Collaborative Manipulation in Clutter Scenes via Dual-Branch Grasping and Stackelberg Pushing

Jianze Ye[1], Chenghao Li[1], Haolan Zhang[1], Peiwen Zhou[2], and Nak Young Chong[1]

[1]School of Information Science, Japan Advanced Institute of Science and Technology
Ishikawa 923-1292, Japan ({s2310176, chenghao.li, s2420423, nakyoung}@jaist.ac.jp)
[2]Southwest Automation Research Institute
Mianyang, China (Zhoupeiwen0722@outlook.com)

Abstract: In cluttered scenes, effective object manipulation often requires both precise grasping and proactive scene rearrangement. We propose a dual-branch reinforcement learning framework that separately predicts grasp position and orientation, trained via supervised pretraining and shaped rewards to ensure stable and sample-efficient learning. To minimize unnecessary pushing, we model the coordination between grasp and push agents as a Stackelberg game, where the push agent acts only when grasp success is unlikely, to enhance downstream grasp success.. Experimental results in simulation show that our method improves grasp success and action efficiency, outperforming existing baselines in both success rate and policy economy.

Keywords: Robotic manipulation; Deep reinforcement learning; Grasp planning; Stackelberg game; Multi-agent coordination; Pushing and grasping; Cluttered environment.

1. INTRODUCTION

Manipulation in cluttered environments constitutes a core challenge in robotic manipulation, where the presence of occlusions, ambiguous object geometries, and physical interactions between objects significantly hinder reliable execution [1] [2][3][4][5]. Unlike isolated settings, cluttered scenes often restrict the robot's ability to directly access feasible grasp configurations, resulting in high failure rates and reduced task efficiency [6][7].

Addressing these challenges requires more than reactive grasp planning. A robotic system must actively modify the environment to expose graspable surfaces or isolate target objects. To this end, various studies have explored the integration of non-prehensile actions—such as pushing—into grasping pipelines. These methods typically aim to improve grasp success by using pushing to rearrange the scene, often within a reinforcement learning framework. While they differ in aspects such as network architecture, reward design, or training strategy, they share the common goal of enabling more effective grasping in dense and cluttered environments [8] [9] [10] [11][12][13][18].

A representative and widely adopted approach in this direction is Visual Pushing for Grasping (VPG), which jointly learns pushing and grasping policies via self-supervised Q-learning in cluttered scenes [13]. VPG demonstrates that enabling a robot to perform pushing actions before grasping can significantly improve overall manipulation success, particularly in densely piled environments. Despite its effectiveness, VPG exhibits two key limitations. First, it discretizes grasp orientations into a fixed set of angles, which restricts the policy's flexibility in handling diverse object configurations [11]. Second, it models pushing and grasping as loosely coupled modules, rather than relying on an explicit coordination mechanism. This often results in redundant or unnecessary pushing actions that do not meaningfully improve graspability. Several follow-up approaches attempt to stabilize training via staged grasp-then-push schemes, but may lack explicit coordination and offer limited gains in action efficiency [12]. In contrast, multi-agent reinforcement learning frameworks, such as those based on Stackelberg games, offer structured paradigms for agent coordination, enabling one agent to optimize its behavior in anticipation of the other's response [14][15][16].

Our work focuses on this challenge within the context of top-down grasping, a common formulation in tabletop manipulation. In this setting, each grasp is defined by a planar position on the heightmap and an in-plane rotation angle (yaw), which captures the essential geometry needed for parallel-jaw grasping while simplifying both perception and control. This abstraction is well-suited to RGB-D based systems and has been widely adopted in recent robotic manipulation literature.

To overcome the limitations of previous push-grasp frameworks, we propose a reinforcement learning approach that integrates fine-grained grasp modeling with structured agent coordination. Specifically, the grasping policy is decomposed into two branches: one predicts pixel-wise grasp positions over the input heightmap, while the other regresses a continuous grasp angle based on a local height patch extracted around the selected location. This decoupled architecture allows spatial and angular reasoning to be learned separately, resulting in more expressive and adaptable grasp behavior across diverse object configurations.

In parallel, we model the interaction between pushing and grasping using a game-theoretic formulation. Rather than relying on heuristic triggers or rigid switching, we treat the pushing policy as a strategic leader in a Stackelberg game, selecting actions that improve the future

This work was supported by JSPS KAKENHI Grant Number JP23K03756.

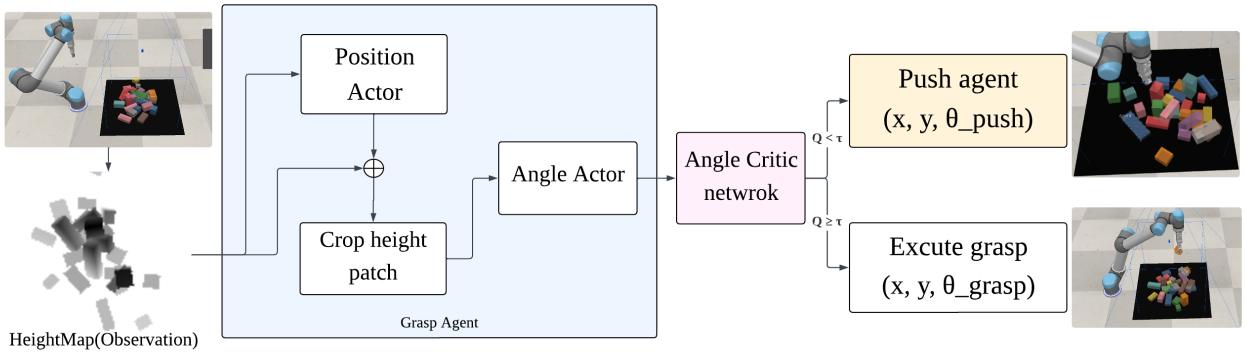


Fig. 1.: Action Selection Pipeline. The grasp agent predicts a grasp position from the full heightmap and a grasp angle from a cropped local patch. The proposed action is evaluated by a pretrained angle critic. If the Q-value is below a threshold τ, a push is triggered at the same location; otherwise, the grasp is executed.

graspability of the scene [16][17]. The grasping agent, acting as the follower, responds by estimating grasp success given the modified environment. This asymmetric decision structure allows the pushing agent to act efficiently—only intervening when a grasp is predicted to fail and when a push is expected to improve downstream outcomes. The complete system architecture is depicted in Fig. 1 and detailed in Section 2.

Together, these components form a flexible and efficient manipulation system capable of handling complex, cluttered environments with reduced redundant actions and improved grasp success.Our work focuses on 2D top-down grasping, where each grasp is defined by a position on the heightmap and an in-plane rotation angle (yaw). This formulation simplifies grasp representation while capturing essential geometric information for parallel-jaw grasping.

Our main contributions are summarized as follows:

- We propose a grasping policy with a dual-branch architecture that decouples position and angle prediction, enabling more expressive and continuous grasp action representation.
- We introduce a structured coordination mechanism between pushing and grasping policies by formulating their interaction as a Stackelberg game, where the push agent anticipates grasp outcomes to guide efficient pre-manipulation.
- We pretrain the grasp policy components using successful grasp-only experience collected in simulation, improving policy initialization and training stability.
- Our method demonstrates improved grasp success and manipulation efficiency in cluttered scenes, reducing redundant actions compared to existing push-grasp baselines.

Underlying these design choices is a key insight: robust grasping capabilities can substantially reduce the need for preparatory actions such as pushing. While pushing is useful for resolving occlusions or repositioning objects, excessive reliance on it can introduce inefficiencies. To mitigate this, we enhance the grasping agent's spatial and angular reasoning through architectural decoupling and targeted pretraining. By enabling the agent to handle complex configurations independently, our system minimizes unnecessary interventions and achieves more efficient.

2. METHOD

2.1 Overall Framework

We propose a grasp-centric reinforcement learning framework that decouples position and angle prediction into two separate branches via a modular grasp agent. This design enables the network to specialize in spatial and angular reasoning independently, which is critical for robust robotic manipulation in cluttered environments. To stabilize training, both branches are first pretrained using self-supervised data collected in CoppeliaSim. To minimize unnecessary interactions, the system triggers a pushing action only when the predicted grasp quality is low, a pushing action is conditionally triggered to improve future graspability, thereby minimizing unnecessary interaction.

Fig. 1 illustrates the complete inference procedure of our system. At each time step, a heightmap is captured from the CoppeliaSim environment. The grasping agent consists of two branches, each with its own convolutional encoder: the position branch processes the full heightmap to select a grasp point, while the angle branch takes a cropped local patch centered at this location to predict the grasp orientation. These two outputs jointly define the candidate grasp action. To decide whether this grasp should be executed, the system queries the angle critic network trained during reinforcement learning. The local patch and predicted angle are passed through the frozen critic to estimate the expected Q-value. When the angle critic predicts low Q-value for a proposed grasp, the corresponding local patch is reused as input to the push agent, which predicts a push direction. The push action is then executed from the same (x, y) location, with the predicted angle applied as the direction of motion. Otherwise, the grasp is executed directly.

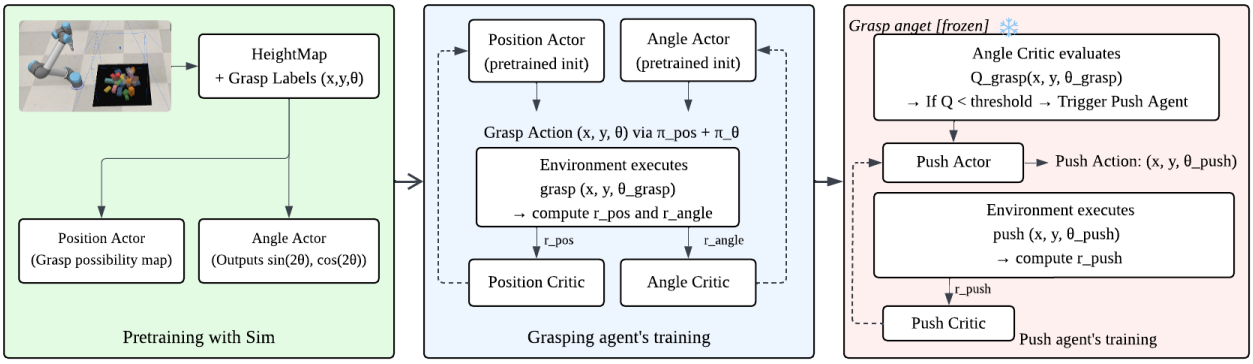


Fig. 2.: Training pipeline of our method. The procedure consists of three stages: self-supervised pretraining using labeled simulation data, grasping agent training with branch-specific reward signals, and delayed push agent training triggered only when the grasping critic predicts low Q-values. Each component is optimized using a TD3-based actor-critic architecture.

2.2 Grasping and Pushing Networks

Our system consists of three neural networks: grasping actor's two branches and a pushing angle prediction network.

The grasping actor comprises a Position branch and an Angle branch. The Position branch predicts per-pixel grasp quality scores using two lightweight convolutional layers applied to the full feature map. The Angle branch takes a local patch centered at the selected grasp location and regresses the grasp orientation as a 2D vector $[\sin(2\theta), \cos(2\theta)]$, allowing continuous and symmetric angle representation.

The pushing angle prediction network shares the same structure as the Angle branch but predicts the pushing direction as $[\sin(\theta), \cos(\theta)]$. Since pushing lacks symmetry, this formulation preserves the full directional resolution. All modules are kept lightweight to enable fast inference and seamless integration with the robot control loop.

2.3 Reward Formulation

To encourage cooperative behavior between the grasping and pushing agents, we formulate the reward design under a Stackelberg game framework. In this setup, the pushing agent acts as the leader, making decisions that anticipate the optimal response of the grasping agent, which serves as the follower.

Grasping reward. The grasping agent is supervised via a reward vector $R = [R_{\text{pos}},\ R_{\text{ang}}] \in \mathbb{R}^{\text{K}}$, where each component supervises one branch of the grasp policy.

Position reward. The position branch receives a dense reward signal that evaluates the spatial plausibility of the selected grasp location:

$$R_{\text{pos}} = \begin{cases} 1.0 & \text{success} \\ w_{\text{dist}} \cdot e^{-d_{\min}} + w_{\text{open}} \cdot (1 - D_{\text{patch}}) & \text{otherwise} \end{cases} \tag{1}$$

Here, $d_{\min}$ denotes the distance to the nearest object center, encouraging the selection of positions physically close to objects, and D_{patch} quantifies the local density within the cropped patch, penalizing grasp attempts in cluttered regions. The two terms are combined with weights $w_{\text{dist}} = 0.5$ and $w_{\text{open}} = 0.3$.

We adopt an exponential decay for the distance term to impose higher sensitivity near object centers—where small deviations can significantly affect grasp success. In contrast, the openness component uses a linear function to provide a stable and interpretable gradient reflecting spatial clearance. This hybrid formulation ensures that the overall reward remains differentiable and additive, while enabling distinct gradient behaviors: the exponential term yields sharp gradients in close-range scenarios, facilitating fine-tuned grasp localization; the linear term maintains consistent feedback about surrounding clutter. Together, they promote grasp locations that are both reachable and minimally obstructed.

Angle reward. The angle branch receives a sparse, occlusion-aware reward signal defined as:

$$R_{\text{ang}} = \begin{cases} 1.0 & \text{success} \\ \lambda \cdot \left(1 - \mathbb{E}_{\text{ı} \in \mathbb{L}(\theta)} \left[\mathbb{I}(\text{F}_{\text{ı}} \geq \text{F})\right]\right) & \text{otherwise} \end{cases} \tag{2}$$

In Eq. (2), $L(\theta)$ refers to a sampled line segment along the predicted grasp direction centered at the grasp location. For each pixel $p \in L(\theta)$, z_p is the height at p, and z_c is the height at the grasp center. The expectation computes the fraction of occluding pixels where $z_p \geq z_c$. A small scaling factor $\lambda = 0.2$ is used to maintain smooth gradients while avoiding undue influence on the critic's learning signal. This formulation encourages the agent to prefer grasp orientations that approach from less occluded directions, improving feasibility and robustness in cluttered scenes.

Pushing reward The pushing agent does not receive immediate feedback from the environment. Instead, its reward is computed based on grasp success over a look-ahead window T:

$$R_{\text{push}} = \frac{1}{T} \sum_{t=1}^{T} (\text{grasp}_t = \text{success}) \tag{3}$$

This formulation quantifies how many objects are successfully grasped within $T = 3$ steps following a push action.

2.4 Training Procedure

Our training procedure is composed of three sequential stages, each designed to improve stability and efficiency in learning cooperative behavior between agents. The overall training pipeline is illustrated in Fig. 2.

1. **Self-supervised pretraining.** We first train the grasping agent via self-supervised learning using labeled data collected in simulation. Grasp success or failure serves as supervision for both the position scoring and angle prediction networks. We use cross-entropy loss for the position heatmap and regression loss for the angle output.

We construct a structured pretraining dataset by executing grasp attempts in simulation using a fixed policy that always targets the tallest object in the scene. The center pixel location and in-plane orientation (yaw) of the topmost object are extracted from the simulator and used as supervision targets, with the corresponding heightmap as input. Each attempt is executed, and the outcome (success or failure) is recorded automatically, enabling self-supervised labeling without human annotation.

To ensure the effectiveness of pretraining, we retain only successful grasp examples. This design choice is motivated by two key considerations. First, failure cases generated by a fixed policy tend to be repetitive—often resulting from identical infeasible configurations—and thus contribute limited diversity. Second, successful examples offer diverse, high-quality demonstrations of feasible grasps, guiding the network toward actionable patterns. This positive learning bias improves the stability of subsequent reinforcement learning.

2. **Grasping agent** After pretraining, the grasping policy is further optimized via reinforcement learning and is composed of two decoupled branches: a position network that evaluates grasp quality over the entire heightmap, and an angle network that predicts an optimal grasp orientation from a local patch centered at the selected location. Although both branches contribute to the final action, their decoupling introduces a structural credit assignment challenge: a suitable grasp position might still result in failure if paired with a poor angle prediction, making it difficult for the position network to receive appropriate feedback. To mitigate this, we apply the shaped rewards described in Subsection 2.3. The position reward is weighted such that its components sum to approximately 1.0 ($w_{\text{dist}} = 0.5$, $w_{\text{open}} = 0.3$), ensuring well-scaled gradients and preserving learning signal integrity. This design adheres to the stationarity principle in reinforcement learning [19], promoting stable training dynamics, and supports more accurate credit assignment [20] by isolating each branch's contribution to the final outcome. Overall, this structure enhances training robustness and enables the agent to learn more resilient grasp strategies in cluttered environments.

3. **Pushing agent** Once the grasping agent is sufficiently trained, the pushing agent is activated only when the angle critic predicts a low Q-value for the current grasp candidate. Inspired by Stackelberg game theory [21], we model the pushing agent as a strategic leader and the grasping agent as a reactive follower. In this asymmetric framework, the push policy selects actions that reshape the scene in anticipation of the follower's grasp response. To reflect this hierarchy, the pushing agent is optimized using a delayed reward signal based on the grasp outcomes within a look-ahead window. Specifically, we evaluate the average grasp success over the subsequent T timesteps and backpropagate this signal through the pushing policy using TD3. This design aligns with the principles of Stackelberg multi-agent reinforcement learning frameworks [17], allowing the leader to improve downstream task performance while avoiding redundant interventions.

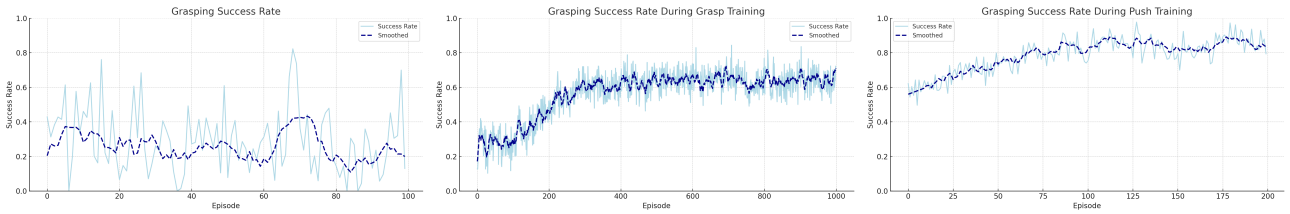
The proposed dual-agent framework integrates grasp position and angle reasoning with strategic push-grasp coordination under a Stackelberg formulation. Each component—pretraining, branch-specific reward shaping, and hierarchical policy learning—has been designed to enhance sample efficiency and decision robustness in cluttered manipulation tasks. In the following section, we evaluate our approach through a series of simulation-based experiments, examining both individual component performance and the effectiveness of the overall system.

3. EXPERIMENTS

3.1 Experimental Setup

All experiments are conducted in the CoppeliaSim simulation environment. A fixed RGB-D camera is mounted 0.5 meters above the workspace center, capturing orthographic top-down views. The depth image is projected into a 100×100 heightmap, which serves as the primary input to both the grasp and push policies. Each training episode corresponds to one simulated scene. At the start of an episode, 25 blocks with randomized dimensions are dropped from a height of 0.3 meters, forming cluttered configurations. The episode ends when either all objects are successfully removed from the workspace or 10 consecutive grasp attempts fail, at which point the episode is considered complete.

The robot performs top-down parallel-jaw grasps using a fixed gripper width of 6 cm, with the grasp point and in-plane rotation predicted by the learned policy. Push actions are defined as 10 cm linear motions centered at grasp candidates with low predicted Q-values; the direction of the push is predicted by the push agent. To support grasp angle prediction, a 24×24 local heightmap patch is cropped around each candidate position. This patch size reflects the physical scale of the task: the gripper width corresponds to roughly 12 pixels, and the push length spans about 20 pixels. The expanded receptive field allows the network to consider not only the target object but also surrounding context, which is crucial for inferring feasible grasp orientations.



(a) Validation after pretraining. (b) Grasp agent training (c) Push agent training

Fig. 3.: Grasp success rate curves for each stage of our framework.

3.2 Evaluation Metrics

We evaluate our method based on several metrics that capture both the effectiveness and efficiency of the learned policies:

- **Grasp Success Rate** In one episode the proportion of successful grasp attempts over all grasp executions.
- **Action Efficiency** Defined as the ratio of successfully grasped objects to the total number of executed actions (grasp + push). This metric captures the policy's ability to complete the task with minimal redundant actions.
- **Completion** The percentage of episodes in which all objects are successfully removed from the workspace. An episode is marked incomplete if no object is grasped after 10 consecutive attempts.

3.3 Quantitative Results

We evaluate the manipulation performance across the three learning stages of our framework using grasp success rate curves.

Validation after Pretraining Fig. 3a shows the grasp success rate evaluated over 100 episodes after pretraining. While the average success rate remains around 25%, indicating that the network has learned a basic grasping strategy, performance exhibits high variance. In some episodes, the success rate drops to zero due to cluttered scenes and the lack of adaptability. This instability stems from the deterministic nature of the pretraining policy, which always targets the tallest object, making the network highly sensitive to scene randomness.

Grasp Agent Training Fig. 3b illustrates the grasp success rate during Grasp agent training. The training begins with a noticeable performance advantage, indicating that the pretrained position and angle networks provide a good initialization for policy learning. As training progresses, the success rate steadily improves, eventually stabilizing around 65%. This suggests that the combination of branch-specific rewards and a dual-critic TD3 architecture enables effective and stable policy refinement. Overall, the grasp agent successfully learns to handle cluttered scenes through a decoupled, well-initialized learning structure.

Push Agent Training Fig. 3c shows that once the push agent begins training, the grasp success rate steadily increases from around 60% to approximately 80%. This indicates that the learned push actions effectively enhance graspability by improving scene conditions, ultimately leading to more successful grasp executions.

Comparison with Baselines Table 1 summarizes system performance across various configurations. The *Pretrained Only* baseline achieves 27.3% grasp success, indicating that while fixed-policy pretraining offers a useful initialization, it lacks adaptability in complex environments. Introducing reinforcement learning in the *Grasp Agent Only* setup significantly improves grasp success to 63.8%, with a task completion rate of 93.2%. These results highlight the benefits of our dual-branch architecture and dense reward design in enabling robust grasp behavior. Further gains are realized by incorporating the push agent. The *Full System-0.7* configuration, using a Q-threshold of 0.7, achieves 81.5% grasp success and 76.4% action efficiency—demonstrating that the agent learns to intervene selectively and meaningfully, enhancing graspability while minimizing redundant actions. Raising the threshold to 0.8 (*Full System-0.8*) increases grasp success to 86.0%, but also leads to more frequent pushing and reduced action efficiency (68.2%). This reflects a fundamental trade-off between reliability and operational economy. Our selected threshold strikes a practical balance, reinforcing the value of Stackelberg coordination in regulating intervention frequency based on grasp confidence.

Fig. 4 presents representative simulated cases. In examples (1) and (2), the grasp agent successfully selects and executes grasps on isolated blocks. In contrast, examples (3) and (4) depict scenarios where the grasp candidates are located in cluttered regions. In these cases, the angle critic predicts low graspability, prompting the push agent to rearrange the scene. These examples demonstrate how the system prioritizes direct grasps when feasible and resorts to pushing only when necessary to improve grasp conditions.

In summary, our experiments confirm that fine-grained grasp modeling combined with structured push-grasp coordination leads to improved performance and efficiency in cluttered environments. We now conclude with a summary of our contributions and insights.

4. CONCLUSION

We presented a dual-branch reinforcement learning framework for robotic manipulation in cluttered environments, addressing both fine-grained grasp modeling and strategic push-grasp coordination. By decoupling grasp position and orientation into separate prediction branches, our method enabled more expressive and flex-

Table 1.: Performance comparison with baseline, ablated versions, and prior methods under push and no-push settings.

Method	Completion (%)	Grasp Success (%)	Action Efficiency (%)
VPG (Grasp Only)	90.5	55.8	55.8
VPG	100	67.7	60.9
Pretrained Only (Ours)	38.6	27.3	27.3
Grasp Agent Only (Ours)	93.2	63.8	63.8
Full System-0.7 (Ours)	100	81.5	**76.4**
Full System-0.8 (Ours)	100	**86.2**	73.9

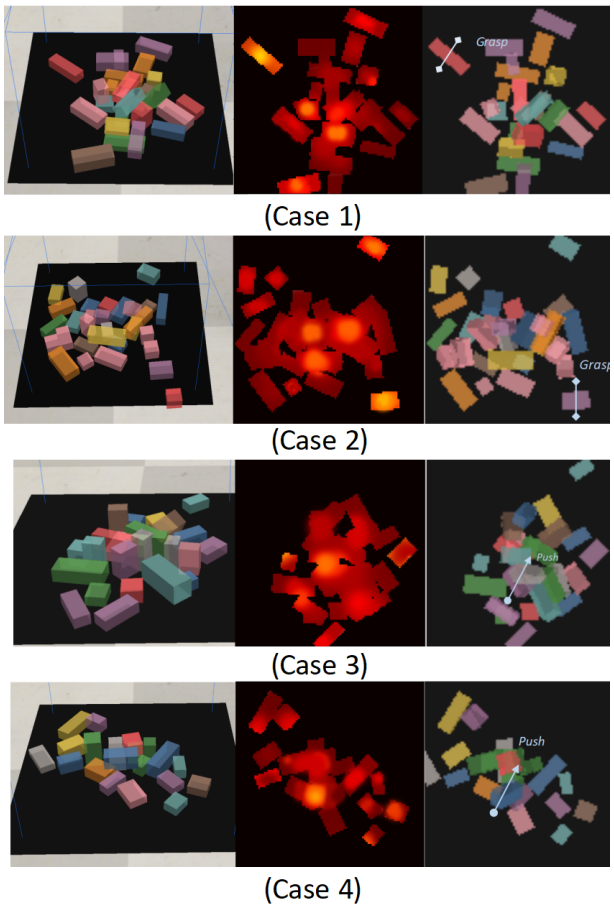


Fig. 4.: Action visualization for grasp and push. Each case shows CoppeliaSim environment, predicted heatmap, and the final action.

ible grasping. Self-supervised pretraining further enhances stability and provides a better initialization for reinforcement learning.

To minimize unnecessary pushing, we introduced a Stackelberg game-based formulation, where the push agent acts as a strategic leader and intervenes only when the grasp Q-value is low. This structured coordination improved both grasp success and action efficiency. Extensive experiments in simulation demonstrated that our approach outperforms existing push-grasp baselines in terms of grasp success, scene completion, and policy economy. Qualitative results confirmed that the system selectively pushes only when necessary, reducing redundant actions and improving task efficiency.

REFERENCES

[1] Mohammed, M.Q.; Kwek, L.C.; Chua, S.C.; Al-Dhaqm, A.; Nahavandi, S.; Eisa, T.A.E.; Miskon, M.F.; Al-Mhiqani, M.N.; Ali, A.; Abaker, M. et al. Review of Learning-Based Robotic Manipulation in Cluttered Environments. Sensors; 2022; 22, 7938. [DOI: https://dx.doi.org/10.3390/s22207938]

[2] R. Newbury et al., "Deep learning approaches to grasp synthesis: A review," IEEE Trans. Robot., to be published, doi: 10.48550/arXiv.2207.02556.

[3] C. Li, N. Y. Chong,"Monozone-Centric Instance Grasping Policy in Large-Scale Dense Clutter" *IEEE/ASME Trans. Mechatron.*, earliy access, 2025.

[4] P. Zhou, Z. Gao, C. Li, and N. Y. Chong, "An efficient deep reinforcement learning model for online 3d bin packing combining object rearrangement and stable placement," in 2024 24th International Conference on Control, Automation and Systems (ICCAS), 2024, pp. 964–969.

[5] H. Zhang, J. Tang, S. Sun, X. Lan, Robotic Grasping from Classical to Modern: A Survey, arXiv Preprint arXiv:2202.03631, 2022.

[6] S. Kumra, S. Joshi, and F. Sahin, "GR-ConvNet v2: A real-time multi-grasp detection network for robotic grasping," Sensors, vol. 22, no. 16, 2022, Art. no. 6208.

[7] M. B. Imtiaz, Y. Qiao, and B. Lee, "Prehensile and non-prehensile robotic pick-and-place of objects in clutter using deep reinforcement learning," Sensors, vol. 23, no. 3, p. 1513, Jan. 2023, doi: 10.3390/s23031513.

[8] M. Zhao, G. Zuo, S. Yu, D. Gong, Z. Wang, and O. Sie, "Positionaware pushing and grasping synergy with deep reinforcement learning in clutter," CAAI Trans. Intell. Technol., 2023, to be published, doi: 10.1049/CIT2.12264.

[9] H. Zhang et al., "Reinforcement learning based pushing and grasping objects from ungraspable poses," 2023, arXiv:2302.13328.

[10] 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 Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Macau, China, Jan. 2019, pp. 619–626, doi: 10.1109/IROS40897.2019.8967899.

[11] A. A. Shahid, L. Roveda, D. Piga, and F. Braghin, "Learning continuous control actions for robotic

grasping with reinforcement learning," in Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC), Oct. 2020, pp. 4066–4072.
[12] Y. Wang, K. Mokhtar, C. Heemskerk, and H. Kasaei, "Self-supervised learning for joint pushing and grasping policies in highly cluttered environments," in Proc. IEEE Int. Conf. Robot. Automat., 2024, pp. 13840–13847.
[13] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with selfsupervised deep reinforcement learning," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 4238–4245.
[14] M. Wen et al., "Multi-agent reinforcement learning is a sequence modeling problem," in Proc. Int. Conf. Neural Inf. Process.Syst., 2022,pp. 16509–16521.
[15] J. J. Koh, G. Ding, C. Heckman, L. Chen, and A. Roncone, "Cooperative control of mobile robots with stackelberg learning," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS). IEEE, 2020, pp. 7985–7992.
[16] Zhang B, Li L, Xu Z et al (2023a) Inducing stackelberg equilibrium through spatio-temporal sequential decision-making in multi-agent reinforcement learning. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI '23.
[17] K. Zhang, Z. Yang, and T. Basar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," in Handbook of Reinforcement Learning and Control. Cham, Switzerland: Springer, 2021, pp. 321–384. [Online].
[18] D. Morrison, P. Corke, and J. Leitner, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," in Proc. Robot.: Sci. Syst. Conf., 2018.
[19] Y. Bengio, M. Delalleau, and A. Le Roux, "Meta-learning for stochastic gradient MCMC," in Proceedings of the 36th International Conference on Machine Learning (ICML), Long Beach, CA, USA, 2019, pp. 524–533.
[20] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[21] H. von Stackelberg, *Market Structure and Equilibrium*. Vienna, Austria: Springer, 1952. (Originally published in German as *Marktform und Gleichgewicht*, 1934.)