Deep Reinforcement Learning-based Construction Robots Collaboration for Sequential Tasks

Lei Huang[1] and Zhengbo Zou[1]

***Abstract*— The integration of robots into the construction industry shows promise in addressing challenges such as stagnant productivity and low efficiency. Recently, an increasing amount of research develops construction robots based on reinforcement learning (RL). However, most existing RL-based construction robots are trained to conduct specific tasks individually without cooperation. This paper proposes an approach that utilizes two RL-based construction robots (an unmanned ground vehicle and a robot arm) to collaboratively finish the task of window panel transport and installation in sequence without human intervention. Our experiment results show that the two construction robots can successfully collaborate to finish all tasks in an end-to-end manner after they are trained separately with a success rate of 79.6%.**

I. INTRODUCTION

The construction industry has reached an output of over $10 trillion dollars worldwide by the end of 2020 and is expected to continue flourishing [1]. However, the construction industry is prone to poor productivity and low efficiency due to skilled labor shortages and labor-intensive tasks [2]. Moreover, the construction industry has been struggling to provide workers with safe working conditions, as it has the highest rate of fatal accidents accounting for nearly 20% of occupational deaths in the U.S. [3]. The solution of utilizing construction robots to tackle these challenges was first proposed as early as the 1980s [4]. By having robots conduct treacherous and arduous construction tasks, the workers' responsibility shifts from operation to supervision [5], which also reduces the possibility of workers being exposed to dangerous situations.

With the rapid development of reinforcement learning (RL) for generating optimal control policies without handcrafted designs [6], RL-based construction robots have recently drawn researchers' attention [5]. Existing works focus on training a single construction robot to conduct each task individually, such as installing a ceiling panel [7], assembling a lap joint [8], and placing a wood building block [9]. However, similar to multiple workers are responsible for different tasks as a team on-site, construction robots should also have the capability of working collaboratively. For example, before having a robot arm install a window panel, we could first have an unmanned ground vehicle (UGV) transport the window panel to a location within the robot arm's working reach.

As a first attempt to explore multiple construction robots conducting tasks collaboratively, we propose an approach that utilizes a UGV and a robot arm to conduct the task of window panel transportation and installation in collaboration after they are separately trained using RL. During the entire inference process, the UGV and the robot arm deliver the tasks in an end-to-end manner without any human intervention such as handcrafted adjustment or manipulation.

II. RELATED WORK

We can roughly divide construction robots into pre-programmed robots and RL-based robots according to how the control policies are produced [5].

Pre-programmed construction robots conduct tasks following a pre-designed control policy. Specialists design the control policy in detail outlining step-by-step instructions for construction robots. Pre-programmed construction robots are mature enough for practical deployment and have achieved success on-site [10]. For example, [11] used pre-programmed construction robots to bolt steel structures; [12] provided trajectories for robot arms to assemble timber frames. However, the control policies of pre-programmed robots are usually deterministic; hence only adaptable to specific working scenarios that were considered during policy design. Consequently, pre-programmed robots cannot generalize to dynamic environments, and tend to fail once the working condition is changed.

To equip construction robots with adaptability and flexibility, researchers recently start to develop control policies using RL methods [13]. RL-based construction robots actively learn control policies by interacting with environments [14]. During repeated interactions, robots receive different rewards while taking actions under various scenarios (e.g., positions, poses, and surrounding elements). Their objective is to learn control policies such that the they maximize the expected total reward. For example, if picking up a window panel will give a robot arm the maximum reward, a properly trained RL-based robot arm should be able to complete the task regardless of the location of the panel, as long as the environment is capable of communicating the current states and the rewards for a specific action. Researchers have trained RL-based construction robots to conduct various tasks. [7] trained a robot arm using an RL method that imitated behaviors from video demonstrations [15] to install ceiling panels. [8] trained a robot arm using a variant of the Deep Deterministic Policy Gradient algorithm (DDPG)

*We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), [funding reference number: ALLRP 570442-2021]

[1]Lei Huang and Zhengbo Zou are with the Department of Civil Engineering, Faculty of Applied Science, University of British Columbia, 6250 Applied Science Lane, Vancouver, BC V6T 1Z4, CANADA lei.huang@ubc.ca, zhengbo@civil.ubc.ca

[16] to assemble lap joints for timber frames in simulation and then migrated the control policy to a real robot arm. [9] trained a robot arm using Twin Delayed DDPG [17] to place a building block for assembly. However, these RL-based construction robots [7]–[9] were trained to conduct tasks alone without collaborating or communicating with other robots, which can be problematic when we incorporate multiple robots to achieve a higher level of automation.

To enable construction robots to work collaboratively, we propose a novel RL-based approach that allows a UGV and a robot arm to conduct a sequence of tasks consisting of window panel transportation (by UGV) and installation (by robot arm) in an end-to-end manner without external instructions. Due to the extensive time and resource requirements when training using real robots, we train our robots in simulation following the norm of the existing RL-based construction robots [7]–[9].

III. METHODOLOGY

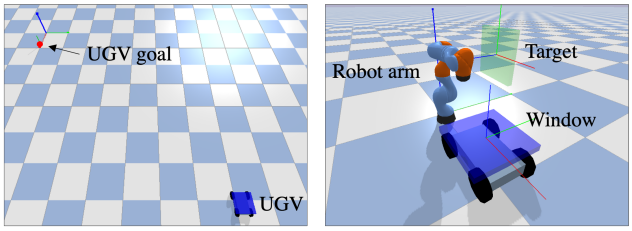
Our approach trains a UGV agent for window panel transportation and a robot arm agent for window installation in two separate RL environments using the proximal policy optimization (PPO) algorithm [18]. We then test these two robots to conduct tasks sequentially in collaboration using control policy inference in a joint environment that contains both the UGV and the robot arm.

A. Building Virtual Environments

The first step is to build two different environments for the UGV and the robot arm in Pybullet [19], which is a physics simulation engine widely used in RL community. As shown in Fig.1(a), the UGV's environment contains a simplified UGV, a blue transparent window panel on the UGV, and a goal for UGV. The starting point of the UGV is randomly and uniformly distributed in an area away from the goal point, while the goal point is fixed. The observation space has eight dimensions including the position, orientation, and velocity of the UGV, and the position of the goal, all of which are in the X-Y plane. The control policy generates two-dimensional continuous actions to control the steering angle of the front wheels and the driving speed of the four wheels. The objective is for the UGV to navigate towards the goal point such that the mass center of the window panel is as close to the goal point's coordinates as possible. For the navigation task, we are only concerned about if it arrives within the robot arm's reach. Thus, we make the environment setting and the reward function concise and straightforward. The reward function for training the UGV agent is as follows:

$$R_t^{UGV} = \begin{cases} 2, & \text{reaching goal} \\ \frac{\Delta d}{C}, & \Delta d > 0 \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where R_{UGV} represents the immediate reward, Δd represents the change of distances from the window panel to the goal point between the current timestep and the previous timestep, and C is a scaling factor. Essentially, in the transportation process, a scaled positive reward would be returned only if the window panel was getting closer to the goal; otherwise, no reward (zero) would be returned.



(a) Environment for UGV training (b) Environment for robot arm training

Fig. 1. Environments for training construction robots

As shown in Fig.1(b), the robot arm's environment contains a robot arm, a window panel on the UGV, and a green transparent cuboid marking the target where the window is expected to be installed. We use the seven-axis KUKA LBR iiwa robot as the robot arm due to its wide application for RL tasks such as object pick and place and human-robot collaboration [20]. Following general poses on real robot arms, the robot arm is initialized such that the rotations of the fourth joint and the sixth joint are 1.57 radians and -1.57 radians, respectively, and the rest of the joint rotations are all zeros. To ensure the robot arm can conduct the tasks of pickup and installation, the window panel and the target should be both within the range of the robot arm. Considering that the UGV trained in Fig.1(a) might reach the destination with small offsets, we randomize the initial location of the window panel so that the robot arm learns to pick up window panels considering uncertainties of the initial location distributed in an area. The observation space for the RL agent has 21 dimensions, including seven-dimensional joint rotations of the robot arm, seven-dimensional position and orientation of the window, and seven-dimensional position and orientation of the target. The control policy generates seven-dimensional continuous actions to control the rotation increments of the seven joints. The objective of this task is for the robot arm to first pick up the window panel and then move it towards the target opening. We use the distance between the mass center of the window panel and the target for measuring successful installations. The reward function for training the robot arm is as follows:

$$R_t^{Arm} = \begin{cases} 1, & \text{pick} \\ 2, & \text{install} \\ -1, & \text{collision} \\ -\frac{1}{3000}, & \text{otherwise,} \end{cases} \tag{2}$$

where R_{arm} represents the immediate reward. The reward function is designed this way so that the cumulative reward in each trial (a maximum of 3000 timesteps) is always bounded between -2 and 3.

B. Training Construction Robots

After building the virtual environments, we train the UGV for navigation and the robot arm for window panel pickup and installation using policy gradient algorithms, which are

Algorithm 1 Vanilla Policy Gradient Algorithm

Initialize policy and value function parameters θ_0, ϕ_0.
for $k = 0, 1, 2, ...$ **do**
  Collect $\mathcal{D}_k = \{\tau_i\}$ by running $\pi_k = \pi(\theta_k)$.
  Compute rewards-to-go $\hat{R}_t$.
  Compute advantage estimates $\hat{A}_t$ based on V_{ϕ_k}.
  Estimate policy gradient as

$$\hat{g}_k = \frac{1}{|\mathcal{D}_k|} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}_t.$$

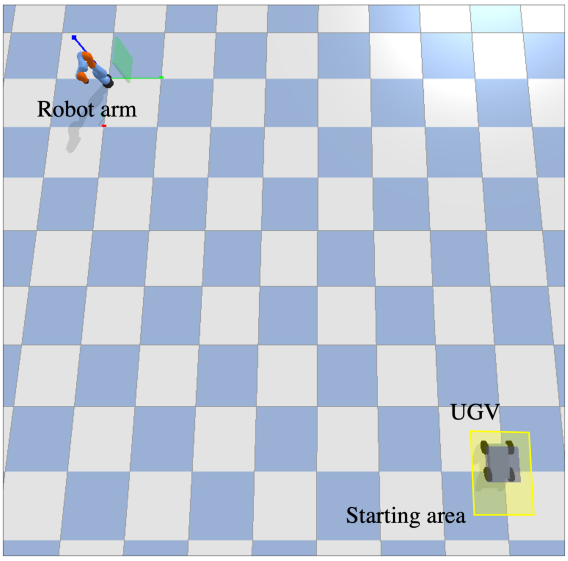
  Compute policy update, $\theta_{k+1} = \theta_k + \alpha_k \hat{g}_k$.

$$\theta_{k+1} = \theta_k + \alpha_k \hat{g}_k.$$

  Fit value function:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} (V_\phi(s_t) - \hat{R}_t)^2.$$

end for



Environment for UGV and robot arm collaboration

Fig. 2. The environment for testing robot collaboration

suitable for continuous control of RL agents deployed on robots [14]. Policy gradient methods use parameterized functions such as neural networks to model control policies π. The policy functions are optimized using gradient ascent so as to generate actions (a) that maximize cumulative rewards for agents.

Algorithm 1 shows the process of how a general policy gradient algorithm works in detail. First, we collect a set of trajectories $\mathcal{D}_k$ by having the agent interact with the environment using the current control policy π_k. For each trajectory, we compute the rewards-to-go. We then compute the advantage estimations based on the current value function V_{ϕ_k}. After calculating the policy gradients based on the objective function, we update the control policy using gradient ascent methods such as Adam [21]. Lastly, we fit the value function by regression on mean-squared error via gradient descent. This process is repeated until convergence.

The vanilla policy gradient algorithm's performance is susceptible to collapse since it does not constrain how much the new control policy deviates from the previous policy. To avoid this limitation, we specifically adopt PPO-Clip [18], which constrains the gradients. Meanwhile, compared to other policy gradient algorithms that tackle the gradient issue such as trust region policy optimization (TRPO) [22], PPO is more widely used for its easier implementation and tuning. PPO is similar to the vanilla policy gradient algorithm except for the objective function. PPO's objective function [18] is as follows:

$$L = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a), g(\epsilon, A^{\pi_{\theta_k}}(s,a))\right), \quad (3)$$

where

$$g(\epsilon, A) = \begin{cases} (1+\epsilon)A & A \geq 0 \\ (1-\epsilon)A & A < 0 \end{cases}.$$

Instead of using KL-divergence or other constraint terms, PPO-Clip achieves the constraint by clipping in the objective function.

C. Collaborating Using Inference

After training the UGV and the robot arm in their respective environments using PPO-Clip, we build a third environment (the collaboration environment, as shown in Fig.2) for robot collaboration as testing. The collaboration environment accommodates the trained UGV and robot arm, the window panel, and the non collidable marker used as the robot arm's target. Without any human intervention, a successful collaboration between the two robots would require that the UGV has to navigate to the area around the goal point first; only then will the robot arm start operating for window panel pickup and installation.

IV. EXPERIMENTS AND RESULTS

To validate our approach, we tested the trained UGV and the robot arm in the collaboration environment by inference from their learned control policies. In the experiment, we randomized the starting point of the UGV in the yellow area as shown in Fig.2. We conducted experiments repeatedly for 10 times, each of which we ran 100 trials and counted the number of successful attempts to arrive, pick, and install.

TABLE I presents the experiment result. Out of 1000 trials, the success rate of the UGV arriving in the designated area is 97.5%; the success rate of the robot arm picking up the window panel is 94.8%, and the success rate of the robot arm installing the window panel is 79.6%. Given that the UGV has arrived in the area, the success rate of the robot arm picking up the window panel is 97.2%; and the success rate of the robot arm installing the window panel is 81.6%. Given that the UGV has arrived, and the robot arm has picked up the window panel, the success rate of the robot arm installing the window panel is 84.0%.

TABLE I

NUMBERS OF SUCCESSFUL ARRIVALS, PICKUPS, AND INSTALLATIONS.

Group Index	Trial	Arrive	Pick	Install
1	100	98	93	78
2	100	97	95	79
3	100	98	95	79
4	100	96	92	80
5	100	97	95	82
6	100	97	94	75
7	100	100	97	79
8	100	98	96	85
9	100	97	95	80
10	100	97	96	79
Total	1000	975	948	796

V. CONCLUSIONS AND DISCUSSIONS

In this paper, we proposed an approach aiming to have a UGV and a robot arm conduct a sequence of tasks (i.e., window transportation and installation) in collaboration after training two RL agents separately in two different environment settings. Results showed that the trained robots could successfully finish the sequence of tasks with considerable success rates. In the construction industry, most tasks are carried out in sequence and are usually connected to each other. Studying how construction robots can partition the tasks and work collaboratively is crucial to achieving a higher level of automation in construction. Our approach makes a first attempt to have multiple robots trained using RL to carry out construction tasks in collaboration. For future work, we will explore simultaneously training multiple construction robots for working in collaboration and migrate the control policies to real construction robots.

REFERENCES

[1] G. Robinson, J. Leonard, and T. Whittington, "Oxford economics: Future of construction a global forecase for construction to 2030," sep 2021.

[2] H. Karimi, T. R. Taylor, G. B. Dadi, P. M. Goodrum, and C. Srinivasan, "Impact of skilled labor availability on construction project cost performance," *Journal of Construction Engineering and Management*, vol. 144, no. 7, p. 04018057, 2018.

[3] OSHA, "Occupational safety and health administration, united states department of labor: Commonly used statistics," 2019.

[4] C. Haas, M. Skibniewski, and E. Budny, "Robotics in civil engineering," *Computer-Aided Civil and Infrastructure Engineering*, vol. 10, no. 5, pp. 371–381, 1995.

[5] C.-J. Liang, X. Wang, V. R. Kamat, and C. C. Menassa, "Human–robot collaboration in construction: Classification and research trends," *Journal of Construction Engineering and Management*, vol. 147, no. 10, p. 03121006, 2021.

[6] B. Eysenbach, R. R. Salakhutdinov, and S. Levine, "Robust predictable control," in *Advances in Neural Information Processing Systems*, 2021.

[7] C.-J. Liang, V. R. Kamat, and C. C. Menassa, "Teaching robots to perform quasi-repetitive construction tasks through human demonstration," *Automation in Construction*, vol. 120, p. 103370, 2020.

[8] A. A. Apolinarska, M. Pacher, H. Li, N. Cote, R. Pastrana, F. Gramazio, and M. Kohler, "Robotic assembly of timber joints using reinforcement learning," *Automation in Construction*, vol. 125, p. 103569, 2021.

[9] B. Belousov, B. Wibranek, J. Schneider, T. Schneider, G. Chalvatzaki, J. Peters, and O. Tessmann, "Robotic architectural assembly with tactile skills: Simulation and optimization," *Automation in Construction*, vol. 133, p. 104006, 2022.

[10] T. Salmi, J. M. Ahola, T. Heikkilä, P. Kilpeläinen, and T. Malm, "Human-robot collaboration and sensor-based robots in industrial applications and construction," in *Robotic building*. Springer, 2018, pp. 25–52.

[11] B. Chu, K. Jung, M.-T. Lim, and D. Hong, "Robot-based construction automation: An application to steel beam assembly (part i)," *Automation in construction*, vol. 32, pp. 46–61, 2013.

[12] K. Iturralde and T. Bock, "Integrated, automated and robotic process for building upgrading with prefabricated modules," in *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 35. IAARC Publications, 2018, pp. 1–8.

[13] X. Xu and G. B. De Soto, "On-site autonomous construction robots: A review of research areas, technologies, and suggestions for advancement," in *37th International Symposium on Automation and Robotics in Construction: From Demonstration to Practical Use-To New Stage of Construction Robot, ISARC 2020*. International Association on Automation and Robotics in Construction (IAARC), 2020, pp. 385–392.

[14] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. MIT press, 2018.

[15] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, "Imitation from observation: Learning to imitate behaviors from raw video via context translation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1118–1125.

[16] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. van Hasselt, and D. Silver, "Distributed prioritized experience replay," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=H1Dy—0Z

[17] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[19] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.

[20] O. Kyjanek, B. Al Bahar, L. Vasey, B. Wannemacher, and A. Menges, "Implementation of an augmented reality ar workflow for human robot collaboration in timber prefabrication," in *Proceedings of the 36th international symposium on automation and robotics in construction (ISARC)*. International Association for Automation and Robotics in Construction (IAARC), 2019, pp. 1223–1230.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015. [Online]. Available: https://arxiv.org/abs/1412.6980

[22] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1889–1897. [Online]. Available: https://proceedings.mlr.press/v37/schulman15.html