2025 IEEE/SICE International Symposium on System Integration (SII)
January 21-24, 2025, Munich, Germany

Task-priority Intermediated Hierarchical Distributed Policies: Reinforcement Learning of Adaptive Multi-robot Cooperative Transport

Yusei Naito[1,2], Tomohiko Jimbo[3,1], Tadashi Odashima[1], and Takamitsu Matsubara[2]

***Abstract*— Multi-robot cooperative transport is crucial in logistics, housekeeping, and disaster response. However, it poses significant challenges in environments where objects of various weights are mixed and the number of robots and objects varies. This paper presents Task-priority Intermediated Hierarchical Distributed Policies (TIHDP), a multi-agent Reinforcement Learning (RL) framework that addresses these challenges through a hierarchical policy structure. TIHDP consists of three layers: task allocation policy (higher layer), dynamic task priority (intermediate layer), and robot control policy (lower layer). Whereas the dynamic task priority layer can manipulate the priority of any object to be transported by receiving global object information and communicating with other robots, the task allocation and robot control policies are restricted by local observations/actions so that they are not affected by changes in the number of objects and robots. Through simulations and real-robot demonstrations, TIHDP shows promising adaptability and performance of the learned multi-robot cooperative transport, even in environments with varying numbers of robots and objects.**

I. INTRODUCTION

Multi-robot cooperative transport, where robots collaborate to move objects, is gaining traction across logistics, household chores, and disaster response [1][2]. Take room tidying, for instance: lighter objects may be moved individually, while heavier ones require teamwork. However, predicting object weights beforehand is difficult due to the frequent replacement of objects, necessitating decisions on cooperation or individual handling. Hence, this study centers on multi-agent reinforcement learning for distributed control strategies, enabling multiple robots to cooperatively transport objects of various unknown weights in environments with wireless communication among robots.

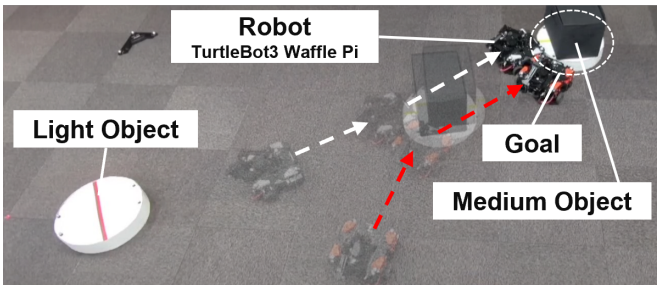
Each robot repeatedly selects and transports an object in an environment with multiple objects. Therefore, the problem can be naturally divided into two parts: the task allocation problem for choosing which object to transport and the robot control problem for physical transport. An appropriate solution to solving these two problems simultaneously may be to learn a hierarchical policy: the higher policy for selecting objects and the lower policy for controlling the robot consistently.

*This work was not supported by any organization

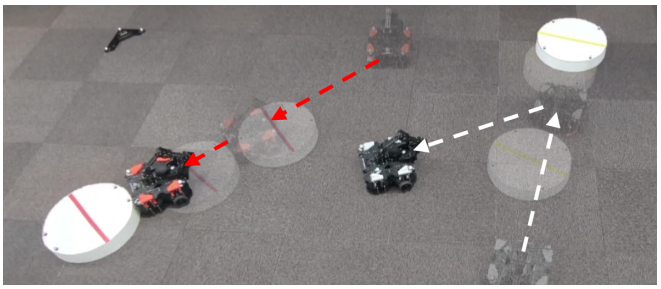
[1]Y. Naito, T. Jimbo and T. Odashima are with the R-Frontier Division, Frontier Research Center, Toyota Motor Corporation, 1, Toyota-cho, Toyota, Aichi 471-8571, Japan

[2]Y. Naito and T. Matsubara are with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Nara, Japan.

[3]T. Jimbo is with Toyota Central R&D LABS., inc., Aichi, 480-1192, Japan



(a) Cooperative action



(b) Independent action

Fig. 1: Real-robot demonstrations by our method. (a) Robots have high priority on one medium object and cooperate to transport it. (b) Each robot has a high priority for separate objects and transports them independently.

Applying Reinforcement Learning (RL) with such a hierarchical policy is challenging in environments where objects of various weights are mixed and the number of robots and objects varies, which is common in logistics, housekeeping, and disaster response applications. The higher policy inputs information about all entities and outputs object IDs for robot transportation. However, RL typically assumes fixed observation and action dimensions, complicating policy reuse with varying entity counts [3][4]. To address this, approaches with local observations and actions have been explored [5][6]; however, relying solely on local observations can hinder task allocation performance, especially in large-scale environments. Additionally, local actions limit object selection, complicating adaptability versus performance balance.

This paper introduces Task-priority Intermediated Hierarchical Distributed Policies (TIHDP), a multi-agent RL framework that addresses these challenges through a hierarchical policy structure. Each robot within this framework operates on a distributed policy structured into three layers: 1) task allocation policy (higher), 2) dynamic task priority (intermediate), and 3) robot control policy (lower). To maintain

performance as the number of objects and robots in the environment changes, the task allocation and robot control policies are both restricted by local observations/actions so that they are not affected by changes in the number of objects and robots. The dynamic task priority layer, on the other hand, receives global object information and communicates with other robots to manipulate the priority of any object. The task allocation policy manages the priorities of objects in the neighborhood of the robots, while the task priority layer determines the highest priority object globally based on higher levels of influence and inter-robot communication. Finally, the robot control policy controls the behavior of the robot with respect to the target object. Through simulations and real-robot demonstrations (Fig. 1), TIHDP shows promising adaptability and performance in learning multi-robot cooperative transport tasks, even in environments with varying numbers of robots and objects.

II. RELATED WORK

A. Combinatorial Optimization

When employing deterministic methods in cooperative transportation, the task allocation can be formulated as a combinatorial optimization problem to minimize total travel distance under specified constraints on robot numbers. Techniques like the Hungarian algorithm [7] and integer linear programming [8] are utilized for this purpose. While these methods offer optimal solutions for minimizing travel distance, they often assume the availability of prior information regarding the required number of robots for each object. However, this assumption may not always hold. For instance, objects with frequent changes might necessitate manual data registration or estimation through cameras. Although cameras can provide object shape information, estimating required robot numbers based on weight poses challenges.

In contrast, the proposed method's strategy is crafted to determine subsequent actions relying on the robot's behavior and resulting changes in the position and speed of both the robot and object. This approach operates independently of prior knowledge about the necessary number of robots for transportation.

B. Multi-agent Reinforcement Learning (MARL)

Multi-agent Reinforcement Learning (MARL) has been explored for task allocation purposes [3][4], utilizing Markov Decision Processes and Multi-agent Deep Deterministic Policy Gradient [9]. However, these methods assume fixed numbers of robots and tasks, limiting their applicability beyond training scenarios. Addressing this, distributed policy models have been proposed, setting minimum thresholds for robots and tasks [5]. In multi-robot cooperative transport, frameworks employing dynamic task priorities and global communication have been effective [6], proving superior to using simplified rule-based algorithms for robot control. Cooperation in robot control, including force coordination and collision avoidance, is vital for real-world object transport. Existing research focuses on cooperative control and communication to position large objects [10].

Our proposed method employs MARL with a distributed policy model, limiting minimum robots and tasks. Distinguishing itself, it learns to handle multiple objects and specific robot controls for each transport, obviating the need for manual rule development.

C. Hierarchical Reinforcement Learning

Cooperative multi-object transport presents challenges due to the complexity of having each robot handle multiple objects. Learning such tasks, known for their long processes, is difficult [11][12]. Hierarchical reinforcement learning has been explored to address this, with goal-conditioned methods training lower-level policies to approach goal states set by higher-level policies and option framework-based methods training higher-level policies to switch between lower-level policies. These approaches have shown effectiveness in tasks like navigation [13].

Our method deviates from prior work by introducing the dynamic task priority layer between the two policies, treating object IDs as targets in lower-level policies. Object IDs are represented discretely, inform lower-level observations and rewards, and enable the lower layer to focus on transporting individual objects as a reinforcement learning task.

III. PROBLEM FORMULATION

The transport task targeted in this study consists of N robots and M objects, along with corresponding goals for these objects. Here, both the robots and goals are assumed to be point mass models on a plane.

We first define the position and orientation, and the velocity and angular velocity of robot i as $\boldsymbol{x}_i \in \mathbb{R}^3$ and $\dot{\boldsymbol{x}}_i \in \mathbb{R}^3$, respectively. Next, for object l, its position, goal position, velocity, and weight are defined as $\boldsymbol{z}_l \in \mathbb{R}^2$, $\boldsymbol{z}_l^* \in \mathbb{R}^2$, $\dot{\boldsymbol{z}}_l \in \mathbb{R}^2$, and $m_l \in \mathbb{R}$, respectively. Additionally, the radius of the goal is denoted as D. Using these definitions, the transport of object l is considered complete when $||\boldsymbol{z}_l - \boldsymbol{z}_l^*|| \leq D$.

Each robot is capable of observing the above-mentioned states for all robots. Furthermore, they can observe all states of each object except for weight w. Each robot's policy outputs the control command $\boldsymbol{u}_i$ based on this information.

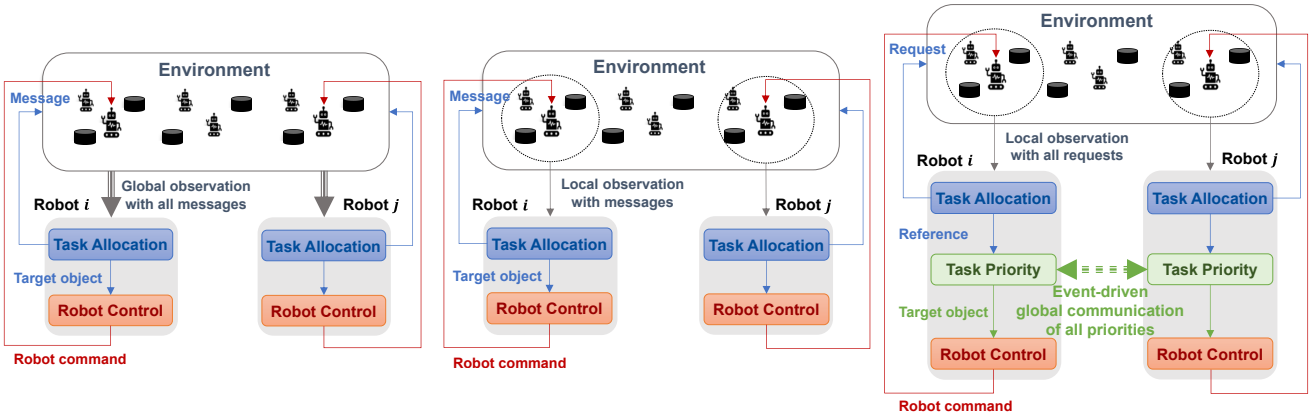
Additionally, the following assumptions are made:

- Robots know M and N
- Robots can communicate globally

IV. METHOD

This section describes our TIHDP-based method for adaptive cooperative transport. Fig. 2 shows the difference between global and local observation methods and our method.

In Fig. 3, Robot i initially receives a high-level observation $\boldsymbol{o}_i^{\text{hi}}$ that includes information about nearby robots and objects. The task allocation policy, π_i^{hi}, then generates a higher-level action $\boldsymbol{a}_i^{\text{hi}}$ based on this observation. Subsequently, in the dynamic task priority layer, the higher-level action is utilized to adjust the dynamic task priority $\boldsymbol{\phi}_i := [\phi_i^1, ..., \phi_i^M] \in \mathbb{R}^M$ through the robot's priority operation command $\boldsymbol{c}_i$. Then, by comparing its request signal α_i with the response signal β_j from robot j, the priority of robot j's top-priority object l_j^* in



(a) Two-layered-global (b) Two-layered-local (c) Ours

Fig. 2: Overview of frameworks. (a) Robots observe globally and select an object. (b) Robots observe locally and select an object from the vicinity. (c) Robots observe locally and update the task priority that possesses global memory.

its list $\boldsymbol{\phi}_j$ is increased if a match occurs. Each robot focuses on transporting its highest-priority object l_i^* from its list $\boldsymbol{\phi}_i$, using its observation $\boldsymbol{o}_i$ to derive a lower-level observation $\boldsymbol{o}_i^{\mathrm{lo}}$ necessary for this task. Based on this observation, the lower-level policy π_i^{lo} generates the robot command $\boldsymbol{a}_i^{\mathrm{lo}} = \boldsymbol{u}_i$.

With such structured policies, both adaptability and performance to changes in the number of robots and objects can be achieved. More details are shown below.

A. Distributed Partially Observable Markov Decision Process

We utilize a distributed partially observable Markov decision process to apply MARL to multi-robot cooperative transportation. First, we define the state of the environment as $\boldsymbol{s} = [\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{z}, \boldsymbol{z}^*, \dot{\boldsymbol{z}}, \boldsymbol{w}]$. This includes information about all robots and objects.

Next, for robot i, we define its observation and action as $\boldsymbol{o}_i$ and $\boldsymbol{a}_i$, respectively. The action $\boldsymbol{a}_i$ includes the robot's control command $\boldsymbol{u}_i$ and, if necessary, signals to other robots. Each robot's distributed policy can observe information in the state $\boldsymbol{s}$, excluding the weight of the objects $\boldsymbol{w}$. However, to accommodate changes in the number of robots and objects, the number of input dimensions to the neural network must be constant. Therefore, the actual information about the robots and objects used is always selected in a fixed number according to each method.

Finally, we describe the reward design for evaluating actions that contribute to object transportation. The reward for object l, r_l, is given as follows:

$$r_l = \dot{\boldsymbol{z}}_l \cdot \frac{\boldsymbol{z}_l^* - \boldsymbol{z}_l}{\|\boldsymbol{z}_l^* - \boldsymbol{z}_l\|}. \tag{1}$$

As shown in Equation (1), the reward uses the component of the object's velocity in the direction of the goal. Therefore, a positive reward is obtained for actions that bring a robot closer to the object faster, while a negative reward (penalty) is obtained for actions that take it away from the object.

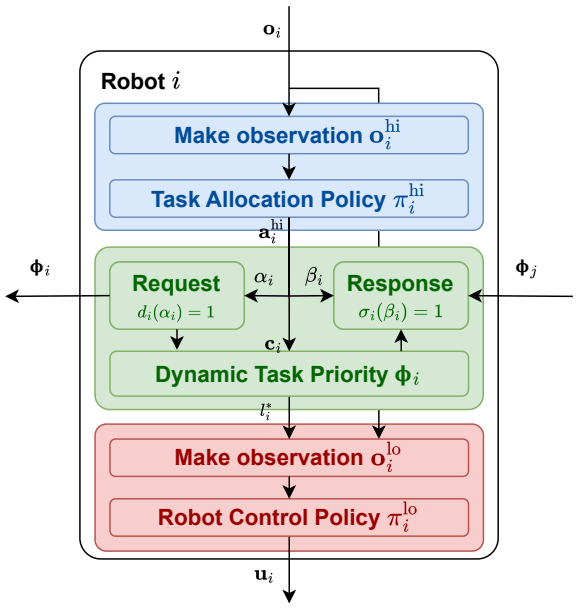


Fig. 3: Task-priority Intermediated Hierarchical Distributed Policy. In robot i, the two policies of higher and lower use local observations $\boldsymbol{o}_i^{\mathrm{hi}}$ and $\boldsymbol{o}_i^{\mathrm{lo}}$. In the intermediate layer, task priorities $\boldsymbol{\Phi}_i$ are maintained while conducting global communication and the task priorities fluctuate when a request $d_i(\alpha_i)$ and response $\sigma_i(\beta_i)$ are established. The policy ultimately outputs the robot's control command $\boldsymbol{u}_i$.

To maximize the acquired reward, robots must always aim to acquire action policies that transport more objects to the goal faster.

B. Task-priority Intermediated Hierarchical Distributed Policy Model

1) Task Allocation Layer: Based on the method of Shibata et al. [6], each robot has a task allocation policy responsible for manipulating its dynamic task priorities and determining

cooperative timings based on global communication.

As shown in Fig. 2c, the policy always observes a certain number of robots and objects in order of proximity. Specifically, as shown in Fig. 3, the task allocation layer of robot i uses the positions $\bar{\boldsymbol{x}}_i$, velocities $\dot{\bar{\boldsymbol{x}}}_i$ of the nearby J robots, and the positions $\bar{\boldsymbol{z}}_i$, goal positions $\bar{\boldsymbol{z}}_i^*$, and velocities $\dot{\bar{\boldsymbol{z}}}_i$ of the nearby K untransported objects to create the higher-level observation $\boldsymbol{o}_i^{\text{hi}} = [\boldsymbol{u}_i, \bar{\boldsymbol{x}}_i, \dot{\bar{\boldsymbol{x}}}_i, \bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}_i^*, \dot{\bar{\boldsymbol{z}}}_i]$. J and K are hyperparameters and are set at about the lowest number present in the environment. Therefore, the distributed policy model for task allocation in this local observation is given as $\boldsymbol{a}_i^{\text{hi}} = [\boldsymbol{c_i}, \alpha_i, \beta_i] = \pi_i^{\text{hi}}(\boldsymbol{o}_i^{\text{hi}})$.

Here, $\boldsymbol{c}_i := [c_i^1, ..., c_i^K] \in \{-1, 1\}^K$ represents the manipulation of priorities ϕ_i^l for nearby objects, $\alpha_i \in \{0, 1\}$ is the output for request signals, and $\beta_i \in \{0, 1\}$ is the output for response signals.

2) Dynamic Task Priority: Based on the method of Shibata et al. [6], each robot has a dynamic task priority layer that updates the priorities for transporting each object.

As shown in Fig. 3, the dynamic task priority layer manipulates the priorities of each object based on higher-level actions and communication with other robots, identifying the object that should be transported at that moment. All priorities are initialized with the same value.

First, we explain the operation of global communication. For robot i, request signals d_i^l and response signals σ_i for object l are calculated using α_i and β_i included in the higher-level actions as follows:

$$d_i^l(\alpha_i) = \begin{cases} 1, & \text{if } \alpha_i = 1 \text{ and } l = l_i^* \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

$$\sigma_i(\beta_i) = \begin{cases} 1, & \text{if } \beta_i = 1 \\ 0, & \text{otherwise} \end{cases}. \tag{3}$$

These signals are always shared among all robots and affect the dynamic task priority when a request-response is established between robots.

Using the request signal d_i^l, response signal σ_i, and its own manipulation $\boldsymbol{c}_i$, the dynamic task priority is updated as follows:

$$\phi_i^{l\prime} = \begin{cases} 0, & \text{if } \|\boldsymbol{z}_l^* - \boldsymbol{z}_l\| < D \\ (1 - k_\phi)\phi_i^l + k_\phi \left(\bar{c}_i^l + \sigma_i \sum_{j=1}^{N} d_j^l \right), & \text{otherwise} \end{cases}, \tag{4}$$

$$\phi_i^l = \frac{\phi_i^{l\prime}}{\sum_{l=1}^{M} \phi_i^{l\prime}}. \tag{5}$$

Here, $k_\phi > 0$ is a constant that limits the amount of priority change per operation, and $\bar{\boldsymbol{c}}_i := [\bar{c}_i^1, ..., \bar{c}_i^M] \in \{-1, 0, 1\}^M$ is a mapping of $\boldsymbol{c}_i$ in the order of object IDs, and values beyond the nearest K neighbors are zero.

As shown in Equation (4), the priority of an object that has been transported at that point is set to 0. The determination of whether the transportation is complete is constantly made, so an object that has been transported once will return as a transportation target if it moves away from the goal more than D due to being touched by a robot.

Based on the manipulation $\boldsymbol{c}_i$ of the priorities of nearby objects, the priority of any object either increases or decreases. Furthermore, based on global communication, if the request signal d_j^l for object l from robot j and the response signal σ_i from robot i satisfy $d_j^l = \sigma_i = 1$, the priority ϕ_i^l of object l held by robot i increases. In other words, it is possible to raise the priority of objects not included in the higher-level observation $\boldsymbol{o}_i^{hi}$ in response to requests from other robots. Finally, as shown in Equation (5), all task priorities are normalized so that their sum equals 1.

The object l_i^* that robot i should transport at that moment is selected as the one with the highest priority among all objects, as shown in the following equation:

$$l_i^* = \underset{l}{\text{argmax}}\ \phi_i^l. \tag{6}$$

3) Robot Control Layer: Each robot has a robot control layer for selecting the robot's control commands $\boldsymbol{u}_i$.

As shown in Fig. 3, the robot control layer for robot i uses the currently required object l_i^* for transportation to create the lower-level observation $\boldsymbol{o}_i^{\text{lo}}$. This includes information about the nearest object to avoid collisions. In other words, a single robot control policy is used for all objects. This allows the experience of trial and error in transportation to be shared across all objects, potentially improving sample efficiency during learning. For the robots, only the nearest information is observed. Therefore, the lower-level observation $\boldsymbol{o}_i^{\text{lo}}$ used by the robot control policy is given as $\boldsymbol{o}_i^{\text{lo}} = [\boldsymbol{u}_i, \boldsymbol{x}_{\text{nearest}_i}, \dot{\boldsymbol{x}}_{\text{nearest}_i}, \boldsymbol{z}_{l_i^*}, \boldsymbol{z}_{l_i^*}^*, \dot{\boldsymbol{z}}_{l_i^*}, \boldsymbol{z}_{\text{nearest}_i}, \dot{\boldsymbol{z}}_{\text{nearest}_i}]$. Thus, the distributed policy model for robot control in this observation $\boldsymbol{o}_i^{\text{lo}}$ is given as $\boldsymbol{a}_i^{\text{lo}} = \boldsymbol{u}_i = \pi_i^{\text{lo}}(\boldsymbol{o}_i^{\text{lo}})$.

C. Hierarchical Reward Design

Reinforcement learning is conducted in both the task allocation layer and the robot control layer in a synchronized way; the task allocation layer uses a reward aimed at optimization for the entire team, while the robot control layer uses another reward focused on optimizing the transportation of a single object.

1) Reward for Task Allocation Policy: The reward r_i^{hi} for the task allocation policy of robot i is given as:

$$r_i^{\text{hi}} = \sum_{l=1}^{M} r_l. \tag{7}$$

This ensures that all robots receive the same reward at the same time. To maximize the acquired reward, cooperative behavior that considers other robots is necessary.

2) Reward for Robot Control Policy: The reward r_i^{lo} for the robot control policy of robot i is given as follows:

$$r_i^{\text{lo}} = r_{l_i^*} + \min(0, r_{\text{nearest}}) + e_i, \tag{8}$$

where,

$$e_i = \begin{cases} 1, & \text{if } \dot{\boldsymbol{x}}_i \cdot (\boldsymbol{z}_{l_i^*} - \boldsymbol{x}_i) > 0 \text{ or } \|\boldsymbol{z}_{l_i^*} - \boldsymbol{x}_i\| < E \\ 0, & \text{otherwise} \end{cases}. \quad (9)$$

Here, $r_{l_i^*}$ is the reward for the object to be transported and r_{nearest} is the reward for the nearest object. E is a constant representing the distance at which an object is considered sufficiently close.

Thus, the robot control policy is trained with the goal of transporting the currently required object l^* and avoiding collisions with the nearest object.

D. Centralized Training with Decentralized Execution

We used the Multi-agent Proximal Policy Optimization (MAPPO) in the learning of each policy [14]. MAPPO is an extension of Proximal Policy Optimization (PPO) [15], a reinforcement learning algorithm, to a centralized training with decentralized execution structure. The Actor network corresponding to the policy is distributed using local observations. The Critic network used during learning can uniquely use the observations of other robots as well, facilitating the acquisition of cooperative actions.

V. EXPERIMENT

A. Experimental Overview

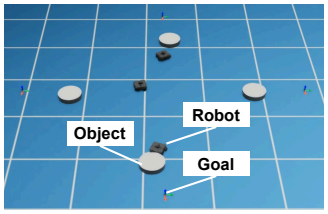
Simulations were used to learn and validate the proposed method. The experiment had two main objectives. The first objective was to verify whether the proposed method could effectively learn the action policy for the cooperative transport task. To achieve this, a physical simulation environment that closely mimicked the actual characteristics of the robots and objects was constructed. The second objective was to evaluate the performance of the learned policy. The transport performance of the post-learning policy was compared with two ablation studies where the task allocation layer was omitted, using either local communication or global communication. Additionally, performance evaluations were conducted in environments with different numbers of robots and objects than those used during the learning phase.

B. Simulation Environment

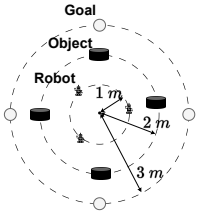
NVIDIA's physics simulator, Isaac Sim, was used to construct a parallel simulation environment for cooperative transport, as shown in Fig. 4a. Furthermore, an environment for learning and executing the proposed method was implemented, based on Isaac Orbit [16]. Parallel simulation environments and their corresponding reinforcement learning libraries enable faster collection of experience data compared to traditional learning environments, as reported by [17][18].

C. Task Setting

We used the TurtleBot3 Waffle Pi robot from ROBOTIS. TurtleBot3 has two drive wheels, one on each side. In this experiment, no actuators other than the drive wheels are used and the object is transported by pushing. The control command space is $\boldsymbol{u} = [u_{\text{move}}, u_{\text{turn}}]^T$, where $u_{\text{move}} \in \{\text{forward}, \text{backward}, \text{none}\}$ and $u_{\text{turn}} \in \{\text{right}, \text{left}, \text{none}\}$,



(a) Rendered image



(b) Initial placement

Fig. 4: Simulation environment. Robots, objects, and goals are arranged in a circular pattern from the inside out.

allowing for combined movements of forward-backward motion and turning.

Objects are disc-shaped. All objects have the same shape but three different weights: light, medium, and heavy (which cannot be carried even with cooperation). The robot cannot distinguish the weight of an object by its shape, so it has to decide whether to cooperate or share the work only when it actually pushes the object.

The placement of robots, objects, and goals is shown in Fig. 4b. Initially, robots are randomly positioned within a $0.5\,\text{m}$ radius from a reference point on the circumference of a $1\,\text{m}$ radius circle. Objects are randomly placed within a $0.3\,\text{m}$ radius from a reference point on the circumference of a $2\,\text{m}$ radius circle. Each object's goal is positioned on the circumference of a $3\,\text{m}$ radius circle.

D. Evaluation Metrics

To quantitatively evaluate the performance of the method, the following metrics are used:

- Task Completion Ratio (TCR): The proportion of objects successfully transported out of the total transportable objects in the environment.
- Full Task Completion Ratio (FTCR): A binary metric where a value of 1 is assigned if the TCR equals 1 (indicating all transportable objects were successfully transported), and a value of 0 is assigned otherwise.

E. Comparison Methods

The following two types, consisting of a task allocation layer and a robot control layer, were used for comparison:

- Two-layered-global: with global observation and action (Fig. 2a)
- Two-layered-local: with local observation and action (Fig. 2b)

F. Results

1) Transport Performance: The rewards for the proposed method and two comparison methods during training using the settings in Table I and Table II are shown in Fig. 5. Furthermore, the results of evaluating transport performance are presented in Table III. As shown in Table 3, our proposed method and Two-layered-local can be applied to any scenario different from the training scenarios, whereas Two-layered-global only applies to scenarios with the same number of robots and objects as in the training phase. Moreover, it was

TABLE I: Simulation parameters for training

Parameter		Value
Number of robots	N	3
Number of objects	M	4
Lightweight objects		2
Medium-weight objects		1
Heavyweight objects		1
Goal radius	D	0.1 m
Episode length		400 steps

TABLE II: Policy parameters

Parameter		Value
Task Allocation Policy Hidden Layers		[256, 128, 64] with LSTM
Robot Control Policy Hidden Layers		[256, 128, 64]
Task Priority Time Constant	k_ϕ	0.1

found that the proposed method has a higher transport performance compared to the two methods. The Two-layered-global method may have encountered difficulties in learning due to the size of the search space and action space. Moreover, it appears that the Two-layered-local method, lacking the capability to globally comprehend the environment, faced challenges in making appropriate decisions.

2) Cooperative and Divided Actions: We examined the occurrence of cooperation and division in transport tasks. The trajectories of robots and objects during transport are shown in Fig. 6.

In Fig. 6a, each robot starts transporting objects separately. Here, Robot 0, encountering a lightweight object (L), can complete its transport. However, Robot 1, encountering a heavyweight object (H), cannot proceed with the transport. Therefore, Robot 1 abandons the transport of this object and joins Robot 2, who is transporting a medium-weight object. Next, in Fig. 6b, Robot 0 abandons the transport of the heavyweight object and heads towards the remaining lightweight objects. Robots 1 and 2 cooperate to advance the transport of objects. Finally, from Fig. 6c to Fig. 6d, all objects except the heavyweight ones reach their destination. Thus, the action policy by the proposed method appropriately utilizes cooperation and division according to the situation, indicating that cooperation in task allocation was achieved.

Furthermore, from Fig. 6b to Fig. 6c, the two robots were observed to transport objects in a coordinated manner by appropriately combining their forces. Compared to solitary transport, cooperative transport requires the other robot to be taken into account to adjust the force and direction applied to the object. The proposed method was confirmed to be capable of acquiring such coordination in robot control.

3) Effectiveness of Global Communication: We examined the effectiveness of global communication in the proposed method. From Table III, it is evident that scenarios with communication enabled higher transport performance in all cases. Global communication primarily aids in decisions about cooperation and division for robots and objects outside local observation, suggesting it is more effective in scenarios with many robots and objects.

TABLE III: Evaluation of transport performance where L represents lightweight objects, M medium-weight objects, and H heavyweight objects.

Objects					Two-layered		Ours	
# L	# M	# H	# Robots	Metrics	global	local	w/o com	w/com
2	1	1	3	Mean TCR	0.638	0.737	0.805	**0.815**
				Mean FTCR	0.180	0.281	0.430	**0.500**
3	1	1	3	Mean TCR	-	0.757	0.791	**0.805**
				Mean FTCR	-	0.172	0.335	**0.367**
3	2	1	3	Mean TCR	-	0.584	0.634	**0.641**
				Mean FTCR	-	0.0234	0.0313	**0.0625**
3	2	1	4	Mean TCR	-	0.652	0.686	**0.705**
				Mean FTCR	-	0.00781	0.0938	**0.109**

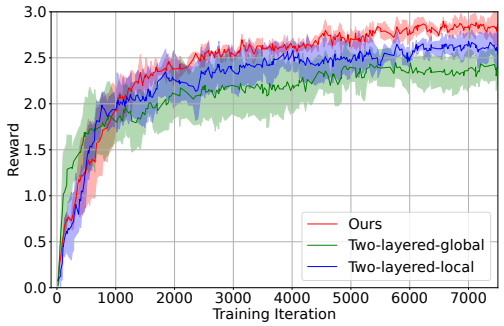


Fig. 5: Cumulative rewards of evaluated methods

VI. Demonstration

The aim of this demonstration was to verify that the policy learned in the simulation could transport the object to the target position using multiple real robots.

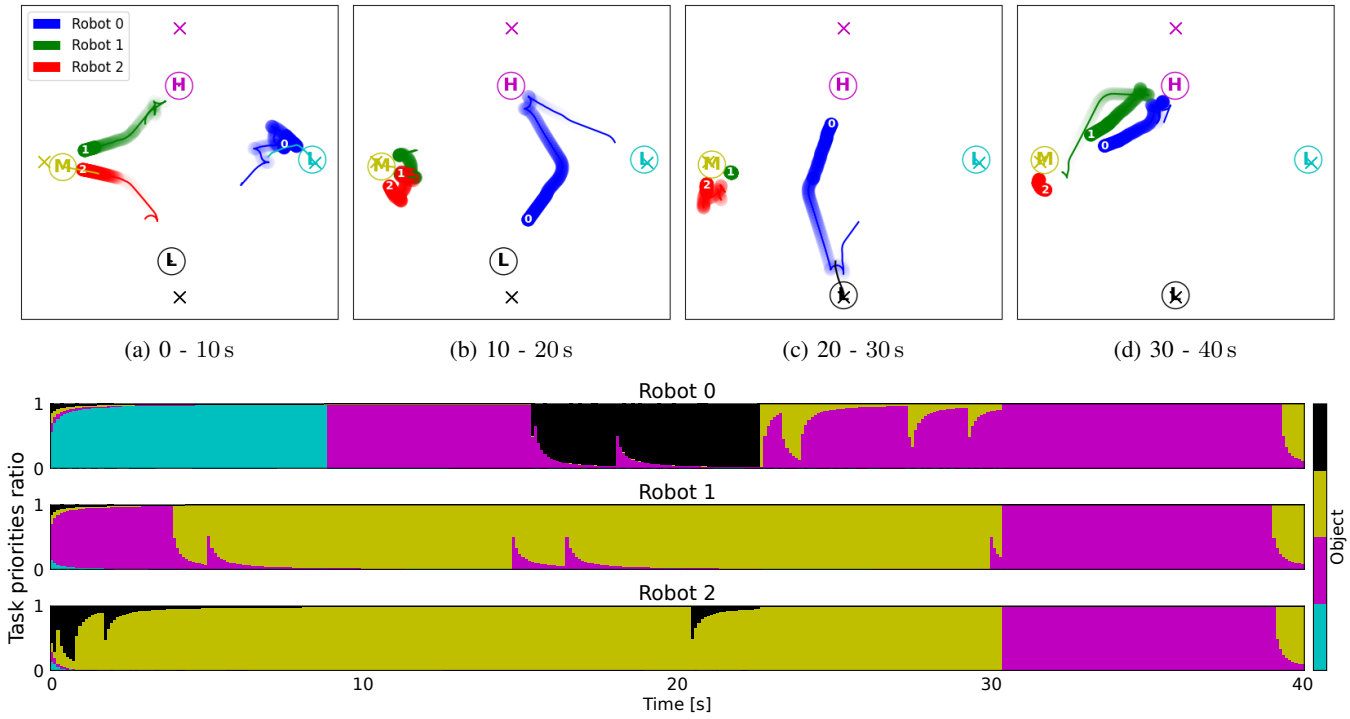
The positions and yaw angles of the robot and the object were observed using a motion capture system operating at 120 Hz. The velocity and angular velocity were calculated using the measured positions and yaw angles. The control inputs were computed on a control PC and transmitted to each robot at 10 Hz via Wi-Fi communication, using the policy learned in the simulation from these measurements.

The results are shown in Fig. 1a and Fig. 1b. Although there may be some gap between the simulation and the actual experiment, it was possible to control the object to a position close to the desired position after several trials.

VII. Discussion

This paper introduces a reinforcement learning framework, Task-priority Intermediated Hierarchical Distributed Policies, for coordinating multiple robots in the cooperative transport of objects of varying weights. The structured policies allow for learning control policies applicable to diverse scenarios with varying numbers of objects and robots.

In future work, our focus may shift towards refining practical robot control. This could involve delving into more complex controls, like cooperative grasping with arms capable of handling diverse shapes, for real-world application. Additionally, scalability investigation is vital for deploying these methods in large-scale logistics environments.



(a) 0 - 10 s (b) 10 - 20 s (c) 20 - 30 s (d) 30 - 40 s

(e) Task priorities

Fig. 6: Trajectories of robots and objects. (a)-(d) The colored circles, cross marks, and letters indicate the objects, their goals, and the weights. Robots should transport each object to the goal with the same color. (e) The color bar represents the proportion of the priority of each color object at each time step.

REFERENCES

[1] A. Nath, A. AR, and R. Niyogi, "A distributed approach for autonomous cooperative transportation in a dynamic multi-robot environment," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, ser. SAC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 792–799.

[2] E. Tuci, M. H. M. Alkilabi, and O. Akanyeti, "Cooperative object transport in multi-robot systems: A review of the state-of-the-art," *Frontiers in Robotics and AI*, vol. 5, 2018.

[3] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, "Joint optimization of multi-uav target assignment and path planning based on multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 146 264–146 272, 2019.

[4] T. Niwa, K. Shibata, and T. Jimbo, "Multi-agent reinforcement learning and individuality analysis for cooperative transportation with obstacle removal," in *Distributed Autonomous Robotic Systems: 15th International Symposium*. Springer, 2022, pp. 202–213.

[5] C. D. Hsu, H. Jeong, G. J. Pappas, and P. Chaudhari, "Scalable reinforcement learning policies for multi-agent control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 4785–4791.

[6] K. Shibata, T. Jimbo, T. Odashima, K. Takeshita, and T. Matsubara, "Learning locally, communicating globally: Reinforcement learning of multi-robot task allocation for cooperative transport," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 11 436–11 443, 2023.

[7] L. Liu and D. A. Shell, "Assessing optimal assignment under uncertainty: An interval-based algorithm," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 936–953, 2011.

[8] L. Sabattini, V. Digani, C. Secchi, and C. Fantuzzi, "Optimized simultaneous conflict-free task assignment and path planning for multi-agv systems," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1083–1088.

[9] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, 2017, pp. 6382–6393.

[10] K. Shibata, T. Jimbo, and T. Matsubara, "Deep reinforcement learning of event-triggered communication and control for multi-agent cooperative transport," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8671–8677.

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

[12] W. B. Powell, *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons, 2007, vol. 703.

[13] O. Nachum, S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18, 2018, pp. 3307–3317.

[14] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 611–24 624, 2022.

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

[16] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[17] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, *et al.*, "Dextreme: Transfer of agile in-hand manipulation from simulation to reality," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5977–5984.

[18] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance GPU based physics simulation for robot learning," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.