# CraftEnv: A Flexible Collective Robotic Construction Environment for Multi-Agent Reinforcement Learning

Rui Zhao*
Tencent Robotics X Lab
Shenzhen, China
reyzhao@tencent.com

Xu Liu*
Shanghai Jiao Tong University,
Tencent Robotics X Lab
Shanghai, China
liu_skywalker@sjtu.edu.cn

Yizheng Zhang*
Tencent Robotics X Lab
Shenzhen, China
yizhenzhang@tencent.com

Minghao Li
Tencent Robotics X Lab, Sun Yat-sen University
Shenzhen, China
limh83@mail2.sysu.edu.cn

Cheng Zhou
Tencent Robotics X Lab
Shenzhen, China
mikechzhou@tencent.com

Shuai Li
Shanghai Jiao Tong University
Shanghai, China
shuaili8@sjtu.edu.cn

Lei Han
Tencent Robotics X Lab
Shenzhen, China
lxhan@tencent.com

## ABSTRACT

CraftEnv is a flexible Collective Robotic Construction (CRC) environment for Multi-Agent Reinforcement Learning (MARL) research. CraftEnv can be used to study how artificial intelligent agents may learn to cooperate and solve complex real world tasks, such as collective construction and intelligent warehousing. The environment contains a set of collective construction tasks, which require a group of robotic vehicles to cooperate and learn to build different constructions efficiently. There are different elements in the CraftEnv, such as smartcars, blocks, and slopes. The smartcars can use the blocks and slopes to build different structures. The CraftEnv is highly flexible and simple to use, which enables creative and quick task-designs. The environment is written in python and can be rendered using PyBullet. The simulation is built based on real world robotic systems, designed with real-world constraints in mind. The learned policy can be transferred to the real world robotic system. CraftEnv is tailored for effective use by the research community and pushing forward collective intelligence and swarm technology.

## KEYWORDS

Multi-Agent Reinforcement Learning; Collective Intelligence

**ACM Reference Format:**
Rui Zhao*, Xu Liu*, Yizheng Zhang*, Minghao Li, Cheng Zhou, Shuai Li, and Lei Han. 2023. CraftEnv: A Flexible Collective Robotic Construction Environment for Multi-Agent Reinforcement Learning. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

## 1 INTRODUCTION

In the research of evolutionary robotics, Collective Robotic Construction (CRC) is one of the biggest applications in industry worldwide [9], due to the considerable productivity and sustainability challenges in the industry field. Considering that most construction robots are not fully automated and often require guidance and instructions from the operators, Multi-Agent Reinforcement Learning (MARL), as a possible solution, has become one of the most popular methods for CRC systems [18][1][20]. Casting CRC tasks into the MARL framework is a very difficult game with sparse and delayed rewards, as robots often need to build scaffolding to reach the higher levels of the structure to complete the construction task. Despise the high difficulty, with proper design of the simulator, MARL algorithms could help the construction robots to establish a learning process based on the feedback from the construction site and lead to a near-optimal policy to realize the goal [26].

However, compared with other application fields of MARL, there is no comprehensive evaluation environment in CRC tasks, which greatly limits the evaluation and development of MARL research in CRC. Usually the evaluation of MARL algorithms are focusing on the game environments or simulator of simple tasks, such as SMAC [16], MPE [13] and RWARE [14]. Currently, in the field of CRC, however, the evaluation of MARL methods mainly focus on some over-simplified tasks, such as constructing some goal structure with only blocks [18], or only considers planar construction, where agents are encouraged to move the points into some projected scalar field with given shape [20]. Such a design is not only too simplified to fully consider a large number of physical constraints in real scenarios, but also difficult to deploy in practical applications. In this way, even though MARL has made great progress in the field of swarm intelligence [2], there is a clear gap to apply it to the field of CRC.

Therefore, in order to further promote the application of MARL in CRC environment and also to further promote the practice of

*Equal contribution.

*Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.*

MARL in some complex real-world application scenarios such as collective construction and intelligent warehousing, we introduce CraftEnv, the first comprehensive CRC environment in the MARL domain, and compare 6 MARL algorithms in a diverse set of cooperative multi-agent tasks, including independent learning algorithms [23], centralized multi-agent policy gradient algorithms [7][27] and value decomposition algorithms [22][15]. The algorithms are evaluated in four goal-conditioned building tasks, a free building task, and a breaking barrier task. These tasks are designed as comprehensive modelings for real-world scenarios such as collective construction, smart warehousing. Besides, to further demonstrate the challenges that the flexible environmental design of CraftEnv can bring to various MARL algorithms, and to further test its deployment ability on physical machines, we configured CraftEnv with physical machines and successfully deployed the CraftEnv trained model on the real robots. Besides, we also trained high-complexity tasks of CraftEnv on the cluster with large-scale distributed training [6, 8, 21], which shows that the high flexibility of CraftEnv brings creativity and new possibilities to MARL algorithms. CraftEnv aims to combine the best MARL algorithms with real-world CRC environments, providing inspirations for the future research of MARL from the perspective of application, and promoting the application of reinforcement learning related technology in the field of collective intelligence and swarm technology.

Our main contributions are as follows: (1) We introduce CraftEnv, the first comprehensive MARL CRC environment. CraftEnv is highly flexible and is able to simulate various real-world scenarios, such as collective construction and intelligent warehousing; (2) We design multiple tasks of various difficulties, including goal-conditioned building tasks, free building tasks and breaking barrier tasks. With the comparison of 6 benchmarking algorithms, we provide detailed analysis of their properties from practical perspective; (3) In order to simulate the real application scenarios more accurately, We conduct additional experiments with large-scale distributed training. Besides, physical machines are built for CraftEnv, demonstrating the flexibility of transferring the policy learned by CraftEnv to real-world robotic systems.

# 2 PRELIMINARY

## 2.1 Markov Game

Similar to the setting of single-agent reinforcement learning, MARL also addresses sequential decision-making problems, but with multiple agents involved. Specifically, both the transition of the system state and the reward received by each agent are now affected by the joint action of all agents. In the most general setting, each agent can have its own long-term reward to optimize, making the problem considerably more intractable.

Markov Games (MGs) has been widely used in the literature for developing MARL algorithms. Specifically, a Markov game $\mathcal{G}$ is defined as a tuple $\mathcal{G} = \left(\mathcal{N}, \mathcal{S}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \mathcal{P}, \{\mathcal{R}^i\}_{i\in\mathcal{N}}, \gamma\right)$, in which $\mathcal{N} = \{2, \ldots, N\}$ denotes the set of all agents, $\mathcal{S}$ denotes the finite state space, $\mathcal{A}^i$ denotes the finite action space of agent $i$. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\mathcal{S})$[1] denotes the transition probability from state $s \in \mathcal{S}$ to any state $s' \in \mathcal{S}$ for any joint action $a \in \mathcal{A}$, and denote $R^i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ as the reward function that determines the immediate reward for agent $i$ after a transition from $(s, a)$ to $s'$. $\gamma \in [0, 1)$ is the discount factor.

The interactive process of the agents and the environment is modeled as follows. At each time step $t$, each agent $i \in \mathcal{N}$ choose an action $a_t^i \in \mathcal{A}^i$ from state $s_t$, and the system will then transition to the next state $s_{t+1}$, and the reward of agent $i$ is given by $R^i(s_t, a_t, s_{t+1})$. The ultimate goal of each agent is to optimize its own long-term reward by following the policy $\pi^i : \mathcal{S} \mapsto \Delta\left(\mathcal{A}^i\right)$. Therefore, with the joint policy of all agents $\pi : \mathcal{S} \mapsto \Delta(\mathcal{A})$ defined as $\pi(a|s) := \prod_{i\in\mathcal{N}} \pi^i\left(a^i|s\right)$, we can define the value function for agent $i$:

$$V^i(s) := \mathbb{E}\left[\sum_{t\geq 0} \gamma^t R^i(s_t, a_t, s_{t+1}) \middle| a_t^i \sim \pi^i(\cdot|s_t), s_0 = s\right] \quad (1)$$

and the corresponding Q-function:

$$Q^i(s,a) := \mathbb{E}\left[\sum_{t\geq 0} \gamma^t R^i(s_t, a_t, s_{t+1}) \middle| a_t^i \sim \pi^i(\cdot|s_t), s_0 = s, a_0 = a\right]. \quad (2)$$

The setting of CraftEnv mainly focus on cooperative MARL, where all agents share a common reward function, i.e., $R^1 = R^2 = \cdots = R^N = R$. The model is often referred as multi-agent MDPs (MMDPs) or Markov teams. With this setting in mind, the value functions and Q-functions of each agent are identical, which enables many single-agent RL algorithms to be applied.

## 2.2 Benchmarking Algorithms

Currently, there are three paradigms for MARL: centralized learning, independent learning, and centralized training with decentralized execution (CTDE). Centralized learning treats the whole system as a whole and adopts single-agent reinforcement learning algorithm for training, which solves the problem of non-stationary environment, but cannot solve the problems of no communication, large scale and large action space. Independent learning allows each agent to train its own strategy independently, but it neglects the connection between multiple agents, which sometimes aggravates the learning instability. By contrast, CTDE can not only improve the learning efficiency, but also allow each agent to make independent decisions, which solves the problem of multi-agent learning to a certain extent. However, as the number of agents increases, the solution of the optimal joint value function may become more complicated. Therefore, among the popular MARL algorithms, we choose IQL as a representative of the independent learning MARL algorithms, and COMA [7], VDN [22], QMIX [15], QTRAN [19], MAPPO [27] as the representatives of the CTDE MARL algorithms. The details of our consideration are listed in Appendix F.

# 3 ENVIRONMENT

Now, we specifically elaborate on the structural features of CraftEnv, including the main components of the environment, the specification of the MDP, and the cooperative tasks available based on the environment. The structure of CraftEnv is shown in Figure 1. The MatrixEnv in CraftEnv provides the basic elements and multiple interactive interfaces, such as task specification, Gym-style

[1] Let $\mathcal{A} := \mathcal{A}^1 \times \cdots \mathcal{A}^N$ for clarity.

RL and rendering. CraftEnv inherits from MatrixEnv and provides a comprehensive interface for a number of MARL algorithms and the rendering result.

## 3.1 Basic Elements

In order to better simulate the logistics and transportation scenarios in the real world and improve the flexibility of the environment, CraftEnv sets the environment itself as a $m \times n \times z$ map, which serves as the working environment for agents. Considering the map as a storage space or a building site, blocks and slopes are designed as basic elements, where blocks can be thought as the packages for transportation, and the combination of blocks and slopes can be viewed as basic units for the construction of buildings. Similar to the game setting of the Minecraft game, agents are free to manipulate these components, including picking up, moving and placing them. Besides, the slopes can be folded and unfolded for the agents to construct complex buildings. This flexibility allows agents to explore a variety of ways to cooperate, allowing for more freedom in the design of tasks and further testing the ability of agents to cooperate.

## 3.2 States, Actions and Rewards

*3.2.1 State Space.* In the process of logistics transportation, the current position of each object to be transported is very important to the worker's decision of action. According to the decision of transport location, the worker can choose an object as the target, plan out the path to the object, and move it to the destination. Therefore, it is also critical for the agents in CraftEnv to make decisions on what to carry, where the destination is, how to reach the object and how to carry it to the destination. Since the interactions among agents and the elements of the environment are operated on a 3D map, the position of agents and elements can be represented as coordinates in the map. Therefore, we construct the observation of an agent from:

(1) Self-awareness: the agent's current position in the map. This is consistent with our common sense: the first thing a porter or construction worker needs to know in a work scene is his position, and then he can make a decision;
(2) Position of other agents. In CRC systems, agents are required to cooperate efficiently to complete tasks, but different agents may interfere with each other. For example, during the construction of a building, two agents may want to move a building material to different locations, or two agents may have conflicting paths on their way to move objects. Therefore, it is critical for one agent to be aware of the position of other agents, by which different agents can cooperate with each other to complete harder tasks.

*3.2.2 Action Space.* The actions of an agent in CraftEnv are designed based on the properties of the real-world smartcar models. We have not only built a complete simulation model for smartcars in CraftEnv, but also the corresponding physical machines to support all the available actions available in the simulated environment. Concretely speaking, the available actions designed for smartcars includes:

(1) Moving in horizontal and vertical directions. A total of 4 directions of moving options ensure the freedom of agent movement. The design not only makes it easier for agents to move around flexibly, but also makes it convenient to incorporate various physical constraints when designing action masks and to prohibit dangerous behaviors such as blocking agents from traveling backward uphill;
(2) Interactions with different objects in the environment. The smartcar is designed to possess the ability to interact with objects in the environment in various ways, which are designed to easily simulate the process of cargo handling and building construction in real-life environments. The details about the interactions are introduced in Appendix B.

As we can see, the actions in CraftEnv are designed with discrete settings in mind. This strategy can not only further enhance the stability of the agent training, but also further ensure the flexibility of the environment. CraftEnv provides rich interfaces that make it possible to design richer action spaces beyond the actions mentioned above.

*3.2.3 Reward Setting.* As described above, CraftEnv is a highly flexible MARL environment for CRC systems. Therefore, in order to deploy different simulation tasks such as transportation of packages and construction of buildings, various settings of the specific task is needed for the environment. Therefore, CraftEnv provides an easy-to-use interface for specifying the reward function. As some more concrete examples, here we consider three different kinds of tasks: (1) building with specified shape requirement; (2) building with high complexity; (3) carrying a flag to the goal with breaking barriers. In the first kind of tasks, it is natural to use discrete reward, where some numerical reward is given when the agents success in building part of the blueprint. However, in the second scenario, instead of fixed blueprint, the reward function should encourage the construction of buildings with high complexity, making the function more flexible and requiring customization under different level of complexity. As will be shown later, various reward functions can be designed for different intentions, such as encouraging connecting more blocks together or encouraging constructing higher buildings.

# 4 EXPERIMENT

As a cooperative MARL environment for CRC systems, CraftEnv has an environment design similar to Minecraft and can support rich task designs. Specifically, after designing the components and tasks of the environment elaborately, our primary goal is to test the cooperation ability among agents in this highly flexible environment under different kinds of tasks. Furthermore, as CraftEnv can be conveniently deployed to real-world hardware systems (the results in the deployment step is shown as a video in the supplemental materials), we hope that this new environment can promote the real-world application of MARL and swarm intelligence in scenarios such as CRC systems and smart warehousing. The code is available at https://github.com/Tencent-RoboticsX/CraftEnv.

## 4.1 Task Design

As described before, the task of CraftEnv is constructed in terms of the building scenario and the breaking barrier scenario. Specifically, in the building task, possible bottlenecks of MARL algorithm in

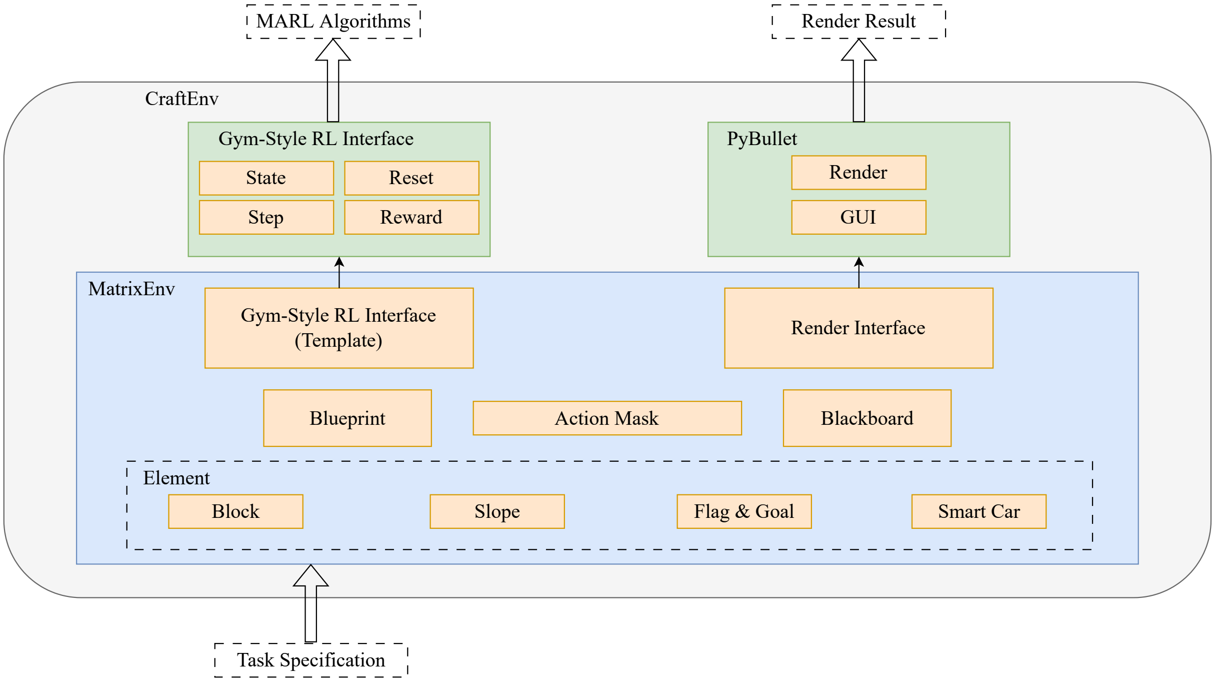

**Figure 1: The Structure of CraftEnv.**

cooperative building will be analyzed by designing various kinds of goal conditioned tasks with different difficulties. The flexibility of CraftEnv ensures the feasibility of the implementation of goal conditioned tasks with free difficulty. In our experiment, we design four kinds of goals with different difficulties – including strip buildings, block buildings and two two-story buildings with different difficulty. With tasks of different difficulty, current SOTA MARL algorithms will show different performances, and specific analysis on this result will be discussed. Besides, the free building tasks without specific blueprint are also designed to encourage the agents to freely explore and construct complex structures.

As a simulation of the obstacles that can arise in CRC tasks, CraftEnv designs tasks for breaking barriers: with the target of carrying the flag to the goal position, CraftEnv supports obstacles of various shapes and difficulties, enabling agents to explore freely, and break down obstacles and complete the goal under cooperation. With such sparse reward, the breaking barrier task not only further improves the difficulty of the simulation environment, but also further stimulates the cooperation ability between agents – otherwise, it will be impossible to complete the transportation task in limited time steps.

#### 4.1.1 *Goal-conditioned Building Tasks.*

In goal conditioned building tasks, we encourage agents to cooperate to achieve the goal building process by specifying the design drawings of the target buildings. It can be seen that the reward in this process is discrete, that is, we can give different rewards for the completion of the construction. In order to test the performance of MARL algorithms at different levels, we designed a variety of experimental tasks, as shown in Figure 2.

In addition, in the goal conditioned building task, we consider sparse reward, that is, we give some rewards based on the completion of some building goals. In order to encourage more complex building processes, we have made some specific settings for the rewards of different components of building. The specific settings are shown in Table 1.

**Table 1: Reward for goal-conditioned building tasks.**

| *Local Task* | *Reward Value* |
|---|---|
| Contribute a first-layer block | 1 |
| Contribute a folded slope | 1 |
| Unfold a slope correctly | 1 |
| Contribute a (simple) second-layer block | 1 |
| Contribute a (complex) second-layer block | 3 |
| Complete the building task | 4 |

#### 4.1.2 *Free Building Tasks.*

Aside from goal conditioned building task, we also choose a more free building task, that is, we do not give a specific building blueprint, but encourage agents to explore more possibilities freely. Specifically, by specifying rewards for different complex architectural forms, CraftEnv can encourage agents to cooperate extensively to build complex buildings. This goal is similar to the attraction of Minecraft itself: encourage players to play their creativity, and use simple basic modules to build buildings with rich shapes.

In addition, this design approach is more accurately in line with the real-world CRC scenario: for a variety of transportation and construction jobs, it is not realistic to fully specify all the details

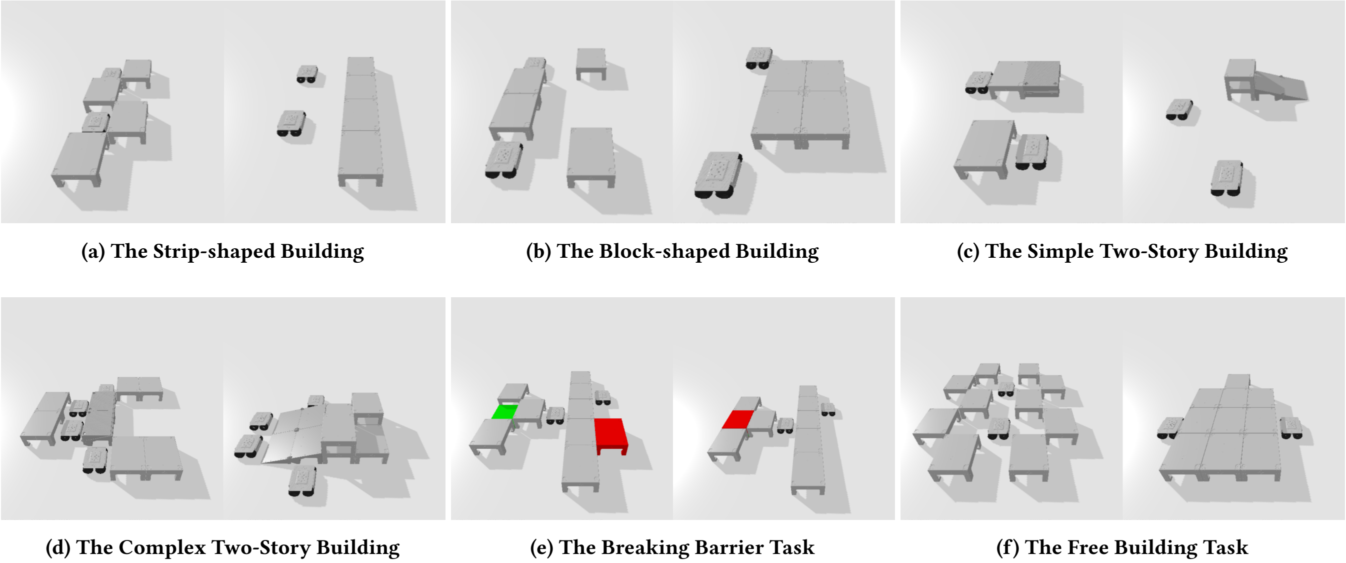
(a) The Strip-shaped Building
(b) The Block-shaped Building
(c) The Simple Two-Story Building
(d) The Complex Two-Story Building
(e) The Breaking Barrier Task
(f) The Free Building Task

**Figure 2: Example of different designs of the buildings tasks. The building difficulty is arranged in increasing order.**

of the target each time. On the other hand, by giving a specific definition of the complexity of the building and training agents, it is not only widely applicable, but also more in line with the requirements of the field of swarm intelligence: agents can autonomously discover the knowledge needed in the environment through collaboration, and achieve the goals through efficient understanding and collaboration.

In this task, the design of the reward function plays the most important role in training, and the complexity evaluation of the building itself takes a variety of forms. For the training on a single machine, we encourage the construction of large-scale platforms. Concretely speaking, we construct the reward function using deep-first graph search [24] for discovering the connected components in the map. Denote the set of all connected components as $C = \{c_1, \ldots, c_n\}$, and $f(c_i) = d_i$ is the number of blocks connected in $c_i$. The reward function is designed as

$$R(C) = \max\{d_i = f(c_i) : c_i \in C\} - 1. \tag{3}$$

Besides, for the training on the cluster with large-scale distributed training, a more complex reward function is designed, where multiple aspects in the construction tasks are considered. The details about the hard tasks are introduced in Appendix C.

*4.1.3 Breaking Barrier Tasks.* As an challenging simulation in the smart warehousing scenarios where the agents may meet unexpected obstacles when interacting with the environment, We design the implementation of the breaking barrier task as the case shown in Figure 2e. The task specified for the agents is to cooperate in carrying the flag to the goal. The wall-shaped barriers on the flag side and embracing-shaped barriers on the goal side are the main obstacles for the task. It is required for the agents to cooperate to break the barriers and find an available path to carry the flag to the goal. However, the skill of clearing the blocks along the way are completely reward-free, thus requiring effective exploration for the agents to achieve the task.

Besides, in the breaking barrier task, since our main goal is to carry the flag to the goal, we choose not to assign explicit reward for removing the barriers, but let the agents explore freely to learn the policy of removing the barriers and reach the goal. Therefore, the setting of reward for the breaking barrier task is designed as:

$$R_t = d(p_{t-1}, p_g) - d(p_t, p_g) + \alpha\mathbb{I}(p_t = p_g) - \beta,$$

where $p_t$ is the position of the flag at time $t$, $p_g$ is the position of the goal, $d$ is a distance metric, and $\mathbb{I}$ is the indicator function for measuring where we have successfully carried the flag to the goal, $\alpha$ is the reward for completing the task, and $\beta$ is the time penalty. In our experiment, we set $\alpha = 10$ and $\beta = 1$.

## 4.2 Computational Requirements

All local experiments presented in this work were executed on one Tesla M40 GPU with 12GB video memory. The main types of CPU models that were used for this work is Intel(R) Xeon(R) Platinum 8255C CPU @ 2.50GHz processor. Some of the benchmarking algorithms are implemented with reference to the PyMARL [17] and ExtendedPyMARL [14]. All the experiments can be executed within 12 hours.

Additionally, in order to exploit the flexibility of CraftEnv, we also design experiment with large-scale distributed training with hundreds of CPUs. As the case of RLLib [10] and TLeague [21], being able to train large models can dramatically improve the performance of the model, making the model capable of solving larger, more difficult problems [5].

## 4.3 Result on Predefined Tasks

Based on the fully-cooperative CraftEnv and the various tasks built on it, here we compare the performance of current benchmarking MARL algorithms, including IQL [23], VDN [22], COMA [7], QMIX [15], QTRAN [19] and MAPPO [27], and analyze the reason behind the experiment result.

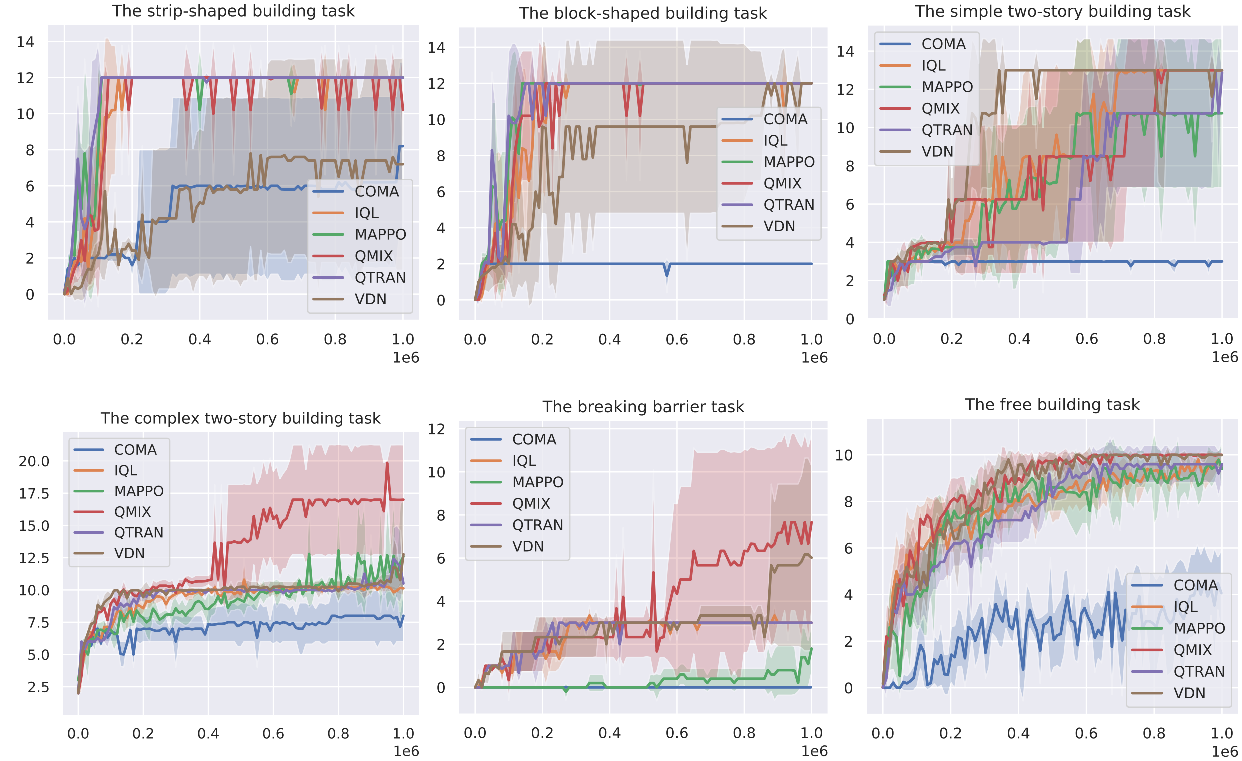

**Figure 3: Averaged return for the benchmarking algorithms under 6 tasks. The X axis represents the time step of the environment, and the Y axis is the averaged return. All the results are averaged under 5 independent runs with random seed.**

*4.3.1 Performance on Goal-Conditioned Building Tasks.* The performance of different MARL algorithms on the four goal-conditioned building tasks is shown in Figure 3, where the performance of different algorithms varies among tasks. Specifically, in the two first-layer tasks, the performance of COMA is apparently lower than other algorithms. By analyzing the parameters in the training of the model, it can be seen that the variance of counterfactual advantages of COMA is significantly higher than other algorithms, leading to its poor performance in these tasks with sparse reward. Besides, since COMA has the assumption that each agent will follow the current policy while fixing the action of other agents, the cooperation of agents in these cooperative tasks will be harmed. This disadvantage has also been observed in other experiments. In the learning process of COMA, different agents often compete for limited blocks and try to transport them to the destination they want to reach, or move the blocks moved by other agents to the desired location. This problem leads to a lot of useless competition in the training process, which harms the performance of the algorithm.

In addition, it can be seen that all the algorithms shows different degrees of instability, which is particularly obvious in the tasks of higher difficulties. This phenomenon is mainly due to the $\varepsilon$-greedy strategy introduced in the training process (detailed discussions are provided in Appendix E). Therefore, comparing the success rates of different algorithms on the same task will be more significant than simply comparing the numerical result of the reward, which is shown in Table 2.

*4.3.2 Performance on Free Building Tasks.* Different from previous environments, for the free building tasks, we do not specify the specific blueprint of the building, but encourage agents to cooperate freely to construct structures with high complexity.

In the task shown in Figure 2, the agents are encouraged to construct a large-scale interconnection branch with blocks. As shown by the cumulative reward in the training procedure (Figure 3) and the comparison of success rates (Table 2), QMIX and VDN performs observably better than other algorithms, which benefits by their simple yet effective decomposition of the value function. Concretely speaking, in the given free building task, the most effective way for the agents to cooperative is to establish an effective collaborative strategy that can allocate a low overhead handling strategy for each agent so that different blocks can be connected quickly. Besides, this strategy should ensure that there is no or as little conflict between paths of different agents as possible. Therefore, both the value decomposition of QMIX according to the monotonicity assumption and the additive decomposition of VDN can find the strategy that maximizes the reward of each agent while ensuring the optimal global reward, thus beneficial for the learning of the agents.

*4.3.3 Performance on Breaking Barrier Tasks.* From the comparison of returns (Figure 3), it can be seen that different algorithms have significant gap in this task. To be specific, the performance of QMIX

**Table 2: The average success rate of the benchmarking algorithms in the tasks.**

| *Task* | *COMA* | *IQL* | *MAPPO* | *QMIX* | *QTRAN* | *VDN* |
|---|---|---|---|---|---|---|
| Strip-shaped building | 0.30 | 0.88 | 0.91 | 0.86 | **0.92** | 0.35 |
| Block-shaped building | 0.00 | 0.85 | **0.90** | 0.86 | 0.89 | 0.63 |
| Simple two-story building | 0.00 | 0.49 | 0.35 | 0.44 | 0.30 | **0.71** |
| Complex two-story building | 0.00 | 0.00 | 0.02 | **0.28** | 0.00 | 0.00 |
| Free building[2] | 0.00 | 0.32 | 0.44 | **0.95** | 0.77 | 0.90 |
| Breaking Barrier | 0.00 | 0.02 | 0.00 | **0.57** | 0.00 | 0.29 |
| **Average** | 0.05 | 0.43 | 0.44 | **0.66** | 0.48 | 0.48 |

algorithm is significantly higher than that of other algorithms, and reaches the highest reward level. In addition, the VDN algorithm has achieved good performance and can complete tasks to some extent. In contrast, algorithms such as QTRAN and MAPPO can only achieve some progress (such as moving the flag to make it closer to the goal), but cannot complete the task. By analyzing the learning procedure of the algorithms, we can draw a conclusion that the reason why QMIX and VDN can complete the task is due to their property of value function decomposition. Under an efficient strategy to complete the task, most of the agents should act as a facilitator: they need to denote themselves in clearing an convenient path for other agents to carry the flag to the goal, which is critical for the success of the task but is not explicitly rewarded. Therefore, the additional flexibility brought by the value decomposition network in QMIX and VDN can help different agents coordinate their strategies to complete the task, which has been observed to be especially effective in difficult tasks [14].

### 4.4 Result with Large-Scale Distributed Training Tasks

In order to exploit the potential and the flexibility of CraftEnv, we also design scenarios with more complex structure to train with large-scale distributed resources, which more accurately simulates the working environment in real-world scenarios such as smart logistics. As MAPPO shows the ability to utilize large-scale samples in complex tasks in practice [3, 27], we choose it as the algorithm used in the large-scale training process. The design of these simulation tasks are shown in Figure 4, and the details are listed in Appendix C. The video recording the result with distributed training is also provided in the supplemental materials. We believe that in future practice, with detailed construction of the simulation task for real-life applications in CRC systems, CraftEnv can promote richer applications of MARL in real-world applications.

### 4.5 Result on Physical Machines

Aside from the simulation, we also have designed and built the physical CraftEnv robotic system in the real world [25]. We successfully deployed the policies learned using CraftEnv on these physical machines, demonstrating the advantage of CraftEnv in connecting the simulated interaction and real-life deployments. Benefit by the comprehensive design of the state space and action space in CraftEnv, this process can be completed directly with the corresponding physical model. Some examples of the result is shown in Figure 5. Besides, the deployment results are also recorded in the supplemental materials. With comprehensive and rich physical constraints, CraftEnv can help to develop stable and easy-to-use strategies for MARL agents to learn in the applications of CRC systems.

## 5 RELATED WORK

To our best knowledge, CraftEnv is the first CRC environment for MARL research, which is designed with the aim of pushing forward the development of collective intelligence and swarm technology. As demonstrated in our experiments, CraftEnv can be used as a simulation environment for real-world scenarios such as smart warehousing and intelligent construction, and can be easily and efficiently deployed to real-world applications. Structurally speaking, CraftEnv is a fully-cooperative MARL environment that is enlightened by the MineCraft game and has high flexibility. Currently, there has been a variety of cooperative MARL environments and CRC simulation scenarios in the research fields, but CraftEnv shows its unique advantages in multiple aspects such as system architecture, task design, and application deployment.

In the context of CRC, there are many studies focusing on the improvement on traditional methods, such as SAPSO [28], SAFER [12] and NAIVE [11]. However, most of these researches only consider simple tasks such as building blocks of specified structure [28], or approximating rigid bodies with linear elements and used finite element analysis (FEA) for structural calculations [11]. Even though there exists some works that use intuitive and easy-to-use engines and game development tools such as Unity3D to implement dynamic simulation environments [12], but the simulation is still restricted in patterns such as unanchored structures and irregular terrains. In comparison, CraftEnv supports various types of elements with comprehensive and detailed physical constraints and interfaces for custom objects. For the actions of agents, CraftEnv also supports action masks that fit the physical constraints of the real world scenarios.

Besides, there have been some studies on the application of MARL in CRC systems [1, 18, 20]. However, the tasks considered in these works are either using goal structures consisting only of blocks to estimate the performance of the trained policy [18] or using point mass boids to test their tuning behavior [1]. However, these tasks are designed only as games that are far from real-world

[2] Strictly speaking, there is no concept of "task completion" in free building tasks, as there is no unique evaluation metric, instead agents are encouraged to use their own creativity to achieve higher rewards. Here the "success" of task is that the agents have found a way to connect all the blocks to form a large platform.

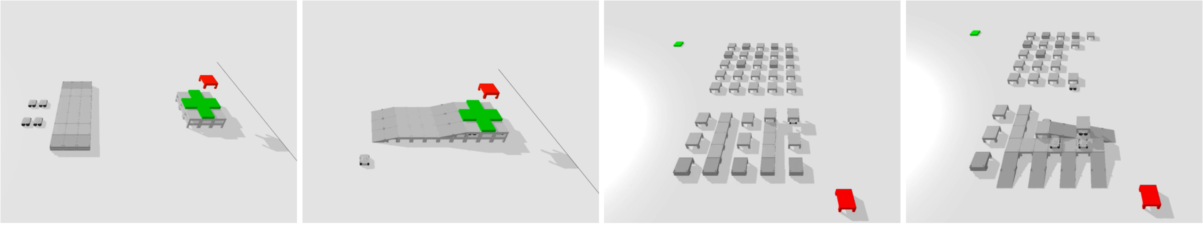

**Figure 4: Example of goal-conditioned building tasks (left) and free building tasks (right) with distributed training.**

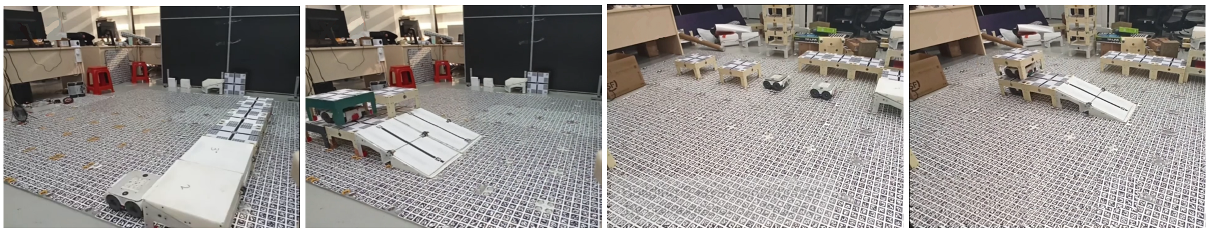

**Figure 5: Experiment on the physical machines [25].**

**Table 3: Comparison of CraftEnv with other popular cooperative MARL environments.** CraftEnv supports the customization of multiple types of tasks based on various elements. CraftEnv's comprehensive physical constraints facilitate the simulation of multiple real-world scenarios and can be tested in real machines with the counterpart entities for the components of the environment.

| *Environment* | *Observability* | *Reward Setting* | *Agent Number* | *Main Difficulty* | *Task Number* |
|---|---|---|---|---|---|
| SMAC | Partial | Dense | 2 – 10 | Large action space | 1 |
| LBF | Partial / Full | Sparse | 2 – 4 | Coordination among agents | 7 |
| MPE | Partial / Full | Dense | 2 – 3 | Non-stationary | 9 |
| RWARE | Partial | Sparse | 2 – 4 | Sparse reward | 3 |
| **CraftEnv** | **Partial / Full** | **Sparse / Dense** | **Flexible** | **Complex and diverse tasks** | **Free to Design** |

applications. Compared with them, CraftEnv is the first comprehensive and rich MARL environment for the simulation of CRC systems. The task settings of CraftEnv are not only closely linked to real-world applications such as smart warehousing, but also provide high flexibility, making the evaluation of the MARL algorithm not only more comprehensive, but also effectively integrated with real-world applications.

Current MARL environments that have connections to real-life applications often try to reflect the cooperation ability among agents with goals that require high degree of collaboration. For example, in the LBF environment [4], agents need to collect randomly scattered food in a grid world, and in RWARE [14], agents are asked to place the shelves into designated workspaces, which is similar to CraftEnv's task design. However, the physical constraints in RWARE on the workspace are relatively simple, and it only considers the two-dimensional case, which is far from the practical application scenarios. CraftEnv, by contrast, provides detailed physical constraints in 3D scenarios with rich elements that can more appropriately simulate real-world tasks such as smart warehousing. In addition, the aforementioned MARL environments are either pure game environments or only simulations of real-world tasks. CraftEnv, however, provides physical machine support, where the trained strategy can be directly deployed on physical models. Detailed comparison with other MARL environments is shown in Table 3.

## 6 CONCLUSION

We propose the first MARL environment for CRC scenarios, CraftEnv, which can model multiple real-world scenarios such as smart warehousing and smart construction from the perspective of reinforcement learning. CraftEnv can not only conveniently and consistently evaluate the performance of various MARL algorithms in real scenes, but also promote the application of MARL technology in real tasks on this basis, thus promoting the development of collective intelligence. In the experiments, we find that the value factorization based method can often achieve better performance in multiple tasks of CraftEnv. Besides, the bottleneck of performance for different algorithms mainly exists in the task allocation of agents and the early exploration of the environment. We hope CraftEnv can shed some light on the relative strengths and limitations of existing MARL algorithms in real-life applications and provide guidance in terms of practical considerations and future research, In this way, swarm intelligence can be further developed in a variety of real situations.

## ACKNOWLEDGEMENT

Shuai Li is supported by National Natural Science Foundation of China (92270201, 62006151, 62076161) and Shanghai Sailing Program.

## REFERENCES

[1] Shadi Abpeikar, Kathryn Kasmarik, Matthew Garratt, Robert Hunjet, Md Mohiuddin Khan, and Huanneng Qiu. 2022. Automatic collective motion tuning using actor-critic deep reinforcement learning. *Swarm and Evolutionary Computation* 72 (2022), 101085.

[2] CS Chen, Yaqing Hou, and Yew-Soon Ong. 2016. A conceptual modeling of flocking-regulated multi-agent reinforcement learning. In *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 5256–5262.

[3] Long Chen, Bin Hu, Zhi-Hong Guan, Lian Zhao, and Xuemin Shen. 2021. Multiagent meta-reinforcement learning for adaptive multipath routing optimization. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[4] Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. 2020. Shared Experience Actor-Critic for Multi-Agent Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[5] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. 2012. Large scale distributed deep networks. *Advances in neural information processing systems* 25 (2012).

[6] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. 2018. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*. PMLR, 1407–1416.

[7] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[8] Lei Han, Jiechao Xiong, Peng Sun, Xinghai Sun, Meng Fang, Qingwei Guo, Qiaobo Chen, Tengfei Shi, Hongsheng Yu, and Zhengyou Zhang. 2020. Tstarbot-x: An open-sourced and comprehensive study for efficient league training in starcraft ii full game. *arXiv preprint arXiv:2011.13729* (2020).

[9] Samuel Leder, HyunGyu Kim, Ozgur Salih Oguz, Nicolas Kubail Kalousdian, Valentin Noah Hartmann, Achim Menges, Marc Toussaint, and Metin Sitti. 2022. Leveraging Building Material as Part of the In-Plane Robotic Kinematic System for Collective Construction. *Advanced Science* 9, 24 (2022), 2201524.

[10] Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph Gonzalez, Michael Jordan, and Ion Stoica. 2018. RLlib: Abstractions for distributed reinforcement learning. In *International Conference on Machine Learning*. PMLR, 3053–3062.

[11] Nathan Melenbrink, Panagiotis Michalatos, Paul Kassabian, and Justin Werfel. 2017. Using local force measurements to guide construction by distributed climbing robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4333–4340.

[12] Nathan Melenbrink and Justin Werfel. 2018. Local force cues for strength and stability in a distributed robotic construction system. *Swarm Intelligence* 12, 2 (2018), 129–153.

[13] Igor Mordatch and Pieter Abbeel. 2017. Emergence of Grounded Compositional Language in Multi-Agent Populations. *arXiv preprint arXiv:1703.04908* (2017).

[14] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. 2021. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS)*. http://arxiv.org/abs/2006.07869

[15] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 4295–4304.

[16] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043* (2019).

[17] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philiph H. S. Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. *CoRR* abs/1902.04043 (2019).

[18] Guillaume Sartoretti, Yue Wu, William Paivine, TK Kumar, Sven Koenig, and Howie Choset. 2019. Distributed reinforcement learning for multi-robot decentralized collective construction. In *Distributed autonomous robotic systems*. Springer, 35–49.

[19] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 5887–5896.

[20] Caroline Strickland, David Churchill, and Andrew Vardy. 2019. A reinforcement learning approach to multi-robot planar construction. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 238–244.

[21] Peng Sun, Jiechao Xiong, Lei Han, Xinghai Sun, Shuxing Li, Jiawei Xu, Meng Fang, and Zhengyou Zhang. 2020. Tleague: A framework for competitive self-play based distributed multi-agent reinforcement learning. *arXiv preprint arXiv:2011.12895* (2020).

[22] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296* (2017).

[23] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[24] Robert Tarjan. 1972. Depth-first search and linear graph algorithms. *SIAM journal on computing* 1, 2 (1972), 146–160.

[25] Qiwei Xu, Yizheng Zhang, Shenghao Zhang, Rui Zhao, Zhuoxing Wu, Dongsheng Zhang, Cheng Zhou, Xiong Li, Jiahong Chen, Zengjun Zhao, et al. 2022. RECCraft System: Towards Reliable and Efficient Collective Robotic Construction. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 8979–8986.

[26] X Xu and B García De Soto. 2022. Reinforcement learning with construction robots: A preliminary review of research areas, challenges and opportunities. In *39th International Symposium on Automation and Robotics in Construction, ISARC 2022*. International Association for Automation and Robotics in Construction (IAARC), 375–382.

[27] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. 2021. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955* (2021).

[28] Henry Zapata, Niriaska Perozo, Wilfredo Angulo, and Joyne Contreras. 2020. A hybrid swarm algorithm for collective construction of 3D structures. *International Journal of Artificial Intelligence* 18, 1 (2020), 1–18.