CAIC: Congestion-Aware Intent Communication for Multi-Agent Reinforcement Learning

Pei Li[1] **Yongkang Zhang**[1] **Zhonglin Lv**[1] **Jinmin Zhu**[1] **Jiangjin Yin**[*1]

[1]College of Informatics, Huazhong Agricultural University, Key Laboratory of Smart Farming for Agricultural Animals, Hubei Engineering Technology Research Center of Agricultural Big Data, and Engineering Research Center of Intelligent Technology for Agriculture, Wuhan 430070, China

Abstract

In multi-agent reinforcement learning, communication is essential for effective cooperation. Most existing methods assume instantaneous message delivery, and the few delay-aware studies consider only external fixed or stochastic delays, neglecting queueing delays caused by competition for a shared channel. We formulate the shared channel as a queueing system with state-dependent service rates, and propose Congestion-Aware Intent Communication (CAIC). To ensure message validity under delay, CAIC employs a temporal masked autoencoder to predict each agent's future trajectory and encode it into a delay-robust intent message, with attention-based fusion integrating asynchronous messages at the receiver. To maintain timeliness, CAIC dynamically adjusts communication frequency based on intent message changes and delay estimates, proactively mitigating congestion. Experiments on Hallway, MPE, and SMAC demonstrate that CAIC outperforms existing baselines on most evaluated scenarios in both no-delay and queueing-delay settings, and ablation studies reveal that stale messages under delay can be even more harmful than no communication.

1 INTRODUCTION

Multi-agent reinforcement learning (MARL) has been widely applied to domains such as autonomous driving [Shalev-Shwartz et al., 2016], game AI [Vinyals et al., 2019], and UAV swarm coordination [Xia et al., 2022]. A fundamental challenge in these settings is partial observability, where each agent can only access local information, making effective coordination difficult. Agent communication addresses this challenge by enabling information sharing, and has become a central research topic in cooperative MARL [Foerster et al., 2016, Sukhbaatar et al., 2016].

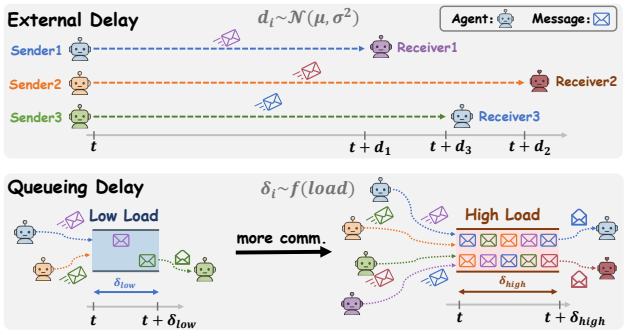

Figure 1: Motivation for investigating queueing delay. Unlike random external delays, queueing delay is endogenously determined by agents' collective communication behaviors.

Existing studies explore efficient communication strategies along three dimensions, namely learned communication scheduling and gating policies to decide *when* to communicate [Singh et al., 2019, Kim et al., 2019, Wang et al., 2020]; dynamic graph structures to select *with whom* to communicate [Niu et al., 2021, Hu et al., 2024, Zhang et al., 2025]; and tailored message content design for *what* to communicate [Kim et al., 2021, Sun et al., 2024]. However, these methods universally assume instantaneous message delivery, neglecting the communication delays inevitable in real-world deployment.

A few studies have begun to consider communication delay [Yuan et al., 2023, Song et al., 2025], but treat it as a fixed or stochastic external factor. In practice, agents typically share a bandwidth-limited communication channel, and simultaneous transmissions lead to congestion and queueing delay. In contrast to external delays that are independent of agents' behavior, queueing delay is endogenously determined by agents' collective communication decisions and may grow rapidly with communication frequency (Figure 1). This endogeneity implies that queueing delay can be

*Corresponding author.

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:3705–3718.

mitigated through coordinated communication scheduling, rather than treated as an exogenous constraint.

In this paper, we adopt a state-dependent service rate model grounded in queueing theory to describe the congestion dynamics of the shared channel, where the service rate degrades nonlinearly once the channel load exceeds a congestion threshold. Based on the insight of delay's endogeneity, we formulate multi-agent cooperation under queueing delay as a Congestion-Aware Decentralized Partially Observable Markov Decision Process (CA-Dec-POMDP) and propose the Congestion-Aware Intent Communication (CAIC) framework. CAIC tackles queueing delay from two perspectives. For message validity, conventional messages such as hidden states become stale upon delayed arrival, whereas predictions of future behavior remain informative. Accordingly, we employ a temporal masked autoencoder to predict each agent's future action trajectory from its historical trajectory and encode it as an intent message that preserves coordination value despite queueing delay. For timeliness, we introduce a time-varying Exponential Moving Average (EMA) delay estimator to track channel delay and dynamically adjust communication frequency based on intent message changes and delay estimates, balancing information demand against congestion cost.

Our main contributions are summarized as follows:

- To our knowledge, we are the first to systematically study endogenous queueing delay induced by shared-channel congestion in MARL communication, and adopt a state-dependent service rate model grounded in queueing theory.
- We introduce a delay-robust intent encoding mechanism that preserves message validity through future trajectory prediction, and a congestion-aware communication decision scheme driven by a time-varying delay estimator.
- We validate CAIC on Hallway, MPE, and SMAC, demonstrating that CAIC outperforms existing baselines on most evaluated scenarios in both no-delay and queueing-delay settings; ablations further reveal that stale messages under delay can be even more harmful than no communication.

2 RELATED WORK

Multi-Agent Reinforcement Learning. Multi-agent reinforcement learning (MARL) extends single-agent reinforcement learning (RL) [Mnih et al., 2015] to settings where multiple agents interact in a shared environment. A core challenge is credit assignment, as agents receive only a shared team reward yet must assess each individual's contribution. The Centralized Training with Decentralized Execution (CTDE) paradigm addresses this challenge by accessing global state and joint actions during training and attributing rewards to individual agents through a centralized critic or value decomposition network. Representative methods include QMIX [Rashid et al., 2020], COMA [Foerster et al., 2018], and MADDPG [Lowe et al., 2017]. However, such methods require agents to rely solely on local observations during execution, limiting effective coordination.

Multi-Agent Communication. Multi-agent communication enables agents to share information during execution. Early methods such as CommNet [Sukhbaatar et al., 2016], DIAL [Foerster et al., 2016], and TarMAC [Das et al., 2019] pioneer end-to-end differentiable communication, allowing all agents to broadcast messages at every timestep. Subsequent work improves communication efficiency along three dimensions. For *when* to communicate, NDQ [Wang et al., 2020] learns to minimize communication through information-theoretic regularization, TMC [Zhang et al., 2020] reduces frequency via temporal windows, and ETC-Net [Hu et al., 2023] introduces event-triggered messaging. For *with whom* to communicate, SOG [Shao et al., 2022] forms self-organized groups through conductor election, CommFormer [Hu et al., 2024] models the communication architecture as a learnable graph, and TGCNet [Zhang et al., 2025] learns dynamic directed communication graphs through a gated adjacency mechanism. For *what* to communicate, IS [Kim et al., 2021] transmits attention-compressed imagined trajectories as messages, CACOM [Li and Zhang, 2024] generates receiver-specific messages through context-aware attention, and T2MAC [Sun et al., 2024] constructs messages from local evidence and integrates them selectively at the receiver. All these methods assume instantaneous message delivery, ignoring communication delay.

Delay in MARL. DAMARL [Chen et al., 2020] formulates a delay-aware Markov game incorporating action and observation delays and proposes a CTDE-based algorithm to mitigate the resulting non-stationarity. RDC [Fu et al., 2025] formulates stochastic observation delays as a DSID-POMDP and mitigates their impact through delay-free observation reconstruction. These methods focus on environment-level delays without addressing communication delay. For communication delay, DACOM [Yuan et al., 2023] is the first to study it in MARL, proposing a receiver-side TimeNet to learn adaptive waiting times, but assumes that communication completes within a single decision interval, failing under multi-interval delays. CoDe [Song et al., 2025] further studies asynchronous communication across decision intervals, encoding the current hidden state into a VAE-compressed latent as the message; however, with only indirect future supervision, its reliability degrades under longer delays; moreover, CoDe broadcasts to all agents, aggravating channel congestion. Both methods treat delay as fixed values or Gaussian distributions, ignoring queueing delay caused by agents' own communication behavior.

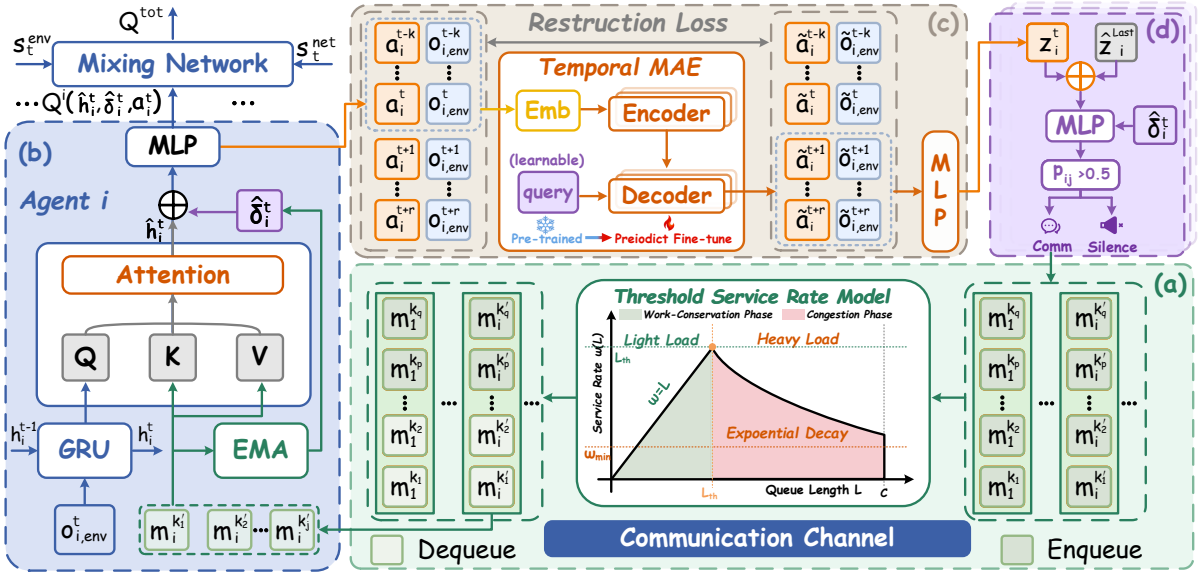

Figure 2: Overview of the CAIC framework. **(a) Network congestion model** (Section 3): shared queueing channel with state-dependent service rates. **(b) Agent decision module** (Section 4.1): message fusion via attention and action selection with delay estimates. **(c) Intent generation module** (Section 4.2): future trajectory prediction via MAE. **(d) Communication decision module** (Section 4.3): selective communication based on intent change and delay cost.

3 PRELIMINARIES

Queueing Theory. Queueing theory studies arrival, waiting, and service processes, where μ denotes the service rate. Classical models assume constant μ; in practice it varies with load. Such state-dependent behavior is well-evidenced in network congestion control, where RED [Floyd and Jacobson, 1993] drops packets at a load-dependent rate beyond a threshold, and REM [Athuraliya et al., 2001] marks packets at a rate exponential in queue length. Abouee-Mehrizi and Baron [Abouee-Mehrizi and Baron, 2016] formalize state-dependent M/G/1 systems with service rate $\mu(L)$ as a function of queue length L.

Network Congestion Model. We model the shared communication channel as a discrete-time queueing system with state-dependent service rates. At each timestep t, agents submit messages to a shared queue, whose length we denote L^t. The queue serves at most $\mu(L^t)$ messages per timestep; excess messages accumulate, inducing delay. Drawing on state-dependent queueing theory, we define the service rate as a function of the queue length:

$$\mu(L^t) = \begin{cases} L^t, & \text{if } L^t \le L_{\text{th}} \\ \omega_{\min} + (L_{\text{th}} - \omega_{\min})\, e^{-\alpha \eta^t}, & \text{if } L^t > L_{\text{th}} \end{cases}, \tag{1}$$

where $\eta^t = (L^t - L_{\text{th}})/(C - L_{\text{th}})$ is the normalized excess load, L_{th} the congestion threshold, α the decay rate, C the queue capacity, and $\omega_{\min}$ the minimum service rate. Under light load the system is work-conserving; once the queue exceeds L_{th}, the service rate decays exponentially toward $\omega_{\min}$, causing delay to grow. To simulate stochastic channel fluctuations, we inject Gaussian noise during execution for non-empty queues: $\tilde{\mu}(L^t) = \text{clip}(\mu(L^t) + \varepsilon,\, 1,\, L^t)$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. This model captures key congestion phenomena including congestion collapse, nonlinear delay growth under load, and endogenous congestion induced by agents' own communication behavior. We do not model multi-hop routing, heterogeneous links, or MAC-layer collisions, as these would increase complexity without providing additional insight into the coordination problem.

CA-Dec-POMDP. We formulate multi-agent cooperation under queueing delay as a Congestion-Aware Decentralized Partially Observable Markov Decision Process (CA-Dec-POMDP), defined by a tuple $\langle \mathcal{N}, S, A, P, \Omega, O, \mathcal{D}, R, \rho_0, \gamma \rangle$. Agents $\mathcal{N}$, observation function Ω, reward R, initial state distribution ρ_0, and discount factor γ retain their standard Dec-POMDP definitions; we extend the remaining elements as follows. The global state $S = S^{env} \times S^{ch}$ augments the environment state s^{env} with a channel state $s^{ch} = \langle L, \mu(L) \rangle$. The action space is extended to $A = A^{env} \times [0,1]^{N-1}$, where each agent's action $a_i = \langle a_i^{env}, \mathbf{p}_i^{comm} \rangle$ includes per-recipient communication probabilities $\mathbf{p}_i^{comm} = (p_{ij})_{j \neq i}$. The observation is augmented to $O = O^{env} \times \mathbb{R}$, with $o_i^t = \langle o_i^{env,t}, \hat{\delta}_i^t \rangle$ incorporating the local delay estimate $\hat{\delta}_i^t$. $\mathcal{D}$ denotes the delay space endogenously determined by s^{ch}, and the transition decomposes as $P(s'|s, \mathbf{a}) = P^{env}(s^{env\prime}|s^{env}, \mathbf{a}^{env}) \times P^{ch}(s^{ch\prime}|s^{ch}, \mathbf{p}^{comm})$, with channel dynamics driven by agents' collective communication actions $\mathbf{p}^{comm}$.

4 METHOD

As illustrated in Figure 2, at each timestep, each agent fuses asynchronously received messages via attention and estimates the current channel delay, jointly informing action selection (Section 4.1). The agent then predicts its future action trajectory via a temporal masked autoencoder and encodes it as a delay-robust intent message (Section 4.2). Finally, the agent schedules transmission based on intent change and estimated delay to mitigate congestion (Section 4.3). The training procedure is detailed in Section 4.4.

4.1 AGENT DECISION MODULE

Under the Network Congestion Model, messages from different senders arrive asynchronously at the receiver. Let $\mathcal{M}_i^t$ denote the set of messages received by agent i at timestep t. The agent first encodes its local environment observation $o_i^{env,t}$ into a hidden state $h_i^t = \text{GRU}(o_i^{env,t}, h_i^{t-1})$, and integrates $\mathcal{M}_i^t$ via cross-attention with h_i^t as the query:

$$\tilde{h}_i^t = W^O \sum_{m_j \in \mathcal{M}_i^t} \text{softmax}\left(\frac{h_i^t W^Q (m_j W^K)^\top}{\sqrt{d_k}}\right) W^V m_j. \tag{2}$$

A learned gate $g_i^t = \sigma(\text{MLP}([h_i^t; \tilde{h}_i^t]))$ then yields $\hat{h}_i^t = g_i^t \odot \tilde{h}_i^t + (1 - g_i^t) \odot h_i^t$; if $\mathcal{M}_i^t = \varnothing$, $\hat{h}_i^t = h_i^t$.

As the channel state is partially observable, agents cannot directly observe channel delay and must estimate it locally. Each message from sender j carries a sending timestamp t_s^j, from which the agent computes the observed delay as the worst-case across received messages, $d_i^t = \max_j(t - t_s^j)$. As inter-arrival intervals are non-uniform, we adopt a time-varying EMA whose decay rate adapts accordingly:

$$\hat{\delta}_i^t = \beta^{\Delta t} \cdot \hat{\delta}_i^{t-\Delta t} + (1 - \beta^{\Delta t}) \cdot d_i^t, \tag{3}$$

where $\beta \in (0, 1)$ is the base decay coefficient and Δt the interval since the last arrival. The factor $\beta^{\Delta t}$ down-weights stale estimates when messages are sparse and maintains smoothness when they are frequent. This estimator satisfies an upper bound property (Theorem 1; proof in Appendix A).

The agent representation $\hat{h}_i^t$ and delay estimate $\hat{\delta}_i^t$ are fed into the Q-network, and actions are selected greedily, with ϵ-greedy exploration during training:

$$a_i^t = \arg\max_a Q_i(\hat{h}_i^t, \hat{\delta}_i^t, a). \tag{4}$$

Incorporating $\hat{\delta}_i^t$ enables the policy to account for channel congestion during action selection, favoring less communication-dependent actions under high delay.

4.2 INTENT GENERATION MODULE

Observation-based messages become stale under queueing delay. We therefore encode each agent's predicted future trajectory as the message, capturing its behavioral intent.

At timestep t, after selecting action a_i^t, agent i constructs its recent trajectory:

$$\boldsymbol{\tau}_i^{t-T+1:t} = \{(o_i^{env,t-T+1}, a_i^{t-T+1}), \ldots, (o_i^{env,t}, a_i^t)\}. \tag{5}$$

We propose a Temporal Masked Autoencoder (MAE) to predict each agent's future trajectory. The MAE adopts an encoder-decoder architecture, preserving full temporal context via self-attention rather than compressing history into a single hidden state. Unlike random masking [He et al., 2022, Kang et al., 2025], during training we employ suffix masking, where the preceding T steps serve as input and the final K steps are masked as prediction targets.

The encoder maps each environment-observation-action pair $(o_i^{env,t}, a_i^t)$ through linear embedding and positional encoding, and processes the sequence with self-attention:

$$c_i^t = \text{Encoder}(\boldsymbol{\tau}_i^{t-T+1:t}). \tag{6}$$

The decoder takes K learnable query vectors $\hat{q} \in \mathbb{R}^{K \times d_e}$ as input and uses cross-attention to selectively retrieve relevant history from c_i^t, predicting the future K-step trajectory:

$$\hat{\boldsymbol{\tau}}_i^{t+1:t+K} = \text{Decoder}(\hat{q},\, c_i^t). \tag{7}$$

The predicted future trajectory

$$\hat{\boldsymbol{\tau}}_i^{t+1:t+K} = \{(\hat{o}_i^{env,t+1}, \hat{a}_i^{t+1}), \ldots, (\hat{o}_i^{env,t+K}, \hat{a}_i^{t+K})\} \tag{8}$$

contains joint predictions of observations and actions, which are vectorized and projected through a Multi-Layer Perceptron (MLP) to a compact intent vector z_i^t, reducing the message size transmitted through the shared channel:

$$z_i^t = \text{MLP}(\text{vec}(\hat{\boldsymbol{\tau}}_i^{t+1:t+K})). \tag{9}$$

Compared with transmitting instantaneous observations or hidden states, intent encoding is more robust to delay, as predicted future trajectories retain coordination value even after delayed arrival. Sender-side prediction is also computationally efficient, requiring only N MAE inferences per step in total, whereas a receiver-side design where each receiver reconstructs future intent from the transmitted hidden state would scale to $N(N-1)$ inferences under full communication.

4.3 COMMUNICATION DECISION MODULE

Unrestricted communication exacerbates channel congestion under limited bandwidth, inducing queueing delay. Selective communication is therefore critical, since each agent should transmit only when informational gain outweighs the channel cost.

We design the communication decision from two complementary perspectives: communication value and communication cost. The value of a message is determined by the deviation of the agent's current intent z_i^t from z_{ij}^{last}, the intent last transmitted to recipient j, where a larger change indicates a significant shift in behavioral plan, yielding higher informational gain. The cost is reflected by the delay estimate $\hat{\delta}_i^t$, where higher delay signals greater channel congestion, implying that additional transmissions would further intensify it. Balancing these two factors, the communication decision module computes a communication probability for each potential recipient j:

$$p_{ij}^t = \sigma\Big(\text{MLP}\Big([\text{MLP}_v([z_i^t;\, z_{ij}^{\text{last}}]);\, \hat{\delta}_i^t]\Big)\Big), \tag{10}$$

where σ denotes the sigmoid function. During training, $b_{ij}^t \sim \text{Bernoulli}(p_{ij}^t)$ to maintain exploration; at test time, $b_{ij}^t = 1$ if $p_{ij}^t > 0.5$, and 0 otherwise. When $b_{ij}^t = 1$, agent i transmits z_i^t to recipient j and updates $z_{ij}^{\text{last}} \leftarrow z_i^t$. When multiple messages are enqueued simultaneously, they are prioritized by sender-receiver distance, with closer pairs served first. The communication decision module is trained via a Value-of-Information (VOI) objective (Section 4.4).

4.4 TRAINING

CAIC follows the CTDE paradigm. Training involves three objectives optimized with separate optimizers: (1) the reinforcement learning loss, which optimizes cooperative policies via QMIX value decomposition; (2) the MAE loss, which trains future trajectory prediction; and (3) the communication value loss, which trains the communication decision module.

Reinforcement Learning Loss. Each agent computes Q-values $Q_i(\hat{h}_i^t, \hat{\delta}_i^t)$ for all actions. The local Q-values are mixed into a global Q_{tot} via QMIX, conditioned on the environment state $\mathbf{s}^{env}$ and the channel state $\mathbf{s}^{ch} = \langle L^t, \mu(L^t)\rangle$:

$$Q_{tot} = f_{mix}\big(Q_1, \ldots, Q_N;\, \mathbf{s}^{env}, \mathbf{s}^{ch}\big). \tag{11}$$

Following double Q-learning [Van Hasselt et al., 2016], action selection uses the current network, $a_i^* = \arg\max_{a_i'} Q_i(a_i')$, while value evaluation uses the target network f_{mix}^-, reducing overestimation bias. The loss is computed over a mini-batch $\mathcal{B}$ sampled from the replay buffer:

$$y = r + \gamma\, f_{mix}^-\big(Q_1^-(a_1^*), \ldots, Q_N^-(a_N^*);\, \mathbf{s}^{env\prime}, \mathbf{s}^{ch\prime}\big),$$
$$\mathcal{L}_{TD} = \frac{1}{|\mathcal{B}|}\sum_{b\in\mathcal{B}}\big(y^b - Q_{tot}^b\big)^2. \tag{12}$$

MAE Loss. The MAE is trained to reconstruct the masked suffix of observation-action trajectory windows sampled from the replay buffer, using the Mean Squared Error (MSE) loss:

$$\mathcal{L}_{MAE} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\left\|\hat{\boldsymbol{\tau}}_i^{t+k} - \boldsymbol{\tau}_i^{t+k}\right\|^2\right], \tag{13}$$

where $\hat{\boldsymbol{\tau}}_i^{t+k} = (\hat{o}_i^{env,t+k}, \hat{a}_i^{t+k})$ is the predicted observation-action pair at the k-th future step and $\boldsymbol{\tau}_i^{t+k}$ is the corresponding ground truth. Since the MAE's input distribution is determined by the policy while policy learning depends on the intent representations produced by the MAE, simultaneous optimization creates mutually dependent non-stationary targets. We therefore adopt a two-stage training pipeline. In the **offline stage**, we roll out 100 episodes under the initial policy and train the MAE until convergence, providing RL with an intent encoder of sufficient quality before training begins. In the **online stage**, we clear the replay buffer once RL starts and fine-tune the MAE every 5,000 environment steps to track policy distribution drift. Its parameters are frozen during RL updates.

Communication Value Loss. The communication decision module requires a supervisory signal indicating when communication is valuable. We define the Q-value gain ΔQ_j^t from message fusion as the Value of Information (VOI):

$$\Delta Q_j^t = \text{softplus}\Big(\max_a Q_j(\hat{h}_j^t, \hat{\delta}_j^t, a) - \max_a Q_j(h_j^t, \hat{\delta}_j^t, a)\Big), \tag{14}$$

where $\text{softplus}(x) = \log(1 + e^x)$ is a smooth non-negative surrogate for $\max(0, x)$. ΔQ_j^t is informative only when agent j has received messages. When no messages arrive, it provides no transmission incentive and would suppress necessary communication. We therefore augment it with message novelty Δz_{ij}^t and channel congestion κ_i^t to form a complete supervision signal. We normalize the three signals as follows:

$$\widehat{\Delta Q}_j^t = \sigma(\Delta Q_j^t), \quad \Delta z_{ij}^t = \frac{\|z_i^t - z_{ij}^{\text{last}}\|_2}{\sqrt{d_m}}, \quad \kappa_i^t = \frac{\hat{\delta}_i^t}{\delta_{\max}}, \tag{15}$$

where d_m is the intent vector dimension and $\delta_{\max}$ is a fixed delay normalization constant. The three terms are combined into a communication value target:

$$v_{ij}^t = \text{clip}\Big(\lambda_v\, \widehat{\Delta Q}_j^t + \lambda_d\, \Delta z_{ij}^t - \lambda_c\, \kappa_i^t,\ 0,\ 1\Big). \tag{16}$$

Communication value is maximized when receiver benefit is high, message content is novel, and the channel is uncongested. The module is trained to approximate this target via binary cross-entropy:

$$\mathcal{L}_{comm} = -\lambda_{comm}\sum_t\sum_{i\neq j}\big[v_{ij}^t \log p_{ij}^t + (1 - v_{ij}^t)\log(1 - p_{ij}^t)\big], \tag{17}$$

where v_{ij}^t is a stop-gradient target. This module uses a separate optimizer to prevent gradient interference with policy learning. The complete training procedure is detailed in Algorithm 1 (Appendix B).

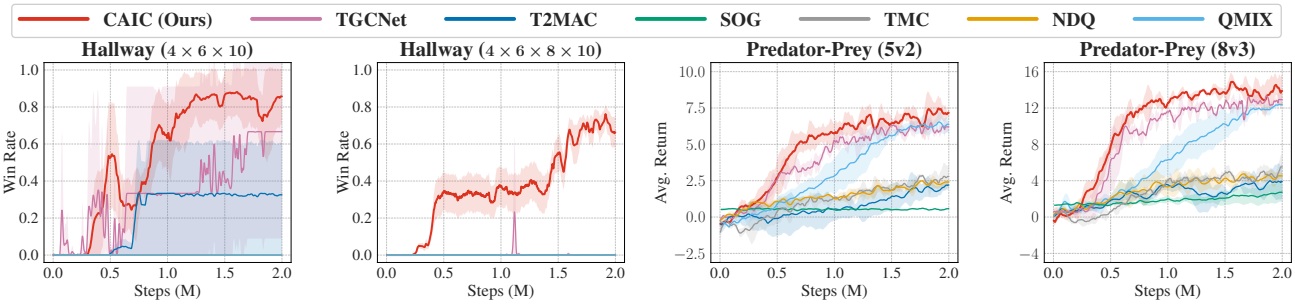

Figure 3: Main results on Hallway and Predator-Prey environments without communication delay.

Table 1: Results under queueing delay; each cell shows mean(std). **Bold**: best, underline: second best.

Task	CAIC	CoDe	TGCNet	T2MAC	SOG	TMC	NDQ	QMIX
CN N=6	−1.37(0.00)	−3.37(0.03)	**−1.12**(0.08)	−1.38(0.04)	−1.70(0.22)	−3.34(0.43)	−2.71(1.17)	−1.79(0.20)
CN N=10	**−1.83**(0.03)	−2.84(1.28)	−1.84(0.07)	−1.88(0.02)	−4.17(1.59)	−5.41(0.09)	−2.56(0.12)	−2.62(0.60)
PP 5v2	**7.70**(0.29)	3.71(0.96)	6.69(0.73)	1.69(1.42)	0.64(0.07)	0.87(0.10)	2.26(0.20)	6.39(0.42)
PP 8v3	**12.63**(1.70)	4.55(1.77)	10.78(2.70)	3.61(1.36)	2.24(1.05)	1.88(0.22)	4.20(0.64)	12.36(0.09)
SMAC 2s3z	**91.8**(3.9)	51.2(18.6)	67.9(5.9)	0.0(0.0)	70.7(8.3)	27.5(17.5)	69.1(4.6)	51.9(22.9)
8m	**89.5**(3.8)	70.6(4.8)	59.4(8.8)	29.7(42.1)	84.2(9.7)	47.5(28.0)	80.8(10.4)	82.9(5.7)
MMM	**88.6**(5.9)	60.9(12.8)	56.8(8.1)	0.0(0.0)	54.6(7.1)	38.0(12.0)	57.4(19.4)	76.3(3.5)
5m_vs_6m	**70.9**(6.5)	0.0(0.0)	2.1(1.6)	62.9(6.2)	37.0(9.6)	15.4(6.8)	14.1(6.5)	51.0(6.6)
3s5z	**39.7**(10.1)	22.1(24.1)	35.3(4.8)	1.0(1.5)	25.1(4.1)	0.1(0.1)	0.0(0.0)	28.3(8.0)

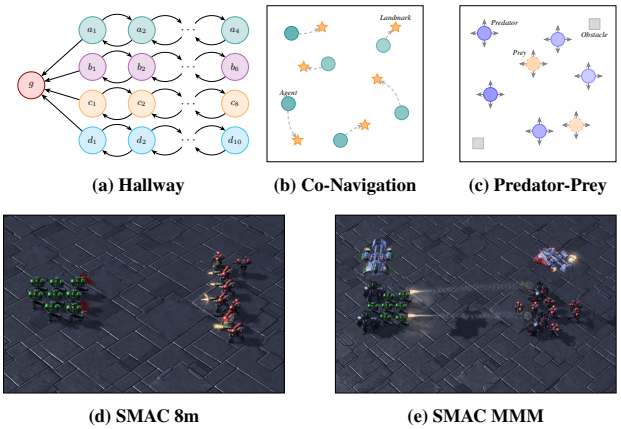

Figure 4: Multi-agent environments in our experiments.

5 EXPERIMENTS

We evaluate CAIC on Hallway, MPE, and SMAC, focusing on two aspects: *message validity*, whether intent messages remain effective for coordination under delay, and *communication timeliness*, whether congestion-aware decisions regulate transmissions to alleviate congestion. Cooperative Navigation and Predator-Prey (the two MPE tasks) and SMAC are evaluated under queueing delay, and Hallway and Predator-Prey in the no-delay setting; under delay, all methods share the same delay channel for fair comparison.

5.1 EXPERIMENTAL SETUP

Hallway [Wang et al., 2020] is a Dec-POMDP coordination task (Figure 4) where multiple agents navigate through parallel corridors of different lengths, each observing only its own position, while all agents must reach the goal state simultaneously.

MPE [Lowe et al., 2017] is a 2D multi-agent particle environment. We select two tasks: Cooperative Navigation (CN), where agents cooperatively cover landmarks, and Predator-Prey (PP), where predators coordinate to capture randomly moving prey. We evaluate CN with N=6 and N=10, and PP with 5v2 and 8v3, covering different agent scales and communication-load regimes.

SMAC [Samvelyan et al., 2019] is a micromanagement benchmark derived from StarCraft II, in which each unit is controlled by an independent agent under partial observability and agents must learn cooperative tactics. We evaluate on five maps spanning difficulty levels and team scales.

We compare against QMIX [Rashid et al., 2020] (no-communication baseline) and six communication methods: NDQ [Wang et al., 2020], TMC [Zhang et al., 2020], SOG [Shao et al., 2022], T2MAC [Sun et al., 2024], TGCNet [Zhang et al., 2025], and CoDe [Song et al., 2025]. We report test win rate for SMAC and Hallway, and normalized average return for MPE.

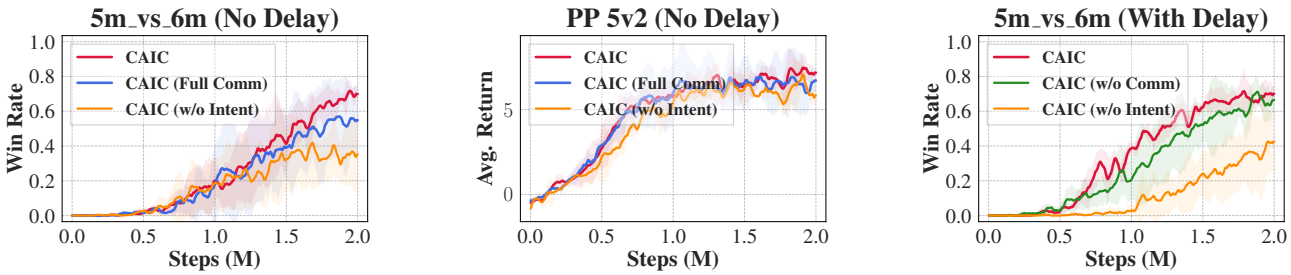

Figure 5: Ablation study results of CAIC, w/o Intent, w/o Comm, and Full Comm.

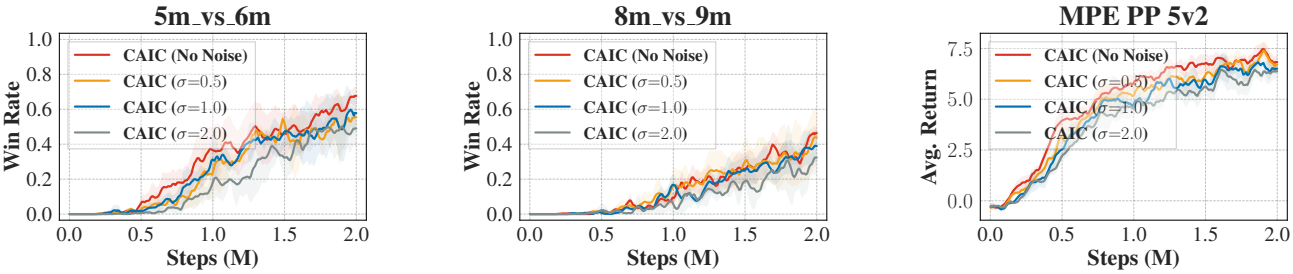

Figure 6: Robustness to delay estimation noise. Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is injected into the EMA delay estimate $\hat{\delta}$.

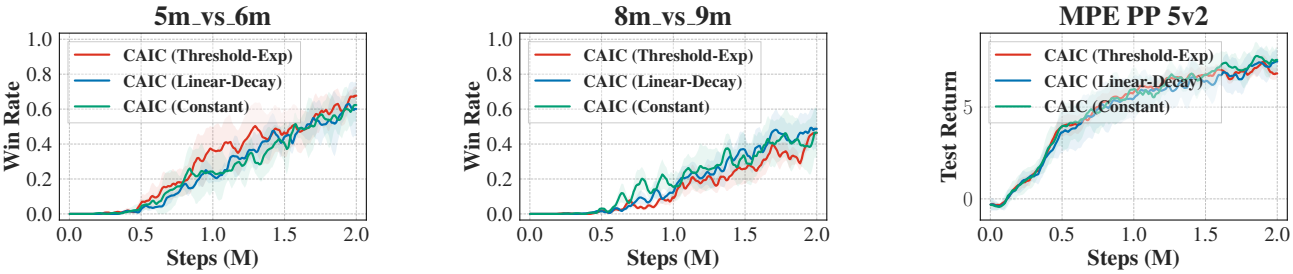

Figure 7: Performance under three service-rate forms: Threshold-Exp (ours), Linear-Decay, and Constant.

5.2 MAIN RESULTS

No-Delay Results. CAIC outperforms all baselines on the evaluated Hallway and Predator-Prey scenarios (Figure 3). Although the hidden state h_i^t contains at least as much information as the intent vector z_i^t in the information-theoretic sense, the coordination-irrelevant content in h_i^t, such as observation noise and unfiltered history details, dilutes the truly coordination-relevant signal at the receiver. Existing methods compress messages to reduce this redundancy, such as NDQ via an information bottleneck and T2MAC via key/value projections, but these compressions lack explicit guidance toward future behavioral content. CAIC's future-trajectory supervision directly steers the encoding toward what teammates will do next, which is especially valuable in future-oriented tasks such as synchronized arrival in Hallway and coordinated encirclement in Predator-Prey.

Queueing Delay Results. Under queueing delay (Table 1), CAIC achieves the best or near-best performance on both Cooperative Navigation and Predator-Prey. In Cooperative Navigation, intent messages reveal teammates' movement targets, preventing redundant landmark coverage. Most baseline methods suffer notable performance degradation, while CAIC maintains stable performance, validating the delay robustness of intent messages. On SMAC, CAIC achieves the highest win rate on all five maps. Baseline communication methods degrade substantially, as their messages encode instantaneous information that becomes outdated after delay. Decisions based on stale messages can even mislead behavior, explaining why some communication methods perform worse than the no-communication QMIX. T2MAC is particularly sensitive, since its evidence-driven integration relies on alignment between received messages and the recipient's current state, and delayed messages introduce outdated evidence that undermines decision reliability. The delay-aware CoDe also lags behind CAIC. On one hand, CoDe compresses the current state into a VAE latent as its message and captures future information only indirectly, so its reliability degrades under longer delays. On the other hand, CoDe broadcasts to all other agents, which itself congests the shared queue and further amplifies message delay.

Table 2: Communication analysis of CAIC on SMAC; each cell shows mean(std).

Task	Rate (%)	VOI	Cost
2s3z	40.3(3.8)	0.67(0.05)	0.33(0.08)
8m	36.9(0.4)	0.80(0.04)	0.41(0.00)
MMM	47.9(0.1)	0.74(0.02)	0.22(0.00)
5m_vs_6m	32.9(0.4)	0.75(0.04)	0.34(0.01)
3s5z	45.5(1.5)	0.77(0.04)	0.22(0.01)

Communication Analysis. Table 2 reveals that CAIC learns differentiated communication strategies across scenarios. On MMM (10 agents), CAIC exhibits the highest communication rate (47.9%) yet incurs the joint-lowest delay cost (0.22), indicating effective congestion management under high communication demand. On 5m_vs_6m, the communication rate is the lowest (32.9%), showing that agents transmit only at critical moments under numerical disadvantage. These results demonstrate that CAIC balances coordination benefits against congestion costs.

5.3 ABLATION STUDY

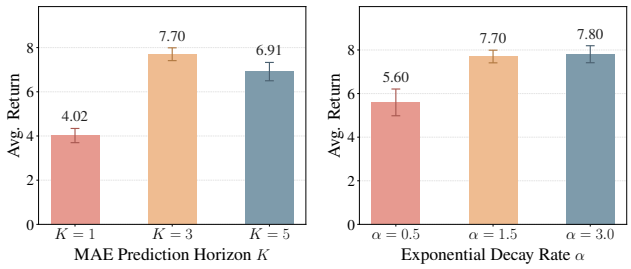

Figure 8: Sensitivity analysis on MPE Predator-Prey 5v2 under queueing delay.

Component Analysis. Figure 5 reports the ablation study on SMAC 5m_vs_6m and MPE PP 5v2 under both no-delay and queueing-delay settings, comparing CAIC against three variants.

w/o Intent replaces the intent vector z_i^t with the raw GRU hidden state h_i^t as the message throughout, while VOI computation, attention fusion, and training remain identical, isolating the effect of message content. It is the weakest variant across all settings. Even without delay it trails CAIC, confirming that explicitly encoded intent carries more coordination value than raw hidden states. Under delay it degrades most, on 5m_vs_6m even falling below *w/o Comm*.

w/o Comm removes communication entirely, serving as a no-communication baseline. Its advantage over *w/o Intent* on 5m_vs_6m under delay arises because outdated hidden-state messages actively mislead decision-making, making

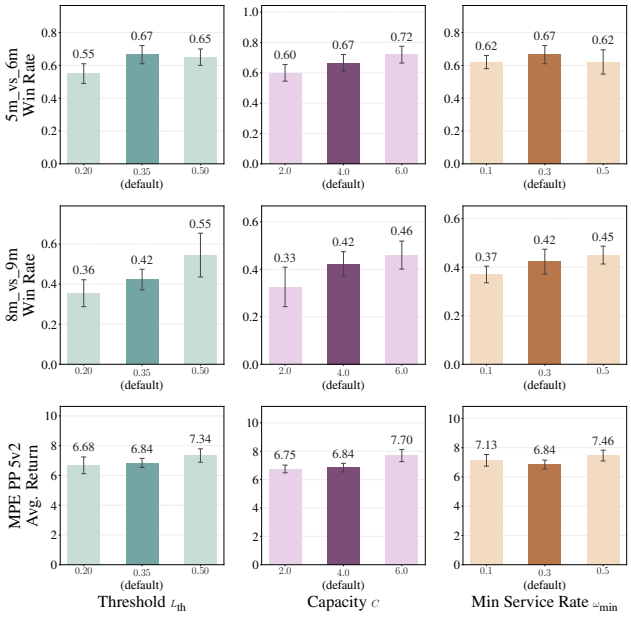

Figure 9: Queue parameter sensitivity: congestion threshold L_{th}, queue capacity C, and minimum service rate ω_{min}.

silence preferable. This confirms that what matters under delay is the temporal robustness of message content rather than communication alone.

Full Comm broadcasts at every step. Even without delay it underperforms CAIC on 5m_vs_6m, since flooding the receiver with redundant messages dilutes the informative ones and undermines attention-based fusion; under delay it degrades further, as transmitting at every step raises channel load and queueing delay for all messages, creating a vicious cycle of congestion and staleness. This validates the necessity of the communication decision module in regulating transmission frequency.

Robustness Analysis. We stress-test two idealized assumptions of CAIC: accurate delay estimation and dependence on a specific queue form. First, we inject Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ into the delay estimate $\hat{\delta}$ at evaluation (Figure 6). On communication-critical maps (5m_vs_6m, 8m_vs_9m), performance degrades as σ grows, confirming that the policy genuinely relies on $\hat{\delta}$; on the lower-demand PP 5v2 it barely changes. Even under the strongest noise σ=2.0, performance does not collapse, demonstrating robustness to estimation error. Second, we replace the threshold-exponential service rate with two alternative forms, Linear-Decay and Constant (Figure 7). CAIC achieves comparable performance across all three, indicating that its effectiveness stems from the generic "congestion→delay" feedback rather than any hand-crafted queue form. Indeed, CAIC's communication decisions depend solely on the EMA-estimated delay $\hat{\delta}$ and never directly observe the queue state or the service-rate function.

Sensitivity Analysis. We evaluate the sensitivity of CAIC to the TemporalMAE prediction horizon K and the exponential decay rate α of the state-dependent service rate on MPE Predator-Prey 5v2 under queueing delay (Figure 8). Setting $K{=}1$ substantially reduces performance, as single-step predictions provide insufficient future coverage. The default $K{=}3$ performs best, while $K{=}5$ shows a slight decline, possibly owing to the difficulty of long-horizon prediction. For the decay rate α, $\alpha{=}0.5$ slightly degrades performance, while $\alpha{=}1.5$ and $\alpha{=}3.0$ achieve comparable results, indicating robustness to α provided sufficient congestion pressure exists.

We also assess robustness to the three queue-model parameters (Figure 9): the congestion threshold L_{th}, the queue capacity C, and the minimum service rate $\omega_{\min}$. Across all three scenarios, CAIC remains stable, dropping by only 10–15% under the most stringent settings, confirming that CAIC does not rely on finely tuned queue parameters.

6 CONCLUSION

We presented CAIC, a framework that addresses queueing delay in multi-agent communication through intent encoding and congestion-aware decisions. Ablations reveal that transmitting stale messages can be more harmful than no communication, underscoring the necessity of temporally robust message content. This work models channel congestion with a single shared queue, whereas practical systems may involve multi-hop forwarding and heterogeneous channels. Moreover, intent encoding relies on relatively stable policies; in adversarial or non-stationary settings, prediction accuracy may degrade. Future work can introduce more realistic network models and explore adaptive prediction horizons to handle rapid policy changes.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62302189 and in part by the Fundamental Research Funds for the Central Universities under Grant 2662022XXQD002.

References

Hossein Abouee-Mehrizi and Opher Baron. State-dependent M/G/1 queueing systems. *Queueing Systems*, 82(1-2): 121–148, 2016.

Sanjeewa Athuraliya, Steven H. Low, Victor H. Li, and Qinghe Yin. REM: Active queue management. *IEEE Network*, 15(3):48–53, 2001.

Baiming Chen, Mengdi Xu, Zuxin Liu, Liang Li, and Ding Zhao. Delay-aware multi-agent reinforcement learning for cooperative and competitive environments. *arXiv preprint arXiv:2005.05441*, 2020.

Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. TarMAC: Targeted multi-agent communication. In *International Conference on Machine Learning*, pages 1538–1546, 2019.

Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, 1993.

Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 2974–2982, 2018.

Songchen Fu, Siang Chen, Shaojing Zhao, Letian Bai, Hong Liang, Ta Li, and Yonghong Yan. Rainbow delay compensation: A multi-agent reinforcement learning framework for mitigating delayed observation. In *Advances in Neural Information Processing Systems*, 2025.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

Guangzheng Hu, Yuanheng Zhu, Dongbin Zhao, Mengchen Zhao, and Jianye Hao. Event-triggered communication network with limited-bandwidth constraint for multi-agent reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(8):3966–3978, 2023.

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

Sehyeok Kang, Yongsik Lee, Gahee Kim, Song Chong, and Se-Young Yun. MA^{2}E: Addressing partial observability in multi-agent reinforcement learning with masked auto-encoder. In *International Conference on Learning Representations*, pages 20145–20165, 2025.

Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi. Learning to schedule communication in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2019.

Woojun Kim, Jongeui Park, and Youngchul Sung. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations*, 2021.

Xinran Li and Jun Zhang. Context-aware communication for multi-agent reinforcement learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 2024.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Yaru Niu, Rohan R. Paleja, and Matthew C. Gombolay. Multi-agent graph-attention communication and teaming. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 2021.

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, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2186–2188. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

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

Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. In *International Conference on Learning Representations*, 2019.

Shoucheng Song, Youfang Lin, Sheng Han, Chang Yao, Hao Wu, Shuo Wang, and Kai Lv. CoDe: Communication delay-tolerant multi-agent collaboration via dual alignment of intent and timeliness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23304–23312, 2025.

Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Chuxiong Sun, Zehua Zang, Jiabao Li, Jiangmeng Li, Xiao Xu, Rui Wang, and Changwen Zheng. T2MAC: Targeted and trusted multi-agent communication through selective engagement and evidence-driven integration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15154–15163, 2024.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, pages 2094–2100, 2016.

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.

Tonghan Wang, Jianhao Wang, Chongyi Zheng, and Chongjie Zhang. Learning nearly decomposable value functions via communication minimization. In *International Conference on Learning Representations*, 2020.

Zhaoyue Xia, Jun Du, Jingjing Wang, Chunxiao Jiang, Yong Ren, Gang Li, and Zhu Han. Multi-agent reinforcement learning aided intelligent UAV swarm for target tracking. *IEEE Transactions on Vehicular Technology*, 71(1):931–945, 2022.

Tingting Yuan, Hwei-Ming Chung, Jie Yuan, and Xiaoming Fu. DACOM: Learning delay-aware communication for multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11763–11771, 2023.

Sai Qian Zhang, Jieyu Lin, and Qi Zhang. Succinct and robust multi-agent communication with temporal message control. In *Advances in Neural Information Processing Systems*, volume 33, pages 17271–17282, 2020.

Zhuohui Zhang, Bin He, Bin Cheng, and Gang Li. Bridging training and execution via dynamic directed graph-based communication in cooperative multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23395–23403, 2025.

CAIC: Congestion-Aware Intent Communication for Multi-Agent Reinforcement Learning (Supplementary Material)

Pei Li[1] **Yongkang Zhang**[1] **Zhonglin Lv**[1] **Jinmin Zhu**[1] **Jiangjin Yin**[*1]

[1]College of Informatics, Huazhong Agricultural University, Key Laboratory of Smart Farming for Agricultural Animals, Hubei Engineering Technology Research Center of Agricultural Big Data, and Engineering Research Center of Intelligent Technology for Agriculture, Wuhan 430070, China

A THEORETICAL ANALYSIS OF DELAY ESTIMATION

We establish that the EMA delay estimator $\hat{\delta}_i^t$ (Eq. 3) provides an upper bound on the true network delay $D_i(t)$. The analysis proceeds in three steps: (1) bounding the true delay by the observed delay, (2) bounding the tracking error of the EMA estimator, and (3) combining both results. Throughout, we consider the nominal (noiseless) channel dynamics of Eq. 1, under which a non-empty queue serves $\mu(L^t)$ messages per timestep; service is treated as fluid, and integer rounding adds at most one timestep to the bounds below.

Assumption 1 (Bounded Queue Variation)**.** *There exists a constant $\Delta_{\max} > 0$ such that $|L^{t+1} - L^t| \leq \Delta_{\max}$ for all t. This holds because both the number of arrivals per step (each of the at most N broadcasting agents enqueues one message per receiver, i.e., at most $N(N-1)$ messages) and the service rate $\mu(L^t) \leq L_{\text{th}}$ are bounded, so the net queue change per step is bounded by $N(N-1) + L_{\text{th}}$.*

Assumption 2 (Bounded Delay Variation)**.** *There exists a constant $\Delta_d > 0$ such that between any two consecutive message arrivals at times t_{k-1} and t_k, the observed delay satisfies $|d_i^{t_k} - d_i^{t_{k-1}}| \leq \Delta_d$. This is motivated by Assumption 1, since bounded queue variation implies bounded changes in waiting time over bounded inter-arrival intervals.*

Lemma 1 (True Delay vs. Observed Delay)**.** *Under Assumption 1, the true network delay for a message entering the queue at time t satisfies*

$$D_i(t) \leq \frac{\mu_{\max} + \Delta_{\max}}{\mu_{\min}} \cdot d_i^t,$$

where $\mu_{\max} = \max_L \mu(L)$, $\mu_{\min} = \min_{1 \leq L \leq C} \mu(L) > 0$ (restricted to non-empty queues), and $d_i^t = \max_j (t - t_s^j)$ is the maximum observed delay among messages received at time t.

Proof. Let τ^* denote the sending time of the message that achieves the maximum observed delay d_i^t, so its end-to-end delay is exactly $t - \tau^* = d_i^t$, and let L^{τ^*} be the queue length at time τ^*. This message must wait for the L^{τ^*} messages ahead of it to be served, and the fastest possible service rate is $\mu_{\max}$, so its delay is at least $L^{\tau^*}/\mu_{\max}$. Since this delay equals d_i^t, we have $d_i^t \geq L^{\tau^*}/\mu_{\max}$, i.e.,

$$L^{\tau^*} \leq d_i^t \cdot \mu_{\max}. \tag{18}$$

By Assumption 1, the queue length at the current time t is bounded by:

$$L^t \leq L^{\tau^*} + \Delta_{\max} \cdot (t - \tau^*) = L^{\tau^*} + \Delta_{\max} \cdot d_i^t. \tag{19}$$

A message entering the queue at time t experiences delay at most $L^t/\mu_{\min}$, where L^t denotes the queue length immediately after the message joins, since $\mu_{\min}$ is the minimum service rate throughout the draining process. Substituting (18) and (19):

$$D_i(t) \leq \frac{L^t}{\mu_{\min}} \leq \frac{d_i^t \cdot \mu_{\max} + \Delta_{\max} \cdot d_i^t}{\mu_{\min}} = \frac{\mu_{\max} + \Delta_{\max}}{\mu_{\min}} \cdot d_i^t.$$

□

Proposition 1 (EMA Tracking Bound)**.** *Under Assumption 2, let $\{t_k\}$ be the sequence of message arrival times with inter-arrival intervals $\Delta t_k = t_k - t_{k-1}$, and let $\Delta t_{\min} = \min_k \Delta t_k > 0$. Define $\rho = \beta^{\Delta t_{\min}} \in (0,1)$, and suppose the estimator is initialized with $\hat{\delta}_i^{t_0} \geq d_i^{t_0}$. Then the EMA estimator satisfies:*

$$d_i^{t_k} \leq \hat{\delta}_i^{t_k} + \varepsilon_{\text{track}}, \quad \textit{where} \quad \varepsilon_{\text{track}} = \frac{\rho \cdot \Delta_d}{1-\rho}.$$

Proof. Define the tracking error $e_k = \hat{\delta}_i^{t_k} - d_i^{t_k}$. By the EMA update rule (Eq. 3):

$$\hat{\delta}_i^{t_k} = \beta^{\Delta t_k} \cdot \hat{\delta}_i^{t_{k-1}} + (1 - \beta^{\Delta t_k}) \cdot d_i^{t_k}. \tag{20}$$

Subtracting $d_i^{t_k}$ from both sides:

$$\begin{aligned} e_k &= \beta^{\Delta t_k} \cdot (\hat{\delta}_i^{t_{k-1}} - d_i^{t_k}) \\ &= \beta^{\Delta t_k} \cdot \big(e_{k-1} + d_i^{t_{k-1}} - d_i^{t_k}\big). \end{aligned} \tag{21}$$

Write $\delta_d^k = d_i^{t_{k-1}} - d_i^{t_k}$, so that $\delta_d^k \in [-\Delta_d, \Delta_d]$ by Assumption 2, and note that $\beta^{\Delta t_k} \in (0, \rho]$ because $\Delta t_k \geq \Delta t_{\min}$ and $\beta \in (0,1)$ imply $\beta^{\Delta t_k} \leq \beta^{\Delta t_{\min}} = \rho$. We prove by induction that $e_k \geq -\varepsilon_{\text{track}}$ for all k, where $\varepsilon_{\text{track}} = \rho\Delta_d/(1-\rho)$; the initialization hypothesis $\hat{\delta}_i^{t_0} \geq d_i^{t_0}$ gives $e_0 \geq 0$.

Base case: $e_0 \geq 0 \geq -\varepsilon_{\text{track}}$.

Inductive step: assume $e_{k-1} \geq -\varepsilon_{\text{track}}$. From (21), $e_k = \beta^{\Delta t_k}(e_{k-1} + \delta_d^k)$. If $e_{k-1} + \delta_d^k \geq 0$, then $e_k \geq 0 \geq -\varepsilon_{\text{track}}$. Otherwise $e_{k-1} + \delta_d^k < 0$, and since multiplying a negative quantity by the larger factor $\rho \geq \beta^{\Delta t_k}$ can only decrease it,

$$e_k = \beta^{\Delta t_k}(e_{k-1} + \delta_d^k) \geq \rho\,(e_{k-1} + \delta_d^k) \geq \rho\,(e_{k-1} - \Delta_d) \geq \rho\,(-\varepsilon_{\text{track}} - \Delta_d) = -\varepsilon_{\text{track}},$$

where the final equality substitutes $\varepsilon_{\text{track}} = \rho\Delta_d/(1-\rho)$ and simplifies. Hence $e_k \geq -\varepsilon_{\text{track}}$ for all $k \geq 0$. Rearranging $e_k = \hat{\delta}_i^{t_k} - d_i^{t_k} \geq -\varepsilon_{\text{track}}$ gives $d_i^{t_k} \leq \hat{\delta}_i^{t_k} + \varepsilon_{\text{track}}$. □

Theorem 1 (Upper Bound Property of EMA Delay Estimator)**.** *Under Assumptions 1 and 2, define $\kappa_\mu = (\mu_{\max} + \Delta_{\max})/\mu_{\min}$ and $\varepsilon_{\text{track}} = \rho\Delta_d/(1-\rho)$ with $\rho = \beta^{\Delta t_{\min}}$. Suppose the estimator is initialized with $\hat{\delta}_i^{t_0} \geq d_i^{t_0}$. Then at every message arrival time t, the true network delay satisfies:*

$$D_i(t) \leq \kappa_\mu \cdot \big(\hat{\delta}_i^t + \varepsilon_{\text{track}}\big).$$

In the work-conserving regime ($L^t \leq L_{\text{th}}$), the service rate satisfies $\mu(L^t) = L^t$ (Eq. 1), so every queued message is cleared within a single timestep and the true delay is trivially bounded by $D_i(t) \leq 1$; the estimator bound then holds with substantial slack.

Proof. By Lemma 1, $D_i(t) \leq \kappa_\mu \cdot d_i^t$ at every arrival time t. By Proposition 1, $d_i^t \leq \hat{\delta}_i^t + \varepsilon_{\text{track}}$ at every arrival time t. Combining the two bounds gives $D_i(t) \leq \kappa_\mu \cdot (\hat{\delta}_i^t + \varepsilon_{\text{track}})$. □

Remark. The tracking error $\varepsilon_{\text{track}} = \rho\Delta_d/(1-\rho)$ decreases as β decreases (more responsive EMA) or, for a fixed Δ_d, as $\Delta t_{\min}$ increases (less frequent arrivals). With typical values $\beta = 0.75$ and $\Delta t_{\min} = 1$, $\rho = 0.75$ and $\varepsilon_{\text{track}} = 3\Delta_d$. The initialization condition $\hat{\delta}_i^{t_0} \geq d_i^{t_0}$ is easily satisfied in practice by initializing the estimator to a conservatively large value. Between consecutive arrivals, both d_i^t and $\hat{\delta}_i^t$ are held constant, and the same argument yields the bound with an additional additive term $(\Delta_{\max}/\mu_{\min}) \cdot (t - t_k)$, where t_k is the most recent arrival time; the bound thus degrades gracefully with the staleness of the last observation. We note that the bound can be conservative when the number of agents N is large, as $\Delta_{\max} = N(N-1) + L_{\text{th}}$ increases κ_μ. However, the practical role of the delay estimator does not depend on the tightness of this bound: it serves as an input feature to the Q-network and communication decision module, providing a signal that is monotonically correlated with the current congestion level, enabling agents to perceive and respond to changes in network conditions.

B CAIC TRAINING PROCEDURE

Algorithm 1 CAIC Training Procedure

Input: Environment env, max timesteps T_{max}, replay buffer $\mathcal{B}$, MAE loss threshold ϵ_{MAE}, MAE fine-tune interval f_{MAE}, target update interval f_{target}

Initialize Q-networks, GRU, attention module, intent generator, communication decision module, mixer, and their target networks
Initialize optimizers: main (Adam), communication (Adam), MAE (SGD)
— Phase 1: MAE Pre-training —
Collect initial episodes into $\mathcal{B}$
repeat
    Sample batch from $\mathcal{B}$, extract observation-action windows
    Unfreeze MAE; compute $\mathcal{L}_{MAE} = \frac{1}{NK} \sum_{i,k} \|\hat{\boldsymbol{\tau}}_i^{t+k} - \boldsymbol{\tau}_i^{t+k}\|^2$ via suffix masking
    Update MAE with SGD; freeze MAE
    Soft update target MAE: $\theta_{MAE}^- \leftarrow \tau_{MAE}\, \theta_{MAE} + (1 - \tau_{MAE})\, \theta_{MAE}^-$
until $\bar{\mathcal{L}}_{MAE} < \epsilon_{MAE}$ or max pre-training episodes reached
Clear $\mathcal{B}$, reset $t_{env} \leftarrow 0$
— Phase 2: Main Training Loop —
while $t_{env} \leq T_{max}$ **do**
    Reset env, initialize hidden states h_i^0 for all agents
    for each timestep t in episode **do** ▷ Episode Collection
        Receive delayed messages $\mathcal{M}_i^t$ from shared queue
        Update delay estimate $\hat{\delta}_i^t$ via time-varying EMA (Eq. 3)
        Encode: $h_i^t = \mathrm{GRU}(o_i^{env,t},\, h_i^{t-1})$
        Fuse: $\hat{h}_i^t = \mathrm{Attention}(h_i^t,\, \mathcal{M}_i^t)$
        Select action: $a_i^t = \epsilon\text{-greedy}(Q_i(\hat{h}_i^t, \hat{\delta}_i^t))$
        Predict future trajectory via MAE (no gradient), compress into intent z_i^t
        Compute $p_{ij}^t = \sigma(\mathrm{MLP}(z_i^t, z_{ij}^{\mathrm{last}}, \hat{\delta}_i^t))$
        Obtain broadcast decision b_{ij}^t from p_{ij}^t (Sec. 4.3); if $b_{ij}^t = 1$, enqueue z_i^t and update $z_{ij}^{\mathrm{last}} \leftarrow z_i^t$
        Service queue at rate $\mu(L^t)$ and deliver messages
    end for
    Store episode in $\mathcal{B}$, $t_{env} \leftarrow t_{env} + T$
    Sample batch from $\mathcal{B}$ ▷ Parameter Update
    Compute current and target Q-values (double Q-learning)
    Compute $\mathcal{L}_{TD}$; update GRU, attention, Q-networks, mixer via main optimizer
    Compute $\mathcal{L}_{comm}$ (VOI-based); update comm. decision module via comm. optimizer
    if $t_{env} \bmod f_{MAE} = 0$ **then**
        Unfreeze MAE; compute $\mathcal{L}_{MAE}$; SGD update; freeze MAE
        Soft update target MAE: $\theta_{MAE}^- \leftarrow \tau_{MAE}\, \theta_{MAE} + (1 - \tau_{MAE})\, \theta_{MAE}^-$
    end if
    if episode $\bmod f_{target} = 0$ **then**
        Hard update target Q-networks and mixer
    end if
end while

C EXPERIMENTAL DETAILS

C.1 QUEUEING DELAY PARAMETERS

The state-dependent service-rate model is parameterized as follows for all experiments with queueing delay:

Parameter	Symbol	Value
Congestion threshold	L_{th}	$0.35 \times M$
Exponential decay coefficient	α	1.5
Minimum service-rate ratio	$\omega_{\min}/L_{\text{th}}$	0.3
Queue capacity	C	$4 \times M$

Table 3: Queueing delay model parameters. $M = N(N-1)$ denotes the maximum per-step message load for N agents.

C.2 HYPERPARAMETERS

Hyperparameter	Value
GRU hidden dimension	128
Message fusion attention heads	1 (single-head)
Intent vector dimension	128
QMIX mixing embed dimension	32
QMIX hypernet layers / embed dim	2 / 64
MAE hidden dimension	128
MAE sequence length T	10
TemporalMAE prediction horizon K	3
MAE Transformer heads / layers	4 / 2
MAE learning rate (SGD)	5×10^{-4}
MAE fine-tune interval	5,000 steps
MAE target soft-update rate τ	0.01
MAE pre-train threshold	0.03
Communication decision learning rate	2×10^{-3}
Comm. value weights (λ_v, λ_d, λ_c)	(0.7, 0.3, 0.5)
Comm. loss coefficient λ_{comm}	0.5
Delay normalization constant $\delta_{\max}$	5
Delay EMA decay coefficient β	0.75
Main optimizer learning rate (Adam)	5×10^{-4}
Batch size (SMAC / MPE, Hallway)	32 / 64
Replay buffer size	5,000
Target network update interval	200 episodes
ϵ-greedy anneal	$1.0 \rightarrow 0.05$ over 50k steps
Discount factor γ	0.99

Table 4: Hyperparameters of CAIC.

D COMPUTATIONAL COMPLEXITY

The computational cost of CAIC can be decomposed by component. GRU encoding and MAE inference are per-agent operations with complexity linear in N. Attention-based message fusion depends on the number of actually received messages rather than the total agent count. The communication decision involves $N(N-1)$ agent pairs for computing sending probabilities, yielding a theoretical complexity of $O(N^2 d)$, which is on par with existing pairwise communication methods such as NDQ and TGCNet. In practice, the congestion-aware decision module keeps the communication probability at 30–50% (Table 2), substantially reducing the actual message transmission and fusion overhead. Furthermore, VOI is computed only at the receiver side, and MAE inference runs independently on each agent, both amenable to parallelization.