Acquisition of Cooperative Behavior in Multi-Agent Path Finding

Haruto Sugawara[1] and Hiroyuki Toda[1][0000-0003-4883-527X]

[1]Graduate School of Data Science, Yokohama City University
y255614c,toda.hir.xg@yokohama-cu.ac.jp

Abstract

The Multi-Agent Path Finding (MAPF) problem involves planning collision-free paths for multiple agents traversing from initial to designated positions. Reinforcement learning-based approaches have recently gained attention, demonstrating effective path planning in complex environments under decentralized control. However, these methods encounter a fundamental limitation: individual reward maximization by each agent results in inter-agent interference, degrading performance. This research addresses reward design in reinforcement learning-based MAPF to facilitate cooperative behavior. Our key idea is to incorporate other agents' reward influence into individual reward functions and systematically modulate this influence to enhance cooperation acquisition. Through comparative evaluation against existing methodologies, we demonstrate our approach achieves improved performance.

1. Introduction

Recent advances in robotics have facilitated the widespread deployment of autonomous robots in warehouse automation (Zhang et al. 2023), disaster search and rescue operations (Drew 2021), and aircraft-towing systems (Morris et al. 2016). These applications often require multiple robots operating simultaneously, necessitating sophisticated coordination mechanisms to ensure efficient and safe operation. Such multi-robot coordination problems can be formulated as Multi-Agent Path Finding (MAPF), which determines collision-free paths for multiple agents moving from initial positions to designated destinations while minimizing movement costs. This framework offers practical benefits: reduced operational time in warehouse logistics, rapid deployment in disaster zones, and efficient coordination in airport operations.

MAPF problems are typically represented in grid environments where each cell contains free space, obstacles, or robots, with agents moving to adjacent cells or remaining stationary at each time step (Figure 1). Existing practical MAPF systems commonly employ centralized control (Sharon et al. 2015; Silver 2005), optimizing the system by aggregating information from all agents (Varambally, Li, and Koenig 2022). This centralized approach utilizes global environmental information to derive theoretically optimal solutions.

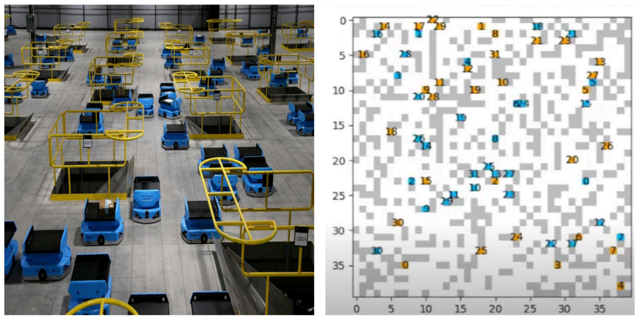

Figure 1: Multiple robots navigating with path planning (left)[1] can be represented as a grid-based Multi-Agent Path Finding problem (right).

However, centralized control faces critical real-world limitations. First, computational scalability becomes problematic as the exponentially growing state space makes real-time processing impractical with increasing agent numbers. Second, the system exhibits brittleness to environmental changes: even localized modifications such as a single obstacle appearing or one agent encountering an unexpected delay require complete path recalculation for all agents, creating substantial computational overhead. Third, the demanding communication infrastructure requirements, where all agents must maintain constant connectivity with the central controller, hinder deployment in environments with unstable or limited network availability(Zhang et al. 2024).

To address these challenges, distributed approaches have recently gained significant attention as promising alternatives to centralized control. These distributed methods offer substantial advantages that address the aforementioned limitations: scalable performance through parallel processing that avoids exponential computational growth, rapid responses to environmental changes without requiring system-wide recalculation, and inherent fault tolerance to communication failures through autonomous decision-making.

Among these distributed approaches, reinforcement learning-based methods have emerged as particularly promising solutions for MAPF (Chung et al. 2024)(Alkazzi and Okumura 2024). Reinforcement learning enables agents

[1]Image source: https://www.bostonglobe.com/2019/12/30/nation/robots-take-over-warehousing-workers-pushed-adapt/

to learn policies through trial-and-error with their environments, making it well-suited for decentralized coordination where agents must adapt to dynamic conditions and other agents' actions. In MAPF, reward functions are designed to reflect problem requirements by providing positive rewards for reaching goals and negative rewards for collisions.

Representative methods in this category include DHC (Distributed Heuristic multi-agent path finding with Communication) (Ma, Luo, and Ma 2021) and DCC (Decision Causal Communication) (Ma, Luo, and Pan 2021) , which enable agents to learn independently based on individual rewards. While these methods offer scalability and simple system structures, they suffer from a fundamental coordination challenge: agents may act selfishly, leading to interference, congestion, or deadlock in complex environments (Song et al. 2024; He et al. 2024).

This research focuses on reinforcement learning-based methods for distributed MAPF and addresses the acquisition of cooperative behavior through reward and curriculum-based training. Specifically, in distributed optimization, this research extends conventional individual reward functions to incorporate the influence of neighboring agents' actions to promote cooperation. To handle the resulting training complexity, we combine this reward design with curriculum learning (Bengio et al. 2009).

The main contributions of this research are as follows:

- We propose a method to promote cooperative behavior in MAPF through the combination of reward design and curriculum learning. This approach effectively addresses the complexity introduced by considering other agents' rewards while maintaining the framework of existing research without significant computational overhead.
- Our evaluation shows that the proposed method achieves superior path planning performance, surpassing existing research across all evaluated scenarios in success rates.

The structure of this research is as follows. Section 2 presents related work and clarifies the positioning of this research. Section 3 defines the MAPF problem and environment. Section 4 explains our proposed method. Section 5 presents experimental results and discussion. Section 6 provides conclusions and future work.

2. Related works

Distributed Reinforcement Learning for MAPF

Multi-Agent Path Finding (MAPF) has been extensively studied as a distributed reinforcement learning problem, where multiple agents must coordinate to reach designated goals while avoiding collisions. While centralized control can theoretically achieve globally optimal solutions, it suffers from critical limitations: exponentially increasing computational costs, limited adaptability to environmental changes, and dependency on communication infrastructure.

In contrast, distributed reinforcement learning addresses these challenges by enabling agents to learn policies independently through local observations and trial-and-error interactions. To facilitate coordination in such distributed settings, existing research has explored two primary approaches: (1) providing heuristic path information to guide agents toward their destinations, and (2) introducing lightweight communication protocols for sharing limited state or action information among neighboring agents.

Representative methods combining both strategies include DHC (Distributed Heuristic multi-agent path finding with Communication) (Ma, Luo, and Ma 2021) and DCC(Decision Causal Communication) (Ma, Luo, and Pan 2021), both based on independent learning mechanisms. In these frameworks, each agent optimizes its own policy based on individual rewards and treats other agents as part of the environment, achieving scalability while maintaining simple system structures.

These frameworks allow parallel and independent learning among agents, effectively mitigating the computational explosion of centralized planning. However, they face a fundamental limitation: each agent primarily maximizes its own reward, which can lead to interference, congestion, or even deadlocks in dense environments (Song et al. 2024; He et al. 2024). Although heuristic and communication mechanisms partially stabilize the learning process, they do not fully resolve the non-stationarity arising from concurrent policy updates.

These findings indicate that while distributed reinforcement learning frameworks are scalable and efficient, explicit mechanisms to promote cooperative behavior are required to prevent performance degradation due to selfish learning.

Cooperative Behavior through Reward Design

In multi-agent reinforcement learning, research has explored cooperative behavior acquisition through reward function design. Peng et al. proposed incorporating neighboring agents' average reward into individual reward functions in traffic simulation (CoPO) (Peng et al. 2021), improving system performance and robustness by making each agent's reward sensitive to surrounding agents' outcomes.

However, this approach faces convergence challenges when agent rewards depend on other agents' actions. To address this, Song et al. proposed considering potential rewards from optimal actions other agents could take, rather than actions actually taken(CoRS) (Song et al. 2024).

Research on Curriculum Learning

Curriculum learning (Bengio et al. 2009) is a methodology that optimizes the learning content of tasks in a stepwise manner. It aims to improve learning speed and performance by applying the human learning process of starting with simple tasks and gradually increasing difficulty to machine learning. Particularly in reinforcement learning, curriculum learning is actively utilized because stepwise learning is effective for problems with large and complex search spaces (Narvekar et al. 2020).

The application of curriculum learning in MAPF has primarily focused on task scale expansion. Methods such as DHC (Ma, Luo, and Ma 2021), DCC (Ma, Luo, and Pan 2021), MAPPER (Liu et al. 2020), and SACHA (Lin and Ma 2023) begin training in small environments and progressively increase map size and the number of agents as performance thresholds are achieved.

Recently, studies have explored curriculum learning from perspectives beyond task scale. Phan et al. proposed progressively expanding the goal assignment radius to facilitate long-distance movement learning (Phan et al. 2024). Zhao et al. introduced a three-stage curriculum that transitions from single-agent to cooperative pathfinding by gradually shifting the reward function from individual to team-based optimization (Zhao et al. 2023). While this approach effectively promotes cooperation, it requires centralized control and global system knowledge, limiting scalability.

These approaches highlight the potential of curriculum learning to promote cooperation, yet scalability under decentralized settings remains an open challenge.

3. Preliminaries

Problem Formulation

This paper addresses a partially observable variant of Multi-Agent Path Finding (MAPF), where agents have only partial observation of the environment but aim to fully cooperate to minimize the average completion time.We model this as a decentralized partially observable Markov Decision Process (Dec-POMDP) (Littman 1994), defined as a 7-tuple $(\mathcal{S}, \mathcal{A}_i, P, \Omega_i, O, R, \gamma)$:

- $\mathcal{S}$: the set of global states
- $\mathcal{A}_i$: the set of actions for agent i, with joint action space $\mathcal{A} = \prod_{i=1}^{n} \mathcal{A}_i$
- Ω_i: the observation space of agent i, with joint observation space $\Omega = \prod_{i=1}^{n} \Omega_i$
- $P : \mathcal{A} \times \mathcal{S} \to \mathcal{S}$: the state-transition function, $P(s'|a, s)$
- $O : \mathcal{A} \times \mathcal{S} \to \Omega$: the observation function, $P(o|a, s)$
- $R : \mathcal{S} \times \mathcal{A} \to \mathbf{R}$: the reward function
- $\gamma \in [0, 1]$: the discount factor

Task Setting Given an undirected graph $G = (V, E)$ and n agents, each agent $i \in \{1, 2, \ldots, n\}$ is assigned a unique start location $s_i \in V$ and goal location $g_i \in V$. At each discrete time step, agents can move to adjacent vertices or remain stationary. Collisions occur when agents occupy the same vertex simultaneously (vertex conflict) or traverse the same edge in opposite directions (edge conflict).

Our simulation uses a discrete 2D grid of size $m \times m$, where agents execute five actions at each time step: move up, down, right, left, or stay. Episodes terminate when all agents reach their goals, or upon collision or timeout.

Each agent has partial observability through a Field-Of-View (FOV) of size $l \times l$ $(l < m)$ centered on itself (Figure 2), corresponding to Ω_i in the Dec-POMDP formulation. Within its FOV, agent i observes positions of other agents, obstacles, and goals, forming local observation $o_i \in \Omega_i$.

The objective is to find a set of conflict-free paths that minimize the average completion time across all agents, where each agent learns a policy $\pi_i : \Omega_i \to \mathcal{A}_i$ to maximize expected cumulative discounted reward $\mathbf{E}[\sum_{t=0}^{\infty} \gamma^t r_t^i]$.

Reinforcement Learning Framework

We employ Deep Q-Network (DQN) (Mnih et al. 2015) in an Independent Q-Learning (IQL) setting, where each agent independently learns its action-value function while treating other agents as part of the environment.

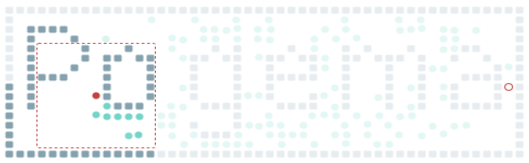

Figure 2: Partial observation with FOV. Agents (red dots) observe information within their FOV (red frame). Gray dots represent obstacles (dark) and other agents (light). Image cited from[2].

At each time step t, agent i observes o_t^i, selects action a_t^i, and maximizes cumulative discounted reward $R_t^i = \sum_{k=0}^{\infty} \gamma^k r_{t+k}^i$. DQN approximates the action-value function $Q^{\pi_i}(o^i, a^i)$ using a neural network $Q(o^i, a^i; \theta_i)$, trained by minimizing:

$$L(\theta_i) = \mathbf{E}_{o^i,a^i,r^i,o'^i}[(Q(o^i, a^i; \theta_i) - y^i)^2] \quad (1)$$

where $y^i = r^i + \gamma \max_{a'^i} Q(o'^i, a'^i; \bar{\theta}_i)$ and $\bar{\theta}_i$ denotes target network parameters updated periodically for stability.

Baseline Framework: DHC & DCC

DHC (Ma, Luo, and Ma 2021) and DCC (Ma, Luo, and Pan 2021) are representative IQL frameworks for distributed MAPF, achieving scalability through independent policy optimization with inter-agent communication. Both share a three-module architecture: observation encoder, attention-based communication block, and dueling Q-network. DCC differs from DHC primarily in its more sophisticated communication mechanism, albeit with increased computational overhead.

Both frameworks use the same reward function design: agents receive a reward of $+3.0$ for reaching their goal, a penalty of -0.5 for collisions, and a small penalty of -0.075 for each movement action or staying at non-goal positions to encourage efficiency. Staying at the goal incurs no penalty. This design aligns with task objectives while providing dense feedback for learning convergence.

Training in both frameworks employs a curriculum learning that progressively increases map size and agent count as learning stabilizes. Starting with a single agent in a 10×10 grid, the curriculum adds agents and expands the grid by 5 cells when success rate exceeds 0.9 over the most recent 200 episodes. DHC's curriculum ultimately reaches 12 agents in a 40×40 grid, while DCC extends to 16 agents in the same grid size. Model parameters are inherited between curriculum stages.

In this study, we adopt DHC and DCC as baseline frameworks and extend their learning strategy to enable the acquisition of cooperative behaviors through reward design and curriculum learning.Detailed network architectures and hyperparameter settings for both baseline methods are provided in the Appendix.

[2]Image :https://github.com/CognitiveAISystems/pogema

4. Proposed Method

Overview and Motivation

DHC and DCC achieve scalability through Independent Q-Learning (IQL), where each agent independently maximizes its own reward. While both frameworks incorporate heuristic guidance and communication mechanisms to partially stabilize learning, agents remain focused on individual rewards. This leads to selfish behavior causing collisions and deadlocks (Song et al. 2024; He et al. 2024), as concurrent policy updates create non-stationary environments.

To address this, we propose a framework combining reward design and curriculum learning (Figure 3). We incorporate neighbors' rewards to promote cooperation and use a two-stage curriculum to manage training complexity. Our approach extends the learning strategy of these IQL frameworks while maintaining their architectural simplicity and computational efficiency, requiring no modifications to the neural network structure or communication mechanisms.

Reward Design for Cooperative Behavior

To promote cooperation, we incorporate neighbors' rewards into each agent's reward function:

$$R_t^i = (1-\alpha) r_t^i + \alpha \frac{1}{|A^{-i}|} \sum_{j \in A^{-i}} r_t^j \tag{2}$$

where r_t^i is agent i's individual reward at time t, A^{-i} denotes neighboring agents within its FOV, and $\alpha \in [0, 1]$ is the cooperation coefficient controlling the balance between self-interest and cooperation. When $\alpha = 0$ the agent behaves completely selfishly as in standard DHC and DCC, while $\alpha = 1.0$ represents complete altruism.

The intuition behind this design is straightforward. High average neighbor rewards indicate that agent i's action did not harm others and suggest cooperative behavior. By maximizing R_t^i, agents learn to balance self-interest with social impact. The averaging mechanism captures the net social effect across multiple neighbors while maintaining $O(|A^{-i}|)$ computational efficiency.

This reward design draws inspiration from CoPO(Peng et al. 2021), which successfully improved traffic flow by incorporating neighboring agents' average rewards. However, direct application of such approaches to MAPF faces a critical challenge: non-stationarity. Since neighbor rewards depend on their evolving policies, the second term in Eq. (2) changes as other agents learn, making Q-value convergence difficult especially when α is large (Song et al. 2024).

To address this convergence challenge, an alternative approach, CoRS(Song et al. 2024), addresses this by considering counterfactual rewards—potential rewards from optimal actions other agents could take rather than actions actually taken. While this reduces dependency on other agents' actual policies and improves convergence, it requires additional computation to estimate counterfactual actions by exploring other agents' action spaces at each training step. This computational overhead becomes significant in large-scale MAPF scenarios with numerous agents.

Given the need to balance cooperation promotion and computational efficiency, we adopt the simpler averaging approach of Eq. (2) and address the convergence challenge through curriculum learning rather than counterfactual reasoning. This pragmatic choice maintains the computational scalability essential for distributed MAPF while still enabling cooperative behavior acquisition.

Toward Stable Cooperative Learning: IGM

The Individual-Global-Max (IGM) condition (Son et al. 2019) provides theoretical insight into the convergence challenge of our reward design. Let $Q^{tot}(s, \mathbf{a})$ denote the global Q-function and $Q^i(o^i, a^i)$ the individual Q-function. IGM is satisfied when:

$$\arg\max_{\mathbf{a}} Q^{tot}(s, \mathbf{a}) = \Big(\arg\max_{a^1} Q^1(o^1, a^1), \dots, \arg\max_{a^n} Q^n(o^n, a^n) \Big) \tag{3}$$

When IGM holds, each agent's individual reward maximization automatically leads to global optimality—an ideal property for distributed systems. Our reward design (Eq. (2)) aims to encourage such IGM-like cooperative behavior. By incorporating neighbors' rewards, agents learn that actions benefiting others (high $\sum_{j \in A^{-i}} r_t^j$) often lead to better overall outcomes, gradually aligning individual and collective objectives.

However, achieving strict IGM guarantees in practice is extremely challenging due to partial observability, non-stationary environments, and the difficulty of designing provably IGM-satisfying reward functions.

Rather than pursuing computationally expensive theoretical guarantees or centralized training, we take a pragmatic approach through curriculum learning that progressively guides agents toward cooperative behavior while managing the non-stationarity introduced by Eq. (2). This approach maintains the fully decentralized learning paradigm essential for scalability in large-scale MAPF.

Two-Stage Curriculum Learning

To address the non-stationarity challenge in our reward design, we propose a two-stage curriculum that progressively introduces cooperation (Algorithm1). This design is motivated by the observation that directly training with high cooperation coefficients (α) causes learning instability due to the rapidly changing reward landscape as all agents simultaneously update their policies.

Stage 1: Scale Expansion Curriculum. Following the original training procedures of DHC and DCC, we first employ a scale expansion curriculum with individual rewards ($\alpha = 0$) that gradually increases agent numbers and grid size. Training begins with a single agent in a 10×10 grid. When the success rate in the most recent 200 episodes exceeds 0.9, the curriculum adds one agent and expands the grid by 5 cells. This process repeats until reaching the final task configuration: 12 agents in a 40×40 grid for DHC and 16 agents in a 40×40 grid for DCC, following their respective original settings. We adopt these original configurations to maintain consistency with the baseline frameworks and isolate the effect of Stage 2. This stage establishes basic pathfinding skills in a stationary environment with $\alpha = 0$.

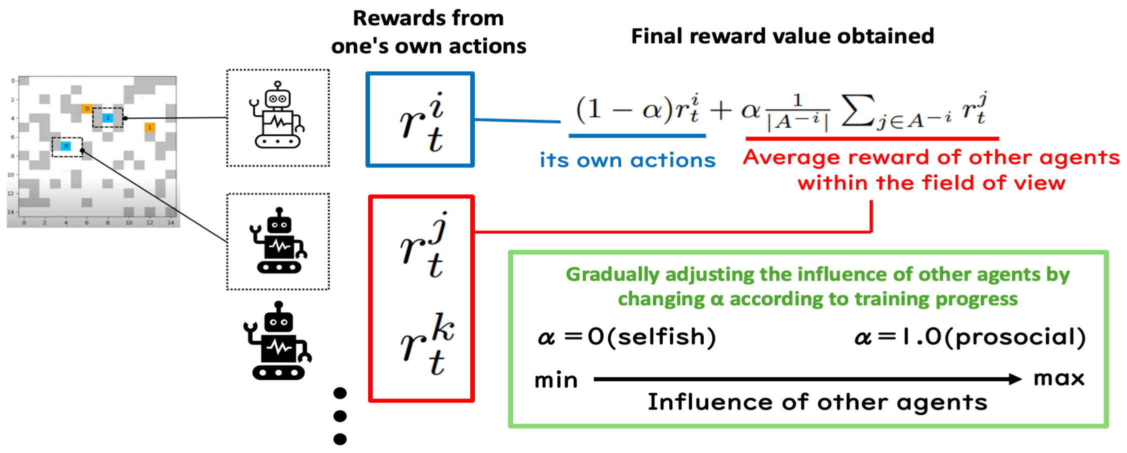
Rewards from one's own actions
Final reward value obtained
r_t^i
$(1-\alpha)r_t^i + \alpha\frac{1}{|A^{-i}|}\sum_{j\in A^{-i}} r_t^j$
its own actions
Average reward of other agents within the field of view
r_t^j
r_t^k
Gradually adjusting the influence of other agents by changing α according to training progress
α = 0(selfish)
α = 1.0(prosocial)
min
max
Influence of other agents

Figure 3: Proposed method

Algorithm 1: Two-Stage Curriculum Learning

```
 1: Stage 1: Scale Expansion Curriculum
 2: Initialize: 1 agent in $10 \times 10$ grid, $\alpha = 0$
 3: while not final task do
 4:     {Final task: DHC (12 agents, $40 \times 40$), DCC (16 agents, $40 \times 40$)}
 5:     Train agents with current configuration
 6:     Evaluate success rate on recent 200 episodes
 7:     if success rate $> 0.9$ then
 8:         Add 1 agent and expand grid by 5
 9:         Inherit model parameters
10:     end if
11: end while
12: Stage 2: Cooperative Exploration Curriculum
13: Initialize $\alpha = 0$, load model from Stage 1
14: while $\alpha \leq 1.0$ do
15:     Train agents with current $\alpha$ using Eq. (2)
16:     Evaluate success rate on validation tasks
17:     if success rate $> 0.9$ then
18:         Save model; $\alpha \leftarrow \alpha + 0.1$
19:     else if converged without threshold then
20:         return $\alpha - 0.1$
21:     end if
22: end while
23: return $\alpha = 1.0$
```

Stage 2: Cooperative Exploration Curriculum. After completing the scale expansion curriculum, we progressively introduce cooperation by increasing α from 0 in 0.1 increments. We choose this increment size to balance fine-grained exploration of cooperation levels with training efficiency. At each α stage, we train in the final environment configuration (12 agents in 40×40 grid for DHC, 16 agents in 40×40 grid for DCC) until the success rate exceeds 0.9 over the most recent 200 episodes, then advance to the next α value. If training converges without reaching the threshold,we terminate and return the previous α.

This progressive approach serves multiple purposes. First, it allows agents to adapt gradually to increasing non-stationarity. As α increases, the influence of neighbors' evolving policies on each agent's reward signal grows, making the learning environment more dynamic. By incrementing α slowly, we ensure agents have sufficient time to adjust their policies at each level before facing greater non-stationarity.

Second, agents incrementally learn that considering neighbors' outcomes (the second term in Eq. (2)) improves overall performance. At low α values, agents maintain primarily self-interested behavior with slight awareness of neighbors, gradually developing cooperative strategies such as yielding in narrow corridors or coordinating to avoid congestion. As α increases, these cooperative behaviors become more pronounced and refined.

Third, the curriculum automatically identifies the optimal cooperation level α^* for each environment—the maximum α at which agents can successfully learn despite non-stationarity. By incrementing α until convergence fails, we efficiently identify this boundary of stable learning, which varies depending on environmental complexity, agent density, and obstacle configuration.

Compared to alternative curriculum strategies, such as Zhao et al.'s three-stage approach (Zhao et al. 2023) that transitions from individual to team-based rewards, our method maintains the decentralized learning paradigm throughout training. While Zhao et al.'s approach effectively promotes cooperation, it requires centralized control and global system knowledge for team reward computation, limiting scalability. Our method achieves similar cooperative behavior acquisition while preserving the computational efficiency and fault tolerance of fully distributed learning.

5. Experiments

All experiments were conducted on a server with an Intel(R) Xeon(R) w5-3435X CPU and an NVIDIA RTX 6000 Ada Generation GPU. We employed the Ape-X (Horgan et al. 2018) distributed learning framework with 16 parallel actors for experience generation.

Table 1: Performance Comparison of DHC and DCC Enhanced by the Proposed Method on 80 $\times$ 80 Maps

Success Rate ↑

Method/Agents		**4**	**8**	**16**	**32**	**64**	**128**
DHC	Baseline	0.990	0.980	0.970	0.880	0.720	0.420
	w/ α=0.1	**1.000**	**0.990**	0.950	0.845	0.700	0.350
	w/ α=0.2	**1.000**	**1.000**	0.960	**0.885**	**0.780**	**0.460**
	w/ α=0.3	**1.000**	**1.000**	**0.990**	**0.910**	**0.825**	**0.615**
	w/ α=0.4	**1.000**	**0.995**	0.945	0.850	0.710	0.405
DCC	Baseline	1.000	0.990	0.975	0.960	0.890	0.820
	w/ α=0.1	**1.000**	**0.995**	**0.995**	0.950	**0.900**	0.800
	w/ α=0.2	**1.000**	**1.000**	**1.000**	**0.970**	**0.930**	**0.865**
	w/ α=0.3	**1.000**	0.985	0.935	0.845	0.785	0.635

Average Time Steps ↓

Method/Agents		**4**	**8**	**16**	**32**	**64**	**128**
DHC	Baseline	96.72	109.24	122.54	138.32	163.5	213.15
	w/ α=0.1	**90.21**	**103.55**	124.81	140.9	159.79	215.67
	w/ α=0.2	**88.27**	**95.31**	**119.08**	142.58	**157.29**	**207.85**
	w/ α=0.3	**89.20**	**97.35**	**116.61**	138.84	**153.44**	**204.03**
	w/ α=0.4	**94.64**	110.22	**120.80**	**134.79**	165.22	217.99
DCC	Baseline	93.89	109.89	122.24	132.99	159.67	192.9
	w/ α=0.1	96.32	**107.34**	**119.45**	135.18	161.86	198.29
	w/ α=0.2	**94.10**	113.76	**115.11**	136.56	**154.39**	**190.06**
	w/ α=0.3	**93.95**	117.44	127.37	139.07	162.24	210.74

Experimental Settings

Following the standard MAPF benchmark (Stern et al. 2019) and the evaluation protocols of DHC (Ma, Luo, and Ma 2021) and DCC (Ma, Luo, and Pan 2021), our experiments consist of training and testing phases. Environmental settings include 9 $\times$ 9 agent FOV. All remaining training hyperparameters and implementation-specific details are summarized in the Appendix for reproducibility.

Training implements the combined scale expansion and cooperative exploration curricula proposed in Section 4. The success rate threshold for the cooperative exploration curriculum is set to 0.9, and the number of recent samples for calculating success rate is set to 200. For testing, we use randomly generated maps of two sizes: 40 $\times$ 40 and 80 $\times$ 80 grids with obstacle density of 0.3.

Evaluation covers 11 scenarios with agents (4, 8, 16, 32, 64, 128) across grid sizes (40 $\times$ 40, 80 $\times$ 80), with 128-agent scenarios only on 80 $\times$ 80 maps due to spatial constraints. Each scenario runs 200 tests with randomly generated initial positions, goals, and obstacles. Time limits are 256 steps (40 $\times$ 40) and 386 steps (80 $\times$ 80), with 0.3 obstacle occupancy. Success requires all agents reaching goals within time limits; failure occurs from collisions or timeout.

Following the evaluation metrics used in DHC and DCC, we measure performance using two indicators:

- **Success Rate**: the percentage of episodes where all agents reach goals within time limits
- **Average Time Steps**: the mean steps to task completion (using time limit for failures)

Results and Discussion

Similar trends were observed for both 40 $\times$ 40 and 80 $\times$ 80 maps, so we present experimental results on 80 $\times$ 80 maps. In the cooperative exploration curriculum experiments, training terminated when the cooperation coefficient reached 0.5 for DHC and 0.4 for DCC, as the success rate did not exceed the threshold of 0.9. This limitation arises because excessively high cooperation coefficients ($\alpha \geq 0.5$) cause the influence of neighbors' evolving policies to dominate individual rewards, hindering convergence (detailed analysis provided in the Appendix). Therefore, we obtained variants of the proposed method with cooperation coefficients α of 0.1, 0.2, 0.3, and 0.4 for DHC, and 0.1, 0.2, and 0.3 for DCC.

Tables 1 compare our proposed method against the DHC and DCC baselines, respectively, showing success rates and average time steps across different numbers of agents and cooperation coefficients.

For DHC, Proposed α=0.3 consistently outperformed the baseline across all tested scenarios in success rates. For DCC, Proposed α=0.2 demonstrated similar superiority, surpassing the baseline in all scenarios. These results demonstrate the effectiveness of our cooperative reward design in distributed independent learning frameworks.

However, not all cooperation coefficients yielded consistent improvements. While some α values showed improvements in certain scenarios, they also exhibited performance below the baseline in others. This suggests the existence of an environment-dependent optimal cooperation coefficient, with performance deteriorating as the value deviates from this optimum. Our results indicate that cooperation coefficients around α=0.2–0.3 appear to be favorable for the tested scenarios, though further investigation across diverse environments would be needed to establish more generalized optimal values. Regarding average time steps, partial improvements were observed across different cooperation coefficients in both frameworks, though gains were modest compared to the improvements in success rates.

Qualitative observations from simulation runs reveal several important behavioral patterns. First, the proposed method largely eliminated failures in narrow corridor passing, where encounters typically involve one-to-one situations. This demonstrates that the cooperative reward design effectively promotes coordination in relatively simple interaction scenarios. However, the method still exhibited limitations in environments with highly concentrated agents, particularly in scenarios where many agents form dense clusters or chains. This suggests that while our approach improves pairwise coordination, fundamental challenges in multi-agent coordination remain, especially in situations requiring complex collective decision-making among numerous simultaneously interacting agents.

Additionally, agents tended to maintain excessive distance when passing each other, indicating potentially circuitous routes. This overly cautious collision avoidance behavior, inadvertently promoted by incorporating other agents' rewards, could explain why time step improvements remained modest despite substantial success rate gains.

Table 2: Ablation Study: Effect of Cooperative Exploration Curriculum on DHC and DCC (80×80 Maps)

Success Rate ↑

	α	**Method**	**4**	**8**	**16**	**32**	**64**	**128**
DHC	0.2	w/o CL	0.960	0.965	0.925	0.850	0.745	0.415
		Proposed	**1.000**	**1.000**	**0.960**	**0.885**	**0.780**	**0.460**
	0.3	w/o CL	1.000	1.000	0.980	**0.925**	0.810	0.535
		Proposed	**1.000**	**1.000**	**0.990**	0.910	**0.825**	**0.615**
	0.4	w/o CL	0.980	**1.000**	**0.960**	**0.900**	0.650	**0.420**
		Proposed	**1.000**	0.995	0.945	0.850	**0.710**	0.405
DCC	0.2	w/o CL	**1.000**	**1.000**	**1.000**	0.935	0.875	0.835
		Proposed	**1.000**	**1.000**	**1.000**	**0.970**	**0.930**	**0.865**
	0.3	w/o CL	**1.000**	**1.000**	0.910	**0.855**	0.610	0.550
		Proposed	**1.000**	0.985	**0.935**	0.845	**0.785**	**0.635**

Average Time Steps ↓

	α	**Method**	**4**	**8**	**16**	**32**	**64**	**128**
DHC	0.2	w/o CL	99.18	113.55	121.76	**139.83**	162.98	219.64
		Proposed	**88.27**	**95.31**	**119.08**	142.58	**157.29**	**207.85**
	0.3	w/o CL	93.89	109.89	122.61	**129.01**	158.92	210.45
		Proposed	**89.20**	**97.35**	**116.61**	138.84	**153.44**	**204.03**
	0.4	w/o CL	102.76	**105.48**	128.92	137.24	189.11	**214.33**
		Proposed	**94.64**	110.22	**120.80**	**134.79**	**165.22**	217.99
DCC	0.2	w/o CL	**93.03**	114.72	**114.69**	140.24	170.23	200.51
		Proposed	94.10	**113.76**	115.11	**136.56**	**154.39**	**190.06**
	0.3	w/o CL	**92.46**	**116.34**	129.75	146.86	180.24	215.61
		Proposed	93.95	117.44	**127.37**	**139.07**	**162.24**	**210.74**

Ablation Study: Effectiveness of Curriculum Learning Table 2 presents ablation studies comparing the proposed method with variants trained without the Cooperative Exploration Curriculum (w/o CL) for DHC and DCC. In the w/o CL variants, fixed cooperation coefficients are applied directly after completing the Scale Expansion Curriculum, without the gradual progression from α=0. Note that α=0.1 results are not available for w/o CL, as this approach does not involve the incremental α transitions of Stage 2.

The experimental results demonstrate clear improvement trends through curriculum learning for both frameworks across different cooperation coefficients. The proposed method consistently outperforms the w/o CL variants across most metrics and agent densities, with performance gaps particularly pronounced in high-density environments. Improvements observed in both success rates and average time steps suggest that curriculum learning enables more stable and efficient path planning by reducing conflicts during the learning process.

These findings highlight the effectiveness of progressively introducing cooperation coefficients in multi-agent reinforcement learning systems. Direct training with fixed coefficients causes learning difficulties due to rapidly changing reward landscapes, as the second term in Eq. (2) introduces non-stationarity through neighbors' evolving policies. The Cooperative Exploration Curriculum addresses this fundamental challenge by facilitating gradual adaptation to cooperative behaviors, allowing agents to progressively adjust to the non-stationarity introduced by considering other agents' rewards.

The curriculum's effectiveness across tested cooperation coefficient ranges for both frameworks confirms that this approach is particularly beneficial for stable learning in distributed independent learning settings where agents optimize policies autonomously. The ablation study thus validates that combining reward design with curriculum learning is crucial for successfully acquiring cooperative behavior in distributed MAPF scenarios. As a supplementary note, our preliminary experiments indicated that excessively high cooperation coefficients (exceeding 0.5 for DHC and 0.4 for DCC) caused performance degradation, likely due to convergence difficulties from excessive dependence on inter-agent rewards, further supporting the importance of maintaining balanced cooperation intensity throughout the training process.

6. Conclusion

This research proposes a method for acquiring cooperative behavior in distributed MAPF through reward design and curriculum learning. We introduce a cooperation coefficient into reward functions and construct a two-stage curriculum combining Scale Expansion and Cooperative Exploration.

We adopt DHC and DCC as baselines to demonstrate effectiveness. Validation shows significant performance improvements with cooperation coefficients around 0.2–0.3, particularly in high-density environments. Comparison with w/o CL variants confirms that progressive cooperation yields more stable learning than fixed coefficients.

The main contribution is demonstrating that cooperative behavior can be acquired through simple modifications—reward design and curriculum learning—without complex architectural changes or computational overhead. Consistent improvements across DHC and DCC indicate our approach is effective for independent learning frameworks. This provides a practical and scalable solution for large-scale MAPF applications.

Several promising directions remain for future work. First, applying our cooperative reward design to other distributed MAPF methods such as SACHA(Lin and Ma 2023),PICO(Li et al. 2022) would validate the generalizability of our approach across diverse architectural paradigms and learning strategies. Second, time efficiency could be improved through refined reward designs that better balance cooperation and goal-reaching speed, as our current approach introduces computational overhead and potentially overly cautious collision avoidance behaviors. Third, investigating dynamic cooperation coefficient adjustment mechanisms could enhance adaptability. Currently, we apply a uniform α to all agents, but allowing individual agents to dynamically adjust their cooperation levels based on local conditions could enable more context-sensitive strategies. Finally, exploring finer curriculum increments may help bridge performance gaps with centralized methods while maintaining computational simplicity. These directions would further strengthen the practical applicability of cooperative learning in distributed multi-agent systems.

Acknowledgments

This work was supported in part by JSPS KAKENHI Grant Number JP24K03046.

References

Alkazzi, J.-M.; and Okumura, K. 2024. A Comprehensive Review on Leveraging Machine Learning for Multi-Agent Path Finding. *IEEE Access*.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48.

Chung, J.; Fayyad, J.; Younes, Y. A.; and Najjaran, H. 2024. Learning team-based navigation: a review of deep reinforcement learning techniques for multi-agent pathfinding. *Artificial Intelligence Review*, 57(2): 41.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Drew, D. S. 2021. Multi-agent systems for search and rescue applications. *Current Robotics Reports*, 2: 189–200.

He, C.; Duhan, T.; Tulsyan, P.; Kim, P.; and Sartoretti, G. 2024. Social behavior as a key to learning-based multi-agent pathfinding dilemmas. *arXiv preprint arXiv:2408.03063*.

Horgan, D.; Quan, J.; Budden, D.; Barth-Maron, G.; Hessel, M.; Van Hasselt, H.; and Silver, D. 2018. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*.

Li, W.; Chen, H.; Jin, B.; Tan, W.; Zha, H.; and Wang, X. 2022. Multi-agent path finding with prioritized communication learning. In *Proc. IEEE ICRA*, 10695–10701.

Lin, Q.; and Ma, H. 2023. SACHA: Soft actor-critic with heuristic-based attention for partially observable multi-agent path finding. *IEEE Robotics and Automation Letters*.

Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, 157–163. Elsevier.

Liu, Z.; Chen, B.; Zhou, H.; Koushik, G.; Hebert, M.; and Zhao, D. 2020. Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments. In *Proc. IEEE/RSJ IROS*, 11748–11754.

Ma, Z.; Luo, Y.; and Ma, H. 2021. Distributed heuristic multi-agent path finding with communication. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 8699–8705. IEEE.

Ma, Z.; Luo, Y.; and Pan, J. 2021. Learning selective communication for multi-agent path finding. *IEEE Robotics and Automation Letters*, 7(2): 1455–1462.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Morris, R.; Pasareanu, C. S.; Luckow, K. S.; Malik, W.; Ma, H.; Kumar, T. S.; and Koenig, S. 2016. Planning, Scheduling and Monitoring for Airport Surface Operations. In *AAAI Workshop: Planning for Hybrid Systems*, 608–614.

Narvekar, S.; Peng, B.; Leonetti, M.; Sinapov, J.; Taylor, M. E.; and Stone, P. 2020. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181): 1–50.

Peng, Z.; Li, Q.; Hui, K. M.; Liu, C.; and Zhou, B. 2021. Learning to simulate self-driven particles system with coordinated policy optimization. *Advances in Neural Information Processing Systems*, 34: 10784–10797.

Phan, T.; Driscoll, J.; Romberg, J.; and Koenig, S. 2024. Confidence-Based Curriculum Learning for Multi-Agent Path Finding. *arXiv preprint arXiv:2401.05860*.

Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. R. 2015. Conflict-based search for optimal multi-agent pathfinding. *Artificial intelligence*, 219: 40–66.

Silver, D. 2005. Cooperative pathfinding. In *Proc. AAAI Conf. Artif. Intell. Interact. Digit. Entertain.*, volume 1, 117–122.

Son, K.; Kim, D.; Kang, W. J.; Hostallero, D. E.; and Yi, Y. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, 5887–5896. PMLR.

Song, Z.; Zheng, R.; Zhang, S.; and Liu, M. 2024. Cooperative Reward Shaping for Multi-Agent Pathfinding. *arXiv preprint arXiv:2407.10403*.

Stern, R.; Sturtevant, N.; Felner, A.; Koenig, S.; Ma, H.; Walker, T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T.; et al. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search*, volume 10, 151–158.

Varambally, S.; Li, J.; and Koenig, S. 2022. Which MAPF model works best for automated warehousing? In *Proc. SoCS*, 190–198.

Vaswani, A. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Wang, B.; Liu, Z.; Li, Q.; and Prorok, A. 2020. Mobile robot path planning in dynamic environments through globally guided reinforcement learning. *IEEE Robotics and Automation Letters*, 5(4): 6932–6939.

Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; and Freitas, N. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, 1995–2003. PMLR.

Zhang, Y.; Fontaine, M. C.; Bhatt, V.; Nikolaidis, S.; and Li, J. 2023. Multi-Robot Coordination and Layout Design for Automated Warehousing. In *Proc. IJCAI*, 5503–5511.

Zhang, Y.; Zhao, W.; Wang, J.; and Yuan, Y. 2024. Recent progress, challenges and future prospects of applied deep reinforcement learning: A practical perspective in path planning. *Neurocomputing*, 608: 128423.

Zhao, C.; Zhuang, L.; Huang, Y.; and Liu, H. 2023. Curriculum Learning Based Multi-Agent Path Finding for Complex Environments. In *2023 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Appendix: Baseline Model Architectures

We detail the model architectures of our baseline methods: DHC and DCC. Both methods consist of three main modules: observation encoder, communication block, and Dueling Q-network(Wang et al. 2016).

Observation Encoder

The observation encoder consists of eight convolutional layers and a Gated Recurrent Unit (GRU)(Chung et al. 2014). Agent i receives a 9×9 FOV as a 6-channel tensor: (1) obstacles, (2) other agents, (3-6) heuristic guidance derived from a shortest path tree rooted at the goal, providing all valid movement options rather than a single path(Wang et al. 2020)(Liu et al. 2020). The processed information is integrated with the previous communication result $e_i''^{t-1}$ via GRU to generate output e_i^t.

Communication Block: DHC vs DCC

DHC adopts broadcast-style communication using Multi-Head Attention(Vaswani 2017). The output e_i^t is transformed into Query, Key, and Value for each head h:

$$q_i^h = W_Q^h e_i^t, \quad k_j^h = W_K^h e_j^t, \quad v_j^h = W_V^h e_j^t$$

Attention scores aggregate neighbor information:

$$\mu_{ij}^h = \text{softmax}\left(\frac{q_i^h \cdot (k_j^h)^T}{\sqrt{d_k}}\right), \quad \text{head}_i^h = \mu_{ii}^h v_i^h + \sum_{j \neq i} \mu_{ij}^h v_j^h$$

All head outputs are concatenated: $\hat{e}_i^t = f_o(\text{concat}(\text{head}_i^1, ..., \text{head}_i^H))$, then integrated by GRU to generate $e_i'^t$. This process repeats twice to obtain $e_i''^t$. DHC communicates with the two nearest agents.

DCC adds a Decision Causal Unit for selective communication. For each neighbor j: (1) generate modified observation $o_{i,-j}^t$ by masking agent j, (2) compute temporary actions $\tilde{a}_i^t$ and $\tilde{a}_{i,-j}^t$, (3) select only agents where $\tilde{a}_i^t \neq \tilde{a}_{i,-j}^t$. The communication scope is: $\mathcal{C}_i = \{j \mid \tilde{a}_i^t \neq \tilde{a}_{i,-j}^t, j \in \mathcal{B}_i\}$.

DCC employs request-reply two-round communication. Agent i sends message e_i^t and position l_i^t to agents $j \in \mathcal{C}_i$. Agents receiving requests update via Multi-Head Attention and reply with $e_j^{[1]t}$. Agent i reintegrates to generate $e_i^{[2]t}$.

DHC achieves lower computational cost with fixed communication scope (nearest two agents), while DCC incurs additional computation for decision causal selection but significantly reduces communication overhead through selective communication. DCC generally demonstrates superior success rates across scenarios, while DHC offers a favorable trade-off between performance and computational efficiency, particularly suitable when computational resources are limited.

Dueling Q-Network

The communication output is used to compute Q-values, which are decomposed into state value $V(s)$ and advantage $A(s,a)$:

$$Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a')\right)$$

The action with maximum Q-value is selected for execution.

Appendix: Hyperparameter Settings

Table 5 summarizes the hyperparameters used in our experiments. Unless otherwise specified, we followed the default hyperparameter settings provided in the official implementations of the original papers[1].

Table 3: Hyperparameters Used in Our Experiments

Hyperparameter	DHC	DCC
Environment Settings		
FOV radius	4 (9×9 FOV)	
Action dimension	5	
Max episode length	256	
Training Settings		
Training steps	600k	150k
Batch size	192	128
Sequence length	16	20
Learning starts	50000	
Target network update freq.	2000	1750
Discount factor γ	0.99	
Gradient clipping threshold	40	
Forward steps (n-step)	2	
Actor update steps	400	200
Replay Buffer		
Episode capacity	2048	-
Buffer capacity	-	262144
Chunk capacity	-	64
Prioritized replay α	0.6	
Prioritized replay β	0.4	
Curriculum Learning		
Initial setting	1 agent, 10×10 map	
Maximum agents	12	16
Maximum map size	40×40	
Transition threshold	0.9	
Network Architecture		
CNN channel size	128	
Hidden dimension	256	
Communication layers	2	
Communication heads	2	
Communication Strategy		
Communication type	Broadcast (fixed)	Selective (dynamic)
Max communication agents	3 (including self)	Dynamic selection

Appendix: Results of 40 × 40 Maps

Tables 4 present experimental results on 40×40 maps, showing trends consistent with the 80×80 results reported in the main text. The proposed method with $\alpha = 0.3$ for DHC and $\alpha = 0.2$ for DCC consistently outperformed baselines in success rates across all agent densities. Regarding average time steps, while partial improvements were observed, the results were less conclusive. This metric assigns the time limit (256 steps for 40×40, 386 steps for 80×80) to failed episodes, which, though consistent with baseline evaluation protocols, may not fully capture path quality. Future work should incorporate complementary metrics such as collision frequency and path optimality to provide more comprehensive performance assessment beyond binary success/failure outcomes.

[1]https://github.com/ZiyuanMa/DHC, https://github.com/ZiyuanMa/DCC

Table 4: Performance Comparison of DHC and DCC Enhanced by the Proposed Method on 40×40 Maps

Success Rate ↑

Method/Agents		**4**	**8**	**16**	**32**	**64**
DHC	Baseline	0.990	0.950	0.950	0.820	0.450
	w/ α=0.1	**1.000**	0.925	0.920	0.810	**0.525**
	w/ α=0.2	**1.000**	**1.000**	**0.970**	**0.860**	**0.595**
	w/ α=0.3	**1.000**	**1.000**	**1.000**	**0.920**	**0.645**
	w/ α=0.4	**1.000**	**0.995**	**0.965**	0.775	0.430
DCC	Baseline	1.000	0.990	0.990	0.950	0.850
	w/ α=0.1	**1.000**	**1.000**	**1.000**	**0.985**	**0.890**
	w/ α=0.2	**1.000**	**1.000**	**1.000**	**0.970**	**0.905**
	w/ α=0.3	0.995	**1.000**	0.980	0.935	0.795

Average Time Steps ↓

Method/Agents		**4**	**8**	**16**	**32**	**64**
DHC	Baseline	52.33	63.90	79.63	110.10	147.26
	w/ α=0.1	58.24	64.91	87.25	**102.33**	**135.21**
	w/ α=0.2	56.58	**59.70**	83.77	**106.31**	**138.21**
	w/ α=0.3	57.15	**62.09**	**81.42**	**99.83**	**134.86**
	w/ α=0.4	60.55	80.26	85.23	237.02	248.93
DCC	Baseline	48.58	59.60	71.34	93.54	135.55
	w/ α=0.1	**47.35**	**58.43**	73.21	**93.74**	**133.73**
	w/ α=0.2	**47.98**	**57.01**	74.79	94.82	**131.70**
	w/ α=0.3	48.99	**56.85**	73.42	97.10	156.28

Appendix: Qualitative Analysis

We provide qualitative observations from simulation results. These observations serve to offer preliminary insights into the mechanisms underlying the performance improvements. A more systematic and quantitative analysis of these behavioral tendencies remains an important direction for future work, such as measuring the frequency of cooperative maneuvers and examining how communication dependencies evolve across different scenarios.

Distance-Keeping Behavior in Path Crossing

When agents' paths intersected at crossing points, the introduction of cooperation coefficients led to a notable tendency for agents to maintain greater separation distances during passage, particularly in environments with sufficient space. While baseline methods exhibited mixed patterns of both close-proximity and distanced passing, the proposed method with $\alpha > 0$ predominantly favored maintaining spatial margins between agents.

This distance-keeping behavior demonstrates effective collision avoidance and enhanced coordination. However, it may also contribute to longer, more circuitous routes, potentially explaining the modest improvements in average time steps despite substantial gains in success rates. The trade-off between safety margins and path efficiency represents an inherent characteristic of the learned cooperative policies.

Remaining Challenges in High-Density Scenarios

Analysis of failure cases revealed persistent challenges, particularly in scenarios involving linear agent formations or chains. The majority of failures under cooperation coefficients were attributed to timeout rather than collisions, with detailed analysis indicating that deadlock states—rather than inadequate pathfinding capabilities—were the primary cause.

Specifically, when agents were constrained from both sides, stationary behavior became dominant, resulting in prolonged deadlock situations. While the proposed framework incorporating other agents' rewards achieved improvements in certain scenarios, fundamental challenges remain in environments with highly concentrated agent densities requiring complex collective decision-making among numerous simultaneously interacting agents.

Appendix: Results with Expanded Cooperation Coefficients

In the cooperative exploration curriculum, we additionally evaluated a variant in which the cooperation coefficient was increased incrementally irrespective of the success-rate threshold — i.e., the coefficient was expanded after apparent learning convergence without conditioning on recent success. The results (Table 5) are presented as a reference and summarize success rates for DHC on 40×40 maps under different cooperation coefficients.

No configuration with a cooperation coefficient of 0.5 or higher exceeded the baseline. In fact, when the coefficient surpassed 0.5 we observed a pronounced degradation in performance: success rates fell dramatically and, for coefficients of 0.6 and above, often dropped to zero. A likely explanation is that, beyond the 0.5 threshold, the reward signal becomes dominated by other agents' actions rather than an agent's own behaviour. This amplifies non-stationarity and complicates credit assignment during learning, which can prevent stable policy convergence and thereby impair overall performance. These findings indicate that naively increasing the cooperation coefficient without regard to training progress may be detrimental, and they motivate curriculum designs that adapt the cooperation coefficient based on empirical learning signals.

Table 5: Success rates of DHC with different cooperation coefficients on 40×40 maps

Method / Agents		**4**	**8**	**16**	**32**	**64**
DHC	w/ α=0.1	1.000	0.925	0.920	0.810	0.525
	w/ α=0.2	1.000	1.000	0.970	0.860	0.595
	w/ α=0.3	1.000	1.000	1.000	0.920	0.645
	w/ α=0.4	1.000	0.995	0.965	0.775	0.430
	w/ α=0.5	0.885	0.840	0.790	0.510	0.000
	w/ α=0.6	0.450	0.385	0.000	0.000	0.000
	w/ α=0.7	0.000	0.000	0.000	0.000	0.000
	w/ α=0.8	0.000	0.000	0.000	0.000	0.000
	w/ α=0.9	0.000	0.000	0.000	0.000	0.000
	w/ α=1.0	0.000	0.000	0.000	0.000	0.000