Interference-Aware K-Step Reachable Communication in Multi-Agent Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Effective communication is pivotal for addressing complex collaborative tasks in multi-agent reinforcement learning (MARL). Yet, limited communication bandwidth and dynamic, intricate environmental topologies present significant challenges in identifying high-value communication partners. Agents must consequently select collaborators under uncertainty, lacking a priori knowledge of which partners can deliver task-critical information. To this end, we propose Interference-Aware K-Step Reachable Communication (IA-KRC), a novel framework that enhances cooperation via two core components: (1) a K-Step reachability protocol that confines message passing to physically accessible neighbors, and (2) an interference-prediction module that optimizes partner choice by minimizing interference while maximizing utility. Compared to existing methods, IA-KRC enables substantially more persistent and efficient cooperation despite environmental interference. Comprehensive evaluations confirm that IA-KRC achieves superior performance compared to state-of-the-art baselines, while demonstrating enhanced robustness and scalability in complex topological and highly dynamic multi-agent scenarios.

1 Introduction

Multi-Agent Reinforcement Learning (MARL) enables collaborative decision-making through interactions between multiple agents and their environment. This paradigm has demonstrated significant potential in autonomous driving (Chen et al., 2025), game AI (Vinyals et al., 2019), UAV coordination (Liao et al., 2025), where effective inter-agent communication is crucial for successful cooperation. However, practical constraints in communication bandwidth and system scalability make fully-connected communication infeasible (Zhu et al., 2024), necessitating efficient distributed communication strategies (Hu et al., 2024). This raises a fundamental research question: How to identify the most valuable communication partners in complex multi-agent systems? Prior work (Siedler, 2021) shows that poor partner selection can not only diminish cooperation benefits but may actually degrade overall system performance. Moreover, real-world environments often possess complex topologies and exhibit highly dynamic group behaviors, further complicating the identification of high-value communication targets.

Existing approaches for selecting communication partners mainly rely on neighborhood-based constraints. Hüttenrauch et al. (2019); Jiang & Lu (2018) use Euclidean distance as the selection criterion, considering spatially nearby agents as potential communication partners. However, in complex environments, Euclidean distance often significantly overestimates actual reachability, resulting in inefficient communication. As shown in Figure 1 (a), when obstacles are present, agents A and B may have a small Euclidean distance (blue dashed line) but be separated by a long traversable path (yellow arrow path), hindering effective cooperation. To overcome this limitation, other works have proposed visual perception-based approaches that establish communication links only between agents with direct line of sight (green region) (Chen et al., 2021; Baker et al., 2019) (Figure 1 (b)). While this method more accurately reflects physical connectivity, it remains limited in complex environments where agents may fail to detect nearby partners that are occluded from view.

Moreover, existing methods often overlook the interference caused by adversarial dynamics and interactions among agents, which can lead to ineffective cooperation even when agents are close neighbors. For instance, enemy attacks can create high-risk zones that severely disrupt the cooperation of friendly agents, thereby significantly increasing the cost of cooperation. As illustrated in Figure 1(c), the red high-risk zone severely disrupts communication between agents A and B, forces them to take a detour to cooperate, resulting in prohibitively high path transition costs. This observation underscores the critical importance of dynamic interference modeling for effective collaborator selection. While recent work by Naderializadeh et al. (2020) attempts to address this challenge using graph neural networks (GNNs) to infer cooperative structures and implicitly capture adversarial interference through end-to-end learning, such approaches typically exhibit limited scalability and performance degradation in large-scale multi-agent scenarios (Gogineni et al., 2023).

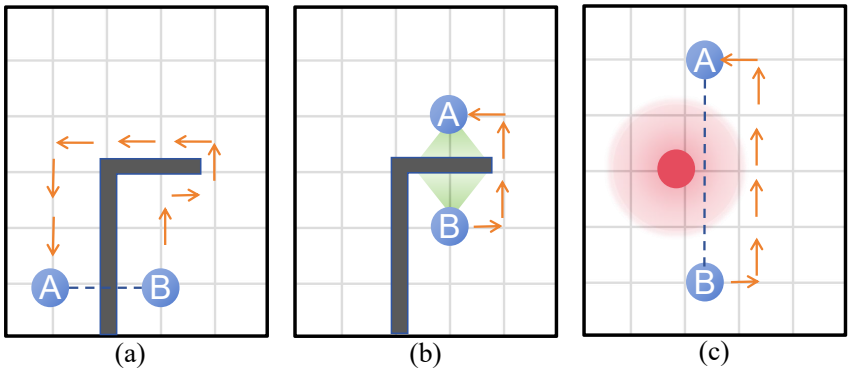


Figure 1: Limitations of common neighborhood constraints in MARL communication. (a) Euclidean distance may misrepresent actual accessibility: agents A and B appear close (blue dashed line) but are separated by a long traversable path (yellow arrow); (b) Vision-based constraints capture physical proximity better but miss reachable agents hidden from view; (c) Even with direct visibility, hostile interference (red region) can block cooperation, forcing detours (yellow arrows).

This paper posits that high-value communication should occur between agents that are both physically reachable and experience minimal interference, thereby enabling sustained and effective collaboration. To this end, we propose the Interference-Aware K-step Reachable Communication (IA-KRC), which consists of two modules as shown in Figure 2. **K-step Reachable Communication**: This module restricts communication to agents that are mutually reachable within K movement steps. This definition considers agents' actual mobility capabilities in complex environments, providing a more accurate characterization of neighborhood relationships compared to Euclidean distance or line-of-sight metrics. **Interference-Prediction Module**: This component evaluates potential interference from agents beyond the K-step neighborhood, including adversarial interference and cooperative conflicts. By integrating these predictions, IA-KRC can further identify low-interference, high-value communication partners within the reachable domain. Furthermore, we develop an IA-KRC-based learning algorithm that efficiently identifies high-value communication and optimizes policy learning in an end-to-end framework.

We validate IA-KRC in challenging multi-agent combat scenarios constructed within the StarCraft Multi-Agent Challenge (SMACv2) framework Samvelyan et al. (2019), featuring dense obstacles and maze-like topologies. Under our self-play framework against strong baselines including CommFormer, Euclid, Vision, RL-Vision, MAPPO, and QMIX, IA-KRC achieves a win rate advantage of at least 4.58× and up to 31.56×. This significant advantage stems from IA-KRC's ability to achieve sustained and effective cooperation in topologically complex and interference-laden environments.

2 Related Work

Learning to communicate in MARL. Effective communication is critical for collaboration in MARL. However, under bandwidth and scalability constraints, determining *whom to communicate with* becomes a central challenge.

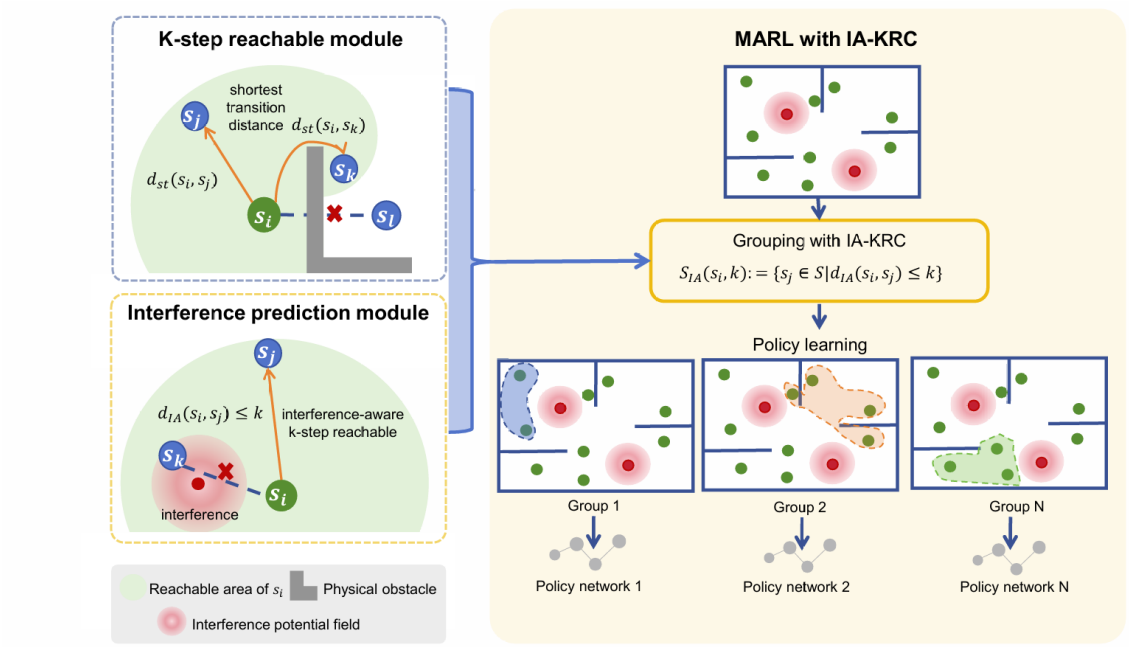


Figure 2: Overview of IA-KRC framework for MARL communication. The K-step Reachability Module restricts communication to agents within a physically reachable domain based on shortest transition distance. The Interference Prediction Module evaluates potential conflicts or adversarial effects, selecting low-interference partners. Combined, these modules enable dynamic grouping and robust leader–follower collaboration in complex environments.

Traditional metrics such as Euclidean distance or line-of-sight visibility often fail to accurately capture the constraints of communication relevant neighborhoods. Recent efforts learn communication topology end-to-end using attention mechanisms or GNNs Naderializadeh et al. (2020); Hu et al. (2024); Niu et al. (2021); Su et al. (2020), such as introducing personalized peer-to-peer communication via learned message routing policies Meng & Tan (2024). Although powerful in small-scale settings, these models lack explicit spatial priors (e.g., physical reachability), do not account for dynamic interference, and often scale poorly with the number of agents Christianos et al. (2021). Beyond spatial cues, some methods explore semantic-level grouping Sunehag et al. (2018), or employ self-supervised learning for message aggregation Guan et al. (2022), but these struggle with dynamic spatial feasibility in topologically constrained tasks. Furthermore, recent advances learn communication graphs via differentiable architecture search and graph–transformer hybrids Cong et al. (2023); Mei et al. (2023), or adopt multi-level asynchronous communication for sequential coordination Ding et al. (2024), to obtain dynamic, sparse topologies with scalable routing. While these methods improve communication efficiency through learned selection, aggregation, or hierarchical policies, most operate on abstract feature spaces without explicitly grounding communication in physical environmental constraints, and consequently may be less effective when spatial accessibility and dynamic physical interference fundamentally constrain collaboration. Our work addresses these limitations by introducing a more faithful spatial foundation for communication learning: we explicitly model both multi-step physical reachability and dynamic environmental interference as structural priors for communication, enabling robust partner selection in physically grounded, dynamic environments.

The concept of K-Step reachability is well-established in single-agent reinforcement learning, where it primarily serves to constrain the selection of subgoals Nachum et al. (2018); Vezhnevets et al. (2017). Our work represents the first application of reachability constraints within multi-agent systems, specifically for the purpose of selecting communication partners. A critical limitation of prior approaches is their difficulty in scaling to multi-agent settings. This is because the inherent interference in such environments—arising from teammate motion, opponent behaviors, and evolving policies Lowe et al. (2017)—means that relying solely on

reachability is insufficient to guarantee sustained and effective cooperation. Addressing this gap, our paper introduces, for the first time, an interference-aware formulation of K-Step reachability. While some works have modeled influence or interference Jiang et al. (2020); Omidshafiei et al. (2022), these concepts have not been integrated with reachability to guide communication. Our approach closes this gap by jointly modeling physical reachability and dynamic interference, enabling robust partner selection in physically grounded, dynamic environments.

3 Preliminaries

In this work, we consider the cooperative MARL scenario formulated as a decentralized partially observable Markov decision process (Dec-POMDP) (Oliehoek et al., 2016), a common framework for modeling cooperation among multiple autonomous agents under uncertainty. Formally, a Dec-POMDP is described as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, r, \gamma, N \rangle$, where $\mathcal{S}$ represents the global state space shared by all agents, $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_N$ is the joint action space of N agents, and $\mathcal{O}$ is the observation space available to each agent. Let $\mathcal{N} = \{1, \ldots, N\}$ denote the set of agents. The state transition probability $\mathcal{P}(s' \mid s, a) \colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ specifies the probability of transitioning to state s' from state s given joint action a. The global reward function $r(s, a) \colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the common objective for all agents. The discount factor $\gamma \in [0, 1)$ determines the relative importance of future rewards. Each agent i independently receives a local observation o_i based on an observation function $O(s, i) \colon \mathcal{S} \times \{1, \ldots, N\} \to \mathcal{O}$.

To facilitate effective decentralized cooperation, we adopt the widely-used leader-follower framework (Soni & Hu, 2018) for agent grouping. Agents are partitioned into cooperative groups, where each group $g \in G$ consists of a leader and several followers. Each group g maintains a joint action-observation history $\boldsymbol{\tau}_g$ and learns a shared group policy $\pi_g(\mathbf{a}_g \mid \boldsymbol{\tau}_g) \colon \mathcal{T}^{|g|} \to \Delta(\mathcal{A}_g)$, where $\mathcal{T}^{|g|}$ denotes the space of joint histories for agents in group g, $\mathcal{A}_g$ is the joint action space of the group, and $\Delta(\mathcal{A}_g)$ is the probability simplex over $\mathcal{A}_g$. Our goal is to efficiently resolve leader election and follower assignment.

4 Method

The objective of IA-KRC is to facilitate sustained and effective cooperation by ensuring that communication occurs between agents that are not only physically reachable but also subject to minimal interference. As illustrated in Figure 2, the IA-KRC framework comprises two main components: the K-Step Reachability Module and the Interference Prediction Module. The former quantifies the physical path accessibility between agents, restricting communication to those in close proximity. The latter assesses the potential communication interference that other agents may impose on a target agent. By integrating these two modules, IA-KRC enables the selection of reliable, long-term cooperative partners within a neighborhood by filtering for low-interference candidates.

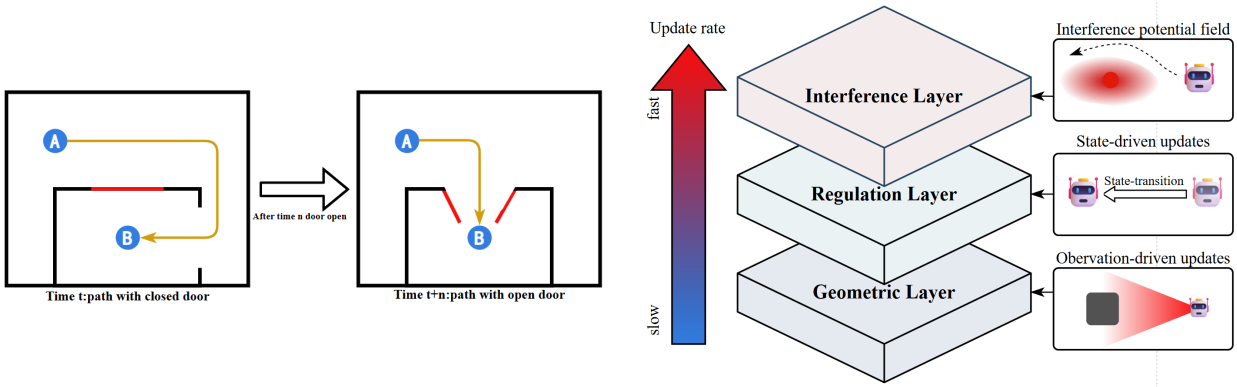
The principle of K-step reachability dictates that an agent must be able to reach the state of another agent within K time steps. In this context, conventional proximity metrics like Euclidean distance are inadequate, as they often fail to capture the underlying topological structure of MDPs. To address this, we introduce the concept of the shortest transition distance as a more suitable metric for evaluating reachability between agents. In a stochastic MDP, the number of steps required for a transition is not deterministic. Consequently, we define shortest transition distance by minimizing the expected first-hitting time over the set of all possible policies, as formally presented in Definition 1.

Definition 1. Let $s_1, s_2 \in S$. Then, shortest transition distance from s_1 to s_2 is defined as:

$$d_{st}(s_1, s_2) := \min_{\pi \in \Pi} \mathbb{E}[T_{s_1,s_2} | \pi] = \min_{\pi \in \Pi} \sum_{t=0}^{\infty} t P(T_{s_1,s_2} = t | \pi),$$

where Π is the complete policy set and T_{s_1,s_2} denotes the first hit time from s_1 to s_2.

Considering the topological complexity, we do not assume the environment is reversible; that is, $d_{st}(s_1, s_2)$ is not necessarily equal to $d_{st}(s_2, s_1)$. Therefore, shortest transition distance is a quasi-metric, as it does not satisfy symmetry.



(a) Impact of time-varying regulation rules on shortest transition distance. Red line: door state changing over time; yellow arrow: shortest transition path.

(b) Illustration of the multi-layer map used to compute K-Step reachability.

Figure 3: Illustration of the time-varying regulation (a) and the multi-layer map structure (b).

Shortest transition distance measures the time steps required to transition from state s_1 to s_2 in the most efficient manner, and it has been a subject of study in several works on single-agent reinforcement learning (Chebotarev & Deza, 2020; Zhang et al., 2020). However, in multi-agent environments, selecting communication partners based solely on this distance is insufficient. This is because actions from other agents can interfere with cooperation, even when two agents are in close proximity. For instance, cooperative agents working together to unlock mechanisms can reduce shortest transition distance, while adversarial agents may block critical pathways, rendering the shortest transition routes practically unreachable. To address this, our proposed Interference Prediction Module explicitly measures the cost of cooperation. Accordingly, the interference-aware shortest transition distance is defined as follows.

Definition 2. Let $C(T_{s_1,s_2} = t|\pi)$ be the cooperation cost incurred when the first-hitting time from s_1 to s_2 is t under policy π. We assume $C(T_{s_1,s_2} = t|\pi) \geq 0$. Then, the interference-aware shortest transition distance is defined as:

$$d_{IA}(s_1, s_2) := \min_{\pi \in \Pi} \sum_{t=0}^{\infty} tP(T_{s_1,s_2} = t|\pi) \times C(T_{s_1,s_2} = t|\pi).$$

Based on the interference-aware shortest transition distance, we introduce the notion of interference-aware K-step reachable region to bound the communication scope among agents.

Definition 3. Let $s_1 \in S$. Then, the interference-aware K-Step reachable region of s_1 is defined as:

$$\mathcal{S}_{IA}(s_1, K) := \{s_2 \in S \mid d_{IA}(s_1, s_2) \leq K\}.$$

To avoid unnecessary global updates, our approach first detects where the environment has actually changed and then refreshes distances only for those local regions. Empirically, shortest transition distances evolve on distinct time scales: obstacle-induced changes are slow, environmental rule-based changes (e.g., door states as shown in Figure 3a) occur at a moderate rate, whereas agent state changes and adversarial interference alter transition costs almost instantly. We therefore introduce a multi-layer map (Figure 3b) that separates information by its rate of change, in which each layer records elements that change on different time scales, thereby allowing us to quickly localize the environmental regions where changes have occurred.

The multi-layer map serves as an abstract representation of the multi-agent environment, comprising three mutually decoupled layers. The Geometric Layer stores static elements and extremely slowly changing dynamic elements, updated through agent observations. The Regulation Layer stores environmental rule-based

information, such as the opening and closing of doors and changes in traffic lights in real-world scenarios. This rule-based information evolves at a moderate rate and is updated based on the actual transitions of agents within the environment. The Interference Layer stores adversarial interference information of entities, dynamically generating real-time interference potential fields from the stored information to precisely track these most rapidly changing attributes. By statistically monitoring the confidence levels of all layers and updating information below confidence thresholds, the map achieves asynchronous updates. **Rather than replicating the full environment, this abstraction discretizes and models only the essential dynamic factors, enabling efficient tracking of non-stationary elements**. Further details of the multi-layer map are provided in the Appendix.

By maintaining the multi-layer map, we can efficiently compute shortest transition distance between any pair of states at any time without requiring agents to execute actions in the actual environment to measure distances. Only when changes are detected in a specific layer (Geometric, Regulation, or Interference) is that layer incrementally refreshed. We only recompute distances for agent pairs within K steps of the updated regions in each layer, thus avoiding expensive global updates. At computation time, the three layers are aggregated into a unified graph representation $\mathcal{G}^{(t)} = \text{Aggregate}(\mathcal{G}^{(t)}_{\text{geo}}, \mathcal{G}^{(t)}_{\text{reg}}, \mathcal{G}^{(t)}_{\text{int}})$, where $\mathcal{G}^{(t)}_{\text{geo}}$, $\mathcal{G}^{(t)}_{\text{reg}}$, and $\mathcal{G}^{(t)}_{\text{int}}$ denote the Geometric, Regulation, and Interference layer graphs at time t, respectively. Given the states of two agents, the shortest path can be computed by applying any shortest path algorithm on the aggregated graph $\mathcal{G}^{(t)}$. Notably, while the shortest path algorithm itself is discrete, the construction of the aggregated graph—particularly the Interference Layer—involves a learnable interference prediction module that continuously refines interference estimates through end-to-end learning, thereby indirectly influencing the shortest path computation. In this work, we employ Dijkstra's algorithm (Dijkstra, 1959), denoting its policy as π_D. In this regard, the distance defined in Definition 2 can be implemented as follows:

$$d_{IA}(s_1, s_2|\mathcal{G}^{(t)}) := \sum_{t=0}^{\infty} tP(T_{s_1,s_2} = t|\pi_D, \mathcal{G}^{(t)}) \times C(T_{s_1,s_2} = t|\pi_D, \mathcal{G}^{(t)}).$$

4.1 Computing Cooperation Cost with an Interference Potential Field

In the course of agent cooperation, interference from other agents or environmental entities is unavoidable, which can undermine the stability and persistence of collaboration. Particularly in multi-agent game-theoretic scenarios, actions such as attacks or threats from adversarial agents frequently disrupt the cooperative process. Consequently, considering the cost of cooperation becomes crucial for selecting high-value partners for interaction.

To quantify the cooperation cost, $C(T_{s_1,s_2} = t|\pi_D)$, this paper draws on the concept of artificial potential fields (Khatib, 1986) to construct an interference potential field (Figure 3b) for each entity within the environment. This potential field quantifies the interference strength exerted by each entity on its surrounding region, while the cooperation cost is the cumulative effect of superimposing the interference fields of all entities. For the interference potential field of a single entity e_i, the interference intensity at state s is denoted as $I(s \mid e_i)$. We adopt the classical exponential decay model (Goldsmith, 2005):

$$I(s \mid e_i) := I_{\text{base}}\, \mathrm{e}^{-\, d_{\text{eff}}\, \lambda_{\text{base}}},$$

where

$$d_{\text{eff}} = d_{\text{actual}}(1 + \alpha(1 - \cos(\theta))).$$

Here, I_{base} represents the base influence strength, typically determined by an agent's attributes such as health and attack power. d_{eff} is the effective interference distance corresponding to the agent's forward-facing angle θ. α is a parameter that modulates the impact of θ on d_{eff}. d_{actual} is the Euclidean distance between the agent and a point in the surrounding area. λ_{base} is the base decay rate of the potential field's strength.

Let $I(s)$ denote the total interference strength at state s, and $I(s|e_i)$ be the interference from entity e_i (for $i \in [1, n]$) at state s. Then, we have:

$$I(s) = \sum_{i=1}^{n} I(s|e_i).$$

The cooperation cost is then defined as the average per-step cost along the trajectory:

$$C(T_{s_1,s_2} = t|\pi_D, \mathcal{G}^{(t)}) = \frac{1}{t} \sum_{s \in S(T_{s_1,s_2}=t|\pi_D,\mathcal{G}^{(t)})} [1 + I(s)],$$
$$\text{for } t > 0; \quad C(T = 0) := 0.$$

where $S(T_{s_1,s_2} = t|\pi_D)$ denotes the set of states on the t-step trajectory from s_1 to s_2 under policy π_D. In the absence of interference ($I(s) = 0$), d_{IA} reduces to the expected path length $\mathbb{E}[T]$. The value of I_{base} is computed based on real-time sampling of the agent's current state (e.g., health and attack power). The term d_{eff} is optimized using a neural network that predicts a vector of attack intent (Casas et al., 2018). This network is trained via supervised learning to minimize the angular error between its predicted attack direction and the agent's actual trajectory. The resulting angle is then used as θ to compute the interference distance. Details are provided in the Appendix.

4.2 MARL with IA-KRC

Building upon the previously established multi-layer map and interference potential field, we integrate the interference-aware transition cost into a leader-follower framework to enable dynamic collaboration. This framework partitions agents into multiple cooperative groups through distributed election and dynamic affiliation, where each group consists of a core leader and several followers.

The process begins with leader election. To align leader selection with the communication horizon and reduce global dependence, we adopt a K-neighborhood centrality. For each agent i, we first define its K-step reachable neighborhood:

$$\mathcal{N}_i^{(K)} := \{j \in \mathcal{N} \mid d_{IA}(s_i, s_j) \leq K\},$$

and then compute the reachable neighbor count as $N_i^{(K)} = |\mathcal{N}_i^{(K)}|$, where $|\cdot|$ denotes the cardinality of a set. Importantly, computing $N_i^{(K)}$ requires only local information within the K-step neighborhood, as our multi-layer map framework pre-computes and maintains pairwise distances for agents within K steps, thereby avoiding global all-pairs distance computation. Larger $N_i^{(K)}$ indicates that agent i can coordinate with more teammates within the K-step reachable domain, making it a better candidate for leader and reducing isolated agents.

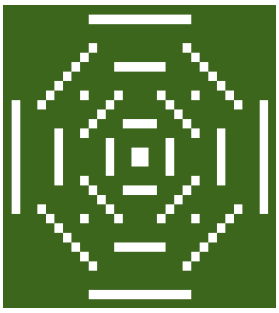
Each agent i computes its own $N_i^{(K)}$ score and exchanges this information with its K-step neighbors. Through local communication within overlapping neighborhoods, agents collectively identify a set of leader candidates with high centrality scores. Since the determination of the K-step reachable domain relies on shortest path computation over the multi-layer map, and the multi-layer map (particularly the Interference Layer) is dynamically updated as the interference prediction module learns, this mechanism can indirectly adapt leader election through learned interference patterns, thereby forming more reasonable communication structures.

Once the set of leaders is established, each non-leader agent (follower) determines its candidate leaders using the same K-step criterion; a leader is a candidate only if $d_{IA}(s_{\text{follower}}, s_{\text{leader}}) \leq K$. Each follower then selects one leader from its candidate set to affiliate with based on a learned assignment policy.

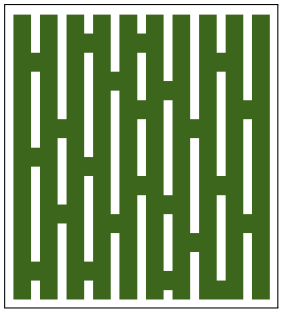
Following group formation, each group $g \in G$ learns a cooperative policy, where G denotes the set of all cooperative groups. This is achieved within a value decomposition framework by training a group-specific joint action-value function Q_{tot}^g to minimize the temporal difference (TD) loss, summed over all groups:

$$\mathcal{L}(\theta) = \sum_{g \in G} \mathbb{E}_{(\boldsymbol{\tau}_g, \mathbf{a}_g, r, \boldsymbol{\tau}'_g) \sim B} \left[(y_g^{\text{tot}} - Q_{\text{tot}}^g(\boldsymbol{\tau}_g, \mathbf{a}_g; \theta))^2 \right].$$

Here, $y_g^{\text{tot}} = r + \gamma \max_{\mathbf{a}'_g} Q_{\text{tot}}^g(\boldsymbol{\tau}'_g, \mathbf{a}'_g; \theta^-)$ is the TD target based on the global reward r, $\boldsymbol{\tau}_g$ and $\mathbf{a}_g$ are the joint history and action for group g, B is the replay buffer, and θ and θ^- are the parameters for the online and target networks, respectively. For more details, please see the Appendix.

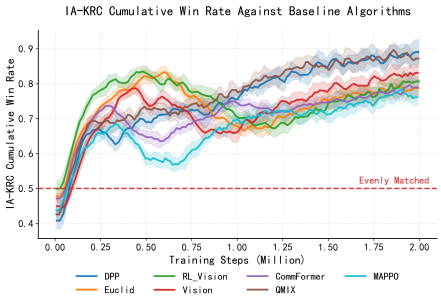


(a) Dense-Obstacle Map.

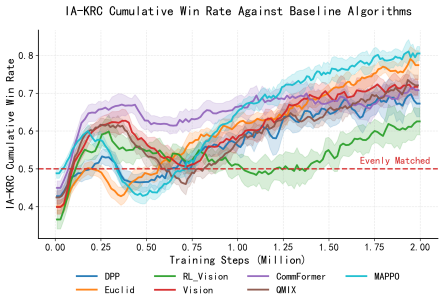


(b) Maze-Structure Map.

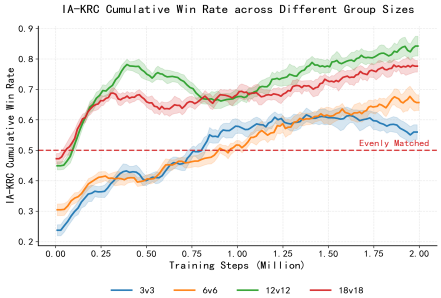
Figure 4: Custom SMACv2 maps. White regions denote obstacles.



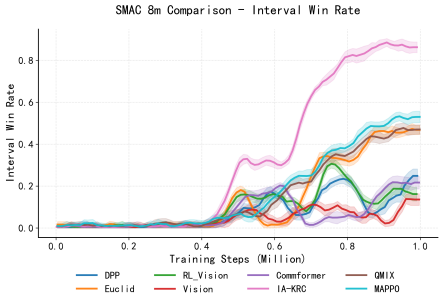
(a) Dense-Obstacle Map.



(b) Maze-Structure Map.



(c) Group Scales.



(d) Standard 8m map.

Figure 5: Learning curves of IA-KRC and baselines over 2.0M steps. (a) Dense-Obstacle. (b) Maze-Structure. (c) Win rate across scales. (d) Standard 8m. (a)-(c): self-play; (d): built-in AI.

5 Experiments

We designed experiments to systematically address the following research questions about IA-KRC's effectiveness, scalability, underlying mechanisms, and generalizability:

- RQ1: (Effectiveness) Does IA-KRC outperform the main baselines in complex dynamic environments?
- RQ2: (Scalability and Structure) How does IA-KRC perform across different team scales and what kind of cooperative structures does it learn?
- RQ3: (Mechanism) What are the relative contributions of the two modules in IA-KRC to the overall performance?
- RQ4: (Generalization) Can IA-KRC retain its advantage in standard scenarios without complex topology?

To answer these questions, unlike existing methods that rely on built-in AI as evaluation baselines, we develop a self-play framework: within the same environment, two teams using different algorithms engage in adversarial play while updating their policies online. We adopt this setup because the built-in AI of the standard SMACv2 environment does not support topological occlusion (vision and attacks are not blocked by obstacles), and its closed-source nature makes rule modifications infeasible, making it difficult to validate cooperative performance under complex topologies. We provide environment and rule details in the supplementary material. The number of surviving agents at the end of each episode determines the winner. The horizon for the K-Step reachability domain is empirically set to $K = 9$ unless otherwise stated. All experiments are repeated 5 times with different random seeds.

5.1 Comparative Performance in Complex Topologies

Experimental Setup. To evaluate the effectiveness of IA-KRC in highly dynamic and topologically complex environments, we construct two custom SMACv2 maps (Figure 4). Figure 4 (a) depicts a symmetrical layout with dense obstacles, requiring complex path planning. Figure 4 (b) adopts a maze-like structure that creates narrow corridors and communication bottlenecks. These maps are specifically designed to amplify the challenges of decentralized coordination and test the adaptability of different communication mechanisms.

We compare IA-KRC against the following baseline methods, among which DPP, Euclid, Vision, RL-Vision, and CommFormer adopt the leader-follower framework for agent grouping:

- **DPP** Yang et al. (2020) groups agents by computing a similarity matrix to ensure diversity.
- **Euclid** Hüttenrauch et al. (2019) uses Euclidean distance between agents as the basis for grouping.
- **Vision(based on SOG)** Shao et al. (2022) forms groups based on mutual visibility and selects leaders randomly.
- **RL-Vision(based on SOG)** Shao et al. (2022) elects leaders via reinforcement learning as an extension of the Vision method.
- **MAPPO** Yu et al. (2022) a centralized-training, decentralized-execution variant of PPO.
- **QMIX** Rashid et al. (2020) a representative value decomposition approach with a monotonic mixing network for cooperative MARL.
- **CommFormer** Hu et al. (2024) employs a graph neural network to learn communication topology end-to-end via the Gumbel-Softmax technique.

Results and Analysis. The learning curves in Figure 5 clearly demonstrate IA-KRC's advantages in both environments. Table 1 reports, for each method, the Dense-side and Maze-side metrics side-by-side (all metrics refer to IA-KRC): FW (final win rate), HW/S (highest win rate/step), and FL (final loss rate), all within 2.0M steps. Since draws can occur in the self-play setting, FW alone may not fully quantify algorithm effectiveness; we therefore also report FL to enable more comprehensive performance comparison.

Method	**Dense-Obstacle Map**			**Maze-Structure Map**		
	FW	**HW/S**	**FL**	**FW**	**HW/S**	**FL**
DPP	88.37 ± 3.12	88.37 ± 3.45 / 0.87M	2.8 ± 1.3	69.63 ± 4.28	69.63 ± 4.10 / 2.00M	15.2 ± 1.2
Euclid	78.81 ± 2.76	83.51 ± 2.94 / 0.61M	2.7 ± 1.2	77.89 ± 3.31	77.89 ± 3.05 / 1.99M	6.3 ± 1.3
Vision	83.51 ± 2.67	83.51 ± 2.43 / 1.99M	2.6 ± 1.1	72.73 ± 3.88	72.73 ± 3.40 / 1.99M	15.0 ± 1.1
RL-Vision	81.11 ± 3.01	83.60 ± 2.58 / 0.49M	7.5 ± 1.4	62.68 ± 4.90	62.68 ± 4.21 / 2.00M	3.8 ± 1.1
CommFormer	79.37 ± 2.25	79.37 ± 2.77 / 1.99M	7.6 ± 1.5	70.40 ± 3.56	70.40 ± 3.12 / 1.99M	15.1 ± 1.2
MAPPO	77.79 ± 2.64	77.79 ± 2.64 / 2.01M	13.2 ± 1.6	81.15 ± 2.8	81.15 ± 2.82 / 2.01M	3.1 ± 1.2
QMIX	88.12 ± 2.51	88.12 ± 2.51 / 2.01M	7.6 ± 1.4	71.98 ± 2.6	71.98 ± 2.69 / 2.00M	3.1 ± 1.1

Table 1: Final cumulative win rate (FW), highest cumulative win rate (HW) within 2.0M steps for Dense-Obstacle and Maze-Structure maps. Additionally, we report final failure rate (FL) corresponding to FW.

On the Dense-Obstacle Map (Figure 5 (a)), IA-KRC achieves final win rates no lower than 77.79% (vs MAPPO), with peaks of 88.37% (vs DPP) and 88.12% (vs QMIX), demonstrating strong late-stage advantage and sustained adaptability.

On the Maze-Structure Map (Figure 5 (b)), IA-KRC remains ahead under harder topology: its final win rate ranges from 62.68% (vs RL-Vision) to 81.15% (vs MAPPO), and it maintains stable margins over Euclid, Vision, CommFormer, and QMIX.

Across both settings, we observe a recurring failure mode in baseline methods: the emergence of *isolated agents* that receive little to no valuable messages, leading to fragmented coordination. This often results in an *avalanche effect*, where early elimination of unsupported agents cascades into total team failure. IA-KRC

mitigates this by explicitly modeling both reachability and interference, dynamically selecting communication peers who are both accessible and strategically viable, thus maintaining cohesive and resilient collaboration throughout the episode.

5.2 Scaling and Structure Study of Multi-Agent Collaboration in IA-KRC

The two central dimensions of multi-agent collaboration are scale and group structure. Scale determines the effectiveness and efficiency of collaboration, whereas organizational structure governs how information is routed, how leader–follower roles are assigned, and whether stable, high-value coalitions can form under environmental dynamics. A mature collaboration framework should preserve a clear organizational form and stable performance at any scale—especially at large scale. Accordingly, this section examines dimensions—scale and organizational structure—and investigates how IA-KRC leverages explicit reachability and interference modeling to sustain high-quality communication and robust coordination as the collaboration setting varies in scale and organization.

Scaling study. To assess scalability, we conduct 2.0 M-step self-play experiments against Vision across team sizes of 3v3, 6v6, 12v12, and 18v18. As shown in Figure 5 (c), IA-KRC achieves 65% (6v6), 86% (12v12) and 79% (18v18) by 2.0 M steps. Notably, the 3v3 scenario peaks at approximately 63% near 1.25 M steps before declining to roughly 56% by 2.0 M steps. This occurs because small team sizes enable both methods to converge toward similar near-optimal grouping structures, thereby reducing IA-KRC's relative advantage. As team size increases, the combinatorial space of group configurations expands factorially. Under this complexity, conventional methods struggle to consistently identify high-quality subgroups, whereas IA-KRC—leveraging reachability filtering and interference-aware priors—systematically constructs robust structures, thereby amplifying its performance advantage at scale.

To further understand this scalability advantage, we conducted a quantitative analysis of the computational complexity in the leader-follower framework. We measured the total computation counts for leader election and follower assignment across varying team sizes (from 4 to 64 agents) on a 64×64 map with obstacles and adversaries. As shown in Figure 6, since our algorithm only computes information within the K-step neighborhood for each agent, the computational complexity remains low. The total computation count grows approximately linearly with team size N, while the per-agent computation remains nearly constant.

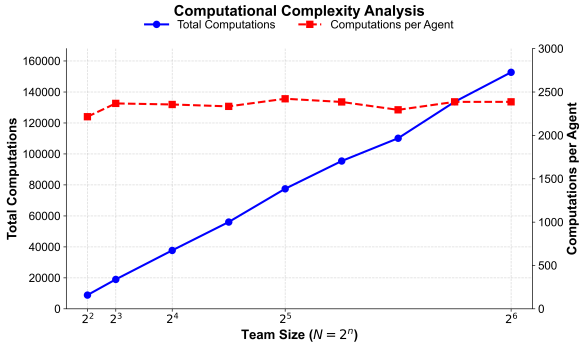


Figure 6: Total computation counts (blue) and computations per agent (red) for leader election and follower assignment.

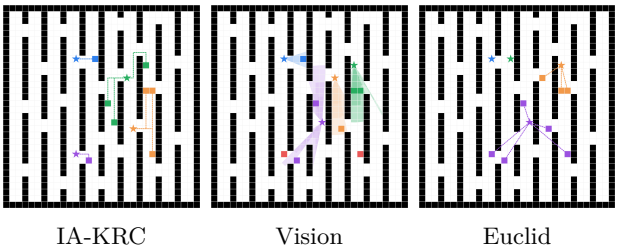


Figure 7: Group structure visualization: (a), (b), and (c) correspond to different algorithms; nodes with same color in same group; star=leader; red=isolated.

Structure study. Beyond scale, we further examine how organizational structure affects cooperation quality. Building on the mainstream hierarchical leader–follower framework (Soni & Hu, 2018; Sheng et al., 2022), we compare alternative leader-election strategies and follower-assignment policies, and quantitatively evaluate the resulting group structures. We adopt two criteria: (1) the proportion of isolated agents, defined as the fraction of timesteps at which an agent has degree 0 in the communication graph, capturing the risk of the avalanche effect; and (2) algebraic connectivity (λ_2), the second-smallest eigenvalue of the Laplacian of each group's communication graph—larger λ_2 indicates stronger connectivity, smoother information flow, and greater robustness (Fiedler, 1973; Mohar, 1991). For completeness, the algebraic connectivity is formally

defined as:

$$\lambda_2(L) = \min_{x \neq 0,\, x^\top \mathbf{1} = 0} \frac{x^\top L x}{x^\top x}.$$

Here, $L = D - A$ is the unnormalized Laplacian of the communication graph, where A is the adjacency matrix and D is the degree matrix; x is any nonzero vector with $x^\top \mathbf{1} = 0$ orthogonal to the all-ones vector $\mathbf{1}$. As shown in Table 2, IA-KRC exhibits a markedly lower isolated-agent ratio than competing methods and achieves a higher mean λ_2 with substantially lower variance, indicating reasonable grouping under complex topology that effectively mitigates the avalanche effect, while demonstrating stronger information connectivity and stable group structures that improve transmission efficiency and yield superior training outcomes. Figure 7 visualizes the grouping structures of different methods: Vision contains many isolated agents, Euclid causes unbalanced resource allocation and inefficient grouping (e.g., cross-obstacle coordination), whereas IA-KRC strikes a better balance and discovers higher-quality structures. RL-Vision reduces variance by learning leader selection, yet vision-based modeling limitations still yield more isolated agents and less efficient groupings. CommFormer learns group assignments via graph neural networks but struggles to quickly discover suitable structures in large, complex environments, resulting in many isolated agents and low spectral connectivity; nevertheless, its inductive bias maintains relatively low variance and comparatively stable structures. Overall, IA-KRC consistently outperforms baselines in the Maze environment;

Method	**Iso Rate**	λ_2 **mean**	λ_2 **var**
IA-KRC	0.0058	0.4622	0.0125
Vision	0.2090	0.2221	0.2295
RL-Vision	0.2009	0.2287	0.1242
Euclid	0.1823	0.2647	0.2198
CommFormer	0.6316	0.0220	0.0714

Table 2: Isolated-agent ratio (Iso Rate), mean algebraic connectivity (λ_2), and its variance (var) in Maze (12v12, 1000 groupings).

Model Variant	K	**FW**
(A) IA-KRC	9	**83.63 ± 2.71**
(B) K-Step Variant	3	75.48 ± 3.02
(C) K-Step Variant	6	79.89 ± 2.95
(D) K-Step Variant	12	73.66 ± 3.87
(E) Without interference	9	74.67 ± 3.15
(F) Without K-Step	None	65.53 ± 4.76

Table 3: FW of IA-KRC ablation variants on the Dense-Obstacle Map at 2.0M training steps, under self-play against Vision.

5.3 Ablation Study of IA-KRC Mechanisms

To assess the contributions of IA-KRC's two core components, we conducted ablation studies on the Dense-Obstacle scenario. Specifically, we vary the reachability horizon K in four settings ($K = 3, 6, 9, 12$) to investigate its impact on collaboration range and prediction stability. In addition, we evaluate the effect of disabling the interference prediction module and disabling the K-step reachability module (replaced by Euclidean distance) respectively, to assess its role in partner selection under dynamic disturbances and complex topologies. All ablation variants were trained in our self-play setting against the Vision baseline; Table 3 reports the IA-KRC variants' final win rate under this setup.

Impact of K-step horizon. Table 3 shows that the final performance at 2.0M steps improves as K increases from 3 (75.48%) to 6 (79.89%), peaking at K=9 (83.63%), but then decreases at K=12 (73.66%). This confirms a nonmonotonic relationship between horizon length and coordination quality. Smaller horizons such as K=3 restrict collaboration to immediate neighbors, limiting access to valued teammates farther away. This leads to lower final performance, even though early-stage results remain competitive. A moderate horizon (K=6) provides a balance of reach and stability, whereas K=9 maximizes midrange partner selection without overextending predictive uncertainty. When K increases to 12, the accumulation of noise in long-range cost estimates and diffuse communication graphs causes performance degradation in later stages. These results indicate that a constrained communication radius is essential for stable and effective grouping under uncertainty.

Impact of interference prediction and K-Step reachability. Model (E), which disables interference prediction while keeping K=9, reaches a final win rate of 74.67%—nearly 9 points lower than the full model. The gap widens as training progresses, as distinguishing safe from risky partners becomes critical. Without interference modeling, agents over-commit to unreliable links (e.g., enemy threats or congestion hotspots).

Model (F), which disables the K-Step reachability module (replacing it with a Euclidean-distance criterion), attains a final win rate of 65.53%, about 18 points lower than the full model. Because K-step reachability degenerates into Euclidean distance, the assessment of actual accessibility becomes overly optimistic in dense-obstacle environments; also, the Euclidean metric weakens the constraining effect of interference prediction on the true transition cost.

By contrast, the full IA-KRC—combining interference-aware partner selection with a K-step reachability constraint on the communication domain—effectively avoids high-cost links, yielding more cohesive and resilient communication structures and markedly stronger late-stage performance and robustness.

The ablation confirms that both the K-Step horizon and interference modeling are indispensable. Horizon length provides temporal foresight, whereas interference prediction safeguards spatial reliability. Together they enable IA-KRC to form high-utility groups, outperforming methods based on proximity or visibility.

5.4 Generalization to Standard Obstacle-Free Environments

To verify generalization, we evaluated IA-KRC on the obstacle-free SMACv2 8m scenario, which lacks topological constraints. We train against the built-in AI instead of using the self-play framework. We compared IA-KRC with Euclid, Vision, and CommFormer using the same training budget and architecture. As shown in Figure 5 (d), IA-KRC maintains superiority even in this simplified environment, where no obstacles exist and communication complexity is reduced. Despite the simplicity of the setting, IA-KRC achieves faster convergence and attains a higher win rate than all baselines.

This result is noteworthy because, in obstacle-free settings, the K-Step reachability constraint often approximates Euclidean distance under continuous, isotropic motion; with grid-based motion primitives it instead aligns with Manhattan or Chebyshev metrics. Nonetheless, IA-KRC continues to outperform Euclid, indicating that its advantage stems not only from geometric constraints but also from its interference-aware modeling. The dynamic influence map enables agents to account for crowding, conflict zones, and temporal instability among teammates—factors that emerge even without physical obstacles.

In contrast, Euclid achieves moderate performance with its aggressive yet static grouping strategy that fails to adapt to evolving dynamics. Vision remains overly conservative, limiting coordination gains. CommFormer improves slowly early on, as end-to-end graph learning introduces overhead in simple environments. RL-Vision exhibits unstable coordination; occasional mid-training peaks are overshadowed by sharp fluctuations and lack of sustained competitiveness. DPP delivers stable but modest results—diversity-based selection prevents severe collapses yet lacks IA-KRC's adaptiveness to exploit favorable configurations. MAPPO and QMIX perform reasonably in the 8m setting but still fall short of IA-KRC. These results confirm that IA-KRC's core mechanisms—reachability filtering and interference prediction—generalize effectively beyond complex topologies, providing robust benefits even under minimal structural constraints.

6 Conclusion

We proposed IA-KRC, a hierarchical communication mechanism for multi-agent reinforcement learning that integrates K-Step reachability and interference-aware modeling to form effective, low-conflict collaboration structures. Central to our approach is a multi-layer map framework that decouples slow-changing environmental topology from rapidly evolving agent adversarial dynamics, ensuring real-time computation of interference-aware K-step reachable distances to enable effective cooperation.

Extensive experiments across diverse SMACv2 scenarios validate the effectiveness and generality of IA-KRC. In complex environments with Dense-Obstacle and Maze-Structure settings, IA-KRC consistently outperformed baselines such as CommFormer and MAPPO, converging faster and achieving higher win rates. Even in the standard 8m scenario without obstacles, IA-KRC maintained an advantage over Euclidean baselines, highlighting its robust generalization beyond structural complexity. These results demonstrate that IA-KRC is not only effective under spatial constraints but also excels in capturing behavioral interference in open environments.

References

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. In *International conference on learning representations*, 2019.

Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *Conference on robot learning*, pp. 947–956. PMLR, 2018.

Pavel Chebotarev and Elena Deza. Hitting time quasi-metric and its forest representation. *Optimization Letters*, 14(2):291–307, 2020.

Jiajia Chen, Bingqing Zhu, Mengyu Zhang, Xiang Ling, Xiaobo Ruan, Yifan Deng, and Ning Guo. Multi-agent deep reinforcement learning cooperative control model for autonomous vehicle merging into platoon in highway. *World Electric Vehicle Journal*, 16(4):225, 2025.

Jingxi Chen, Amrish Baskaran, Zhongshun Zhang, and Pratap Tokekar. Multi-agent reinforcement learning for visibility-based persistent monitoring. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2563–2570. IEEE, 2021.

Filippos Christianos, Georgios Papoudakis, Muhammad A Rahman, and Stefano V Albrecht. Scaling multi-agent reinforcement learning with selective parameter sharing. In *International Conference on Machine Learning*, pp. 1989–1998. PMLR, 2021.

Yuxuan Cong, Shuai Li, Guoliang Zhang, Yang Liu, and Bo Li. Cds: A channel-wise differentiable search for dynamic and sparse communication in marl. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1): 269–271, 1959.

Gang Ding, Zeyuan Liu, Zhirui Fang, Kefan Su, Liwen Zhu, and Zongqing Lu. Multi-agent coordination via multi-level communication. *Advances in Neural Information Processing Systems*, 37:118513–118539, 2024.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.

Kailash Gogineni, Peng Wei, Tian Lan, and Guru Venkataramani. Scalability bottlenecks in multi-agent reinforcement learning systems. *arXiv preprint arXiv:2302.05007*, 2023.

Andrea Goldsmith. *Wireless communications*. Cambridge university press, 2005.

Cong Guan, Feng Chen, Lei Yuan, Chenghe Wang, Hao Yin, Zongzhang Zhang, and Yang Yu. Efficient multi-agent communication via self-supervised information aggregation. *Advances in Neural Information Processing Systems*, 35:1020–1033, 2022.

Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Learning multi-agent communication from graph modeling perspective. In *International Conference on Learning Representations*, 2024.

Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. Deep reinforcement learning for swarm systems. *Journal of Machine Learning Research*, 20(54):1–31, 2019.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *Advances in neural information processing systems*, 31, 2018.

Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph convolutional reinforcement learning. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=HkxdQkSYDB`.

Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

Xiaomin Liao, Yulai Wang, Yang Han, You Li, Chushan Lin, and Xuan Zhu. Heterogeneous multi-agent deep reinforcement learning for cluster-based spectrum sharing in uav swarms. *Drones*, 9(5):377, 2025.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Jiahe Mei, Yang Liu, and Bo Li. Graph-based communication with transformers for multi-agent reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Xiangrui Meng and Ying Tan. Pmac: Personalized multi-agent communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17505–17513, 2024.

Bojan Mohar. The laplacian spectrum of graphs. In Y. Alavi, G. Chartrand, O. R. Oellermann, and A. J. Schwenk (eds.), *Graph Theory, Combinatorics, and Applications*, volume 2, pp. 871–898. Wiley, 1991.

Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Navid Naderializadeh, Fan H Hung, Sean Soleyman, and Deepak Khosla. Graph convolutional value decomposition in multi-agent reinforcement learning. *arXiv preprint arXiv:2010.04740*, 2020.

Yifan Niu, Rohan R Paleja, and Matthew C Gombolay. Multi-agent graph-attention communication and teaming. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2021.

Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

Shayegan Omidshafiei, Daniel Hennes, Marta Garnelo, Zhe Wang, Adria Recasens, Eugene Tarassov, Yi Yang, Romuald Elie, Jerome T Connor, Paul Muller, et al. Multiagent off-screen behavior prediction in football. *Scientific reports*, 12(1):8638, 2022.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

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

Jianzhun Shao, Zhiqiang Lou, Hongchang Zhang, Yuhang Jiang, Shuncheng He, and Xiangyang Ji. Self-organized group for cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 35:5711–5723, 2022.

Junjie Sheng, Xiangfeng Wang, Bo Jin, Junchi Yan, Wenhao Li, Tsung-Hui Chang, Jun Wang, and Hongyuan Zha. Learning structured communication for multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 36(2):50, 2022.

Philipp Dominic Siedler. The power of communication in a distributed multi-agent system. *arXiv preprint arXiv:2111.15611*, 2021.

Aakash Soni and Huosheng Hu. Formation control for a fleet of autonomous ground vehicles: A survey. *Robotics*, 7(4):67, 2018.

Jianyu Su, Stephen Adams, and Peter A Beling. Counterfactual multi-agent reinforcement learning with graph convolution communication. *arXiv preprint arXiv:2004.00470*, 2020.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pp. 2085–2087, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International conference on machine learning*, pp. 3540–3549. PMLR, 2017.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

Yaodong Yang, Ying Wen, Liheng Chen, Jun Wang, Kun Shao, David Mguni, and Weinan Zhang. Multi-agent determinantal q-learning. In *International conference on machine learning*, pp. 10693–10702. PMLR, 2020.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems*, 35:24611–24624, 2022.

Tianren Zhang, Shangqi Guo, Tian Tan, Xiaolin Hu, and Feng Chen. Generating adjacency-constrained subgoals in hierarchical reinforcement learning. *Advances in neural information processing systems*, 33: 21579–21590, 2020.

Changxi Zhu, Mehdi Dastani, and Shihan Wang. A survey of multi-agent deep reinforcement learning with communication. *Autonomous Agents and Multi-Agent Systems*, 38(1):4, 2024.

APPENDIX

This appendix provides detailed specifications for the key components of the Interference-Aware K-step Reachable Communication (IA-KRC) framework, including its core algorithm, the neural network for interference prediction, and the hyperparameters used in the experiments.

A.1 Implementation Details of the IA-KRC Algorithm

Key Variables and Functions. Before presenting the algorithm, we define the key functions and variables used:

Variables:

- $d_{IA}(s_1, s_2)$: Interference-aware shortest transition distance (Definition 2)
- $\mathcal{S}_{IA}(s_1, K)$: Interference-aware K-step reachable region (Definition 3)
- $N_i^{(K)}$: Reachable neighbor count within K-step domain for agent i
- $\mathcal{L}$: Set of elected leaders
- $\mathcal{F}$: Set of followers (non-leader agents)

Functions:

- `update_from_sight`(): Updates geometry layer with visual observations
- `update`() on L_r: Updates regulation layer using agent transitions
- `confidence_refresh`$(\cdot; L_g, L_r)$: Confidence-driven asynchronous updater over layers
- `interference_prediction_module`(): Produces interference costs for L_i
- `aggregate_graph`(): Aggregates geometry obstacles, regulation constraints, and interference weights; sets obstacle $\Rightarrow \infty$, free base $= 1$, and non-obstacle edges $w(u, v) \leftarrow 1 + f_{\text{influence}}(M, v)$ when interference is enabled
- `dijkstra_reachable`(): Shortest-path query on aggregated graph to get $\mathcal{S}_{IA}(s, K)$

The complete IA-KRC algorithm is presented in Algorithm 1.

Algorithm 1 IA-KRC Algorithm

Input: Agent states $\{s_i\}_{i=1}^{N}$, enemy observations E_{obs}, environment observations env_{obs}, parameters K, M
Output: Collaborative groups $G = \{G_1, G_2, ..., G_M\}$
{Phase 1: Multi-Layer Map Update}
Initialize layers: L_g, L_r, L_i
Extract `agent_pos`, `map_matrix` from observations
L_g.`update_from_sight`(`agent_pos`, `map_matrix`, `sight_range`)
L_r.`update`(`agent_transitions`)
`confidence_refresh`(`experience_stats`; L_g, L_r)
{Phase 2: Interference Prediction}
if interference enabled **then**
  `interference_costs` ← `interference_prediction_module`(E_{obs}, `agent_pos`)
  L_i.`update`(`interference_costs`)
end if
{Phase 3: Graph Aggregation and Reachability}
$\mathcal{G}_{agg}$ ← `aggregate_graph`(L_g, L_r, L_i)
for each agent i **do**
  $\mathcal{S}_{IA}(s_i, K)$ ← `dijkstra_reachable`($\mathcal{G}_{agg}$, s_i, K)
  $N_i^{(K)} \leftarrow |\{\, j \neq i : s_j \in \mathcal{S}_{IA}(s_i, K) \,\}|$
end for
{Phase 4: Leader Election and Follower Assignment}
$\mathcal{L}$ ← top M agents with highest $N_i^{(K)}$ scores
Initialize groups: $G_l \leftarrow \{l\}$ for each $l \in \mathcal{L}$
$\mathcal{F} \leftarrow \{1, ..., N\} \setminus \mathcal{L}$
for each follower $f \in \mathcal{F}$ **do**
  $\mathcal{L}_{cand} \leftarrow \{l \in \mathcal{L} : s_f \in \mathcal{S}_{IA}(s_l, K)\}$
  if $\mathcal{L}_{cand} \neq \emptyset$ **then**
    $l^* \leftarrow \arg\min_{l \in \mathcal{L}_{cand}} |G_l|$
    Add f to group G_{l^*}
  end if
end for
return $G = \{G_1, G_2, ..., G_M\}$

A.2 Multi-Layer Map Module

Geometry Layer. Maintains physical connectivity on a grid using agents' visual observations and line-of-sight within a circular perception radius. Newly observed obstacles and free cells update only local neighborhoods; a full grid rebuild is triggered only when novel obstacles are discovered. This layer exposes `update_from_sight` to incrementally refresh obstacle cells, visited cells, and adjacency.

Algorithm 2 Geometry layer: LOS-based incremental update

Input: agent position p, map matrix M, sight range R
for each cell q with $\|q - p\| \leq R$ and LOS(p, q) **do**
  if $M[q] = -1$ **then**
    mark q as obstacle; update local adjacency
  else
    mark q as visited free cell; update local adjacency
  end if
end for
If new obstacles found: rebuild local grid connections; otherwise keep incremental updates

Regulation Layer. Stores environment rule-based connectivity inferred from actual agent transitions (e.g., doors, traffic rules). At each step and for each agent, it logs the tuple $(s_t, a_t, s_{t+1}, \texttt{success})$ derived from the chosen action and observed position change. If `success` is true, it updates a directed edge $s_t \to s_{t+1}$ in the regulation graph; failures are forwarded to the confidence-driven asynchronous mechanism for localized revalidation without directly modifying raw geometry.

Algorithm 3 Regulation layer: transition logging with success/failure

Input: time t, agent id i, current state s_t, action a_t, next state s_{t+1}, flag success
Append $(i, t, s_t, a_t, s_{t+1}, success)$ to transition_log
if success **then**
  if edge (s_t, s_{t+1}) not in $_adj$ **then**
    add directed edge $s_t \to s_{t+1}$ with base weight $w(s_t, s_{t+1}) \leftarrow 1$
  else
    optionally update running statistics for (s_t, s_{t+1})
  end if
  send record to confidence mechanism: update_stats(edge=(s_t, s_{t+1}), succ←1, total←1)
else
  send record to confidence mechanism: update_stats(edge=(s_t, s_{t+1}), succ←0, total←1)
end if

Confidence-Driven Asynchronous Mechanism. Tracks per-edge reliability statistics (success/total), maintains a FIFO of recently blocked candidates, and triggers localized refresh based on a confidence threshold τ_c and decay factor η_{upd}. When reliability for edge (u, v) falls below the threshold (e.g., persistent failures), it marks (u, v) as a blocked candidate and schedules a future revalidation with decayed frequency; otherwise the edge remains usable. This mechanism operates externally to the three layers (Geometry, Regulation, Interference) as an asynchronous updater, supplying obstacle candidates to aggregation without touching raw geometry.

Algorithm 4 Confidence-driven asynchronous update: reliability-based blocking with FIFO

Input: stats map $S[(u, v)] = (succ, total)$, threshold τ_c, decay η_{upd}, FIFO queue Q
for each edge (u, v) in S **do**
  $r \leftarrow \frac{succ}{\max(1, total)}$
  if $r < \tau_c$ **then**
    mark (u, v) as blocked; push (u, v) into FIFO Q (evict oldest if capacity reached)
    schedule revalidation time $t_{\text{next}} \propto \eta_{\text{upd}}$
  else
    keep (u, v) active (unblocked)
  end if
end for

Graph Aggregation and Query. Aggregation combines: (i) geometry-layer obstacles, (ii) regulation-layer constraints and blocked candidates proposed by the confidence mechanism (treated as obstacles), and (iii) interference weights from Section A.3. The base edge weight policy is: obstacle/non-transferable $\Rightarrow \infty$; otherwise base $= 1$. With interference enabled, the influence map M (see Section A.3) assigns costs to all non-obstacle transitions by starting from 1 and then computing $w(u, v) \leftarrow 1 + f_\text{influence}(M, v)$ per the formula, yielding dynamic traversal costs. Consequently, the aggregated result satisfies: free space has base 1, obstacles are ∞, and influenced regions are data-dependent.

Algorithm 5 Aggregate-and-Query under policy π_D (geometry + regulation + interference)

Build obstacle set $\mathcal{O} \leftarrow$ geometry_obstacles $\cup$ regulation_constraints $\cup$ confidence_blocked_candidates

for each candidate edge (u, v) **do**
  if $(u, v) \in \mathcal{O}$ **then**
    $w(u, v) \leftarrow \infty$
  else
    $w(u, v) \leftarrow 1$
  end if
end for
if interference enabled **then**
  obtain influence map M from Section A.3; for non-obstacle (u, v) set $w(u, v) \leftarrow 1 + f_\text{influence}(M, v)$ (cf. A.3)
end if
Run Dijkstra with weights w to obtain costs and $\mathcal{S}_{IA}(s, K)$
Return reachable set and costs

A.3 Interference Prediction Module

The Interference Prediction Module quantifies the traversal risk by modeling the influence of adversarial agents as a potential field. In this field, high-threat enemies generate high-cost regions. The module's core is the calculation of a potential field for each enemy, which incorporates directional influence via a predicted attack intent angle θ. This influence is modulated by a threat level, which is heuristically determined from the enemy's current state and recent actions. As detailed in Algorithm 6, a neural network predicts an attack intent vector for each enemy to derive the angle θ. The resulting path cost map informs the IA-KRC framework's reachability calculations and agent coordination. Figure 8 shows an example of the dynamically computed transition cost map.

Key Variables and Functions for Algorithm 2

Variables:

- E_{obs}: Visual observations of enemies.
- $M_{influence}$: A grid representing the cumulative enemy influence across the map.
- M_{cost}: A grid representing the pathfinding cost for each location.
- e: An individual enemy entity.
- I_{base}: Dynamically computed base influence strength for an enemy.
- λ_{base}: Influence decay rate (hyperparameter).
- α: Angle influence factor (hyperparameter).
- θ_e: Predicted attack intent angle for enemy e.
- $d_{\text{eff}}(p_1, p_2, \theta_e)$: Effective distance considering the predicted attack angle.
- $d_{\text{actual}}(p_1, p_2)$: Euclidean distance between two points.
- *cost_multiplier*: A factor to scale influence into path cost (hyperparameter).

Functions:

- `extract_enemies`(E_{obs}): Parses observations to get a list of enemy entities and their states.

- `calculate_influence`(e): Computes the dynamic base influence strength I_{base} of an enemy based on its attributes (e.g., health, recent actions).
- `predict_attack_intent`(e): Predicts the attack intent angle θ_e for an enemy using a neural network.
- `normalize_map`(M): Normalizes map values to a standard range.

Algorithm 6 Interference Prediction and Cost Calculation

Input: Enemy observations E_{obs}, hyperparameters $\alpha, \lambda_{\text{base}}$
Output: Path cost map M_{cost}
{Phase 1: Initialize Maps}
Initialize $M_{influence}$ with zeros.
Initialize M_{cost} with ones.
{Phase 2: Calculate Enemy Influence}
$Enemies \leftarrow$ `extract_enemies`(E_{obs})
for each enemy e in $Enemies$ **do**
  $I_{\text{base}} \leftarrow$ `calculate_influence`(e) {Compute influence from state}
  $\theta_e \leftarrow$ `predict_attack_intent`(e) {Predict directional intent}
  for each cell p on the map within influence range of e **do**
    $d_{\text{actual}} \leftarrow d_{\text{actual}}(p, e.position)$
    $d_{\text{eff}} \leftarrow d_{\text{actual}} \times (1 + \alpha(1 - \cos(\theta_e)))$ {Calculate effective distance}
    $I(p|e) \leftarrow I_{\text{base}} \times \exp(-\lambda_{\text{base}} \times d_{\text{eff}})$
    $M_{influence}[p] \leftarrow M_{influence}[p] + I(p|e)$
  end for
end for
{Phase 3: Compute Path Cost Map}
$M_{cost} \leftarrow 1.0 + cost_multiplier \times M_{influence}$ {Convert influence to traversal cost}
For each obstacle position p_{obs}: $M_{cost}[p_{obs}] \leftarrow \infty$
$M_{cost} \leftarrow$ `normalize_map`(M_{cost})
return M_{cost}

Neural Network and Parameter Details

This section provides further implementation details for the key components of the Interference Prediction Module.

Neural Network for Attack Intent Prediction. The attack intent angle θ_e for each enemy is derived from a predicted attack vector, which is generated by a dedicated neural network. This network takes an enemy's state as input and outputs a 2D vector representing its most likely direction of attack. The network architecture is a feed-forward Multi-Layer Perceptron (MLP) with two hidden layers:

- **Input Layer:** A flattened vector representing the enemy's state, including its recent trajectory (last 10 positions) and current health.
- **Hidden Layers:** The first hidden layer consists of 128 neurons with a ReLU activation function, followed by a second hidden layer of 64 neurons, also with ReLU activation.
- **Output Layer:** A linear layer with 2 neurons, producing the (x, y) components of the predicted attack intent vector.

Training Process. The attack intent prediction network is trained via supervised learning on data collected during simulation. For each enemy at each time step, we create a training sample consisting of its current state (the network input) and its actual movement vector over the next time step (the ground truth). The network is trained to minimize the angular difference between its predicted attack vector $\mathbf{v}_{\text{pred}}$ and the

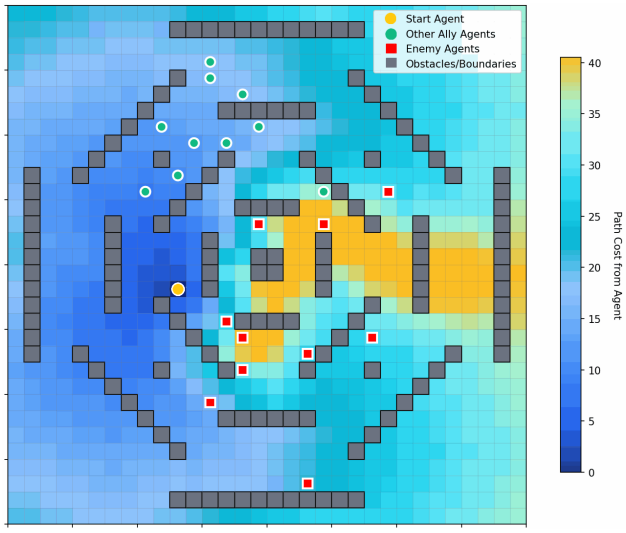


Figure 8: Transition cost map computed from the start agent (yellow) after multi-layer map and interference map processing. Blue regions indicate low traversal cost due to high ally density and minimal enemy interference, while yellow regions represent high traversal cost caused by distant locations and strong adversarial agent interference.

ground truth movement vector $\mathbf{v}_{\text{true}}$. The loss function is defined as the negative cosine similarity:

$$\mathcal{L}_{\text{intent}} = 1 - \frac{\mathbf{v}_{\text{pred}} \cdot \mathbf{v}_{\text{true}}}{\|\mathbf{v}_{\text{pred}}\| \|\mathbf{v}_{\text{true}}\|}$$

Training is performed using the Adam optimizer with a learning rate consistent with the main policy training (see Table 4).

Dynamic Influence Calculation. The base influence strength I_{base} for each enemy is computed dynamically via the `calculate_influence`(e) function in Algorithm 2:

$$I_{\text{base}} = I_{\text{config}} \times T_e$$

where I_{config} is the configured base influence strength (default 2.0), and T_e is the threat level: $T_e = \frac{T_{\text{move}}+T_{\text{attack}}+T_{\text{health}}+T_{\text{mobility}}}{4}$, with $T_{\text{move}} = \min(1.0, |\text{trajectory}|/50.0)$, $T_{\text{attack}} = \min(1.0, |\text{attacks}|/10.0)$, $T_{\text{health}} = \text{normalized_health}$, and $T_{\text{mobility}} = \min(1.0, \bar{d}_{\text{recent}}/3.0)$.

Effective Distance (d_{eff}). The effective distance $d_{\text{eff}} = d_{\text{actual}}(1+\alpha(1-\cos(\theta_e)))$ creates a forward-facing cone of influence, where θ_e is the angle between the predicted attack intent vector and the vector from enemy position to location p. The hyperparameter α controls the directional effect strength.

A.4 IA-KRC Training Details in Multi-Agent Reinforcement Learning

The IA-KRC framework is integrated within a multi-agent reinforcement learning (MARL) system that employs value decomposition methods for cooperative policy learning. This section details the training architecture and implementation specifics of how IA-KRC's dynamic grouping mechanism is incorporated into the MARL training pipeline, extending traditional value-based methods to accommodate interference-aware coordination.

Group-Specific Value Function. The core of our framework is the monotonic decomposition of the group's joint action-value function. Q^g_{tot} is represented as a monotonic combination of the individual utility

functions Q_i for all agents $i \in g$. This property is crucial for decentralized execution, as it guarantees that a greedy action selection by each agent based on its local Q_i corresponds to the maximization of the joint Q^g_{tot}. The relationship is formalized as $Q^g_{\text{tot}} = f_g(\{Q_i\}_{i \in g}, s)$, where f_g is a mixing network specific to group g and conditioned on the global state s. This architecture implies that each group effectively learns its own cooperative policy, tailored to its members and the current state, while still contributing to the global team objective.

Agent Architecture. Each agent utilizes a recurrent neural network (RNN) with an attention mechanism. This network processes the agent's local observation history τ_i to maintain a hidden state h^t_i. At each timestep, the RNN's input includes the agent's current observation, its previous action, and any messages received from its group leader. The resulting hidden state h^t_i is then fed into a feed-forward layer to compute the per-action Q-values $Q_i(\tau_i, \cdot)$.

Mixing Network. The framework employs a mixing network architecture, specified as `flex_qmix` in the configuration, to combine individual Q_i values into the joint Q^g_{tot}. Consistent with QMIX, the weights of this mixing network are generated by a hypernetwork that takes the global state s as input, allowing the mixing function to adapt to different environmental conditions. To enhance training stability, the mixing weights are normalized using a softmax function.

Training Process. The system is trained end-to-end by minimizing the total TD loss, summed across all dynamically formed groups, as defined in the main paper. Experiences are collected using parallel runners and stored in a replay buffer. The learner (`msg_q_learner`) samples mini-batches to perform updates. The TD target for each group g is computed as:

$$y^{\text{tot}}_g = r + \gamma \max_{\mathbf{a}'_g} Q^g_{\text{tot}}(\boldsymbol{\tau}'_g, \mathbf{a}'_g; \theta^-)$$

where θ and θ^- are the parameters of the online and target networks, respectively. The target network is periodically updated with the online network's parameters every 200 episodes.

A.5 Hyperparameter Settings

Table 4: Key Hyperparameter Settings

Parameter	Value	Parameter	Value	Parameter	Value
General RL Parameters					
Learning Rate	5e-4	Optimizer	Adam	Discount Factor (γ)	0.99
Batch Size	32	Replay Buffer Size	5000	Target Update Interval	200 ep.
Epsilon Start	1.0	Epsilon Finish	0.05	Epsilon Anneal Time	500k steps
Experience Buffer Size	10000	Opponent Learning Rate	1e-3		
IA-KRC Framework and Interference Prediction					
K-step Horizon (K)	9	Number of Leaders (N_L)	3	Cost Multiplier	1.5
Interference Decay (λ_{base})	0.3	Angle Influence (α)	0.5	Influence Range	5.0
Base Influence Strength	2.0	Intent Net Hidden Dim 1	128	Intent Net Hidden Dim 2	64
Agent and Mixer Architecture					
RNN Hidden Dim	64	Attention Heads	4	Hypernet Embed Dim	128
Mixing Net Dim	32				

A.6 Avalanche Effect Demonstration

This section demonstrates the avalanche effect through a competitive evaluation between IA-KRC and Vision algorithms after 2.0M training steps. The red team represents Vision agents, while the green team represents IA-KRC agents. The following four stages illustrate how initial tactical advantages can cascade into decisive victory through the avalanche effect mechanism.

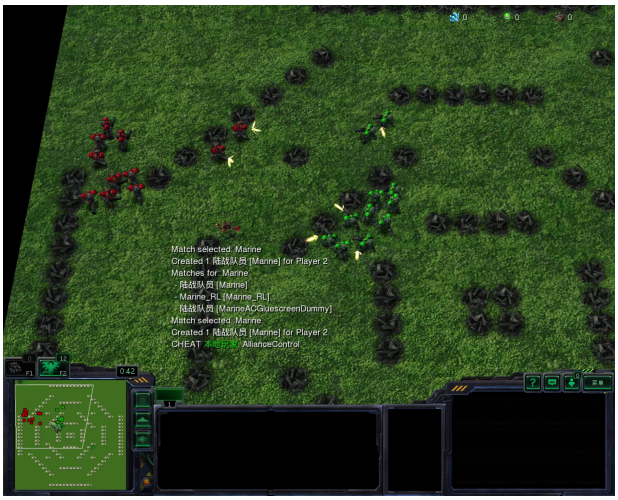


Figure 9: Stage 1: Due to vision limitations, the red team (Vision) forms two isolated agents. These isolated agents cannot communicate or coordinate with other teammates, resulting in low collaboration efficiency and becoming surrounded by the green team (IA-KRC).

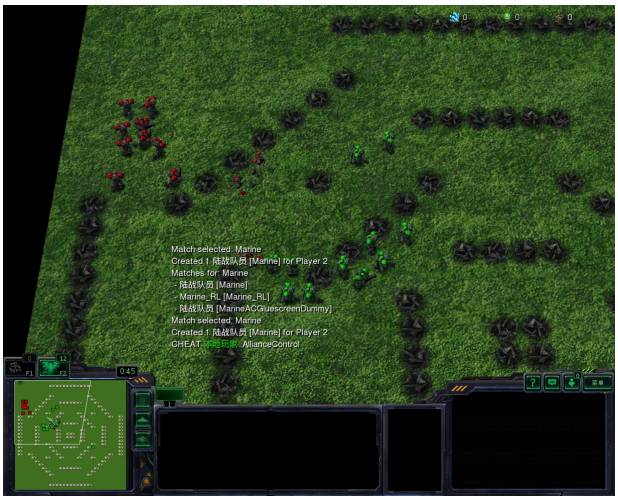


Figure 10: Stage 2: The isolated Vision agents are eliminated through concentrated fire from IA-KRC (IA-KRC loses 1 agent while Vision loses 3 agents), creating an 11v9 situation. The avalanche effect begins to manifest.

Figure 11: Stage 3: Leveraging the numerical advantage, IA-KRC adopts an aggressive strategy. With superior firepower, they continuously expand the combat capability gap.

Figure 12: Stage 4: IA-KRC achieves complete victory, eliminating all Vision agents at the cost of only 4 agents, demonstrating the effectiveness of interference-aware coordination in multi-agent combat scenarios.