SIGMA: Sheaf-Informed Geometric Multi-Agent Pathfinding

Shuhao Liao^{1}, Weihang Xia^{2}, Yuhong Cao^{3}, Weiheng Dai^{3}, Chengyang He^{3}, Wenjun Wu^{1,*}, Guillaume Sartoretti^{3}

***Abstract*— The Multi-Agent Path Finding (MAPF) problem aims to determine the shortest and collision-free paths for multiple agents in a known, potentially obstacle-ridden environment. It is the core challenge for robotic deployments in large-scale logistics and transportation. Decentralized learning-based approaches have shown great potential for addressing the MAPF problems, offering more reactive and scalable solutions. However, existing learning-based MAPF methods usually rely on agents making decisions based on a limited field of view (FOV), resulting in short-sighted policies and inefficient cooperation in complex scenarios. There, a critical challenge is to achieve consensus on potential movements between agents based on limited observations and communications. To tackle this challenge, we introduce a new framework that applies sheaf theory to decentralized deep reinforcement learning, enabling agents to learn geometric cross-dependencies between each other through local consensus and utilize them for tightly cooperative decision-making. In particular, sheaf theory provides a mathematical proof of conditions for achieving global consensus through local observation. Inspired by this, we incorporate a neural network to approximately model the consensus in latent space based on sheaf theory and train it through self-supervised learning. During the task, in addition to normal features for MAPF as in previous works, each agent distributedly reasons about a learned consensus feature, leading to efficient cooperation on pathfinding and collision avoidance. As a result, our proposed method demonstrates significant improvements over state-of-the-art learning-based MAPF planners, especially in relatively large and complex scenarios, demonstrating its superiority over baselines in various simulations and real-world robot experiments.**

I. INTRODUCTION

As intelligent robots advance, the application of large-scale Multi-Agent Path Finding (MAPF) has become increasingly important in scenarios such as warehouse automation, airport management, and robotic fleets [1]–[4]. MAPF involves planning collision-free paths for multiple agents from their start positions to designated goals. This NP-hard problem presents significant challenges in scalability and computational efficiency due to the exponential growth of complexity with respect to the number of agents.

Recently, the MAPF community has started looking to Multi-Agent Reinforcement Learning (MARL) to generate fast and scalable solutions [5]–[7]. Moreover, MARL has gained significant traction in multi-robot systems, where agents collaborate in decentralized settings to achieve global objectives [8]–[13]. These learning-based approaches rely on

^{1}Hangzhou International Innovation Institute, Beihang University, China
^{2}CoreControl Inc, Hangzhou, China
^{3}Department of Mechanical Engineering, National University of Singapore, Singapore
*Corresponding author: Wenjun Wu (wwj09315@buaa.edu.cn).

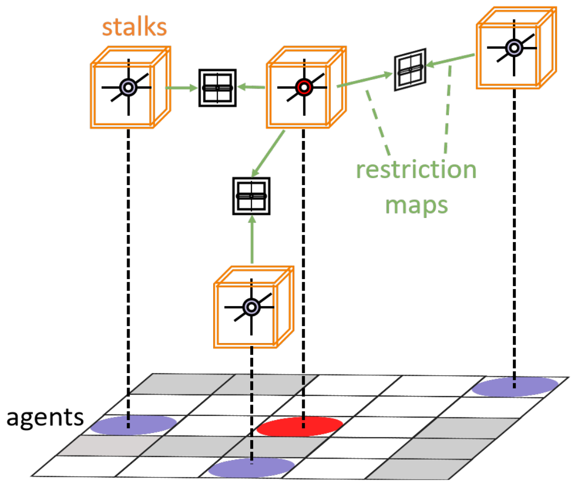


Fig. 1: A sheaf structure in MAPF, where the sheaf structure provides multilayer views of system structure, stalks collate high-dimensional data associated with agents, and restriction maps describe complex relationships between different agents. This sheaf structure aids agents in implementing consensus.

decentralized planning under partial observability (i.e., each agent only observes its nearby environment, usually 11×11 grid world), reducing computational complexity and enabling the network to tackle large-scale scenarios effectively [14]. However, the solutions generated by these learning-based methods are usually suboptimal, since they restrict the information available to agents, hindering their ability to avoid local minima and perform delicate joint behaviors. To improve the performance of learning-based MAPF planners, recent methods tried to augment the information available to agents by incorporating expert paths, designing communication schemes, or providing global map encodings [15]–[17]. These methods show significant improvements over previous learning-based methods, but there is still a remarkable performance gap between learning-based solutions and centralized optimization-based solutions [18].

This work builds upon the observation that existing learning-based planners typically lack the ability to let agents reason about and reach an explicit *consensus* (i.e., a general agreement between agents on their future movements). Such consensus is naturally achieved in centralized methods, which significantly helps agents avoid blockage and deadlock, leading to a better success rate and shorter makespan. However, achieving consensus is non-trivial in decentralized methods, as each agent makes its decision individually. As a result, current state-of-the-art learning-based methods still struggle with dense environments that require tight cooperation among agents.

To address this problem, we introduce SIGMA, a novel sheaf-informed MAPF planner that explicitly trains consensus by learning the underlying geometric cross-dependencies between agents. To the best of our knowledge, this work is the first learning-based method in MAPF that helps agents explicitly achieve consensus among themselves. Our method enhances the capability of the trained planner for modeling complex potential interactions between agents. In particular, we integrate *sheaf theory* [19] into decentralized deep reinforcement learning, enabling agents to learn consensus/global consistency by modeling geometric cross-dependencies between each other, where these geometric cross-dependencies represent the consensus. Sheaf theory broadens the concept of graphs and studies the global consistency of high dimensional data. Particularly, it provides mathematical proof of conditions for achieving global consensus through local observation. Inspired by sheaf theory, we incorporate a neural network to approximately model the consensus in latent space based on sheaf theory and train it through self-supervised learning. By doing so, in addition to normal features for MAPF as in previous works, each agent distributedly reasons about a learned consensus feature for consensus-aware path-finding and collision avoidance. We present exhaustive numerical comparisons with existing conventional and learning-based planners, which show that SIGMA outperforms state-of-the-art learning-based MAPF planners. Notably, SIGMA excels in large-scale scenarios with larger team sizes, where it significantly surpasses existing methods.

II. RELATED WORK

A. Deep Reinforcement Learning-based MAPF

Recent years have seen a growing interest in solving MAPF problems using MARL. The pioneering work, PRIMAL, introduced a combination of RL and imitation learning to plan paths through fully decentralized policies within a partially observable environment [15]. PRIMAL utilized the Asynchronous Advantage Actor-Critic algorithm as underlying RL algoritihm, with all agents sharing the same parameters, while imitation learning was based on behavior cloning from data generated by the ODrM* planner. This approach was later extended in PRIMAL2 to address lifelong MAPF scenarios, incorporating learned conventions to enhance cooperation among agents, particularly in highly structured environments [5]. Subsequent research has explored communication learning as a promising approach to further enhance solution quality. For instance, works like MAGAT and DHC [16], [20] introduced Graph Neural Networks [21] for communication learning, where each agent is treated as a node, and decisions are made based on aggregated information from neighboring agents. DCC, on the other hand, developed a selective communication strategy that determines whether an agent's decision should be influenced by its neighbors [22]. Moreover, PICO integrated planning priorities from a classical coupled planner into an ad-hoc communication topology, aiming to produce policies that reduce collisions [23]. More recently, SCRIMP introduced a scalable method where agents learn from small FOV with a modified transformer for communication, improving performance in dense scenarios [6]. Similarly, ALPHA combined local and global information, using a Graph Transformer to enhance decision-making and cooperation, addressing the limitations of limited FOV [17]. However, these methods often face challenges of scalability and complexity, struggling to handle instances with agent-dense environments.

B. Sheaf Applications

Sheaf theory addresses the local-to-global problem in multi-agent systems by allowing the coherent integration of local data into global structures [24]. Grothendieck then extended its application in algebraic geometry through scheme theory, enabling the handling of complex structures like singularities [25]. In distributed systems, sheaf theory provides a framework for consensus on complex data structures [26]. In signal processing, it manages distributed data with complex dependencies, and in network communication, it optimizes information flow and reduces communication costs [27]. In network science, sheaves provide enhanced descriptions of network structures, capturing the nature of relationships between nodes. Within this framework, opinion dynamics uses discourse sheaves to model how opinions evolve and interact within social networks [28]. Additionally, Bodnar proposed neural sheaf diffusion to learn sheaf laplacians from lower-order data, providing a novel topological perspective on heterophily and oversmoothing in GNNs [29]–[31].

III. MAPF AS AN RL PROBLEM

A. MAPF Problem Statement

The *classical* MAPF problem considers a set of agents $N = \{\alpha_1, \ldots, \alpha_n\}$ and an undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ represents the set of vertices and $\mathbf{E}$ represents the set of edges. Each agent i is assigned a distinct start vertex ($\xi_i \in \mathbf{V}$) and a distinct goal vertex ($g_i \in \mathbf{V}$). Time is discretized into uniform steps. At each time step $t = 0, 1, 2, \ldots$, an agent has the option to either move to an adjacent vertex or remain stationary at its current vertex. A path for agent α_i is defined as a sequence of vertices, either adjacent (indicating movement) or identical (indicating waiting), starting from the agent's start vertex ξ_i and ending at its goal vertex g_i. Collisions between agents are categorized as either vertex collisions, which occur when two agents α_i and α_j occupy the same vertex v at the same time t, or edge collisions, which occur when two agents α_i and α_j simultaneously traverse the same edge (u, v) in opposite directions at time t. A valid solution to the MAPF problem is a set of collision-free paths, one for each agent. The optimality of the solution is typically evaluated by the sum of the arrival times of all agents at their respective goal vertices.

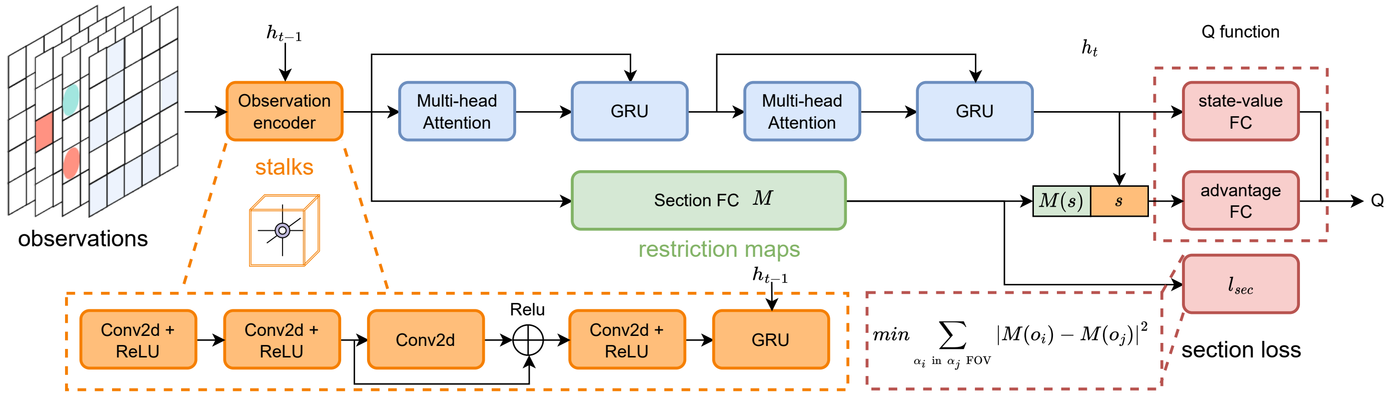


Fig. 2: Network structure of SIGMA. The observation encoder encodes observations in stalks (orange), the section FC learns restriction maps (green), and $M(s)$ is included in the advantage function to enhance action evaluation. A global section loss (red) is then integrated into the network updating to align the policy with the sheaf structure.

B. RL Environment Setup

Remaining consistent with the standard MAPF problem, we use the following setup: the map is a 2D discrete 4-connectivity grid world, where each grid is a vertex connected to its neighbors by edges, and agents can move to the free cell adjacent to their location or stay idle at each time step. An episode terminates when all agents are on their goals at the end of a time step (success) or when the number of time steps reaches the pre-defined limit (failure). We utilize the room-like map generator proposed by ALPHA. The generated room maps contain corridors of varying widths, which closely resemble real offices, warehouses, and other environments. Unlike the temporal impact of loose obstacles in random maps, continuous obstacles in such highly structured maps may significantly affect current decision-making even across substantial distances.

1) Observation: In our settings, each agent can only partially observe the environment limited by the size of the FOV $\ell \times \ell$, where ℓ is smaller than the total environment size m. To ensure the agent remains centered within its FOV, ℓ is selected as an odd number. The observation data is organized into two primary channels: the first channel is a binary matrix that represents obstacles within the FOV, and the second channel is another binary matrix that indicates the positions of other agents when they fall within the FOV. Additionally, the input to our model includes four heuristic channels, each corresponding to one of the possible movement directions: Up, Down, Left, and Right. These heuristic channels share the same dimensions as the FOV, and each cell is marked as 1 if taking the associated action would move the agent closer to its goal from that location. By incorporating these heuristic channels, the agent can infer the best direction to move without the need to explicitly include the goal location in the input [16].

2) Action Space: In our grid-world environment, agents operate within a discrete action space. At each time step, an agent can choose to move to one of the adjacent grid cells or remain stationary. We do not consider diagonal movements thus each agent has a total of five possible actions. During training or execution, agents may occasionally select invalid actions, such as moving into an obstacle or causing a collision with another agent. To handle such situations, invalid actions are not filtered out; instead, if an invalid action occurs, the agent and any involved agents are recursively returned to their previous states until no collisions remain [16].

3) Reward: We follow the DHC setup [16], assigning the same penalty for each agent's movement and for staying away from the goal. The same reward setting ensures a fairer comparison. Our reward structure is shown in Table I.

TABLE I: Reward Structure

Actions	Reward
Move (Up/Down/Left/Right)	-0.075
Stay (on goal, away goal)	$0, -0.075$
Collision (obstacle/agents)	-0.5
Finish	3

IV. LEARNING TECHNIQUES

A. Dynamic Agent Graph

We define a dynamic agent graph $G = (V, E)$ to represent the relationships between agents based on their FOV, as illustrated in Figure 3. Nodes V represent n agent $\alpha_i, i = (1, 2, ..., n)$. An edge $e \in E$ is established between two agents if they are within each other's FOV, indicating that they can potentially interact or need to consider each other's presence while planning their paths. This dynamic graph reflects the changing visibility and proximity of agents as they move through the environment, making it crucial for coordinating their actions and avoiding collisions.

B. Cellular Sheaf in MAPF

A cellular sheaf is an algebraic-topological structure associated with a graph that attaches spaces of data to nodes and edges [32]. To be precise, a cellular sheaf $(G, \mathcal{F})$ on a dynamic agent graph $G = (V, E)$ consists of:

- A vector space $\mathcal{F}(v)$ for each $v \in V$,
- A vector space $\mathcal{F}(e)$ for each $e \in E$,

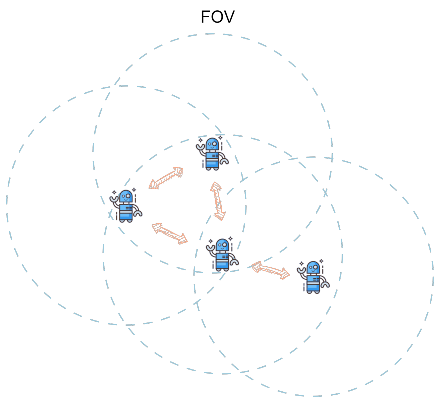


Fig. 3: Dynamic agent graph shows dynamic connections among homogeneous agents, with edges representing mutual visibility within each other's FOV.

- A linear map $\mathcal{F}_{v \trianglelefteq e} : \mathcal{F}(v) \to \mathcal{F}(e)$ for each incident node-edge pair $v \trianglelefteq e$.

According to the sheaf theory, vector spaces is termed as *stalks* and the linear map as *restriction map* [33]. The stalk originates from the analogy where a sheaf in the agricultural sense is a collection of stalks of grain bound together by twine; similarly, in mathematical terminology, a cellular sheaf on a graph is a collection of stalks of data bound together by restriction maps.

Here we focus on the concept of consensus and help readers to understand how it works in MAPF. Agents achieve global consensus via local observations, where the local observations of agents are in node stalks, and the model dependences between agents are in edge stalks. In our case, agents reach consensus when their independent observations map to consistent features via learned restriction maps. Specifically, if for any two agents v and u with observation vectors x_v and x_u agree on the edge e, the condition $\mathcal{F}_{v \trianglelefteq e} x_v = \mathcal{F}_{u \trianglelefteq e} x_u$ should be satisfied. Here, the subspace of direct sum of all the node stalks that satisfy this condition is referred to as the space of global sections $\Gamma(G, \mathcal{F})$ [34]. We regard elements in the space of global sections as representatives of consensus, where the entire agents exhibit no-contradiction behavior in how to map the observations of agents across the graph to global sections. In our work, our target is to utilize self-supervising learning to train a neural networ to model the space of global sections, achieving consensus to avoid congestion and crowding among agents.

C. Sheaf-Informed DQN

As shown in Figure 2, we use a observation encoder to encode the observations into stalks (orange). Since MAPF agents are homogeneous, meaning they have identical characteristics, their corresponding restriction maps are also identical. Thus, we denote the same restriction maps (green) for all agents as M, which is learned through the section FC's training process. Additionally, we incorporate the global section loss(red) directly into the network updating, ensuring that the learned policy respects the underlying sheaf structure and maintains consistency across the observation space.

DQN learns the action value function using neural networks. To incorporate the advantage and value functions, we can define the advantage function $A(s, a)$ and the value function $V(s)$, where $Q(s, a) = V(s) + A(s, a)$. The agents access the current states $s_t \in \mathcal{S}$ and selects an action $a_t \in \mathcal{A}$ according to a policy π at each time step t. The agent's objective is to maximize the expectation of the discounted total return $R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$, where r_t is the reward received at time t.

Q-Learning utilizes an action value function for policy π as $Q^\pi(s, a) = \mathbb{E}[R_t \mid s_t = s, a_t = a]$ and can be recursively defined by $Q^\pi(s, a) = \mathbb{E}_{s'}[r + \gamma \mathbb{E}_{a' \sim \pi}[Q^\pi(s', a')]]$ The optimal action value, $Q^*(s, a) = \max_\pi Q^\pi(s, a)$, satisfies the Bellman optimality equation $Q^*(s, a) = \mathbb{E}_{s'}[r + \gamma \max_{a'} Q^*(s', a') \mid s, a]$. The optimal policy is trained by minimizing the loss $\mathcal{L}_\mathcal{Q}$. Here, the parameters of the target network are updated periodically. In partially observable environments, agents generally need to condition on an state-action history $\mathcal{L}_\mathcal{Q} = \mathbb{E}_{(s,a,r,s')}[(Q(s, a) - y)^2]$, where $y = r + \gamma \max_{a'} Q(s', a')$.

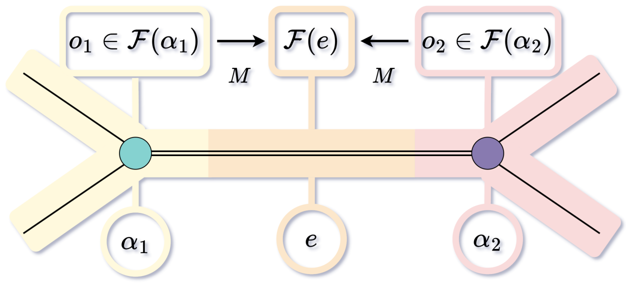

$o_1 \in \mathcal{F}(\alpha_1)$
$\mathcal{F}(e)$
$o_2 \in \mathcal{F}(\alpha_2)$
M
M
α_1
e
α_2

Fig. 4: The observation vectors o_1 and o_2 of neighboring agents α_1 and α_2 are encoded into stalks $\mathcal{F}(\alpha_1)$ and $\mathcal{F}(\alpha_2)$, which are mapped onto the stalk of edge e between them via restriction maps M.

As illustrated in Figure 4, o_1 and o_2 are the observation vectors of neighboring agents α_1 and α_2, and their corresponding stalks $\mathcal{F}(\alpha_1)$ and $\mathcal{F}(\alpha_2)$ should map to the same stalk $\mathcal{F}(e)$ on the edge e via the restriction maps M, and they should match on the edge $\mathcal{F}(e)$ by definition of global sections, i.e., $M(o_1) = M(o_2)$. To satisfy these conditions and measure how close the current observations are to being within the space of global sections, we designed a self-supervise global section loss l_{sec}. Specifically, $l_{\text{sec}}^{\alpha_1}$ can be expressed as follows:

$$l_{\text{sec}}^{\alpha_1} = \sum_{\alpha_i \text{ in } \alpha_1 \text{ FOV}} \left| M(o_i) - M(o_1) \right|^2 \tag{1}$$

$$l_{\text{sec}} = \frac{1}{n} \sum_{i=1}^{n} l_{\text{sec}}^{\alpha_i} \tag{2}$$

Where o_i represents the observation vectors of agent α_i within the FOV of agent α_1, and M denotes the restriction map, n is the number of agents. The function l_{sec} sums the mapping discrepancies between stalks, and by minimizing

TABLE II: Experimental Results. The symbol ”↑” indicates that a higher value is desirable, and vice versa.

Method	**EL↓**						**AR↑**						**SR↑**					
	20 $\times$ 20 room-liked environment with 4, 8, 16, 32, 64, 128 agents																	
ODrM*	30.58	43.19	97.25	292.93	512.00	512.00	100%	98.00%	88.00%	47.00%	0.00%	0.00%	100%	98%	88%	47%	0%	0%
PRIMAL	201.32	275.93	439.95	506.83	512.00	512.00	93.50%	90.88%	88.63%	81.72%	66.79%	35.74%	79%	67%	30%	2%	0%	0%
MAPPER	79.81	101.05	246.69	427.33	512.00	512.00	98.75%	97.53%	95.75%	89.71%	53.92%	6.96%	97%	97%	82%	41%	0%	0%
DHC	45.50	73.86	175.22	354.43	509.69	512.00	99.00%	98.62%	96.56%	90.69%	69.70%	20.09%	98%	93%	77%	45%	1%	0%
DCC	45.73	47.65	129.96	262.21	506.89	-	99.00%	99.50%	98.56%	95.91%	73.55%	-	97%	98%	86%	71%	4%	-
SCRIMP	43.42	61.56	186.34	214.32	488.98	–	99.25%	99.37%	98.87%	97.53%	**82.32%**	–	98%	96%	93%	75%	15%	–
ALPHA	**37.39**	52.26	120.65	310.21	503.87	512.00	**100%**	**100%**	**99.75%**	**97.69%**	70.12%	**25.71%**	**100%**	**100%**	96%	78%	8%	0%
SIGMA	38.04	**40.54**	**66.59**	**136.56**	**398.26**	512.00	**100%**	**100%**	99.12%	96.75%	61.62%	8.46%	**100%**	**100%**	**98%**	**92%**	**39%**	0%
	40 $\times$ 40 room-like environment with 4, 8, 16, 32, 64, 128 agents																	
ODrM*	56.73	69.34	91.89	146.88	375.37	512.00	100%	100%	97.00%	85.00%	32.00%	0.00%	100%	100%	97%	85%	32%	0%
PRIMAL	285.55	384.93	463.86	492.82	511.80	512.00	91.00%	87.62%	85.56%	82.69%	73.14%	61.71%	73%	47%	23%	11%	1%	0%
MAPPER	104.82	157.12	218.35	348.95	491.58	512.00	**100%**	99.37%	98.00%	93.71%	76.60%	51.02%	**100%**	96%	91%	66%	16%	0%
DHC	104.19	127.78	188.62	263.81	427.02	512.00	97.75%	98.00%	97.88%	95.94%	91.17%	72.53%	92%	91%	80%	65%	28%	0%
DCC	63.25	102.40	142.88	201.43	338.16	-	99.75%	99.25%	99.12%	98.72%	96.47%	-	99%	95%	89%	85%	58%	-
SCRIMP	58.53	91.84	116.05	183.54	396.93	484.76	**100%**	99.62%	99.56%	99.21%	94.10%	85.09%	**100%**	97%	95%	84%	42%	12%
ALPHA	64.04	88.75	140.96	206.85	392.23	506.48	**100%**	**100%**	99.75%	99.34%	93.46%	73.99%	**100%**	**100%**	97%	93%	60%	7%
SIGMA	**57.23**	**77.10**	**88.33**	**128.70**	**223.73**	**380.48**	**100%**	**100%**	**100%**	**99.56%**	**98.50%**	**92.11%**	**100%**	**100%**	**100%**	**95%**	**89%**	**69%**
	60 $\times$ 60 room-liked environment with 4, 8, 16, 32, 64, 128 agents																	
ODrM*	84.71	98.43	106.46	163.53	228.95	457.17	100%	100%	99.00%	88.00%	72.00%	14.00%	100%	100%	99%	88%	72%	14%
PRIMAL	363.45	465.35	495.85	508.17	512.00	512.00	84.75%	78.37%	79.75%	73.62%	71.51%	62.83%	54%	25%	11%	3%	0%	0%
MAPPER	177.61	241.31	280.69	388.55	490.02	512.00	99.50%	97.75%	98.31%	93.87%	85.96%	62.47%	97%	89%	90%	61%	17%	0%
DHC	131.59	203.71	186.66	323.19	406.40	496.70	97.75%	96.75%	98.88%	95.16%	93.30%	87.79%	91%	77%	86%	54%	35%	7%
DCC	113.64	145.24	170.38	268.39	331.01	-	98.25%	99.00%	98.88%	97.75%	95.50%	-	95%	93%	89%	69%	58%	-
SCRIMP	106.79	166.37	125.50	211.03	421.65	498.72	99.50%	99.25%	99.61%	98.73%	96.79%	88.08%	98%	95%	97%	81%	31%	8%
ALPHA	110.82	158.59	173.02	263.74	357.17	485.23	99.50%	99.25%	99.75%	99.03%	97.91%	89.16%	98%	97%	97%	86%	67%	25%
SIGMA	**91.02**	**103.63**	**109.53**	**147.76**	**187.96**	**332.35**	**100%**	**100%**	**100%**	**100%**	**99.78%**	**98.52%**	**100%**	**100%**	**100%**	**98%**	**96%**	**80%**

l_{sec}, we ensure that the global section conditions are satisfied. When training converges, the global section loss is minimized, fulfilling the conditions set by sheaf theory, and ensuring that the agents achieve **consensus**.

We incorporate l_{sec} into the network updating to assist the agent in making decisions that adhere to the section conditions. As a result, the learned consensus among agents should be considered during learning the advantage value, we include $M(s)$ in the advantage function to enable better evaluation of actions by the agents. The resulting Q-function in SIGMA is expressed as follows:

$$Q(s,a) = V_{\theta_1}(s) + A_{\theta_2}(s',a) - \frac{1}{|\mathcal{A}|}\sum_{a'\in\mathcal{A}} A_{\theta_2}(s',a') \quad (3)$$

$$s' = [M(s), s] \quad (4)$$

$$\mathcal{L}_{SIGMA} = \mathcal{L}_Q + \Lambda \cdot l_{\text{sec}} \quad (5)$$

Here, $Q(s,a)$ represents the Q value for action a in state s, $V_{\theta_1}(s)$ denotes the value function of state s, which is parameterized by θ_1, and $A_{\theta_2}(s',a)$ signifies the advantage function of action a in the enhanced state s', parameterized by θ_2. The term $|\mathcal{A}|$ represents the size of the action space, $\mathcal{L}_{SIGMA}$ is the loss function, and Λ is a hyperparameter.

V. EXPERIMENTS

In this section, we evaluate SIGMA through comprehensive simulation experiments, comparing its performance with SOTA baselines. We also conduct ablation studies to assess the impact of each component of our approach. Additionally, we test the robustness of the trained model by deploying it in simulation and real world environment.

A. Main Results Comparison

For our experiments, we train our models on structured environments of varying sizes, randomly chosen from a uniform distribution between 10 and 40, while consistently deploying 5 agents. During testing, we explore environments of 20, 40, and 60 sizes, scaling the number of agents from 4 up to 128.

Our evaluation include a comparison with 6 SOTA MAPF solutions: PRIMAL [15], MAPPER [35], DHC [16], DCC [22], SCRIMP [6], and ALPHA [17]. Additionally, we benchmark against the searchbased, bounded-optimal centralized planner ODrM* with an inflation factor of $\epsilon = 2.0$ [18]. For a fair comparison, each planner was tested on the same set of 200 randomly-generated environments.

We employ three metrics to assess performance: 1) **Episode Length(EL)**: This measures the efficiency of a solution by counting the number of actions agents take to reach their goals within a single episode. 2) **Arrival Rate(AR)**: This is the percentage of agents that reach their goals across all episodes. 3) **Success Rate(SR)**: This metric evaluates a planner's ability to completely fulfill a task. Notably, learning-based methods may show a low SR but still have a high AR, which highlights the importance of AR in evaluating episodes that nearly reach completion without being deemed total failures. The results are presented in Table II.

In our experiments, SIGMA consistently outperforms other learning-based planners in terms of SR across all tasks. Notably, as the number of agents increases, SIGMA's improvement in SR significantly exceeds that of the baseline planners. For instance, in complex scenarios where most learning-based planners struggle or fail to solve the problems, such as with 128 agents on a 40×40 map, SIGMA achieves a

SR of 69%. Even more impressively, on a larger 60×60 map, SIGMA's SR reaches 80%. These results highlight the effectiveness of our approach where the agents, through achieving consensus, successfully avoid congestion and overcrowding, demonstrating robust performance even under challenging conditions. Additionally, it is noteworthy that on a smaller 20x20 map, although SIGMA exhibits a high SR, the AR isn't as impressive. This indicates that while consensus effectively prevents congestion, it does not necessarily aid in navigating out of such congested scenarios efficiently.

B. Ablation Analysis

Our method focuses on encoding the sheaf structure(stalks and restriction maps) and integrating global section loss. These elements are applied to both the input of the advantage function and the loss function for updating the network to ensure the correctness of the sheaf structure. To analyze the importance of these elements, we experimented with three ablation variants of SIGMA: 1) **Encoded Stalk(ES)**: This variant tested only the stalks' encoding effectiveness by removing global section loss and restriction maps. 2) **Weighted Penalty(WP)**: This setup assessed the influence of global section loss within the loss calculation by excluding them. 3) **Feature Impact(FI)**: We evaluated the impact of removing restriction maps from the advantage function while keeping other elements constant.

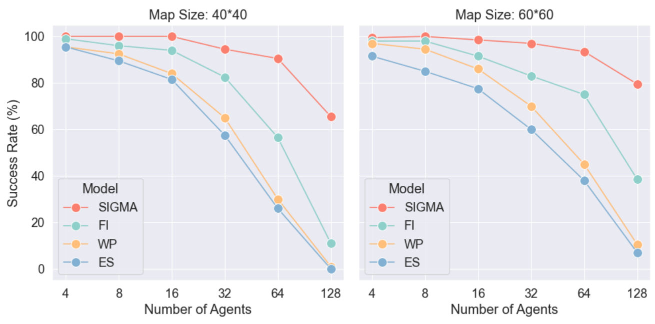
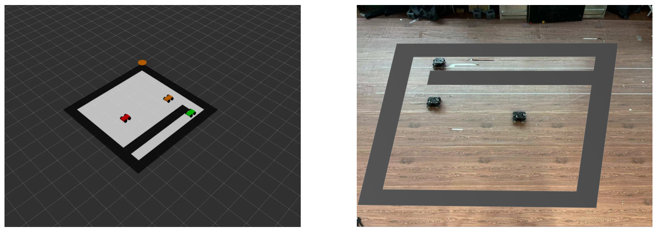


Fig. 5: Success rates of ablation variants on 40x40 and 60x60 maps.

The results of the ablation variants are shown in Figure 5. Encoded stalk results in performance similar to the baselines, indicating that this component alone doesn't significantly influence success rates. The introduction of the restriction maps leads to a slight improvement in success rates, suggesting a beneficial but limited role in the overall system's performance. A notable enhancement is observed with the incorporation of global section loss, particularly as the number of agents increases. This enhancement significantly boosts success rates, demonstrating that global section loss effectively guide agents towards explicit consensus. When all components of the SIGMA framework are utilized, including both stalks, restriction maps, and global section loss, there is a further increase in performance. This indicates that the full integration of the sheaf structure is crucial for achieving consensus.

C. Experimental Validation



(a) Simulation Environment (b) Real World

Fig. 6: Experiments with real robots on room-like map.

As shown in Figure 6, Figure 6a represents the simulation environment, and Figure 6b represents the real world environment. In this setup, each cell in the real world environment measures 0.3m on each side, slightly larger than the agents to ensure each agent occupies only one cell on the map. We employ 3 robots, each equipped with Mecanum wheels and measuring approximately 0.23m × 0.2m. The accurate positions of these robots are tracked using the *OptiTrack Motion Capture System*. The starting and goal positions of the agents are randomly configured. In this experiment, although the robots recognize the virtual positions of obstacles and are programmed to avoid these areas, there are no physical obstacles in the real environment, preventing any interference with the line of sight needed for the OptiTrack motion capture system.

VI. CONCLUSIONS

In this paper, we introduce SIGMA, a novel MAPF planner that pushes beyond the constraints of limited FOV commonly found in existing learning-based MAPF planners by utilizing sheaf theory to let all agents reason about and reach a team-wide consensus. Our approach efficiently encodes the sheaf structure within MAPF, incorporating global section loss to measure consistency. Through extensive experiments conducted on highly structured maps with varying team sizes and environmental complexities, SIGMA consistently outperforms SOTA learning-based MAPF planners and a bounded-optimal search-based planner in most scenarios. By rigorously proving the transition from local observation to global consensus, this work proposes a novel perspective to the MAPF research community.

In future work, we plan to focus on enriching the intricate relationships among agents by building consensus on more complex graphs. This will involve exploring advanced models and techniques that can handle and interpret the interactions and dependencies within larger, more complex network structures.

ACKNOWLEDGMENT

This work was supported by the National Science and Technology Major Project(Grant No.2022ZD0116401), the National Natural Science Foundation of China(Grant No.62441617), and an Amazon Research Award.

References

[1] W. Hönig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Persistent and robust execution of mapf schedules in warehouses," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1125–1131, 2019.

[2] I. Rekik and S. Elkosantini, "A multi agent system for the online container stacking in seaport terminals," *Journal of Computational Science*, vol. 35, pp. 12–24, 2019.

[3] J. Li, A. Tinka, S. Kiesel, J. W. Durham, T. S. Kumar, and S. Koenig, "Lifelong multi-agent path finding in large-scale warehouses," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, 2021, pp. 11 272–11 281.

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

[5] M. Damani, Z. Luo, E. Wenzel, and G. Sartoretti, "Primal_2: Pathfinding via reinforcement and imitation multi-agent learning-lifelong," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2666–2673, 2021.

[6] Y. Wang, B. Xiang, S. Huang, and G. Sartoretti, "Scrimp: Scalable communication for reinforcement-and imitation-learning-based multi-agent pathfinding," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 9301–9308.

[7] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile robot path planning in dynamic environments through globally guided reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6932–6939, 2020.

[8] X. Yu, R. Shi, P. Feng, Y. Tian, J. Luo, and W. Wu, "Esp: Exploiting symmetry prior for multi-agent reinforcement learning," in *ECAI 2023*. IOS Press, 2023, pp. 2946–2953.

[9] X. Yu, Y. Tian, L. Wang, P. Feng, W. Wu, and R. Shi, "Adaptaug: Adaptive data augmentation framework for multi-agent reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 10 814–10 820.

[10] X. Yu, R. Shi, P. Feng, Y. Tian, S. Li, S. Liao, and W. Wu, "Leveraging partial symmetry for multi-agent reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, 2024, pp. 17 583–17 590.

[11] X. Yu, W. Wu, P. Feng, and Y. Tian, "Swarm inverse reinforcement learning for biological systems," in *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2021, pp. 274–279.

[12] P. Feng, R. Shi, S. Wang, J. Liang, X. Yu, S. Li, and W. Wu, "Safe and efficient multi-agent collision avoidance with physics-informed reinforcement learning," *IEEE Robotics and Automation Letters*, 2024.

[13] P. Feng, X. Yu, J. Liang, W. Wu, and Y. Tian, "Mact: Multi-agent collision avoidance with continuous transition reinforcement learning via mixup," in *International Conference on Swarm Intelligence*. Springer, 2023, pp. 74–85.

[14] P. Feng, J. Liang, S. Wang, X. Yu, X. Ji, Y. Chen, K. Zhang, R. Shi, and W. Wu, "Hierarchical consensus-based multi-agent reinforcement learning for multi-robot cooperation tasks," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 642–649.

[15] G. Sartoretti, J. Kerr, Y. Shi, G. Wagner, T. S. Kumar, S. Koenig, and H. Choset, "Primal: Pathfinding via reinforcement and imitation multi-agent learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2378–2385, 2019.

[16] Z. Ma, Y. Luo, and H. Ma, "Distributed heuristic multi-agent path finding with communication," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8699–8705.

[17] C. He, T. Yang, T. Duhan, Y. Wang, and G. Sartoretti, "Alpha: Attention-based long-horizon pathfinding in highly-structured areas," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 576–14 582.

[18] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artificial intelligence*, vol. 219, pp. 1–24, 2015.

[19] J. Hansen and R. Ghrist, "Toward a spectral theory of cellular sheaves," *Journal of Applied and Computational Topology*, vol. 3, pp. 315–358, 2019.

[20] Q. Li, W. Lin, Z. Liu, and A. Prorok, "Message-aware graph attention networks for large-scale multi-robot path planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5533–5540, 2021.

[21] T. Peng, W. Wu, H. Yuan, Z. Bao, Z. Pengru, X. Yu, X. Lin, Y. Liang, and Y. Pu, "Graphrare: Reinforcement learning enhanced graph neural network with relative entropy," in *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 2024, pp. 2489–2502.

[22] Z. Ma, Y. Luo, and J. Pan, "Learning selective communication for multi-agent path finding," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1455–1462, 2021.

[23] W. Li, H. Chen, B. Jin, W. Tan, H. Zha, and X. Wang, "Multi-agent path finding with prioritized communication learning," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 695–10 701.

[24] H. Miller, "Leray in oflag xviia: the origins of sheaf theory, sheaf cohomology, and spectral sequences," *Kantor 2000*, pp. 17–34, 2000.

[25] A. Grothendieck, *A general theory of fibre spaces with structure sheaf*. University of Kansas, Department of Mathematics, 1955, no. 4.

[26] J. Hansen, "Laplacians of cellular sheaves: Theory and applications," Ph.D. dissertation, University of Pennsylvania, 2020.

[27] S. Sardellitti and S. Barbarossa, "Topological signal processing over generalized cell complexes," *IEEE Trans. Signal Process.*, vol. 72, pp. 687–700, 2024.

[28] J. Hansen and R. Ghrist, "Opinion dynamics on discourse sheaves," *SIAM Journal on Applied Mathematics*, vol. 81, no. 5, pp. 2033–2060, 2021.

[29] C. Bodnar, F. Di Giovanni, B. Chamberlain, P. Lio, and M. Bronstein, "Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18 527–18 541, 2022.

[30] T. Peng, H. Yuan, Y. Zhang, Y. Li, P. Dai, Q. Wang, S. Wang, and W. Wu, "Tagrec: Temporal-aware graph contrastive learning with theoretical augmentation for sequential recommendation," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2025.

[31] T. Peng, Y. Liang, W. Wu, J. Ren, Z. Pengrui, and Y. Pu, "Clgt: A graph transformer for student performance prediction in collaborative learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 13, 2023, pp. 15 947–15 954.

[32] J. Hansen and T. Gebhart, "Sheaf neural networks," *CoRR*, vol. abs/2012.06333, 2020.

[33] G. E. Bredon, *Sheaf theory*. Springer Science & Business Media, 2012, vol. 170.

[34] J. Hansen and R. Ghrist, "Toward a spectral theory of cellular sheaves," *Journal of Applied and Computational Topology*, vol. 3, no. 4, pp. 315–358, 2019.

[35] Z. Liu, B. Chen, H. Zhou, G. Koushik, M. Hebert, and D. Zhao, "Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 748–11 754.