*. Based on above analysis, we model our pushing reward function as:

$$R_p = \begin{cases} 0.5, & Q_g^{\text{improved}} > 0.1 \text{ and change detected} \\ -0.5, & \text{no change detected} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $Q_g^{\text{improved}} = Q_g^{\text{after pushing}} - Q_g^{\text{before pushing}}$. And change detected means the surrounding of the goal object is changed and $o_g^{\text{decreased}} > 0.1$, where $o_g^{\text{decreased}}$ represents the decrease of occupy ratio surrounding the goal object and is computed from the depth heightmap. As for grasp actions, we follow the reward function (1) and our reward scheme is the combination of R_g and R_p.

Same as grasping training, a goal-conditioned mechanism is introduced for pushing by goal relabeling. If grasping a non-goal object when the goal object is occluded by others, goals of the sequential push actions in the whole episode will be set to the grasped object which means such push sequence is beneficial for its grasping.

C. Alternating Training

In previous stages our grasping discriminator is trained in a scattered scene only with grasp actions while pushing generator is learned in a relatively cluttered scene with push-grasping collaboration, which may lead to the distribution mismatch problem. That is, the grasping discriminator may fail to distinguish some novel scenes for grasping in push-grasping collaboration. Moreover, the training of grasp net has a great impact on the push net since our pushing reward function is based on the grasp net. To deal with this problem, we further train pushing and grasping policies based on networks ϕ_p and ϕ_g trained in previous stages. In this stage, grasping and pushing policies are alternately trained with the other net's parameters fixed. By alternating training, pushing and grasping policies restrict the positive samples of each other. During this training process, we also match the data distribution of the two policies for their synergy application. Therefore, the coordination performance of pushing and grasping is further improved.

Episode setting is the same for pushing and grasping training with c objects randomly dropped in the workspace. For each episode, the robot executes push actions until the maximal Q value within the goal mask exceeds Q_g^*. The reward functions remain the same as the former stages.

D. Implementation Details

In the first two stages, $a = 5$ objects are randomly dropped into the workspace for grasping training and $b = 10$ objects for pushing training. For the last stage, since our pushing policy has been former trained in clutter with push-grasping collaboration, we train only 500 episodes for pushing while 1000 for grasping, and both with $c = 10$ objects in the

workspace. Besides, we set the grasp Q value threshold $Q_g^* = 1.8$. Also, since perception is not the main concern of our method, we simplify the perception problem of goal object masks. In our work, goal object masks are obtained by projection of existing 3D models in simulation and color segmentation in the real world. If the goal object is invisible, then all pixels in the mask heightmap are of the value 0.

Both push and grasp nets (*i.e.* ϕ_p and ϕ_g) share the same network architecture, which involves three parallel 121-layer DenseNets [28] pretrained on ImageNet [29] to extract concatenated features of color heightmaps, depth heightmaps and goal masks, followed by a FCN [30] which contains two 1×1 convolutional layers with ReLU [31] and batch normalization [32], and then bilinearly upsampled. Both of them are trained using the same loss function as VPG [2]. At each iteration, gradients are only passed through the single pixel where the action was executed and other pixels backpropagate with 0 loss [2].

We train the networks with Adam optimizer, using fixed learning rates 10^{-4}, weight decay 2^{-5}, and betas $(0.9, 0.99)$. We use ϵ-greedy as our exploration strategy, and ϵ is initialized as 0.5 then annealed to 0.1. Our future discount γ is set as a constant at 0.5.

IV. EXPERIMENT

In this section, we carried out a series of experiments to evaluate our system. The goals of the experiments are: 1) to demonstrate our training approach is capable of speeding up the training process effectively. 2) to indicate that our policy is effective and efficient for the task of goal-oriented push-grasping with unseen random and challenging arrangements. 3) to investigate whether our approach can extend to goal-agnostic conditions. 4) to test that our system can successfully transfer from simulation to the real world without any fine-tuning.

We compare the performance of our system to the following baseline approaches:

Grasping-only is a greedy deterministic grasping policy using a single FCN network supervised with binary classification (from trial and error) for grasping. Under this policy agent executes grasping only. In our experiment, the network is replaced by ours. For fair comparisons, this policy is trained with the same simulation environments.

Goal-conditioned VPG is an extended version of VPG [2] by adding the goal mask as input to learn goal-oriented grasping and pushing policies. VPG is a method which predicts Q maps of pushing and grasping for goal-agnostic tasks. Pushes with maximal Q value and grasps with maximal Q value within the goal mask will be executed.

Grasping the Invisible is a target-oriented approach which chooses to push or grasp according to a binary classifier, and applies DQN to train a push-grasping policy [7]. A segmentation mechanism is used to detect whether the target is visible so as to switch between exploration and coordination. That is, if the target is invisible, the agent executes pushes to explore it and when the target can be seen pushes will coordinate with grasps to catch it.

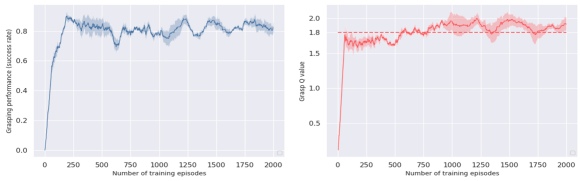


Fig. 3. Training performance of grasp net. After around 600 training episodes, grasp success rate (left) of goal object becomes stable above 80% while grasp Q-value (right) stays stably above $Q_g^* = 1.8$.

A. Evaluation Metrics

We evaluate the methods with a series of test cases where the robot must break dense clutter to pick the goal object. Each test contains n runs and performances are measured with 3 metrics similar to those in VPG [2] but with goal-oriented definitions:

- **Completion**: the average percentage of completion rate over n test runs. If the policy can pick up the goal object without consecutive 10 fail attempts in grasping the goal object in a run, then the task is considered completed.
- **Grasp success rate**: the average percentage of goal grasp success rate per completion.
- **Motion number**: the average motion number per completion. Motion number reflects action efficiency especially pushing efficiency.

B. Simulation Experiments

Our simulation environment is kept the same with [2] for fair comparison, which mainly involves a UR5 arm with an RD2 gripper in V-REP [33].

Parameter Verification. Our first experiment verifies the threshold Q_g^* in Sec III. We report the training performance of grasp net in Fig. 3, where the grasping performance is measured by the percentage of goal grasp success rate over the last $j = 30$ grasp attempts (the numbers reported at earlier training episodes *i.e.* iteration $i < j$ are weighted by $\frac{i}{j}$). After around 600 training episodes, grasping success rate of goal object becomes stable above 80% while grasp Q-value stays stably above $Q_g^* = 1.8$. Thus it is reasonable to regard $Q_g^* = 1.8$ as a threshold to evaluate whether is suitable for goal grasping.

Ablation Studies. We compare our methods with several ablation methods to test 1) whether our techniques can improve sample efficiency and 2) whether alternating-training can relieve the the problem of distribution mismatch.

We record goal grasp success rate versus training episodes over the last 30 grasp attempts to indicate performance during the push-grasping training process and make comparisons with two ablation methods: our system without dense reward is analogous to our method but without any reward for pushing and without any goal-conditioned mechanism, while our system without goal condition is a method without any goal-conditioned mechanism but with rewards for pushing.

To be rigorous, all methods are trained three times and results are plotted in Fig. 4. Note that in this figure performance of our system only involves the first two training stages since alternating training is started after 1500 training episodes (*i.e.* our system without alternating training). The

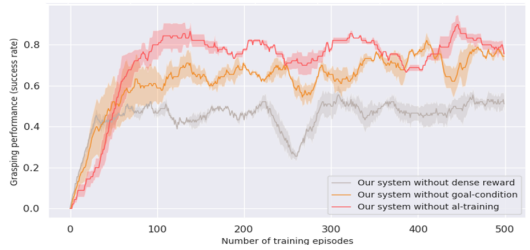


Fig. 4. Comparing training performance of our method with two ablation methods of our system.

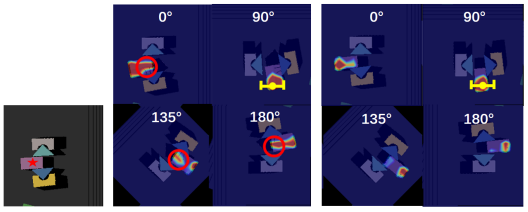


Fig. 5. Example distributions of grasp Q values within goal object (the purple block) before (middle) and after (right) alternating training. We choose four typical grasp Q maps corresponding to grasp orientation 0°, 90°, 135°and 180°respectively to reflect the effect. The full version can be seen in Appendix-F here. Color tends to be red as the Q value increasing. Before alternating training, some unreasonable grasps (highlighted with red circles) show high q values. Such low quality grasps are greatly reduced after alternating training. The chosen grasp is represented by a jaw orientation and a middle point of the gripper.

curves in the figure show that our system without alternating training is capable of converging to 80% grasp success rate in just 350 training episodes, which shows the highest training efficiency. This suggests that our adversarial training and goal-conditioned mechanism can effectively improve the sample efficiency so as to speed up the training process. We can see that policy of our system without dense reward can keep a similar pace as that of our system without alternating training at the early training but slows down after around 50 episodes. This may be because at early training the pushing policy is raw and does not play an important role for grasping, but after several training episodes, pushing policy with dense rewards has been trained to the extent of easing some grasps while that without dense rewards is still immature. With dense pushing rewards, policy of our system without goal condition can achieve a faster learning process and finally reach a similar performance with policy of our system without alternating training in 500 episodes, but still fails in the comparison of the training efficiency because of the lower sample efficiency.

Also we report a case to qualitatively demonstrate the effect of alternating training (more quantitative results are shown in the comparisons to the baseline methods) in Fig. 5, which shows example distributions of grasp Q values within goal object (the purple block) before and after alternating training. Obviously, grasping the goal block at positions between the two triangular objects is unreasonable, but it shows a high Q value before alternating training (highlighted with red circles in Fig. 5). By means of alternating training, such low quality grasps are removed. Moreover, the grasp chosen before alternating training is at the edge of the goal block which might lead to a fail catching. In contrast, the grasp chosen after alternating training is more stable.

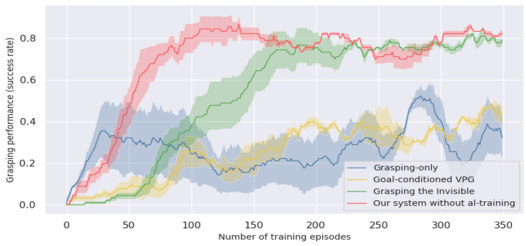


Fig. 6. Comparing training performance of our method with baselines.

TABLE I

SIMULATION RESULTS ON ALL ARRANGEMENTS

Method	Completion		Grasp Success		Motion Number	
Arrangement	r	c	r	c	r	c
Grasping-only	90.0	78.8	60.3	29.0	4.20	7.48
Goal-conditioned VPG	93.8	89.3	62.3	41.7	**3.93**	5.78
Grasping the Invisible	96.7	95.0	54.6	70.4	5.37	4.33
Our system w/o al-training	**97.8**	97.7	83.7	87.6	4.50	3.20
Our system	**97.8**	**99.0**	**90.0**	**90.0**	4.82	**2.77**

* The second row of table represent arrangement type. r, c correspond to random and challenging arrangements respectively.

** More detailed tables with metrics of each case can be seen in Appendix-F here.

Comparisons to baselines. In the following experiments we compare our system with baselines both in training and testing performances. Fig 6 shows training performance of our methods and baselines. Compared with policy of **Grasping the Invisible**, our system without alternating training reaches 80% grasp success rate in less training episodes. Performance of **Goal-conditioned VPG** policy is capable of increasing steadily but at a slower pace. As for the **Grasping-only** policy, interestingly it improves its grasping performance at the fastest pace early in training but ultimately achieves the lowest average performance. This may be due to the fact that while the grasping-only policy is busy refining itself to detect harder grasps, others focus on learning pushes that can make grasping easier.

We conduct test experiments with both random and challenging arrangements. For random arrangements, 30 objects are randomly dropped into the workspace whose shapes and colors are randomly chosen during each run. This scene is similar to that of training, but contains 30 objects rather than 10 so as to demonstrate the generalization of policies to more cluttered scenes. Results are reported in Table I. We observe that our system has much better performance in task completion rate and grasp success rate using a bit more motion number. It is interesting to note that **Grasping the Invisible** achieves a low grasp success rate at 50-60%. This is likely due to its tendency to grasp when the edge of the goal object is unoccupied but the free space is not enough for the gripper insertion. Besides, although **Goal-condtioned VPG** achieves the least motion number, it does not mean that it has a high efficiency since its grasp success rate is low at 60-70%.

Challenging arrangements involves 10 designed cases (shown in Fig. 7) with adversarial clutter where pushes are essential for goal grasping since the goal object are arranged either close side by side with others, or encircled by clutter, or even at the edge of the workspace, and it is rather difficult to pick the goal object without de-cluttering. Compared results to all baseline methods after $n = 30$ runs can be seen in Table I.

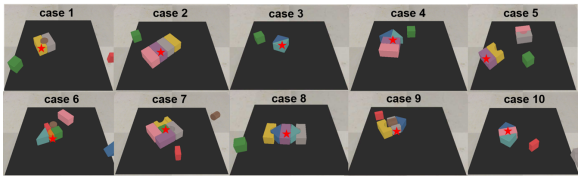


Fig. 7. Challenging arrangements in simulation experiments which involve 10 designed scenes with adversarial clutter. The goal object of each scenario is labeled with a star.

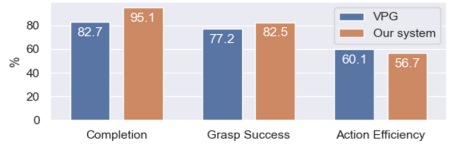


Fig. 8. Performance of goal-agnostic conditions including completion (left), grasp success rate (middle) and average motion number (right).

It is shown in the table that our method outperforms all baselines across all metrics. **Grasping-only** policy achieves the lowest completion rate, and even when it can complete the task, its average grasp success rate remains low at 20-30%. For push-assisted methods, each of them can achieve a much better completion rate, which suggests that push actions effectively assist goal object grasping in adversarial clutter. However, goal grasp success rate of **Goal-conditioned VPG** policy remains low at 40-50%, especially in scenes where objects are arranged closely side by side. Although **Grasping the Invisible** policy performs better in grasp success rate than **Goal-conditioned VPG**, it tends to grasp when space around goal objects is still narrow, thus its grasp success rate performance is also unsatisfactory. **Grasping the Invisible** policy achieves the lowest grasp success rate and highest number of motion in case 5. This is due to the fact that in this case it tries to grasp the object directly without pushing. And after several continuous failed grasps it tends to push the surroundings but unfortunately, push actions are not succinct and make little difference to the scene.

By training pushing policy with direct supervision of grasp success probability, our policy can perform better in both grasp success rate and motion number. Without alternating training, our policy can achieve an average grasp success rate of 87.6% (with above 90% success rate for 5 of 10 test cases) and an average motion number of 3.20, which suggests our push actions are efficient for goal grasping. After alternating training, our policy achieves a higher grasp success rate of 90.2% and a lower motion number of 2.77, which is apparent in case 6, case 8 and case 10. In case 6, the policy with alternating training can predict grasps at the plane while policy without alternating training tends to grasp at the curved surface.

Goal-agnostic Conditions. We further test whether our approach can extend to goal-agnostic conditions. We evaluate our method on 10 challenging cases in Fig. 7 without setting a goal. In this experiment, we mask all objects in the goal mask heightmap and the policy chooses to grasp the one with the highest grasp q value, finally picking up all objects. Results shown in Fig. 8 indicate that our policy can perform well in goal-agnostic conditions too. Compared with **VPG**, our method outperforms in completion rate by 12.4% and grasp success rate by 5.3% while action efficiency (defined as $\frac{objects\ in\ test}{actions\ before\ completion}$) seems a bit lower. This might be due to the fact that our policy chooses an object to grasp itself, and seldom turns to grasp others until this object is successfully caught, thus it tends to execute more push actions to create a suitable state for grasping.

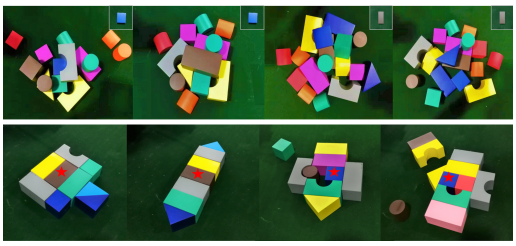


Fig. 9. Testing arrangements in real-world experiments. The upper row shows 4 random arrangements in which the goal object shown on the top right corner is partially or fully occluded. Note that the goal object in the second case (*i.e.* the blue cube) is fully occluded by the brown block. The lower row shows 4 setting challenging arrangements with each goal object labeled with a star.

Fig. 10. Real-world testing cases containing arrangements with novel objects which is unseen during training process.

C. Real-World Experiments

In this section, we test our system in real-world experiments. Our real-world platform includes a UR5 robot arm with a ROBOTIQ-85 gripper. RGB-D images of resolution 1280×720 are captured from an Intel RealSense D435. Our test cases consist of 4 random arrangements in which the goal object is partially or fully occluded and 4 setting challenging arrangements (shown in Fig. 9). We compare our methods with the method **Grasping the Invisible** which has shown a better performance than other baselines in our simulation experiments and other related works in [7].

In the real-world experiments, we set $n = 15$ runs for each test. Note, models of both methods are transferred from simulation to the real world without any retraining. Details about task completion rate, grasp success rate and average motion number of each case are reported in Table II. Overall, our policy outperforms the baseline across all metrics in random and challenging settings. This suggests that our policy can effectively generalize to the real world setting. It is worth noting that **Grasping the Invisible** uses a classifier to decide the actions, which largely depends on the accuracy of the depth images. This might partly explain why their policy is usually unable to accurately judge when to grasp.

Furthermore, we test the generalization capability of novel objects unseen during training. Each scenario contains objects of different height and more complex shape (shown in Fig. 10). Results in Table II confirm that our method can generalize to some unseen objects with a similar object shape distribution as that of the training objects, and has better performance than other methods. Failed cases involve

TABLE II

REAL-WORLD RESULTS ON ALL ARRANGEMENTS

Method	Completion			Grasp Success			Motion Number		
Arrangement	r	c	n	r	c	n	r	c	n
Grasping the Invisible	86.7	85.0	88.3	75.2	70.3	65.1	6.92	6.81	**3.81**
Our system	**93.3**	**95.0**	**90.0**	**81.7**	**86.6**	**81.0**	**5.67**	**4.62**	4.45

* The second row of table represent arrangement type. r, c, n correspond to random, challenging and novel object arrangements respectively.

** More detailed tables with metrics of each case can be seen in Appendix-F here.

catching an object with completely unseen complex shape, which might due to the lack of grasping experience of such kind of shape during training process.

V. CONCLUSION

In this work, we study the task of grasping a goal object in clutter, which is formulated as a goal-conditioned hierarchical reinforcement learning problem. We train the policies in simulation with three stages and evaluate our system in both simulation and real-world platforms. Simulation experiments show that our system can speed up the training process compared with other methods and achieve better performance in both random and challenging cases. Real-world experiments suggest our system is capable of effectively generalizing from simulation to the real world. It is worth mentioning that our system also achieves a better performance in goal-agnostic conditions, which further extends application scenarios of our method. More analysis and experiments are access here.

REFERENCES

[1] D. Batra, A. X. Chang, S. Chernova, A. J. Davison, J. Deng, V. Koltun, S. Levine, J. Malik, I. Mordatch, R. Mottaghi, M. Savva, and H. Su, "Rearrangement: A challenge for embodied ai," *arXiv preprint arXiv:2011.01975*, 2020.

[2] 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 (IROS)*. IEEE, 2018, pp. 4238–4245.

[3] A. Boularias, J. Bagnell, and A. Stentz, "Learning to manipulate unknown objects in clutter by reinforcement," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[4] Y. Deng, X. Guo, Y. Wei, K. Lu, B. Fang, D. Guo, H. Liu, and F. Sun, "Deep reinforcement learning for robotic pushing and picking in cluttered environment," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 619–626.

[5] Y. Chen, Z. Ju, and C. Yang, "Combining reinforcement learning and rule-based method to manipulate objects in clutter," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–6.

[6] B. Huang, S. D. Han, A. Boularias, and J. Yu, "Dipn: Deep interaction prediction network with application to clutter removal," in *IEEE International Conference on Robotics and Automation*, 2021.

[7] Y. Yang, H. Liang, and C. Choi, "A deep learning approach to grasping the invisible," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2232–2239, 2020.

[8] M. R. Dogar, K. Hsiao, M. Ciocarlie, and S. S. Srinivasa, "Physics-based grasp planning through clutter," *Robotics: Science and Systems VIII*, p. 57, 2013.

[9] K. Marios and M. Sotiris, "Robust object grasping in clutter via singulation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 1596–1600.

[10] I. Sarantopoulos, M. Kiatos, Z. Doulgeri, and S. Malassiotis, "Split deep q-learning for robust object singulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6225–6231.

[11] M. Danielczuk, J. Mahler, C. Correa, and K. Goldberg, "Linear push policies to increase grasp access for robot bin picking," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2018, pp. 1249–1256.

[12] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2013.

[13] H. Liang, X. Ma, S. Li, M. Görner, S. Tang, B. Fang, F. Sun, and J. Zhang, "Pointnetgpd: Detecting grasp configurations from point sets," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3629–3635.

[14] A. Rodriguez, M. T. Mason, and S. Ferry, "From caging to grasping," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 886–900, 2012.

[15] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3d object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2012.

[16] C. Choi, W. Schwarting, J. DelPreto, and D. Rus, "Learning object grasping for soft robot hands," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2370–2377, 2018.

[17] 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," *Robotics: Science and Systems (RSS)*, 2017.

[18] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Conference on Robot Learning*, 2018, pp. 651–673.

[19] A. ten Pas and R. Platt, "Using geometry to detect grasp poses in 3d point clouds," in *Robotics Research*. Springer, 2018, pp. 307–324.

[20] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 3406–3413.

[21] J. Mahler and K. Goldberg, "Learning deep policies for robot bin picking by simulating robust grasping sequences," in *Conference on robot learning*, 2017, pp. 515–524.

[22] A. Eitel, N. Hauff, and W. Burgard, "Learning to singulate objects using a push proposal network," in *Robotics research*. Springer, 2020, pp. 405–419.

[23] T. Novkovic, R. Pautrat, F. Furrer, M. Breyer, R. Siegwart, and J. Nieto, "Object finding in cluttered scenes using interactive perception," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8338–8344.

[24] A. Kurenkov, J. Taglic, R. Kulkarni, M. Dominguez-Kuhne, A. Garg, R. Mart'in-Mart'in, and S. Savarese, "Visuomotor mechanical search: Learning to retrieve target objects in clutter," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8408–8414, 2020.

[25] E. Ratner, D. Hadfield-Mennell, and A. Dragan, "Simplifying reward design through divide-and-conquer," in *Robotics: Science and Systems*, 2018.

[26] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. D. Dragan, "Inverse reward design," in *NIPS*, 2017.

[27] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Advances in neural information processing systems*, 2017, pp. 5048–5058.

[28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[30] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[31] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[33] E. Rohmer, S. P. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1321–1326.

APPENDIX

A. More analysis of the threshold Q_g^*

In our system, Q^* serves as an estimated value for a successful grasp. With this threshold, we can encourage the early ending of an episode if pushes have created a graspable scene, which guides the robot to execute push actions as few as possible. This setting makes our learning more simple and efficient. Without this setting, the robot must execute a fixed number of pushes before a grasp, or just compare the Q value of pushing and grasping (*i.e.* VPG), or require additional training of a action classifier to choose between pushing and grasping (*i.e.* Grasping the Invisible). Lots of qualitative and quantitative results in our paper have also confirmed that our method is more efficient and has a better performance.

Also, we set a fixed value because we think Q^* will not change with the scenario or the length of action sequence, but mostly based on the state of the goal object. In Stage II, we fix the grasp net with a fixed threshold Q^* to training pushes to create a graspable state. In this stage, the grasp net and the threshold Q^* can be regarded as the environment for the pushing policy. Similarly, in alternating training stage, the threshold Q^* is also fixed for learning a strategy to reach a stable graspable state as soon as possible in each episode. This threshold can also be regarded as a part of environment. Compared with other methods which uses a graspable probability through softmax, this setting may be more strict.

Besides, Q^* depends the graspability of the goal object we want to achieve. That is, if we set $Q^* = 2.0$, then finally Q value of successful grasps will converge to 2.0. But more push actions will be needed to build this system.

B. Completely handcrafted vs. data-driven pushing rewards

In this section, we will elaborate why completely handcrafted pushing reward for goal-oriented grasping task is limited and defective, thus confirming the necessity of a data-driven pushing rewards.

First, we agree that we can completely handcraft a reward function if our intuition is already good on the problem. But unfortunately, our intuition about how pushes enable future grasps is still not mature enough. Intrinsic pushing reward in VPG [2] does not explicitly consider whether a push enables future grasps. Rather, it simply encourages the system to make pushes that cause change. Also, GTI [7] develops denser rewards by assigning rewards for pushing totally with handcraft to encourage moving objects and removing clutter. These rewards are designed from human intuition that pushes might be supposed to move objects and remove clutter around a goal object. However, it's difficult to quantify the amount of space around the object for grasping. Encouraging pushes to crate space might lead to push unnecessarily for many times without a grasp when it is not confident enough to grasp because the occupancy rate is not low enough. In some cases where the goal object is higher than the surroundings, the object can be directly grasped without any space around in the top-down view. Besides, even the occupancy around the goal object is low, it does not mean that there are enough space for the gripper insertion. In [11], the authors design a series of handcrafted pushing reward functions and conduct studies about how these rewards assist future grasp, which suggests that the graspable probability is not always correlated with object separation.

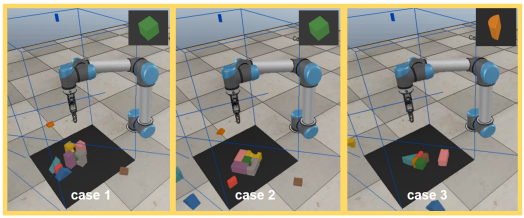


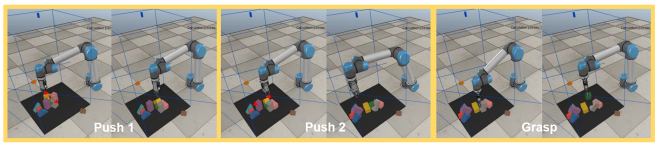
Fig. 11. Testing cases to show the weakness of totally handcrafted reward. The goal object of each case is shown on the top right corner.

Also, our pushing reward function is handcrafted in form. However, we use a grasp net to measure the push rewards, thus the value of pushing reward is essentially data-driven. Our insight is to mainly use a pre-trained grasp net to guide the training of the pushing policy, combined with a proxy reward designed from human intuition. The handcrafted part serves as prior knowledge which leverages human intuition to encourage pushes to remove clutter around the goal object, which can effectively speed up the training process. More importantly, we encourage pushes that directly improve the graspable probability of the goal object predicted by the pre-trained grasp net, which is the primary target of push actions. In this way, we explicitly consider whether a push enables future grasps and utilize posterior experience learned by the grasp net to optimize the system.

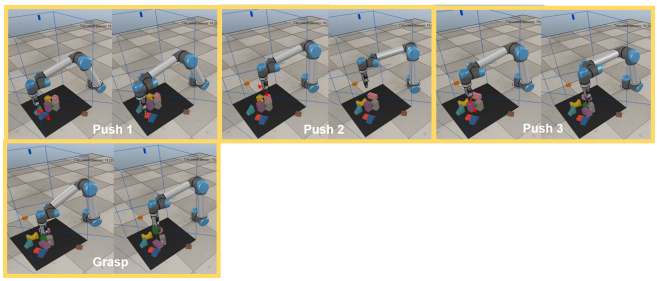
To be more specific, we conduct some case studies to verify the above points. We compare our method with GTI (Grasping the Invisible) [7] with 3 testing cases shown in Fig. 11. We can see that GTI, which uses complete handcrafted pushing rewards to encourage pushes to move or create space around the goal object, is always not confident for grasping and tend to push many times to make the surrounding of the goal object free enough (Fig. 12-14). Also, in case 1, GTI executes a redundant push action (Push 1 in Fig. 12) which changes the environment but is useless for goal object grasping. In contrast, our method can detect effective grasps even if the free space around the goal object is narrow (Fig. 12-13).

C. Comparison of data requirement

Also, to measure the data requirement of our method and other baselines, we record goal grasp success rate over the number of training actions in Fig. 15. For each action step, the goal grasp success rate is measured by the average goal grasp success rate over the last 30 training actions. Results shows that our method needs the minimum number of training actions (pushes and grasps) to reach 80% grasping success rate, which means the minimum requirement of data. Besides, combining Fig. 6, we can draw the conclusion that our push actions are more effective than other methods,

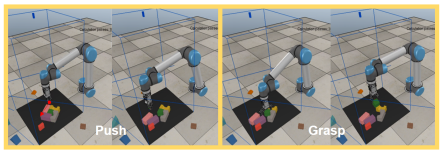


(a) Ours

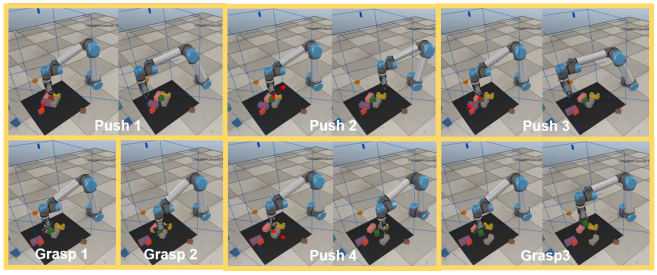


(b) Grasping the Invisible

Fig. 12. Example action sequences in testing case 1 of two methods.

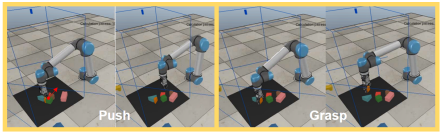


(a) Ours

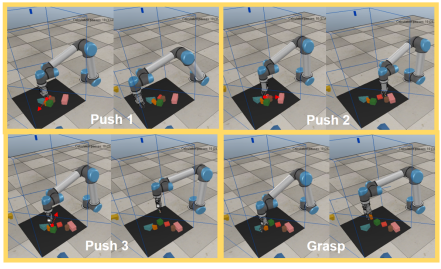


(b) Grasping the Invisible

Fig. 13. Example action sequences in testing case 2 of two methods.



(a) Ours



(b) Grasping the Invisible

Fig. 14. Example action sequences in testing case 3 of two methods.

TABLE III
RESULTS ON ARRANGEMENTS WITH DIFFERENT HEIGHT OBJECTS

Method	Completion	Grasp Success	Motion Number
Goal-conditioned VPG	**98.7**	59.4	5.58
Grasping the Invisible	90.7	23.8	7.89
Our system	97.3	**87.3**	**5.02**

TABLE IV
SIMULATION RESULTS ON NOVEL OBJECT ARRANGEMENTS

Method	Completion	Grasp Success	Motion Number
Grasping-only	86.7	44.9	8.14
Goal-conditioned VPG	**100**	60.2	6.63
Grasping the Invisible	87.3	57.9	5.81
Our system w/o al-training	97.3	73.2	5.85
Our system	97.3	**86.2**	**4.75**

especially compared to Grasping the Invisible. Although it can converge to a 80% goal grasp success rate in around 300 grasp attempts, it consumes much more push actions.

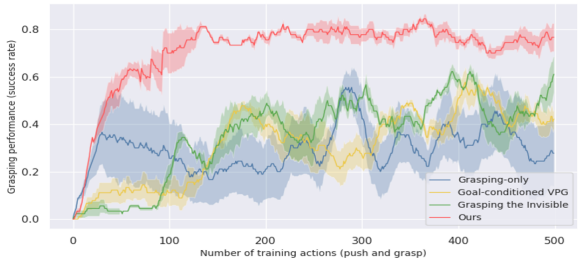


Fig. 15. Training curves of goal grasp success rate over the number of training actions (push and grasp).

D. Simulated experiments of object generalization

Objects of different heights. Five additional testing cases (shown in Fig. 16) which contain objects with different heights are conducted. The comparison shows the advantage of our method (shown in Table III).

Objects of complex shape. Furthermore, we conduct experiments with novel objects of more complex shape than those during training. Cases are shown in Fig. 17) and results are reported in Table IV. Overall, our method can generalize to some unseen objects with a similar object shape distribution as that of the training objects. Failed cases involve catching an object with completely unseen complex shape, which might due to the lack of grasping experience of such kind of shape during training process.

Example failed cases. Typical failed cases are shown in Fig. 18), which involve catching an object with unseen complex shape out of the object shape distribution as that of training objects. By applying some advanced object pose estimation methods or enriching the object shape ranges during training process, we think our method can generalize to more complex object shapes.

E. Example action sequence in real-world experiments

Two example action sequences in real-world experiments are shown in Fig. 19.

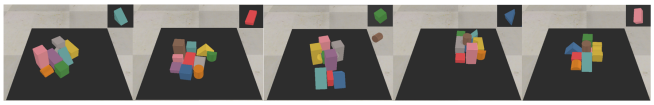


Fig. 16. Testing cases containing objects with different heights. The goal object of each scenario is shown on the top right corner.

Fig. 17. Simulated testing cases containing arrangements with novel objects which is unseen during training process. Also, objects in these arrangements are of relatively complex shape and different height.

Fig. 18. Examples of failed cases. The goal object is a toy cat of unseen complex shape.

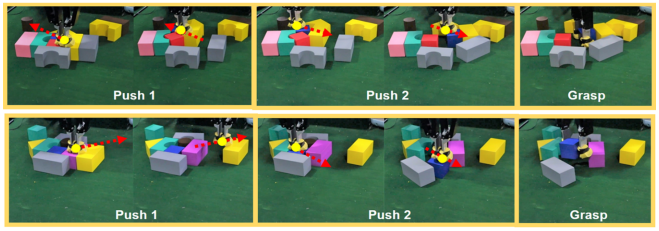


Fig. 19. Example action sequences of real-world experiments. The upper row corresponds to the third challenging arrangement while the lower row to the fourth one.

F. Detailed tables and clearer figures

In this section we show detailed tables with metrics of each case. Random and challenging arrangements of Table II corresponds to Table V and Table VI respectively. Challenging arrangements of Table I corresponds to Table VII. Also, a full version of Fig. 5 is shown in Fig. 20.

TABLE V
REAL-WORLD RESULTS ON RANDOM ARRANGEMENTS

	Grasping the Invisible	Our system
case1	93.3 / 68.8 / 6.88	93.3 / 78.6 / 5.50
case2	86.7 / 71.4 / 6.33	100 / 83.3 / 3.87
case3	86.7 / 85.7 / 7.14	93.3 / 88.1 / 5.86
case4	80.0 / 75.0 / 7.33	86.7 / 76.9 / 7.46
average	86.7 / 75.2 / 6.92	**93.3** / **81.7** / **5.67**

* Number of each cell is presented as "Completion / Grasp Success / Motion Number" to display the detailed testing performance of all the cases in the upper row of Figure 9.

TABLE VI
REAL-WORLD RESULTS ON CHALLENGING ARRANGEMENTS

	Grasping the Invisible	Our system
case1	86.7 / 73.3 / 6.75	93.3 / 85.7 / 4.78
case2	80.0 / 70.4 / 6.83	100 / 90.0 / 3.40
case3	80.0 / 64.8 / 7.11	100 / 90.0 / 5.33
case4	93.3 / 72.7 / 6.54	86.7 / 80.8 / 4.97
average	85.0 / 70.3 / 6.81	**95.0** / **86.6** / **4.62**

* Number of each cell is presented as "Completion / Grasp Success / Motion Number" to display the detailed testing performance of all the cases in the lower row of Figure 9.

TABLE VII

SIMULATION RESULTS ON CHALLENGING ARRANGEMENTS

	Grasping-only	Goal-conditioned VPG	Grasping the Invisible	Our system w/o al-training	Our system
case1	93.3 / 22.9 / 5.50	100 / 23.8 / 6.03	100 / 68.0 / 4.37	100 / 95.0 / 2.07	100 / 97.8 / 2.07
case2	63.3 / 23.8 / 5.32	96.6 / 43.1 / 4.70	100 / 93.3 / 2.20	100 / 89.5 / 3.43	100 / 83.5 / 3.40
case3	90.0 / 26.1 / 5.22	100 / 35.5 / 4.93	100 / 90.4 / 1.21	100 / 92.8 / 2.27	100 / 96.7 / 2.10
case4	23.3 / 10.8 / 18.7	82.8 / 31.4 / 7.46	100 / 80.0 / 5.27	100 / 92.6 / 3.69	100 / 98.3 / 2.03
case5	93.3 / 62.3 / 3.36	68.4 / 84.6 / 2.79	66.7 / 31.6 / 9.87	76.7 / 94.4 / 3.52	96.7 / 88.3 / 2.53
case6	100 / 32.3 / 6.33	100 / 53.3 / 3.32	100 / 74.2 / 3.00	100 / 76.7 / 4.55	100 / 83.3 / 3.40
case7	44.8 / 11.3 / 11.5	55.2 / 32.5 / 8.46	100 / 81.6 / 4.30	100 / 86.3 / 3.20	100 / 83.4 / 4.00
case8	100 / 44.8 / 2.73	93.3 / 16.4 / 10.6	100 / 67.1 / 3.02	100 / 84.2 / 3.13	100 / 95.0 / 2.17
case9	86.7 / 21.7 / 11.5	100 / 74.7 / 2.90	83.3 / 63.3 / 6.90	100 / 81.0 / 3.87	93.3 / 78.5 / 3.92
case10	93.3 / 34.1 / 4.61	96.7 / 21.5 / 6.58	100 / 54.5 / 3.15	100 / 83.3 / 2.30	100 / 96.7 / 2.07
average	78.8 / 29.0 / 7.48	89.3 / 41.7 / 5.78	95.0 / 70.4 / 4.33	97.7 / 87.6 / 3.20	**99.0** / **90.2** / **2.77**

* Number of each cell is presented as "Completion / Grasp Success / Motion Number" to display the detailed testing performance of all the cases in Figure 7.

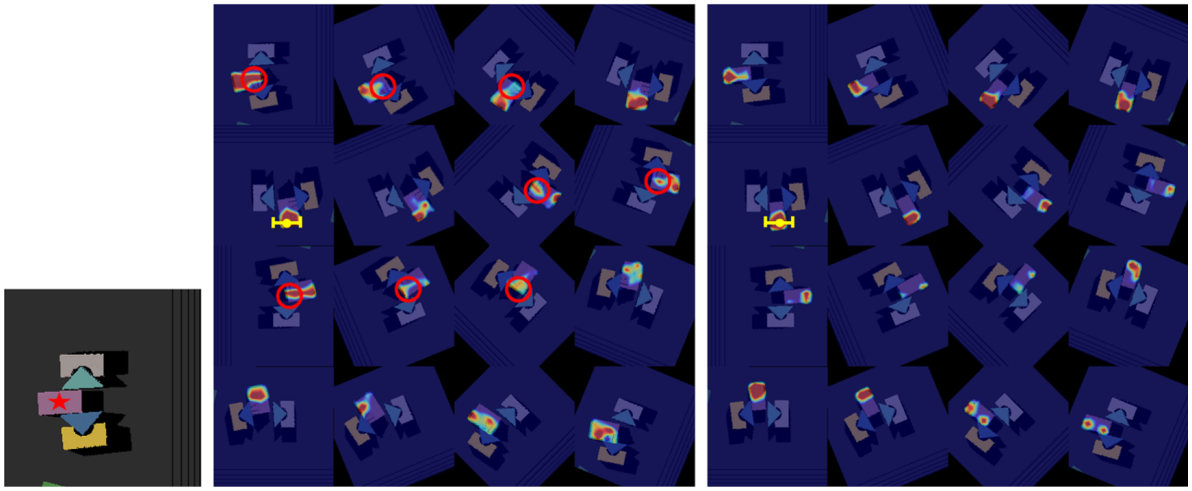


Fig. 20. Example distributions of grasp Q values within goal object (the purple block) before (middle) and after (right) alternating training. Color tends to be red as the Q value increasing. Before alternating training, some unreasonable grasps (highlighted with red circles) show high q values. Such low quality grasps are greatly reduced after alternating training. The chosen grasp is represented by a jaw orientation and a middle point of the gripper.