Kalpa Publications in Computing

Volume 22, 2025, Pages 486–498

Proceedings of The Sixth International Conference on Civil and Building Engineering Informatics

Teaching Robot End Effectors to Grasp Construction Tools Based on Deep Reinforcement Learning

Xiaohu Yu[1], Yantao Yu[2] , Zaolin Pan[3]

1) Research Assistant, Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology, Hong Kong, SAR. Email: xiaohutruth@gmail.com
2) Ph.D., Assis. Prof., Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology, Hong Kong, SAR. Email: ceyantao@ust.hk
3) Ph.D. Candidate, Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology, Hong Kong, SAR. Email: zpanaq@connect.ust.hk

Abstract: There are numerous delicate tasks that require skilled workers to perform in onsite construction. A key challenge in automating these tasks is developing motor skills in robots trained through reinforcement learning (RL), as manipulating irregular and delicate objects like hammers, scaffolding, and drills remains difficult. To address this issue, this paper proposes an RL-based approach to perform delicate tasks using a robotic arm with grippers. We present a simulation-based policy learning framework utilizing the Critic-Actor algorithm in Pybullet to control the robotic arm. In experimental trials, the learned policy was used to grasp six different types of construction tools, and the results demonstrated the feasibility of training with randomly shaped objects to manipulate the construction tools with a reasonable success rate. This method provides a foundation for enhancing the manipulative skills of construction robots, potentially reducing labor costs in the industry.

Keywords: Construction robot; Reinforcement learning; Construction tools; Grasping

1. INTRODUCTION

The construction industry needs adequate workers to meet the construction demand [1]. However, construction tasks purely based on manual labor are time-consuming and inefficient [2,3]. Meanwhile, onsite construction is dangerous for workers, and the monotonous physical activities affect their health status [4]. Therefore, improving the automation level in construction is critical to this industry in the future. To this end, construction robots show huge potential to tackle this urgent issue across a wide range of fields, from off-site prefabrication [5] to on-site construction [6]. Although construction robots can solve these problems, the technology is not widely used in civil engineering and on-site construction due to the complexity of construction scenarios. Currently, most robots utilized in onsite construction are preprogrammed [7] or motion-planned [8], which are only suitable under a limited number of working conditions [9]. These robots perform poorly in unseen construction environments because it is impossible to plan all different possibilities in one program.

Recent review papers [10] discussing and envisioning robotics in construction synchronously share the perspective that the emergence of reinforcement learning (RL) provides a promising avenue. With the prospect of machine learning, RL is promising for developing construction robots with adaptability and flexibility, which can handle the problems mentioned above. RL can learn a specific policy (from observations to actions) for conducting different tasks by interacting with given construction

environments through trial and error [11].

Next, we explain how RL components operate within a specific context—training a robotic arm with an RGB-D camera to perform object pickup tasks. The robot arm is identified as the agent, which interacts with its surroundings. The environment encompasses the entire experimental setup, including anything that the robot interacts with during its task. Observations consist of visual data collected by the RGB-D camera, providing the robot arm with the necessary information to make decisions about its actions. Actions refer to the commands sent to the robot arm to control its joints and grippers, dictating how the arm moves and interacts with objects in its environment. The reward function is crucial in RL as it provides the incentive for the agent to learn [12]. It is used to update the control policy based on the actions taken. Crafting an effective reward function is particularly challenging in complex, long-horizon tasks, which involve multiple steps or subtasks. The control policy dictates the robot arm's movements based on the input from visual data. A well-trained policy enables the robot to successfully complete tasks like picking up objects.

In construction, researchers have conducted many attempts to train a robot using RL in various fields, such as installation [13], assembly [14], navigation [15] and so on. A more comprehensive summary of RL-based robots in construction is provided in recent literature [16]. On the one hand, existing RL-based works have concentrated on heavy construction tasks involving equipment such as robotic arms, cranes, wheel loaders, and unmanned ground vehicles. There is limited attention in delicate work, such as drilling, bolt insertion, and nut tightening, which need to be carried out by either humans or light small-scale robots. On the other hand, RL has emerged as a model-free approach that enables end-to-end robot grasping by learning through continuous trial-and-error interactions with the environment. This allows for self-supervised learning in robotic grasping tasks. However, a major challenge in learning-based robot grasping is how to realize the generalization—whether a robot can apply the knowledge it has learned in self-supervised settings to new environments and unfamiliar objects. While RL has been widely applied in video games and simple simulated robots, its application to complex robotic grasping remains limited. For example, successful grasping of new objects in prior research required extensive training, such as 580,000 grasp attempts over several weeks [17] or 1 million grasp attempts in other studies, making the process costly and difficult to replicate [18].

To address this challenge, this paper introduces a simulation-based policy learning framework utilizing the Critic-Actor algorithm. By training for just 12 hours with randomly shaped objects, this approach achieves a 50-60% grasp success rate on a test set on unseen construction tools. Moreover, the on-policy DRL algorithm Proximal Policy Optimization (PPO) is employed, which offers stable and efficient training by iteratively updating the actor network with a small number of samples over multiple training rounds. This method addresses the issues of step-size selection and variance in traditional policy gradient algorithms.

The key contributions of this paper are:

1. We propose a model-free robotic grasping approach based on an on-policy reinforcement learning (RL) algorithm that enables generalization within a shorter training time, without requiring prior knowledge of target objects or large amounts of experience data.
2. In the experiment, six types of construction tools were used as objects for robotic manipulation, including a hammer, pipe, drill, and others.
3. The experimental results demonstrate that our approach successfully trains the robotic hand to grasp construction tools, achieving success rates of approximately 50-60%.

This article is organized as follows: we proposed a simulation-based policy learning framework utilizing the Critic-Actor algorithm in Section.2. The experiment settings using robot arms to grasp construction tools are detailed in Section 3. Experiment results and conclusion are in Section 4 and 5.

2. METHOD

2.1 Reinforcement learning algorithms

Reinforcement learning (RL) is a novel computational approach. Different from other

computational approaches [19], the agent (Construction robot with gripper) in RL learns by directly interacting with its environment, without the need of supervision or prior knowledge of the environment. The RL process is the formal framework of a Markov decision process (MDP). which can be defined by a quadruplet (s_t, a_t, s_{t+1}, r_t) during the interaction between agents and the environment. During interaction at each episode, the agent based on its specific policy function $\pi_\theta(a \mid s)$ which generates an action a_t. The action next acts on the environment in the state s_t, then the environment represents as a next state s_{t+1}. The interaction result can be assessed by the reward $r(a_t, s_t) = r_t$ (see Fig. 1). Normally, the agent considers a finite-horizon, which means agent explores finite steps T at each episode. The final goal of the agent is to understand the task and learn an optimal policy $\pi_\theta(a \mid s)$ to maximize the discounted cumulative returns G_t ,which is expressed by:

$$G_t = \sum_{k=t}^{T} \gamma^{k-t} r_k(a_k, s_k) \tag{1}$$

Where $t \in [1, T]$, $\gamma \in [0, 1]$ is the discount factor.

In this paper, an actor-critic algorithm is introduced, which combines two different networks. The actor network is responsible for actions a_t through a specific policy function $\pi_\theta(a \mid s)$. The critic network generates the estimation of the value function $V(s)$ or the action value function $Q(s, a)$ to evaluate the actor network.

Firstly, the critic network will be updated by minimizing the time difference error (TD-error). The action value function $Q(s_t, a_t)$ estimated by the critic network can be given by:

$$Q(s_t, a_t) = E_{s_{t+1}, a_{t+1}, \ldots}(G_t \mid s_t, a_t) \tag{2}$$

Which Q represents the long-term accumulated return given a specific action choice a_t at each state s_t.

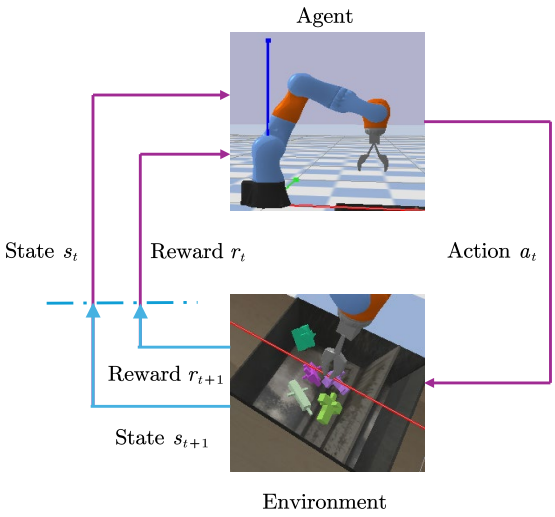


Fig. 1 Interaction between agent and environment in reinforcement learning

The state-value function $V(s_t)$ can be expressed by:

$$V(s_t) = E_{a_t, s_{t+1}, \ldots}(G_t \mid s_t) \tag{3}$$

Unlike the action value function, the value function does not take the action into account and only considers the state.

Furthermore, the loss function for the critic network is defined by:

$$L(\omega_t) = r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \omega_{t-1}) - Q(s_t, a_t; \omega_t) \tag{4}$$

Where $a_t \sim \pi_\theta(a_t \mid s_t)$, $s_{t+1} \sim P(s_{t+1} \mid (s_t, a_t))$. P is the transition probability distribution. The critic network updates parameters of ω_t to decrease $L(\omega_t)$ so that the real return by take the a_{t+1} approaches the expected return to take the action a_t.

Secondly, the actor network is trained by a strategic gradient method, in which θ is updated according to the ascending direction of the gradient to maximize the critic output:

$$\theta \leftarrow \theta + \nabla_\theta log \pi_\theta(a_t | s_t) V(s_t) \tag{5}$$

Where θ represents the parameters of the Actor network, $\pi_\theta(a_t|s_t)$ is the probability of taking action a_t in state s_t, and $V(s_t)$ is provided by the Critic network. The update rule indicates that the parameters of the Actor network are updated in the direction of increasing the output of the Critic network, meaning that the Actor is learning to take actions that the Critic network will lead to higher rewards.

2.2 Model structure

The actor-critic structure utilized in this paper is shown in Fig. 2. A common feature-network is shared by both actor and critic networks to extract the input s_t features, where s_t is an RGB image of size $(48, 48, 3)$. To be more specific, the actor network generates continuous actions represented by Cartesian displacements $(dx, dy, d\theta)$, where dx, dy represent the x-axis offset, y-axis offset, respectively. $d\theta$ denotes the orientation of the gripper, which rotates along the local coordinate system's z-axis.

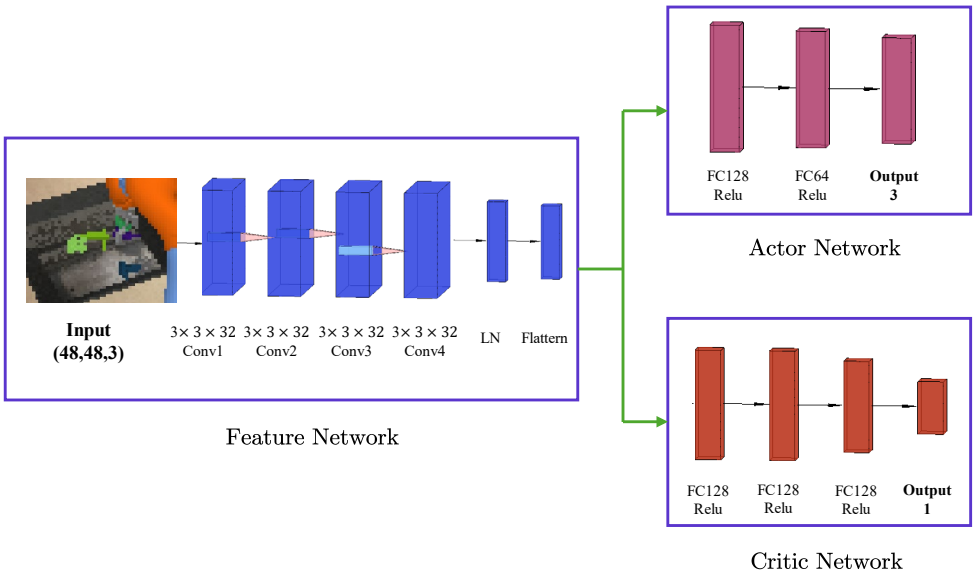


Fig. 2 Actor-critic architecture

2.3 Proximal Policy Optimization

PPO [20] is a widely used algorithm in the field of RL. It aims to optimize policy gradients in a more stable and efficient way than traditional policy gradient algorithms [21] by introducing a clipped objective function and a trust region. The core concept of PPO is to limit the amplitude of policy updates in each iteration using the probability ratio between the current policy and the old policy to avoid significant fluctuations in performance. This probability ratio is implemented through a clipping function, which is usually set to a small value, such as 0.1 or 0.2, to ensure the similarity between the new and old policies. The objective function of PPO can be expressed as:

$$L^{CLIP}(\theta) = E_t\left[\min\left(r_t(\theta)\hat{A}_t,\ clip\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right] \tag{6}$$

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \tag{7}$$

$$A_\pi(s_t, a_t) = Q(s_t, a_t) - V(s_t) \tag{8}$$

where $r_t(\theta)$ represents the ratio between the probability of action a_t given state s_t under the different policies (new and old one). $\hat{A}_t$ denotes an estimate of the advantage, calculated using generalized advantage estimation, which can be calculated by Eq.(8). ϵ represents the hyper-

parameter usually set as 0.2. Here, the expectation $\hat{E}_t$ indicates the empirical average over a finite batch of samples, in an algorithm that alternates between sampling and optimization.

3. EXPERIMENTS

3.1 Experimental environment

The experimental environment was created using PyBullet [22], a real-time physics engine suitable for simulating robotic tasks. To be more specific, The Kuka-Diverse-Object-Env contains 8 different functions, which is designed for reinforcement learning experiments in object manipulation (see Fig. 3). It allows the robot to interact with randomly placed objects in both discrete and continuous action spaces. The environment includes customizable parameters for object placement, camera angles, and collision detection. Each episode terminates when the robot either attempts a grasp or exceeds the maximum step count.

There is a dataset including 900 various objects in Pybullet, which is used as training dataset (see Fig. 3). Each grasping system randomly places 5 different objects in a box for a 7-axis Kuka robot arm to grasp. 3D models of typical construction tools (e.g., hammer, drill, scaffolding) were loaded into the environment. In this study, we built a small dataset of construction tools with around 20. These models firstly were converted into URDF files, which allowed the definition of physical properties such as mass, friction, and dimensions. Furthermore, in order to characterize the common properties of the construction tools, they were divided into six shape-based categories (see Fig. 4). Then, we select one object in each type as the test object, as shown in Fig. 4.

Table 1 Different types of construction tools

Shape-based category	Construction tools dataset
Rod Shape	Hammer, Wrench, Crowbar
Cylindrical Shape	Flashlight, Paint Roller, Pipe
Flat Shape	Brush, Scraper, Scissors
Clamp Shape	Clamp, Pliers, Needlenose Pliers
Slender Shape	Scaffolding pin, Rebar, Screwdriver
Drill Shape	Hammer, Wrench, Crowba

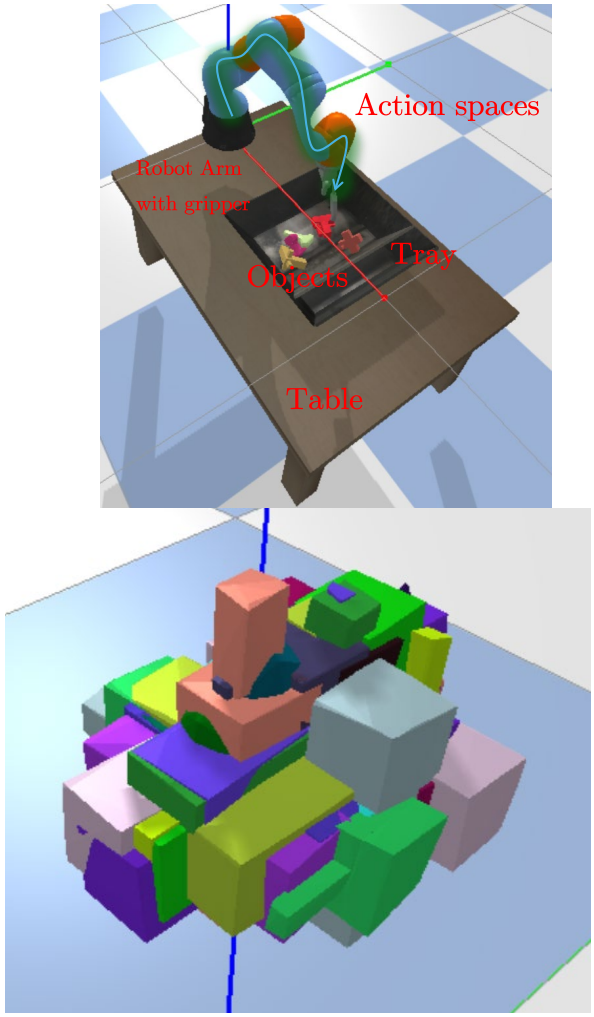


Fig. 3 Experimental environment (Left), Random object dataset (Right)

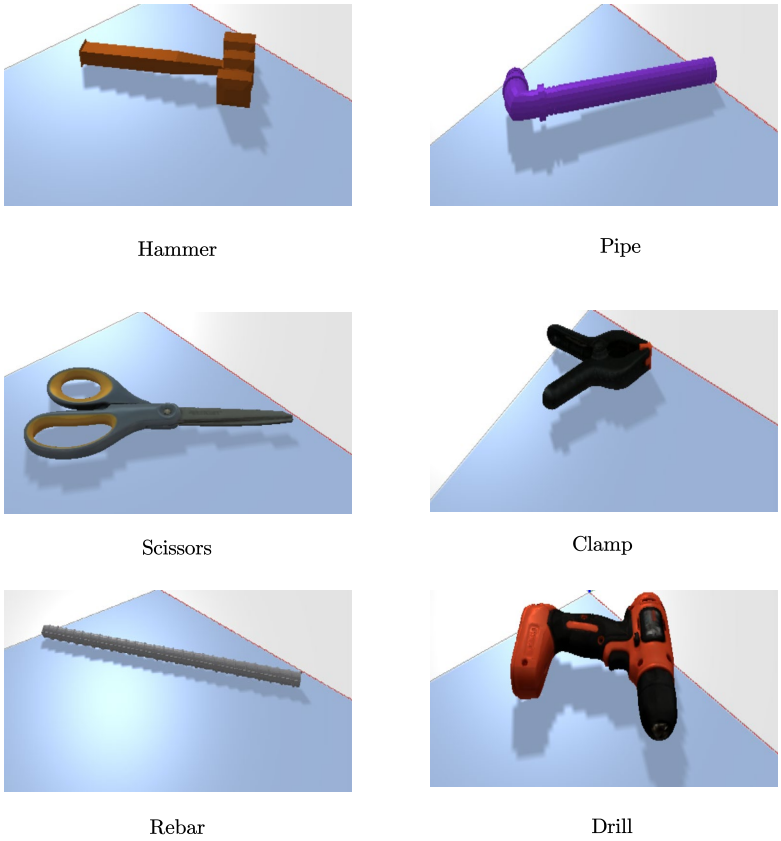


Fig. 4 Selected 6 types of test objects

3.2 Algorithm setting

The algorithm is based on Python 3.10 and Torch 2.4.1 and runs on Ubuntu 22.04 Linux system. Challenges with reward functions include sparse reward functions [12], which provide a reward only when a subtask is completed. While easy to define, they often result in limited guidance for policy updates, requiring a large amount of interaction data and leading to numerous failures. Continuous reward functions offer feedback for every action. This model adopts the sparse reward to decrease exploring time. When grasping an object in one episode, the r_t is 1, otherwise it is 0. Table 2 shows the hyper-parameters used by PPO. The training took place in the PyBullet simulation, where a robotic arm was assigned the task of grasping various objects. Each season consisted of approximately 2,800 steps, corresponding to about 140 episodes, with each episode having a maximum of 20 steps. At the beginning of each episode, the robotic arm was randomly positioned, and the tools were randomly placed within the workspace. The agent learned to adjust its movements and grip based on feedback from the environment. In the resulting graph, the score represents the average of all episode scores within a season.

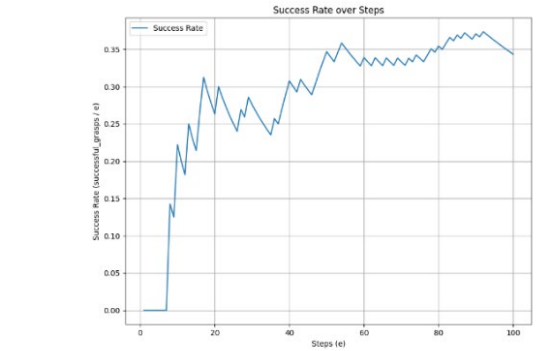
Table 2 Hyper-parameters used for PPO

Parameters	Value
Discount factor γ	0.993

Learning rate α	0.0004
Clip factor in PPO ϵ	0.07
Discount factor in GAE λ	0.95
Trajectory length N	2800
Training epochs	10
Batch size	128

3. RESULTS

The RL model was initially trained on a set of 900 randomly shaped objects, none of which were construction tools, over a short training period of approximately 12 hours. Following this, the model was tested in 100 trials to grasp unseen construction tools in order to evaluate its generalization capabilities. Here, if the construction tools can be lifted the position 0.2 m higher than the table, we assume the case is a successful grasping. The success ratios for each category of tools were assessed to determine the model's performance across these diverse types. The corresponding results for each category are shown in Figures 5–10.



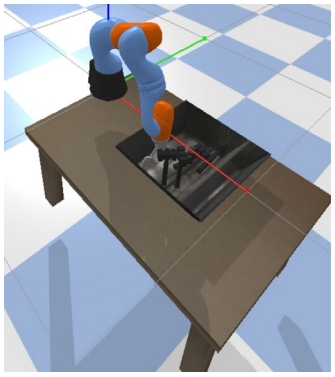
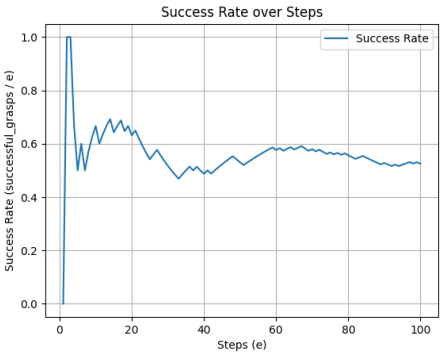


Fig. 5 The performance of RL model for hammer.



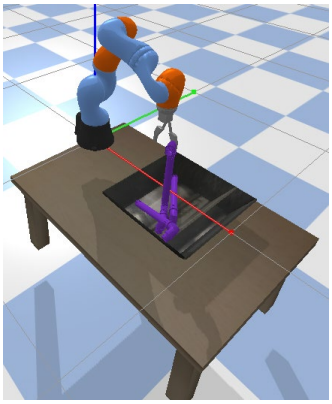
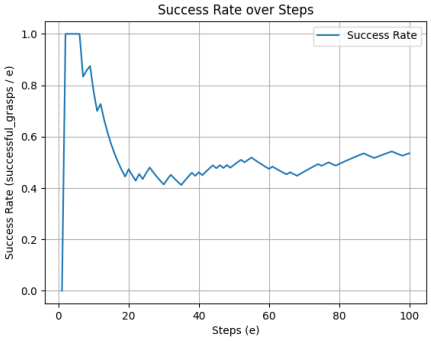


Fig. 6 The performance of RL model for pipe.



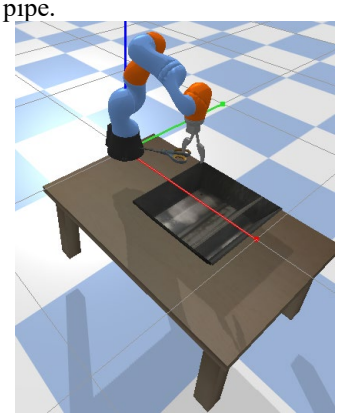
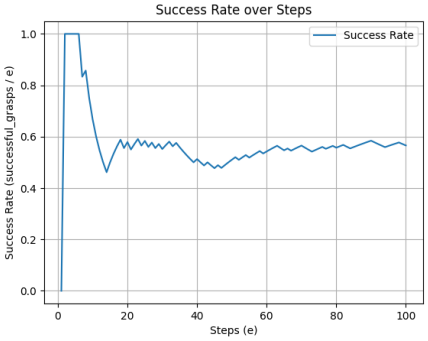


Fig. 7 The performance of RL model for scissors.



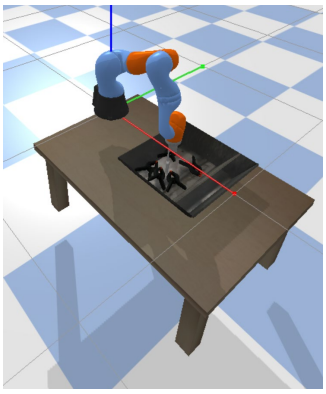
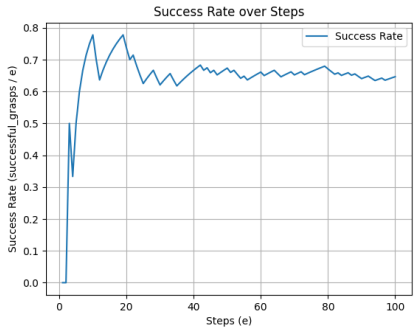


Fig. 8 The performance of RL model for clamp.



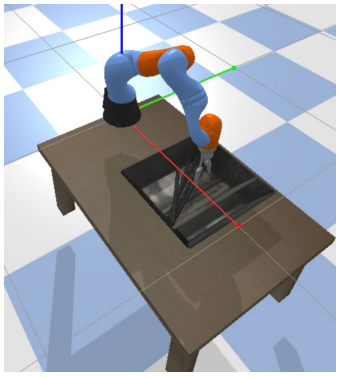
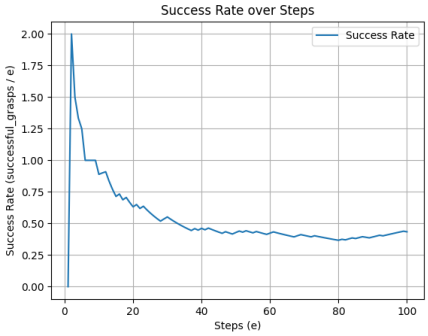


Fig. 9 The performance of RL model for rebar.



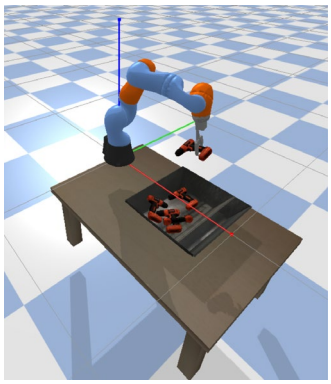


Fig. 10 The performance of RL model for drill.

Overall, as the number of test trials increased, the success ratios tended to converge towards specific values, which can be used to characterize the performance of the trained model. Notably, the model demonstrated the highest success rate for rebar grasping, achieving around a 65% success ratio, followed by pipe grasping as the second highest. The relatively high success rate for these two types of tools can likely be attributed to their cylindrical shape, which aligns well with the two-fingered gripper and lacks protruding features that might obstruct the grasping process. Although these construction tools are irregular, the gripper can exert sufficient frictional force to secure a grasp.

In contrast, the success rates for hammer and drill grasping were around 35%, lower than those for scissors (approximately 57%) and clamp (55%). The primary reason for the lower performance in grasping hammers and drills is the limited number of available grasping points for the gripper, as observed in failed trials. The RL model struggled to effectively learn how to grasp tools with fewer accessible grasping points. Nevertheless, these results provide valuable insights into the model's generalization capability and its potential to accurately grasp construction tools based on their shape characteristics.

4. CONCLUSIONS

The results from the RL model's testing across various construction tools highlight both its potential and the challenges involved in grasping irregular objects. While the model demonstrated a reasonable ability to generalize when grasping cylindrical objects like rebar and pipes, achieving success rates of 65% and slightly lower for pipes, it struggled with tools that have fewer or more complex grasping points, such as hammers and drills, which resulted in a lower success rate around

35%. Tools like scissors and clamps showed intermediate performance, with success rates of 57% and 55%, respectively. These findings suggest that the model's performance is closely linked to the geometric properties of the objects, with simple, cylindrical shapes being easier for the gripper to handle. The challenges in grasping tools with more complex shapes highlight the limitations of the current model in handling tools with fewer accessible grasping points or irregular designs.

From the perspective of RL, several promising avenues for improvement and future research can be considered. One potential enhancement involves integrating a self-attention mechanism into the RL framework by adding new layers to the model architecture. This would allow the model to better focus on relevant features during the learning process and improve its ability to handle complex grasping tasks. Additionally, by changing the input from a single camera to four camera images, the model can gain a more comprehensive view of the environment, reducing the limitations imposed by a restricted field of view and improving overall grasping performance. Another promising approach involves leveraging imitation learning to obtain a pretrained model, which can accelerate the training process. By using datasets generated from videos of construction tool grasping tasks and virtual reality (VR) environments, the model could learn from human demonstrations in tasks where real-world data is scarce or unavailable. This combination of self-attention mechanisms, multi-camera inputs, and imitation learning could significantly enhance the model's generalization ability and robustness, paving the way for more reliable and efficient robotic systems in real-world construction environments.

ACKNOWLEDGMENTS

This work was supported by the Collaborative Research Fund (C6044-23GF) supported by University Grants Committee of The Government of the Hong Kong Special Administrative Region, 30 for 30 Scheme (3030007) funded by The Hong Kong University of Science and Technology. The opinions presented in this paper are those of the authors and do not necessarily reflect the views of the sponsoring organizations.

REFERENCES

[1] revisecond, New Report Shows Construction Workforce Shortage Will Top Half a Million in 2023, Condustrial (2023). https://condustrial.com/2023/03/02/new-report-shows-construction-workforce-shortage-will-top-half-a-million-in-2023/ (accessed October 5, 2024).
[2] R.E. Chapman, D.T. Butry, A.L. Huang, Measuring and Improving U.S. Construction Productivity, (n.d.).
[3] Z. Pan, Y. Yu, Learning multi-granular worker intentions from incomplete visual observations for worker-robot collaboration in construction, Autom. Constr. 158 (2024) 105184. https://doi.org/10.1016/j.autcon.2023.105184.
[4] G. Biswas, A. Bhattacharya, R. Bhattacharya, Occupational health status of construction workers: A review, Int. J. Med. Sci. Public Health 6 (2017) 1. https://doi.org/10.5455/ijmsph.2017.0745302112016.
[5] B. Ter Haar, J. Kruger, G. van Zijl, Off-site construction with 3D concrete printing, Autom. Constr. 152 (2023) 104906. https://doi.org/10.1016/j.autcon.2023.104906.
[6] N. Melenbrink, J. Werfel, A. Menges, On-site autonomous construction robots: Towards unsupervised building, Autom. Constr. 119 (2020) 103312. https://doi.org/10.1016/j.autcon.2020.103312.
[7] E. Gambao, C. Balaguer, F. Gebhart, Robot assembly system for computer-integrated construction, Autom. Constr. 9 (2000) 479–487. https://doi.org/10.1016/S0926-5805(00)00059-5.
[8] L.E. Bernold, Control schemes for tele-robotic pipe installation, Autom. Constr. 16 (2007) 518–524. https://doi.org/10.1016/j.autcon.2006.09.002.
[9] B. Xiao, C. Chen, X. Yin, Recent advancements of robotics in construction, Autom. Constr. 144

(2022) 104591. https://doi.org/10.1016/j.autcon.2022.104591.
[10] H. Wu, H. Li, X. Fang, X. Luo, A survey on teaching workplace skills to construction robots, Expert Syst. Appl. 205 (2022) 117658. https://doi.org/10.1016/j.eswa.2022.117658.
[11] R.S. Sutton, A. Barto, Reinforcement learning: an introduction, Second edition, The MIT Press, Cambridge, Massachusetts London, England, 2020.
[12] Grasping in the Wild: Learning 6DoF Closed-Loop Grasping From Low-Cost Demonstrations | IEEE Journals & Magazine | IEEE Xplore, (n.d.). https://ieeexplore.ieee.org/document/9126187 (accessed October 5, 2024).
[13] L. Huang, Z. Zhu, Z. Zou, To imitate or not to imitate: Boosting reinforcement learning-based construction robotic control for long-horizon tasks using virtual demonstrations, Autom. Constr. 146 (2023) 104691. https://doi.org/10.1016/j.autcon.2022.104691.
[14] B. Belousov, B. Wibranek, J. Schneider, T. Schneider, G. Chalvatzaki, J. Peters, O. Tessmann, Robotic architectural assembly with tactile skills: Simulation and optimization, Autom. Constr. 133 (2022) 104006. https://doi.org/10.1016/j.autcon.2021.104006.
[15] J. Cai, A. Du, X. Liang, S. Li, Prediction-Based Path Planning for Safe and Efficient Human–Robot Collaboration in Construction via Deep Reinforcement Learning, J. Comput. Civ. Eng. 37 (2023) 04022046. https://doi.org/10.1061/(ASCE)CP.1943-5487.0001056.
[16] Reinforcement Learning in Construction Engineering and Management: A Review | Journal of Construction Engineering and Management | Vol 148, No 11, (n.d.). https://ascelibrary.org/doi/10.1061/%28ASCE%29CO.1943-7862.0002386 (accessed October 5, 2024).
[17] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, S. Levine, QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation, (2018). https://doi.org/10.48550/arXiv.1806.10293.
[18] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, S. Levine, Deep Reinforcement Learning for Vision-Based Robotic Grasping: A Simulated Comparative Evaluation of Off-Policy Methods, arXiv.Org (2018). https://arxiv.org/abs/1802.10264v2 (accessed September 30, 2024).
[19] X. Yu, A. Chen, H. Chang, Peridynamic differential operator-based nonlocal numerical paradigm for a class of nonlinear differential equations, Comput. Part. Mech. 10 (2023) 1383–1395. https://doi.org/10.1007/s40571-023-00568-z.
[20] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal Policy Optimization Algorithms, (n.d.).
[21] J. Schulman, S. Levine, P. Moritz, M. Jordan, P. Abbeel, Trust Region Policy Optimization, (n.d.).
[22] Bullet Real-Time Physics Simulation | Home of Bullet and PyBullet: physics simulation for games, visual effects, robotics and reinforcement learning., (2022). https://pybullet.org/wordpress/ (accessed October 4, 2024).