2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

MPGNet: Learning Move-Push-Grasping Synergy for Target-Oriented Grasping in Occluded Scenes

Dayou Li, Chenkun Zhao, Shuo Yang, Ran Song, Xiaolei Li, and Wei Zhang*

***Abstract*— This paper focuses on target-oriented grasping in occluded scenes, where the target object is specified by a binary mask and the goal is to grasp the target object with as few robotic manipulations as possible. Most existing methods rely on a push-grasping synergy to complete this task. To deliver a more powerful target-oriented grasping pipeline, we present MPGNet, a three-branch network for learning a synergy between moving, pushing, and grasping actions. We also propose a multi-stage training strategy to train the MPGNet which contains three policy networks corresponding to the three actions. The effectiveness of our method is demonstrated via both simulated and real-world experiments. Video of the real-world experiments is at https://youtu.be/S_QKZqkh0w8.**

I. Introduction

Grasping is a foundational action for versatile robotic tasks. Researchers have proposed many methods [1]–[8] for robotic grasping, which usually leverage deep learning and/or reinforcement learning to generate grasping policies and show good performance in scenes without occlusion. However, in real-world grasping scenarios, objects are often randomly placed in occluded scenes, bringing difficulties to grasping action. To solve this problem, many researchers have studied pre-grasp synergy, most of which extend non-prehensile pushing action to action space. Some methods [6]–[8] combined pushing and grasping policies for sequential manipulation as pushing can help rearrange the objects, making it easier for performing grasping. Such methods focus on target-agnostic grasping tasks, and cannot perform grasping according to user specification.

Target-oriented grasping aims to grasp user-specified objects. It is particularly challenging for occluded scenes. This is because it is non-trivial to accurately find and grasp the target object based only on partial observations. Most existing methods [9]–[13] for target-oriented grasping used a two-branch Q-network architecture to learn the push-grasping synergy to facilitate target-oriented grasping in occluded scenes. For example, Xu et al. [9] combine RGB-D heightmap and goal segmentation mask as input and train two separate Q-networks to learn the synergy between pushing and grasping actions. Instead of using goal segmentation masks, Yu et al. [12] design a target similarity network (TSN) to estimate goal localization. They also pass the visual feature into two deep Q-networks for making pushing-or-grasping decisions. Li et al. [13] adopt a similar two-branch deep Q-network to estimate the Q value distribution while decoupling the position and angle predictions.

All the authors are with the School of Control Science and Engineering, Shandong University, Jinan, 250061, China.
*Corresponding author: Wei Zhang (email: davidzhang@sdu.edu.cn)

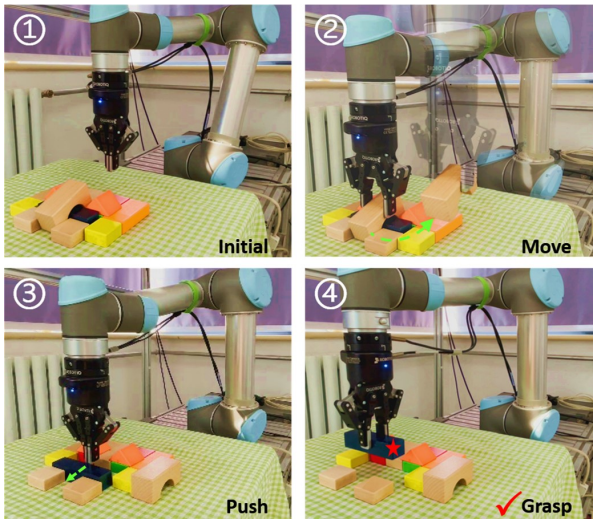

Fig. 1. Through the synergy of moving, pushing, and grasping actions, the robotic arm based on the proposed MPGNet can efficiently grasp the target object in occluded scenes.

One important assumption made by the above methods is that the agent should select the action (pushing or grasping) with a larger Q value at each training step. However, this assumption has poor interpretability and is not always reliable. Besides, most existing methods addressed the challenges of grasping in occluded scenes by pushing the stacked objects in specific directions while the pushing actions sometimes cannot efficiently solve the target-oriented grasping task, and even further mess up the grasping scenarios.

In this paper, we rethink the pipeline for addressing the target-oriented grasping in occluded scenes. First, we design three primitive actions, moving, pushing, and grasping, which work together to complete the tasks. Second, instead of solely selecting actions based on the scale of the Q value, we present a human intuition-guided workflow to coordinate the three primitive actions. Specifically, we design a three-branch deep Q-network to learn a synergic policy for target-oriented grasping in an occluded scene. Since making use of three primitive actions inevitably increases the learning difficulty compared to previous methods based on the push-grasping synergy, we design a multi-stage training strategy to deliver stable and fast policy training. Fig. 1 illustrates the core idea of this work through a real-world example.

To summarize, the main contributions of this paper are:

- We propose a novel three-branch deep Q-network, namely MPGNet, which learns a synergy between moving, pushing, and grasping actions for target-oriented

grasping in occluded scenes.

- We design a multi-stage strategy to train the MPGNet where we first train each action separately and then carry out a joint training to learn the move-push-grasping synergy effectively.
- We perform extensive experiments in the simulation to evaluate MPGNet, and further demonstrate a real-world robotic system that can reliably complete the target-oriented grasping tasks in occluded scenes.

II. Related Work

A. Target-Oriented Grasping without Action Synergy

Some scholars propose to obtain the target object via a single grasping action [5], [14]–[18]. We classify those methods as two-stage and one-stage. The two-stage implies that the target specification and grasp detection are not fused, but rather that task-agnostic grasp detection is first performed, followed by target filtering using a suitable heuristic [5], [14]–[16]. Murali et al. [14] crop the point cloud of the scene using a binary mask of the target object and then perform 6-Dof grasp detection on the target object point cloud. Liri et al. [15] use voice to specify the target object from object detection results and then forward it to the grasp detection module. An alternative, denoted as one-stage, directly generates grasp proposals based on the input target information in the form of images and language instructions [17], [18]. Lin et al. [17] propose to use sketches to represent the target objects and designed an end-to-end network for learning to generate target-oriented grasps. Yang et al. [18] design a multi-object dense descriptor for learning target-oriented grasp affordance via DRL with the targets defined by RGB images.

However, the above methods perform poorly in occluded scenes due to the limited action space. Obtaining the target object with a single grasping action is virtually impossible in some occluded and cluttered scenes. Therefore, it is necessary to explore action synergy algorithms to address the target-oriented grasping tasks in occluded scenes.

B. Target-Oriented Grasping with Action Synergy

Many studies explore the synergy of pre-grasp manipulations such as pushing and grasping to improve the success rate of grasping the target [9]–[13]. Xu et al. [9] propose a goal-conditioned hierarchical reinforcement learning formulation including goal relabeling strategy and alternative training to handle sample inefficiency and accelerate training. Yang et al. [11] propose a Bayesian-based policy to explore the target along with a classifier-based policy for push-grasping synergy. In order to solve the learning problems in large state space, Li et al. [13] decouple the action space by learning the position and the angle of the grasp in a separate way, which means that an additional branch is used to predict grasp angles. Moreover, some new methods are also gradually coming out. In [10], Zuo et al. adopt a graph-based deep reinforcement learning model instead of classical DQN to explore invisible objects so as to achieve better performance for push-grasping synergy.

Currently, the existing action synergy methods for target-oriented robotic grasping are mainly based on pushing and grasping. However, from our experimental test of the methods [6], [9], [11], we discover that in some common occlusion situations (as shown in Fig. 1), push-grasping synergy is not able to handle it efficiently. Firstly, for stacked occlusion, it is difficult for the agent to perform effective pushing actions, which is determined by the definition of the pushing action. Secondly, for push-grasping synergy, in some occasions, multiple action steps are required to successfully grasp the target object, which is time-consuming. Inspired by how humans deal with such occlusion, we expand the action space dimension by building a three-branch action synergy network. It is worth mentioning that Liu et al. propose GE-Grasp [19] to utilize nontarget-oriented grasps to reduce occlusion, however, it involves large amounts of data collection and fails to provide an effective framework to learn the synergy between different actions. Instead, we propose an automatic pipeline to handle this problem by adopting hierarchical reinforcement learning.

III. Method

Fig. 2 illustrates the pipeline of the proposed MPGNet. In the following, we elaborate each of its components.

A. Problem Statement

We define the target-oriented grasping task within the framework of a goal-conditioned Markov decision process. At any time step t, given the current state s_t and goal mask g_t for a specified target object, the agent must select and perform an action a_t based on the policy $\pi(s_t|g_t)$. Subsequently, the system transitions to a new state s_{t+1} and receives an immediate reward $R(s_t, a_t, s_{t+1}|g_t)$. The objective of our deep reinforcement learning task is to discover the optimal policy π^* that maximizes the expected cumulative future rewards, represented as:

$$R_t = \sum_{i=t}^{\infty} \gamma^{i-t} R(s_i, a_i, s_{i+1}|g_i) \tag{1}$$

The discount factor is denoted by γ. In this study, we explore the use of deep Q-learning to train neural networks that approximate the action-value function $Q_\pi(s_t, a_t|g_t)$. The expected reward for taking action a_t in the state s_t at time t is evaluated. A greedy deterministic policy $\pi(s_t|g_t)$ is derived by choosing the action based on the maximum Q values. Moreover, our learning objective is to minimize the temporal difference error δ_t of $Q(s_t, a_t|g_t)$ compared to a target value y_t:

$$\delta_t = |Q(s_t, a_t|g_t) - y_t| \tag{2}$$

$$y_t = R(s_t, a_t, s_{t+1}|g_t) + \gamma Q(s_{t+1}, \arg\max_a(Q(s_{t+1}, a|g_t))) \tag{3}$$

where a is the total set of available actions. This paper focuses on occluded scenes where the target object is occluded by other objects from a heightmap perspective.

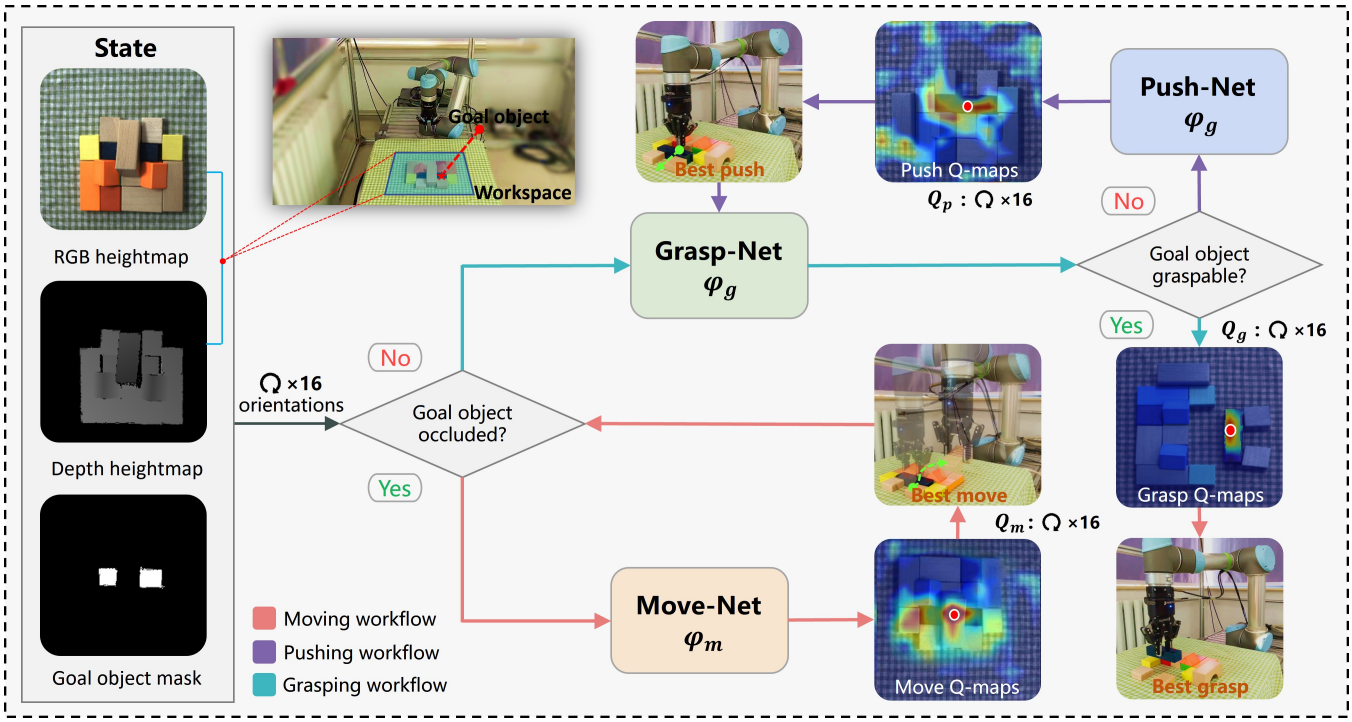

Fig. 2. Overview of MPGNet. The input data of MPGNet are obtained by an RGB-D camera from a top-down view. The heightmaps are rotated at 16 angles to predict different motion orientations and fed into MPGNet. Move-net, grasp-net, and push-net work collaboratively to grasp target objects in occluded scenes. The move-net is designed to remove occluding objects to make the target as graspable as possible. When there is no occlusion in the workspace, the pushing action assists in grasping the target object.

B. Three-Branch Deep Q-Network

1) State Representations: To better represent each state s_t, the RGB images and the depth images captured by a fixed camera are orthographically back-projected upwards in the gravity direction to generate the color heightmap c_t and the depth heightmap d_t. The state of the workspace at time t is represented as $s_t = (c_t, d_t)$. We use a mask heightmap to represent the target object as g_t, which indicates the area occupied by the target object in the workspace. In the simulated environment, the mask heightmap is obtained through the following steps: 1) We obtain the position and shape information of the target object using APIs provided by the simulator and then calculate the area occupied by the target object along the direction of gravity. 2) We populate the mask heightmap with values derived from the spatial data of the target object's occupied area, ensuring the heightmap reflects only the general boundaries of the target object.

2) Action Space: We define each action $a_t \in \{a_m, a_g, a_p\}$ as a distinct motion primitive, where a_m, a_g, and a_p correspond to the behaviors of moving, grasping, and pushing, respectively. The specifics of each motion primitive are detailed below.

Moving. A moving action is designed to move the obstacles, defined as $a_m = (p_m, \theta_m)$. Let $p_m = (x_m, y_m, z_m) \in \mathbf{R}^3$ denote the central position of the two-finger robotic gripper within the workspace, where $\theta_m \in \mathbf{R}$ signifies the rotational angle. The coordinates (x_m, y_m) are derived from the pixel location in Q_m through eye-to-hand calibration. The z_m coordinate is defined as $z_m = h_m - 4cm$, with h_m representing the height at the point (x_m, y_m). During operation, the gripper must descend 4 cm below h_m before the fingers close. The angle θ_m is one of 16 possible orientations around the z-axis, ranging from $0°$ to $360°$, with a $22.5°$ increment between each orientation.

Pushing. A pushing motion executed by the gripper's tip is represented as $a_p = (p_p, d_p)$. Each push has a consistent length of 10 cm, following a straight-line trajectory. The starting point of the push is denoted by $p_p = (x_p, y_p, z_p) \in \mathbf{R}^3$, where $d_p \in \mathbf{R}^3$ specifies the direction of the push. The coordinates (x_p, y_p) are derived from pixel data in Q_p through calibration. To ensure the gripper lightly contacts the object, we set $z_p = h_p - 1cm$ if h_p exceeds 0. Conversely, when h_p is negative, z_p is adjusted to 2 cm to prevent the gripper from touching the ground. The push direction is one of 16 possible orientations within a $0°$ to $360°$ range around the z-axis.

Grasping. A grasping action can be described as $a_g = (p_g, \theta_g)$, where $p_g = (x_g, y_g, z_g) \in \mathbf{R}^3$ indicates the center point of the top-down parallel-jaw grasp, and $\theta_g \in \mathbf{R}$ denotes the grasping angle, which varies from $0°$ to $360°$ around the z-axis, divided into 16 segments. The height at the position (x_g, y_g) is represented by h_g, and the value of z_g is calculated as $h_g - 4$cm. To perform the grasp, the gripper must descend 4 cm below h_g.

3) Network Details: We extend vanilla deep Q-networks [20] by designing our own Q-functions as three feed-forward

fully convolutional networks (FCNs). We name it MPGNet which includes move-net φ_m, push-net φ_p, and grasp-net φ_g. MPGNet takes as input the state representations $s_t = (c_t, d_t)$ with goal g_t and outputs dense pixel-wise Q-value maps $Q = (Q_m, Q_g, Q_p) \in \mathbf{R}^{224\times224\times3}$ with the same resolution of s_t, where $Q_m, Q_g, Q_p \in \mathbf{R}^{224\times224}$ represent Q-values of the corresponding actions in the pixel positions. For each net, we adopt two parallel MobileNetV3 [21] pre-trained on ImageNet to encode color heightmaps and depth heightmaps. A public MobileNetV3 block is used for feature extraction of the goal mask heightmaps. Prior to being input into MobileNetV3, each heightmap undergoes rotation across 16 different angles, facilitating the learning process for motion primitives related to movement, grasping, and pushing actions. The features extracted are initially concatenated along the channel dimension and then processed through two 1×1 convolutional layers. These layers incorporate nonlinear activation functions (ReLU) and are followed by spatial batch normalization. The final step involves bilinear upsampling. The entire network produces 48 pixel-wise Q-value maps, divided into 16 maps for movement at various orientations, 16 maps for pushing in different directions, and 16 maps for grasping at various angles The goal of the moving action is to make enough space around the target object for successive grasping by clearing the occluding objects that the pushing action cannot handle efficiently. We restrict the movement to the vicinity of the target object, allowing the network to concentrate solely on this area. This approach ensures that the movement directly addresses any occlusion affecting the target object.

C. Policy Learning

We model the move-push-grasping synergy for target-oriented grasping as a hierarchical reinforcement learning problem. We set the grasp-net φ_g as the discriminator to score each state s_t for grasping the target object. In the early stage of training the move-net φ_m, it cannot learn how to handle occlusion properly, which causes negative rewards and leads to inefficient learning. Thus, when occlusion occurs, the well-trained grasp-net φ_g is used to score the state s_t changed by the moving action. If the increase of improved grasp Q-values exceeds a threshold, the agent will still receive a positive reward regardless of whether the occlusion is removed by the moving action. When occlusion is absent, pushing actions are initiated, and the robot performs these actions to alter the object's state until it becomes suitable for grasping by the discriminator. We design a multi-stage strategy to train MPGNet. The details are as follows.

Stage I: Target-Agnostic Grasping-Only Training. Since we use the grasp-net φ_g to score the states, we need to ensure that it is well-trained to accurately find the stable grasping positions. At this phase, the robot executes the grasping task without targeting specific objects, which means that the input mask should include all the objects in the scene. The episode concludes once every item in the workspace has been successfully grasped. To reduce the effect of occlusion, we make the locations where m objects are dropped relatively discretely. We define the target-agnostic grasping reward function as:

$$R_{g1} = \begin{cases} 1, & \text{grasping a object successfully} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Stage II: Target-Oriented Grasping-Only Training. In the previous stage, we obtain a relatively stable grasp net for target-agnostic use. In this stage, we aim to train a target-oriented grasp net based on the previous stage. In practice, we randomly drop m objects in the workspace and set the n-th object as our target. For learning efficiently, we make the target object relatively isolated from other objects. We define the target-oriented grasping-only reward function as:

$$R_{g2} = \begin{cases} 1, & \text{grasping the target successfully} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Obviously, there are two occasions where the agent receives zero reward: failed grasps and wrong grasps. After undergoing a substantial number of training episodes, the Q values associated with successful grasp attempts tend to converge to a particular value, denoted as Q_g^*. This threshold indicates that when a state's Q value surpasses this point, the target object is likely graspable. Subsequently, in later stages, the grasp network φ_g from stage II functions as a discriminator.

Stage III: Target-Oriented Moving-Only Training. In this stage, the moving policy is learned to clear occlusions for the target object. To obtain the status of whether the target object is occluded, we propose an occlusion-checking mechanism in the simulation environment: 1) Crop the depth heightmap of the target object through the given target mask. 2) Obtain the position and shape information of the target object via the simulation API. 3) Calculate the highest (z-axis) point on the surface of the target object, and compare the calculated highest point with the highest z value on the cropped depth heightmap to generate the occlusion status. To train the move net, we create some simple occluded scenes at first: drop two objects in sequence and ensure by setting two positions adjacent that the first object is occluded by the other one. We set a flag of moving successfully: after a moving action, if the occlusion of the target object is removed and the target is not grasped by the moving action, the flag is set to true. Consequently, the agent should receive a positive reward. Each episode ends when the flag of moving successfully turns from false to true. We set the maximum threshold as 5 for the number of moving actions, which means that the simulation environment will be reset if the agent executes 5 moving actions without clearing occluding objects. After several training episodes, the agent can easily handle occluded scenes. Next, we add more objects in the workspace to make it more scattered for the agent to learn to handle more occluded scenarios. In the early training stage of occluded scenes, it is usually difficult for the agent to meet the conditions of moving successfully. Thus we give rewards by introducing the method of comparing Q_g before and after moving. The target-oriented moving-only reward

function is defined as:

$$R_m = \begin{cases} 1, & \text{move successfully} \\ 0.5, & Q_g^{\text{im}} > 0.5 \quad \text{and change detected} \\ -0.5, & \text{move the target or no change detected} \end{cases} \tag{6}$$

where 'move the target' denotes that the target object is grasped by the moving action, and 'change detected' denotes that the surrounding of the target object is changed. $Q_g^{\text{im}} = Q_g^{\text{am}} - Q_g^{\text{bm}}$, where Q_g^{im} means the improved grasp Q-value by a moving action, Q_g^{am} means the grasp Q-value after a moving action, Q_g^{bm} means the grasp Q-value before a moving action.

Stage IV: Target-Oriented Pushing-Only Training. In this stage, our primary objective is to train a push network, while keeping the weights of the pre-trained grasp network and move network unchanged. Additionally, the grasp network functions as a discriminator to evaluate and score the pushing actions. Positive rewards will be given to the agent if the pushing actions improve the Q values predicted by the grasp net. In practice, the number of pushing actions is limited to 5 in an episode with a grasp at the end. Each pushing action aims to continuously increase Q_g of the target object until it exceeds Q_g^*. We construct some scenes where the target objects are surrounded by other objects. The target-oriented pushing-only reward function is defined as:

$$R_p = \begin{cases} 0.5, & \text{if} \quad Q_g^{\text{ip}} > 0.1 \quad \text{and change detected} \\ -0.5, & \text{no change detected} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where $Q_g^{\text{ip}} = Q_g^{\text{ap}} - Q_g^{\text{bp}}$. Q_g^{ip} means the improved grasp Q-value by a pushing action, Q_g^{ap} means the grasp Q-value after a pushing action, Q_g^{bp} means the grasp Q-value before a pushing action.

Stage V: Target-Oriented Joint Training. In earlier stages, our grasp net was trained exclusively on grasping actions within a cluttered environment, while our move net was trained in a specifically designed scenario with occlusions. Additionally, our push net was trained in a moderately occluded setting where push-grasp synergy was emphasized. However, when combining them, there will be a problem of distribution mismatch as the training prerequisites differ for each network. Moreover, the move-push-grasping synergy needs further training to better suit real scenarios.

To solve this problem, we further train the move-push-grasping policy based on the previously trained networks φ_m, φ_g, and φ_p. In this stage, all three policies are alternatively trained with other nets' weights frozen. Through this stage, the three actions coordinate with each other better. We set the episode the same for moving, grasping, and pushing as in previous stages. 10 objects are randomly dropped in the workspace to construct a realistic and occluded scene. In each episode, the agent first checks the occlusion status to decide whether moving actions should be executed. Then, the pushing actions are executed till the maximal Q value of the target object exceeds the threshold Q_g^*. The reward functions for each action remain the same as in the previous stages.

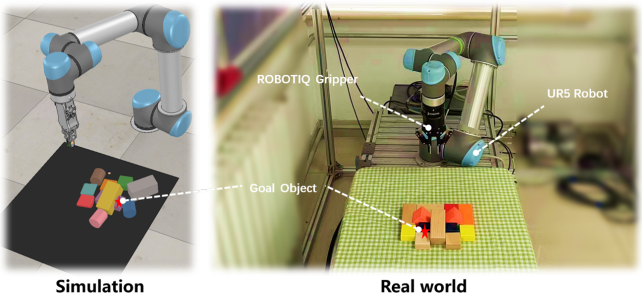

Fig. 3. We train MPGNet in the simulated environment and then transfer it to the real world.

D. Training Details

In the first two stages of the grasping-only training, we set the number of objects in the scene as $m = 5$. To determine Q_g^* for the discriminator, we adopt the methods in [9]. After we finish the first two stages, the Q value becomes stable around $Q_g^* = 1.85$. Thus we set $Q_g^* = 1.85$ as the threshold to determine whether the target object is graspable. Considering that the grasping and the moving positions should be close to the target, we propose to generate the rough target mask by calculating the largest bounding rectangle of the target object, and assign the Q value out of the rough mask to 0. The loss function in our method is the Huber loss function defined as:

$$L = \begin{cases} \frac{1}{2}\delta_t^2, & if \ \delta_t < 1 \\ |\delta_t - \frac{1}{2}|, & \text{otherwise} \end{cases} \tag{8}$$

where δ_t is defined in Section III-A. We also adopt ϵ-greedy exploration strategy with ϵ initialized as 0.5 and annealed to 0.1. We set future discount $\gamma = 0.5$ as constant. The network is trained with Adam optimizer with a fixed learning rate 10^{-4}, a weight decay 2^{-5}, and betas $(0.9, 0.99)$.

IV. Experiments

As shown in Fig. 3, we train MPGNet in the simulated environment and then transfer it to the real world. In our simulated environment (CoppeliaSim), a UR5 robotic arm with an RG2 gripper is mounted in the workspace. Two cameras are mounted at fixed positions with known extrinsic parameters to capture RGB-D images of the workspace. In the real world, we build up a scene similar to the simulation: we use a UR5 robotic arm equipped with a ROBOTIQ two-finger gripper as the agent to execute different actions, and a RealSense D435i RGB-D camera is fixed horizontally above the workspace to capture color and depth images. In the following, we evaluate the proposed method via a series of experiments.

A. Baselines

We compare the performance of our method with the following baselines:

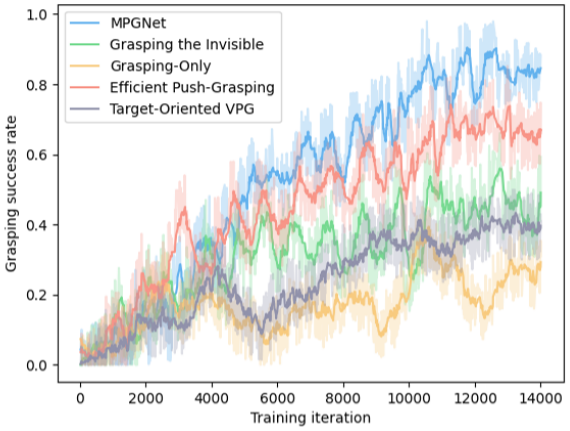

Fig. 4. Learning curves of different methods.

TABLE I
COMPARISON OF DIFFERENT METHODS IN SIMULATION

Method	Task success rate		Average motion number	
	15 objects	30 objects	15 objects	30 objects
Grasping-Only	26%	12%	6.46	8.33
Target-Oriented VPG [6]	32%	26%	5.43	6.54
Grasping the Invisible [11]	72%	66%	4.64	5.33
Efficient Push-Grasping [9]	84%	74%	3.88	4.97
MPGNet	92%	84%	2.85	3.74

Grasping the Invisible [11] is a target-oriented approach that uses a classifier-based policy to coordinate pushing and grasping actions to grasp the target object in clutter. A color segmentation method is used to generate target masks and to detect whether the target is visible for coordination or exploration.

Efficient Push-Grasping [9] uses several techniques to train a target-oriented push-grasping synergy. Goal relabeling is adopted to improve sample efficiency. Moreover, alternative training is applied in the training process which means that pushing and grasping are learned in turn.

Grasping-Only applies a single FCN network to learn a greedy deterministic grasping policy.

Target-Oriented VPG is a target-oriented approach that extends VPG [6] by taking the target mask in addition as input to learn the push-grasping synergy. VPG outputs Q maps of both pushing and grasping for target-agnostic tasks. It executes the action with the highest Q value in the target area.

B. Evaluation Metrics

We evaluate our method in some scenes with occlusion where n objects are randomly dropped at relatively adjacent positions in the simulation environment. We set the first or the second object as the target object to ensure that the target is occluded. The metrics are defined as below:

1) Task Success Rate: In n test experiments, the task success rate is defined as the average percentage of successful tests over n runs. A test is considered successful if the agent grasps the target object successfully within 10 motions.

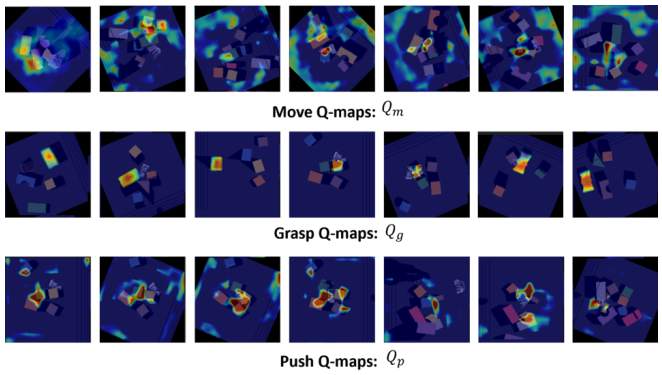

Fig. 5. Visualization of the Q-maps corresponding to the three primitive actions produced by MPGNet.

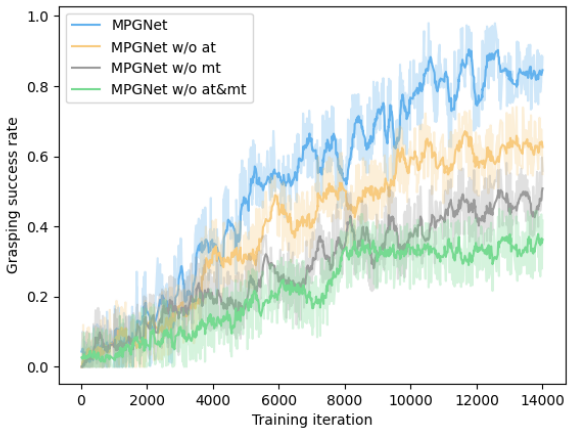

Fig. 6. Learning curves of MPGNet variants.

2) Number of Motions: The number of motions is defined as the mean quantity of actions needed to effectively grasp the target, serving as an indicator of action efficiency. In addition, it also shows the clutter of the scene as more complex scenes often require more actions to grasp the target object in general.

3) Grasp Success Rate: We record the average target grasping success rate for every k training iterations to reflect the grasping capability of the model in the training process. It can be defined as the ratio of the number of successful target grasps to the number of grasp attempts.

C. Simulation Experiments

We compare the performance of MPGNet with other baselines in both training and testing processes. In detail, Grasping the Invisible, Target-Oriented VPG method, Efficient Push-Grasping, and MPGNet have the same training scenario where we place 10 objects in the workspace and the first or the second dropped object is the target object in the occluded scene. For Grasping the Invisible and Target-Oriented VPG methods, we record the training process from scratch. For Efficient Push-Grasping method, we start recording from the joint training.

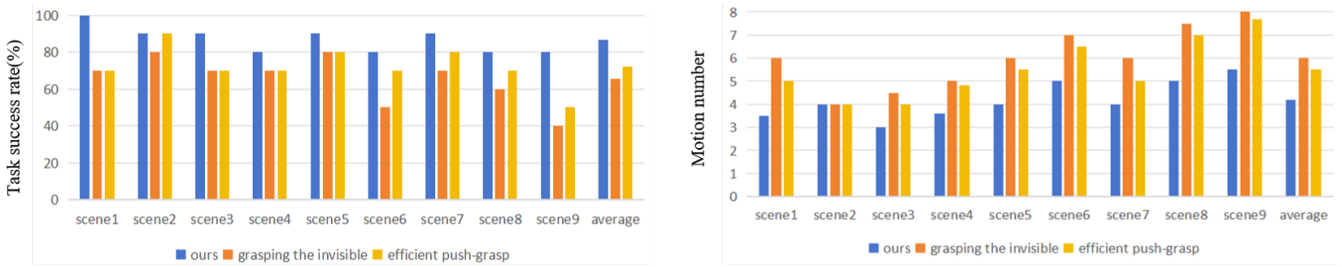

Fig. 7. Results of the real-world tests in terms of (a) the task success rate and (b) the average motion numbers for each test.

We determine the mean success rate of target grasping after every 200 training iterations. During the training process, we limit the number of consecutive grasps to 2 for the methods except Grasping-Only. If this number exceeds 2 without grasping the target object successfully, the system will use a heuristic method to generate a grasp instead of the grasp-net to accelerate training. Also, if the number of total motions exceeds 10 without reaching the target, the scene will be reset. Compared with baselines (shown in Fig. 4), the target grasping success rate of Grasping-Only still has no trend of convergence and stays lower than other methods. We can also see that the success rate of Target-Oriented VPG converges roughly to 0.35 similar to Grasping the Invisible, while the training process of the former one is not stable, indicating that the coordination policy in the latter method is slightly effective in the occlusion scenario. The curve of success rate shows that Efficient Push-Grasping (trained by the scheme proposed in [9]) works better than the other three methods and the success rate convergence is approximately 0.6, which is still lower than the proposed method. For MPGNet, the training process shown in Fig.4 is recorded for stage V. It can be seen that MPGNet achieves the highest grasping success rate compared to other methods, which converges approximately to 0.9. It indicates that MPGNet works effectively and efficiently for target-oriented grasping in occlusion. The visualization of MPGNet's output is shown in Fig. 5.

We also conduct several test experiments in simulation to test whether our method can generalize to other occluded scenes. We test two scenarios with 15 and 30 objects respectively. The test scenes are similar to the joint training scenarios while containing more objects. We randomly select one of the first five objects as the target to ensure that the target is highly occluded, aiming to show the robustness of MPGNet. We run 50 tests for each scenario. The results are reported in Table I, showing that our system achieves the best performance in terms of both task success rate and motion number.

D. Ablation Study

We create three variants of MPGNet, namely MPGNet w/o mt, MPGNet w/o at, and MPGNet w/o at&mt. MPGNet w/o mt denotes that MPGNet is trained without the multi-stage strategy but with the alternative training, indicating that we begin training MPGNet from the joint training stage.

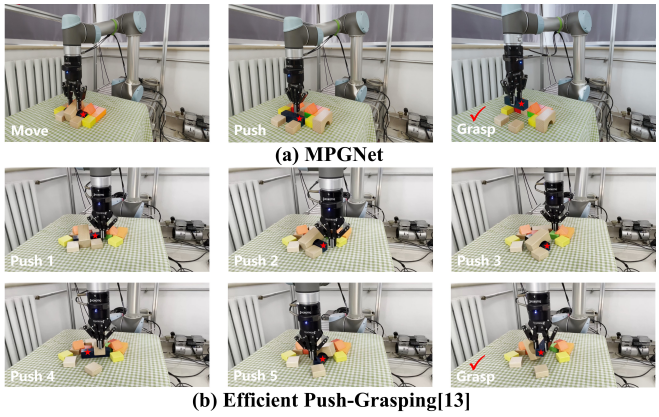

Fig. 8. Comparison of motion sequence. We compare MPGNet with Efficient Push-Grasping [9] through real-world experiments. We can see that our method accomplishes the grasping task with fewer actions.

MPGNet w/o at denotes that MPGNet is trained with the multi-stage strategy but without alternative training, indicating that the action decision in the joint training stage is based on the comparison of the Q values of the three action Q maps instead of the proposed alternative training scheme. MPGNet w/o at&mt denotes that the three action policies are trained jointly without any pre-training and action decisions are made by comparing the Q values. Fig.6 shows that MPGNet outperforms the other three ablated versions in terms of grasping success rate, which demonstrates the effectiveness of the multi-stage and the alternative training strategies.

E. Real-world Experiments

In this section, we test our system in real-world scenarios. We conducted experiments in 9 challenging scenes with seen and unseen objects, in which the target objects are densely occluded. Each scenario contains objects with various textures and shapes. We evaluate our approach by contrasting it with the Efficient Push-Grasping and Grasping the Invisible, both of which have demonstrated commendable results in simulated evaluations. For each test scene, we run 10 rounds. For each round of testing, we ensure consistency in the way the target objects were occluded and adjusted the placement of the remaining objects appropriately. The models we use in real-world experiments are transferred from simulation to reality without any fine-tuning. Results of task success rate and motion number are shown in Fig.7. It can be seen that

MPGNet outperforms the baselines across all the evaluation metrics. Fig.8 shows that our method can grasp the target object with the least motions. As for the generation of the target mask heightmaps, we fine-tune the UOAIS [22] to our scene to generate target mask heightmaps. The output of UOAIS is a set of segmentation masks of the objects in the scene with occlusion labels. In practice, we choose the occluded object with the smallest value in height as the target to ensure that occlusion occurs. Furthermore, in some extra real-world scenarios, we introduce ChatGPT4 [23] as the multimodal large model with the proposed grasp algorithm to realize the occluded object grasping following the language instructions, making our real-world tests more realistic and practical. In detail, we query GPT4 to categorize the detected masks, and the candidate categories are given in advance. Then we prompt the GPT4 to analyze the input language instructions and select the target mask as input for the proposed algorithm. Video demonstration of the real-world tests can be found at: https://youtu.be/S_QKZqkh0w8.

V. Conclusions

In this work, we propose MPGNet, a three-branch network to learn moving, pushing, and grasping synergy for target-oriented grasping in occluded scenes. In practice, we train our network via a multi-stage strategy and combine domain knowledge with Q values to better coordinate the three actions. We compare our method with several baselines, demonstrating that MPGNet converges rapidly with a high grasping success rate. We also evaluate MPGNet via both simulation and real-world experiments, showing that it works well challenging scenes including those that can not be efficiently handled by common push-grasping synergy.

VI. Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grants 61991411 and U22A2057, in part by the National Key Research and Development Plan of China under Grant 2021ZD0112002, in part by the National Natural Science Foundation of China under Grant 62076148, in part by the Shandong Excellent Young Scientists Fund Program (Overseas) under Grant 2022HWYQ-042, and in part by Project for Self-Developed Innovation Team of Jinan City under Grant 2021GXRC038.

References

[1] 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.

[2] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. Aparicio, 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 XIII*, 2017.

[3] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015.

[4] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11444–11453, 2020.

[5] D. Li, P. Wei, C. Zhao, S. Yang, Y. Li, and W. Zhang, "A mobile manipulation system for automated replenishment in the field of unmanned retail," in *2023 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 644–649, IEEE, 2023.

[6] 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)*, pp. 4238–4245, IEEE, 2018.

[7] B. Huang, S. D. Han, A. Boularias, and J. Yu, "Dipn: Deep interaction prediction network with application to clutter removal," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4694–4701, IEEE, 2021.

[8] 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)*, pp. 619–626, Ieee, 2019.

[9] K. Xu, H. Yu, Q. Lai, Y. Wang, and R. Xiong, "Efficient learning of goal-oriented push-grasping synergy in clutter," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6337–6344, 2021.

[10] G. Zuo, J. Tong, Z. Wang, and D. Gong, "A graph-based deep reinforcement learning approach to grasping fully occluded objects," *Cognitive Computation*, vol. 15, no. 1, pp. 36–49, 2023.

[11] 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.

[12] H. Yu, X. Lou, Y. Yang, and C. Choi, "Iosg: Image-driven object searching and grasping," *arXiv preprint arXiv:2308.05821*, 2023.

[13] E. Li, H. Feng, S. Zhang, and Y. Fu, "Learning target-oriented push-grasping synergy in clutter with action space decoupling," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11966–11973, 2022.

[14] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, "6-dof grasping for target-driven object manipulation in clutter," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6232–6238, IEEE, 2020.

[15] F. Liri, H. Lin, K. Lee, B. Fonseca, N. Ruppert, K. George, and A. Panangadan, "Real-time dynamic object recognition and grasp detection for robotic arm using streaming video: A design for visually impaired persons," in *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 0654–0660, IEEE, 2021.

[16] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, *et al.*, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," *The International Journal of Robotics Research*, vol. 41, no. 7, pp. 690–705, 2022.

[17] H. Lin, C. Cheang, Y. Fu, and X. Xue, "I know what you draw: Learning grasp detection conditioned on a few freehand sketches," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 8417–8423, IEEE, 2022.

[18] S. Yang, W. Zhang, R. Song, J. Cheng, and Y. Li, "Learning multi-object dense descriptor for autonomous goal-conditioned grasping," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4109–4116, 2021.

[19] Z. Liu, Z. Wang, S. Huang, J. Zhou, and J. Lu, "Ge-grasp: Efficient target-oriented grasping in dense clutter," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1388–1395, 2022.

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[21] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," 2019.

[22] S. Back, J. Lee, T. Kim, S. Noh, R. Kang, S. Bak, and K. Lee, "Unseen object amodal instance segmentation via hierarchical occlusion modeling," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 5085–5092, IEEE, 2022.

[23] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.