Decoupling Communication from Policy: Robust MARL under Bandwidth Constraints

Alexi Canesse, Benoît Goupil, Jesse Read, Sonia Vanier
École polytechnique (LIX), CNRS,
Institut Polytechnique de Paris,
Palaiseau, France
alexi.canesse@polytechnique.edu

Abstract

Communication enables coordination in multi-agent reinforcement learning (MARL), but many real-world applications, e.g., search-and-rescue with drone swarms, operate under severe bandwidth constraints. Many communication architectures still expose a coupled bottleneck in which a shared latent representation is used for both policy execution and inter-agent communication. Consequently, reducing message size directly limits the policy's latent space, often leading to significant performance degradation. We address this with two contributions. First, we introduce β, a normalised per-agent bandwidth budget that unifies sparsity, rounds, and message dimension into a single comparable constraint. Second, we provide SLIM, a minimal architecture that decouples the communication pathway from the policy's latent representation, allowing us to isolate the effect of bandwidth from the effect of policy capacity while benefiting from in-step communication. We evaluate our method on several partially-observable MARL benchmarks, where communication is essential. Our approach achieves state-of-the-art performance and exhibits scalability and robustness under limited communication, with only marginal degradation as bandwidth is reduced.

1 Introduction

Multi-agent reinforcement learning (MARL) has seen significant advancements in recent years, enabling agents to learn cooperative behaviours in complex environments [37, 4, 44]. One of the main difficulties is effective cooperation between agents, especially in partially-observable settings where each agent has access to only limited information about the environment and other agents. Some methods rely on a central controller, effectively transforming the multi-agent problem into a single-agent one [3, 40]. However, such centralised approaches are often impractical in real-world applications due not only to scalability limitations (with the state and action spaces scaling with the number of agents) but also because such centralisation is simply not feasible; agents may be required to operate in a decentralised fashion. Centralised training with decentralised execution (CTDE) [23, 19] has emerged as a popular paradigm, allowing agents to be trained with access to global information while executing policies based only on local observations. Consequently, communication becomes essential to compensate for limited local observations.

Communication is a mechanism that enables agents to get broader information about the environment and other agents. It enables agents to share information about their local observations, intentions, or learnt knowledge to improve cooperation and overall team performance. However, many real-world applications, such as search-and-rescue missions with drone swarms, autonomous vehicle fleets coordinating in traffic, or underwater exploration robots, involve severe bandwidth constraints. Reducing the size of communicated messages is essential for MARL systems to be deployed in such settings.

Preprint.

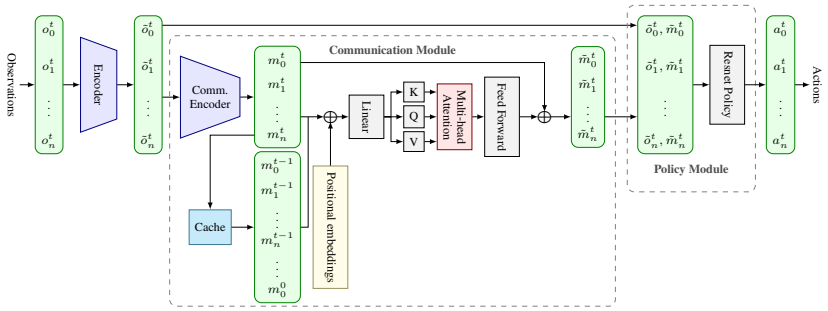


Figure 1: **Architecture of the method.** Observations $(o_i^t)_{i,t}$ of agents $\{i\}_{1,\dots,n}$ are encoded to reduce their dimension and increase their expressiveness before being passed to the Communication Module and the Policy Module. Crucially, encoded observations bypass the communication to reach the policy, allowing the communication module to reduce the dimension of the communication, through messages $(m_i^t)_{i,t}$, without loss of information for the policy.

However, reducing communication without compromising on performance is a significant challenge. Machine learning models typically rely on high-dimensional feature spaces, and forcing them to operate in very low-dimensional spaces can lead to significant performance degradation [46]. In MARL, many existing communication methods use a single shared latent representation both for the agent's policy and for the message it transmits [37, 35, 13]. In that regime, shrinking the message dimension imposes a double penalty: it removes information from transmitted messages and simultaneously reduces the representational capacity available to the policy itself, degrading overall performance. Some methods [4] already use a dedicated communication pathway but they communicate between timesteps, which introduces a lag in the communication. This issue is often solved using multi-round communication, but that is not always possible and increases the communication cost.

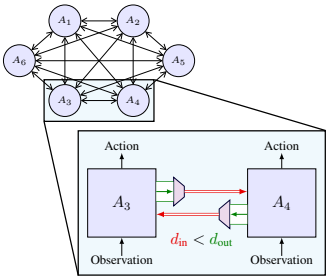


Figure 2: **Problem setting.** Agents interact; receive local observations, exchange messages, and take actions to maximise rewards. We designed a method to reduce the size of the communicated messages with limited loss in performance.

When deploying these systems in the real world, wireless networks impose two primary constraints: contention for medium access and bandwidth limitations [29]. The former requires sparsity in the communication graph (*i.e.*, reducing the frequency or number of connections) while the latter can be addressed by employing either sparsity or smaller messages. The main focus of this paper is on the second constraint (as displayed in Figure 2). We posit that optimising the size of individual messages is a more effective strategy for preserving coordination under bandwidth constraints.

To address this, we introduce Subdivided Lightweight Inter-agent Messaging (SLIM), a method that explicitly decouples communication from the policy input via a dedicated communication module. This separation allows the message dimension to be drastically reduced without restricting the policy's high-dimensional latent representation. We evaluate our approach across several partially observable MARL environments where communication is essential. Our method achieves state-of-the-art performance in high-bandwidth settings, while maintaining robust performance even as the message dimension is constrained to minimal values.

Concretely, we provide the following contributions:

- We establish a standardised evaluation protocol for communication-efficient MARL by defining a normalised bandwidth limit β. By unifying message size, transmission rate, and graph sparsity into one constraint, we provide a framework for systematic benchmarking of communication strategies under identical physical bandwidth limitations.
- We propose a communication architecture tailored to within-timestep MARL communication that isolates message compression within a dedicated module, allowing message dimension

to be reduced without constraining the higher-dimensional latent representation used for action selection.

- We provide an empirical study across four partially observable MARL benchmarks and several established baselines, showing that SLIM is consistently competitive at high bandwidth, notably more robust as bandwidth decreases, and that its cache is useful in environments that are not jointly fully observable.

2 Related Work

Multi-agent Reinforcement Learning. A straightforward approach to MARL treats each agent as an independent learner (IQL), applying standard reinforcement learning algorithms to each agent while treating others as part of the environment dynamics [39, 25]. This often leads to instability because the effective environment becomes non-stationary as other agents' policies change. To address this, the centralised training with decentralised execution (CTDE) paradigm has emerged [23, 8]. It allows the training process to take advantage of global information, typically through a centralised critic or joint value function[44] or value-factorisation methods [30], while ensuring agents rely only on local observations during decentralised execution. Our training framework follows this paradigm, using a PPO [32] actor–critic objective with a centralised value function to compute advantages.

Communication in MARL. Many real-world applications of MARL involve partial observability, making inter-agent communication crucial for effective coordination and decision-making. Several works focus on discrete or interpretable communication [7, 20, 26, 36] to address bandwidth limitations or accommodate low-power hardware. In contrast, our approach belongs to the category of continuous and differentiable communication. Within this domain, other methods have been proposed, such as different pooling techniques [37, 35] and graph neural networks [14, 21, 33, 16, 5, 17] to model agent interactions. Currently, attention mechanisms have emerged as the standard for MARL communication, whether integrated directly into the policy [13] or implemented as a separate module [4].

CommNet [37] introduced continuous communication in MARL by averaging agent hidden states to communicate between time steps. IC3Net [35] builds upon this using individual rewards to mitigate credit assignment challenges [8]. They also introduced a gating mechanism that enables the agents to learn when to communicate, further reducing communication throughput. However, in fully cooperative settings, this mechanism provides limited incentives to suppress unnecessary communication. Gated-ACML [24] addresses this limitation by using Q-value differences to decide when to communicate, facilitating the pruning of uninformative messages.

Similarly to our approach, TarMAC [4] has a dedicated communication module separated from the policy's latent space. They also use an attention mechanism for agent interaction. However, their communication protocol is executed between time steps, which introduces a one-step temporal delay in information exchange. While this can be mitigated through multi-round communication, such a strategy incurs a linear increase in bandwidth consumption relative to the number of rounds.

In contrast to dense communication paradigms, Commformer [13] introduces sparsity in the communication graph. Several works extend this by making the graph dynamic, allowing agents to selectively identify recipients or adapt content [15, 6, 43, 42, 28, 19, 22]. QLBT [9] goes further by taking into account network settings such as latency and saturation to adapt the communication graph between agents. Beyond topological adjustments, Sun et al. [38] adapt message dimensionality to available bandwidth, while SchedNet [18] frames agent selection as a scheduling problem, where only a subset of agents is permitted to transmit at each time step. Event-triggered methods [47, 12, 11] reduce transmission frequency by only communicating during critical state transitions. While effective, these approaches primarily focus on mitigating channel contention rather than reducing the dimensionality of individual messages, which is the central focus of our work. We consider these objectives to be complementary; combining temporal or topological sparsity with our proposed compression framework could further minimise overall communication overhead.

Information Theory. Bandwidth limitations in MARL communication have also been studied from information-theoretic perspectives [41, 5, 45]. Such approaches typically aim to produce compact and informative messages, often motivated by classic results of transmission-rate constraints and

principles of source coding [34]. Methods from this literature operate at the level of message encoding (in contrast to our architectural considerations at the representation and policy level); while not directly within the scope of our work, they could potentially be integrated together in a multi-agent system, for greater communication efficiency.

3 Subdivided Lightweight Inter-agent Messaging (SLIM)

In this section we introduce our SLIM architecture. We first present the notation and preliminaries for MARL with communication. We then describe the SLIM architecture in detail, followed by the training procedure.

3.1 Notation and Preliminaries

We consider multi-agent reinforcement learning (MARL) problems modelled as decentralised partially observable Markov decision processes (Dec-POMDPs) [27], defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, Z, O, n, \gamma)$. This tuple consists of a global state space $\mathcal{S}$, a joint action space $\mathcal{A} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_n$ for n agents, state-transition dynamics P, reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^n$, and a discount factor $\gamma \in [0, 1]$. Partial observability of these processes is modeled by the joint observation space $Z = Z_1 \times \ldots \times Z_n$ and the observation function O.

At discrete time t, each agent i receives a local observation $o_t^i \sim O(\cdot \mid s_t, a_{t-1})$ (where $o_t^i \in Z_i$) and selects an action $a_t^i \sim \pi^i(\cdot \mid \tau_t^i)$ under its policy π^i conditioned on its local action-observation history $\tau_t^i = (o_0^i, a_0^i, \ldots, a_{t-1}^i, o_t^i)$. The MARL objective is to learn a joint policy that maximises the expected cumulative discounted reward (return)

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right].$$

A Dec-POMDP is said to be **jointly fully observable** if the joint observation of all agents at all time t, $z_t = (o_t^1, \ldots, o_t^n)$, uniquely determines the true state s_t of the environment. A jointly fully observable Dec-POMDP is a decentralised Markov decision process (Dec-MDP) [27].

Communication is allowed in these settings to mitigate partial observability. Some methods consider communication as part of the action space while others treat it as a separate mechanism. In order to generalise over different communication protocols, we decompose the agent's decision process into two distinct stages: message generation and transmission, followed by action selection. At each time step t, each agent i first generates a message m_t^i based on its local history via a generation function μ^i. This message is then transmitted to other agents. The agent then selects its action a_t^i conditioned on both its history and the set of messages received from the other agents.

$$m_t^i = \mu^i(\tau_t^i); \qquad a_t^i \sim \pi^i(\cdot \mid \tau_t^i, \{m_t^j\}_{j \neq i}).$$

This formulation unifies various communication protocols, where the received message set can represent direct signals, an aggregated mean field, or the output of a graph neural network. Furthermore, this process can be applied iteratively: steps 1 and 2 can be repeated k times within a single time step to allow for multi-round consensus or intent-based communication.

3.2 The SLIM Architecture

Our proposed SLIM architecture is illustrated in Figure 1. It implements the two-stage decision process using three main components: an observation encoder, a communication module, and a policy network. Non-jointly fully observable settings are specifically addressed through a message history cache integrated within the communication module.

Observation Encoding. The encoder E processes the local observation o_t^i of agent i at time step t to produce a latent representation $\tilde{o}_t^i = E(o_t^i)$. This encoding step reduces the dimensionality of the observation while preserving relevant information for the policy.

Message Generation and Transmission. The representation $\tilde{o}_t^i$ is passed to a communication encoder E_c, which projects it into a compact message vector suitable for transmission defined by $m_t^i = E_c(\tilde{o}_t^i)$. These messages are then broadcasted to the other agents. Simultaneously, the agent receives the set of current messages $\{m_t^j\}_{j \neq i}$ from its peers.

Message History Cache. To address the partial observability of the environment, where the current observation alone is insufficient to determine the true state, each agent maintains a message cache $\mathcal{C}_t^i$ potentially containing knowledge about the environment that could be useful later. This buffer stores the history of all exchanged messages in the system up to time t:

$$\mathcal{C}_t^i = \mathcal{C}_{t-1}^i \cup \{m_t^j\}_{1 \leq j \leq n} = \{m_{t'}^j\}_{\substack{1 \leq t' \leq t \\ 1 \leq j \leq n}}^{1 \leq t' \leq t}.$$

Because the cache stores only transmitted messages rather than full observation embeddings, its memory footprint scales linearly with the communication dimension d.

Temporal Attention Aggregation. To synthesise the available information, we apply a transformer-based attention block over the full content of the cache $\mathcal{C}_t^i$, which contains both the historical log and the messages just received at the current step. This mechanism dynamically weights the importance of each entry, enabling the agent to jointly reason over past and current information to construct a context vector $\tilde{m}_t^i$. Concretely, the transformer input is the sequence of cached messages augmented with two learnt positional embeddings: a temporal embedding that identifies the communication step and an agent embedding that identifies the sender. The current-step messages and the cached messages are processed by the same attention block, so $\tilde{m}_t^i$ is an attention-weighted summary over who sent what and when, rather than a recurrent hidden state that privileges recent messages by construction.

Action Selection. Finally, the policy network π^i selects an action by conditioning on both the local representation and the context vector using

$$a_t^i \sim \pi^i(\tilde{o}_t^i, \tilde{m}_t^i).$$

While the message history cache is critical for Dec-POMDPs, our architecture allows it to be optionally disabled. In Dec-MDP settings, where the current observation is sufficient for optimal decision-making, the cache mechanism can be deactivated. In this configuration, the attention block operates only on the current set of received messages.

3.3 Centralised Training

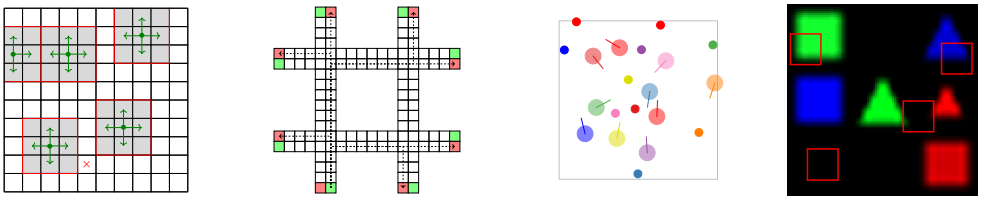
We follow the centralised training with decentralised execution paradigm. During training, we use a shared value function that receives the concatenated inputs from all agents. However, during execution, each agent only has access to its local observation and the messages received from other agents.

We use the standard MAPPO algorithm [44] to train our agents. Specifically, we apply the PPO objective [32] to each agent individually irrespective of parameter sharing. For an agent i having policy parameters θ_k with candidate policy parameters θ, an agent i with parameters θ_k is updated to new parameters θ by optimising the objective $\mathcal{L}_i^p(\theta)$.

$$\rho_{s,a}^{\theta_k} = \frac{\pi_\theta^i(a|s)}{\pi_{\theta_k}^i(a|s)}; \quad \mathcal{L}_i^p(\theta) = -\mathop{\mathbb{E}}_{s,a \sim \pi_\theta} \min\left(\rho_{s,a}^{\theta_k} A_{\pi_{\theta_k}}^i(s,a),\ \text{clip}\left(\rho_{s,a}^{\theta_k}, 1-\varepsilon, 1+\varepsilon\right) A_{\pi_{\theta_k}}^i(s,a)\right)$$
$$- \alpha \mathop{\mathbb{E}}_{s \sim \mathcal{S}} \sum_{a \in \mathcal{A}_i} \pi_\theta^i(a|s) \log \pi_\theta^i(a|s)$$

where $A_{\pi_\cdot}^i(\cdot,\cdot)$ is the advantage function estimated using Generalised Advantage Estimation (GAE) [31], ε is a hyperparameter controlling the size of the policy update, α is a temperature controlling the entropy bonus [10] that encourages exploration. A value estimation is necessary to compute the advantage estimates, and we train the value function $V_\theta^i(\cdot)$ for each agent i by minimising the loss

$$\mathcal{L}_i^v(\theta) = \mathop{\mathbb{E}}_{s,r \sim \pi_\theta} \left(V_\theta^i(s_t) - \hat{R}_t^i\right)^2.$$



(a) Predator-Prey (medium). (b) Traffic Junction (hard). (c) VMAS: Navigation (10 agents) (d) SHAPES

Figure 3: **Environments used in our experiments.** (3a) Predator-Prey: n predators cooperate to capture a fixed prey on a grid, each observing only a window (grey cells). (3b) Traffic Junction: cars navigate an intersection without vision, relying on communication to avoid collisions; green cells are spawn points, red cells goals, dashed arrows possible routes. Cars spawn with probability p, up to a cap of n. (3c) Navigation: n agents in a bounded continuous space must reach individually assigned goals (matching colour) without seeing peers. (3d) SHAPES: agents spawn on a random image of coloured shapes and must reach a target shape while observing only a local patch (red squares).

Since we are using centralised training, the state s_t used in the value function is the joint observation rather than the local observation of agent i. To this end, our value function takes as input the concatenation of all agents' encoded observations and outputs a value estimate for each agent.

We can then combine those two losses to obtain the overall loss function per agent and take the mean over all agents to get the final loss function

$$\mathcal{L}(\theta, \theta_k) = \frac{1}{n} \sum_{i=1}^{n} \left(\mathcal{L}_i^p(\theta) + \mathcal{L}_i^v(\theta) \right).$$

4 Experiments

In this section, we use multi-agent environments to evaluate the performance of our proposed method, the SLIM architecture, against several baselines. Our experiments aim to answer the following research questions: $(\boldsymbol{i})$ Under a high communication bandwidth β, does SLIM match or outperform representative baselines on multi-agent tasks? $(\boldsymbol{ii})$ How does SLIM's performance evolve as the communication bandwidth budget is reduced over a broad range, and is it more robust than baseline methods in partially observable environments? $(\boldsymbol{iii})$ Does the cache help in improving performance in non-jointly observable environments?

4.1 Normalised Agent Bandwidth

We define the normalised agent bandwidth β as the maximum transmission capacity allocated to each agent (in floating-point scalars) per time step divided by the population. This value makes comparison easier between environments of varying number of agents. To ensure equivalent conditions, each model is subjected to the same normalised agent bandwidth. This constraint accounts for the message dimension d, the message frequency k (corresponding to the number of communication rounds per time step as defined in Section 3.1), and the sparsity of the communication graph σ, defined as the fraction of the total population a given agent transmits to, into a single scalar limit. Formally, this bandwidth constraint is defined as

$$\sigma \times k \times d \leq \beta. \tag{1}$$

This standardisation allows us to compare different models at equal bandwidth limits, regardless of their specific communication strategy (e.g., trading sparsity in the communication graph for higher message resolution). Details on the computation of the bandwidth constraints for each model configuration are given in Section A.1.

4.2 Baselines

We compare our SLIM architecture against a diverse set of established MARL baselines to cover a spectrum of communication strategies. We select CommNet [37] to represent a foundational continuous communication approach. We also include IC3Net [35] which improves upon this architecture

by using individual rewards and a gating mechanism. However, consistent with observations in the original paper, we verified that in our fully cooperative scenarios, this gating mechanism remains effectively open, causing IC3Net to operate with a fully dense communication graph. We further include TarMAC [4] because it is the closest prior architecture that already separates communication from the policy pathway. Finally, we benchmark against CommFormer [13], a state-of-the-art transformer-based architecture. We evaluate both the dense variant (i.e., using a complete communication graph) to serve as a high-performance upper bound, and the sparse variant, which learns a fixed sparse communication graph, to assess the effectiveness of communication graph sparsification techniques as a bandwidth accommodation strategy.

4.3 Environments

We evaluate on four standard partially-observable MARL benchmarks (illustrated in Figure 3), with full descriptions in Section B. **Predator-Prey** [35] is a grid world where predators with local vision cooperate to locate a stationary prey; since the prey does not move, past observations carry state information absent from the current joint observation, making the environment non-jointly-observable and motivating the message history cache (ablated in Section 4.7). **Traffic Junction** [35] requires vision-less agents to cross an intersection along pre-assigned routes; concatenating all agents' positions fully determines the global state, so the environment is a Dec-MDP and the cache is disabled. **Navigation** [2] places agents in a bounded continuous space where they must reach individual goals using acceleration actions under momentum, observing only their own position, velocity, and relative goal distance. **SHAPES** [1, 4] spawns agents on an image of coloured shapes that must each reach a target colour while observing only a local patch; differing per-agent goals mean cached peer messages can carry goal-relevant information, providing a second testbed for the cache ablation.

4.4 Experimental Setup

Reproducibility and Compute All experiments are conducted across four seeds $\{1, 2, 3, 4\}$ to ensure statistical robustness. We report mean performance alongside the standard error of the mean $(\text{std}/\sqrt{n_{\text{seeds}}})$ for all graphical representations and alongside the standard deviation for tabular results. The total computational budget for these experiments exceeded 5,000 GPU-hours on NVIDIA H100 GPUs. This is mostly due to the results necessary for Section 4.6. While this number can seem high, it does not reflect inefficiencies, it is driven by the depth of our experiments, which required about 600 trainings (5 algorithm, 5 environments, 7 β values and 4 training seeds).

Hyperparameter Protocol For all baselines, we used the hyperparameters specified in their original publications after verifying their performance are unchanged at $\beta = 64$. For CommFormer, which was not originally evaluated on TrafficJunction, we performed a grid search over the PPO epoch parameter in $\{5, 10, 15\}$ and the PPO clip value in $\{0.05, 0.2\}$. These ranges were selected based on the values used in the original CommFormer experiments on other benchmarks. For SLIM, we performed a grid search, in high bandwidth, over the PPO epoch parameter in $\{1, 5, 10\}$. The final values are provided in Table 2 in the Appendix.

4.5 Performance in High-Throughput Regimes

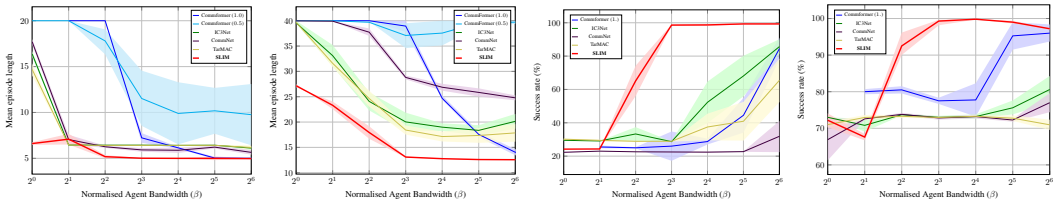
We first establish a performance ceiling for all evaluated architectures by analysing them at the upper bandwidth bound ($\beta = 2^6$). This regime represents a permissive setting where the communication channel is sufficiently wide, allowing us to assess the maximum expressive power of each model's communication protocol. As detailed in Table 1, SLIM consistently achieves state-of-the-art performance across all benchmarks, either outperforming or maintaining statistical parity with established baselines. This verifies that SLIM's architectural design, while optimised for low-resource environments, does not sacrifice absolute performance when bandwidth is abundant.

4.6 Robustness Under Various Bandwidth Constraints

To evaluate the robustness of each architecture, we swept the normalised agent bandwidth β across a logarithmic scale ranging from 2^0 to 2^6. For each value, we trained every model on 4 distinct random seeds, resulting in a total of 28 training runs per model. Figure 4 and Figure 5 illustrate the evolution of performance as a function of β, with shaded regions indicating the standard error.

Table 1: **Results on the different baselines using a bandwidth limit of $\beta = 2^6$.** Performance on Predator-Prey is measured in average steps to capture the prey (lower is better), while performance on Traffic Junction is measured in success rate (higher is better) and in reward for Navigation (higher is better). (†) Results for CommFormer are missing on the Navigation environment because it appears that the method is not able to converge.

Environment	Difficulty	CommNet	IC3Net	TarMAC	CommFormer	**SLIM**
Predator-Prey	Easy	$5.68_{\pm 0.48}$	$6.10_{\pm 0.46}$	$6.18_{\pm 0.50}$	$\mathbf{5.00}_{\pm \mathbf{0.20}}$	$\mathbf{4.97}_{\pm \mathbf{0.04}}$
	Medium	$24.78_{\pm 0.99}$	$20.14_{\pm 2.52}$	$17.86_{\pm 3.46}$	$14.06_{\pm 1.54}$	$\mathbf{12.57}_{\pm \mathbf{0.15}}$
Traffic Junction	Easy	$31.2_{\pm 19.2}$	$85.8_{\pm 10.0}$	$65.5_{\pm 29.6}$	$84.5_{\pm 13.2}$	$\mathbf{99.3}_{\pm \mathbf{0.30}}$
	Medium	$77.0_{\pm 5.34}$	$80.6_{\pm 7.42}$	$71.0_{\pm 2.87}$	$\mathbf{96.0}_{\pm \mathbf{4.31}}$	$\mathbf{97.2}_{\pm \mathbf{0.84}}$
Navigation		$0.49_{\pm 0.06}$	$0.28_{\pm 0.24}$	$0.64_{\pm 0.24}$	†	$\mathbf{0.81}_{\pm \mathbf{0.05}}$



(a) Predator-Prey easy (b) Predator-Prey medium (c) Traffic Junction easy (d) Traffic Junction medium

Figure 4: **Performance of SLIM and baselines across a logarithmic range of normalised agent bandwidth values β (from 2^0 to 2^6).** Top: mean episode length for the Predator-Prey environment (lower values indicate better performance). Bottom: success rate for the Traffic Junction environment (higher is better). Shaded regions denote the standard error of the mean across 4 seeds. One can note that data points for the dense variant of CommFormer are absent in the lowest bandwidth regimes ($\beta = 2^0$) because the architectural constraints of the model prevent it from satisfying such strict bandwidth limits. See details in Section A.1.

CommFormer achieves competitive performance in high-bandwidth regimes; however, this performance degrades significantly as the bandwidth constraint tightens. The sparse variant ($\sigma = 0.5$) exhibits higher variability and lower results. We excluded this variant from Traffic Junction because a fixed sparse communication graph is unsuitable for tasks where specific and dynamic interactions are required (e.g., cars at the same intersection). Similarly, IC3Net and CommNet are less robust, showing either consistently lower returns or quick collapse under limited bandwidth. We observe similar trends for TarMAC, except in Navigation where it maintains a more stable performance. In contrast, SLIM demonstrates robustness under bandwidth constraints. It matches or surpasses the peak performance of the strongest baseline at high bandwidth while maintaining high rewards even under strict constraints where others fail. While a minor performance drop occurs in Traffic Junction under severe bandwidth limitations, SLIM consistently offers robust performances. It validates its ability to learn robust policies without relying on excessive transmission volume.

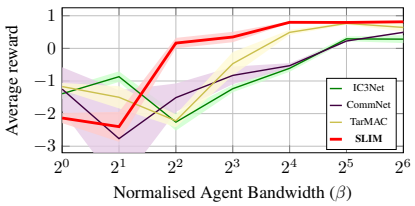
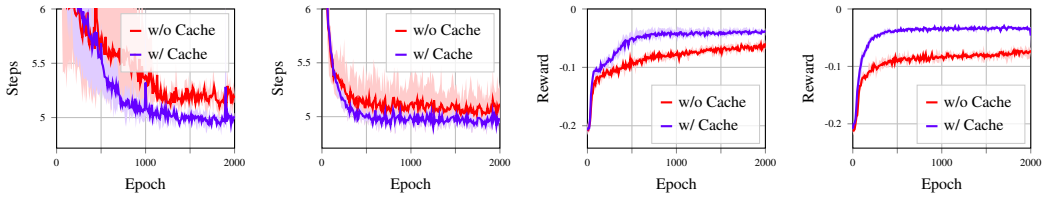


Figure 5: **Performance on a logarithmic range of normalised agent bandwidth values β in the Navigation environment.** Shaded regions denote the standard error. SLIM achieves the best performance in high bandwidth settings and is more resilient to the reduction of the bandwidth.

4.7 Ablation Study: Impact of Message History Cache

To validate the hypothesis that the historical context is beneficial for environments with partial observability, we conducted an ablation study comparing the full SLIM architecture against the variant with the cache deactivated. The results, presented in Figure 6, demonstrate the clear advantage



(a) Low bandwidth (b) High bandwidth (c) Low bandwidth (d) High bandwidth

Figure 6: **Ablation study on the effect of the cache on an non jointly observable environment.** We compare the performance of SLIM with and without the temporal cache mechanism in the Predator-Prey easy environment ((6a) and (6b)) and the SHAPES environment ((6c) and (6d)) under two different communication bandwidths: 2^3 (6a) and (6c)) and 2^6 ((6b) and (6d)). The results demonstrate that the cache significantly improves results along the training process while increasing stability; for non-jointly fully observable environments. The line reported is the mean over 4 seeds, with the shaded area representing the standard error.

of the memory mechanism. Additional detailed results are available in Table 4. In the Predator-Prey and the SHAPES environments, where tracking agent trajectories over time is advantageous, the cache-enabled model consistently outperforms the cache-disabled baseline, achieving the objective in fewer steps in Predator-Prey and higher rewards in SHAPES. The only exception occurs at very low bandwidths in Predator-Prey, where we hypothesise that the limited representational capacity of the latent space prevents the effective disentanglement of spatial features, temporal dependencies, and agent identities.

5 Limitations

Our study targets algorithmic communication efficiency and does not constitute a complete model of real wireless deployment. The normalised bandwidth budget β provides a controlled proxy unifying message dimension, communication rounds, and graph sparsity, but it abstracts away packet headers, quantisation, latency, routing overhead, packet loss, and medium contention. Extending β to capture some of these factors, and evaluating SLIM under more realistic network conditions, is left for future work. We also do not empirically combine SLIM with information-theoretic message compression, which we hypothesise to be complementary but leave to future work. The message history cache had negligible overhead in our experiments, but its memory cost grows linearly with episode length; windowing strategies or compressed memory representations could be considered for *very* long-horizon tasks.

6 Conclusion

We proposed SLIM, a simple and modular approach to communication in multi-agent reinforcement learning that explicitly decouples the architecture responsible for inter-agent messaging encoding, from that dedicated to policy execution (responsible for actions). We introduced an evaluation protocol based on a normalised agent bandwidth metric (β). This unified measure integrates message size, transmission frequency, and graph sparsity into a single constraint, enabling a systematic benchmarking of diverse communication strategies under identical physical limitations. Our empirical results across multiple partially observable benchmarks demonstrate that SLIM not only matches state-of-the-art methods in high-bandwidth regimes but exhibits clear robustness as constraints tighten. While baseline performance degrade rapidly under severe constraints, SLIM maintains effective coordination even with lower bandwidth limits. Finally, ablation studies confirmed the benefit of our proposed message history cache for improving learning in non-jointly observable environments. Future work will further explore connections between modern communication-efficient MARL settings and complementary developments in network science, where transmission rate and information-theoretical views have been widely considered; perhaps allowing further calibration of communication efficiency in a more general sense.

Acknowledgements

This work received financial support from Crédit Agricole SA through the research chair *Trustworthy and Responsible AI* at École Polytechnique.

This work was granted access to the HPC resources of IDRIS under the allocation 2025-AD011017102 made by GENCI.

Finally, we would like to thank Mathis Le Bail, Clément Elliker and Mahammed Elsharkawy for their help and insights throughout the project.

References

[1] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48, 2016.

[2] Matteo Bettini, Ryan Kortvelesy, Jan Blumenkamp, and Amanda Prorok. Vmas: A vectorized multi-agent simulator for collective robot learning. *The 16th International Symposium on Distributed Autonomous Robotic Systems*, 2022.

[3] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998(746-752):2, 1998.

[4] 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. PMLR, 2019.

[5] Shifei Ding, Wei Du, Ling Ding, Jian Zhang, Lili Guo, and Bo An. Robust multi-agent communication with graph information bottleneck optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3096–3107, 2023.

[6] Ziluo Ding, Tiejun Huang, and Zongqing Lu. Learning individually inferred communication for multi-agent cooperation. *Advances in neural information processing systems*, 33:22069–22079, 2020.

[7] Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

[8] 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, 2018.

[9] Ziyang Guo, Zhenyu Chen, Peng Liu, Jianjun Luo, Xun Yang, and Xinghua Sun. Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks. *IEEE Journal on Selected Areas in Communications*, 40(5):1587–1599, 2022.

[10] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[11] Shuai Han, Mehdi Dastani, and Shihan Wang. Model-based sparse communication in multi-agent reinforcement learning. In *Proceedings of the 2023 international conference on autonomous agents and multiagent systems*, pages 439–447. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2023.

[12] 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, 2021.

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

[14] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph convolutional reinforcement learning. 2020.

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

[16] Rui Jiang, Xuetao Zhang, Yisha Liu, Yi Xu, Xuebo Zhang, and Yan Zhuang. Multi-agent cooperative strategy with explicit teammate modeling and targeted informative communication. *Neurocomput.*, 586(C), June 2024.

[17] Rui Jiang, Xuetao Zhang, Yisha Liu, Yi Xu, Xuebo Zhang, and Yan Zhuang. Multi-agent cooperative strategy with explicit teammate modeling and targeted informative communication. *Neurocomputing*, 586:127638, 2024.

[18] 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.

[19] Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2022.

[20] Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. In *International Conference on Learning Representations*, 2017.

[21] Sheng Li, Jayesh K Gupta, Peter Morales, Ross Allen, and Mykel J Kochenderfer. Deep implicit coordination graphs for multi-agent reinforcement learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pages 764–772, 2021.

[22] Xinran Li and Jun Zhang. Context-aware communication for multi-agent reinforcement learning. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 1156–1164, 2024.

[23] 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.

[24] Hangyu Mao, Zhengchao Zhang, Zhen Xiao, Zhibo Gong, and Yan Ni. Learning agent communication under limited bandwidth by message pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5142–5149, 2020.

[25] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012.

[26] Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

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

[28] Murtaza Rangwala and Ryan Williams. Learning multi-agent communication through structured attentive reasoning. *Advances in Neural Information Processing Systems*, 33:10088–10098, 2020.

[29] Theodore S Rappaport. *Wireless communications: Principles and practice, 2/E*. Pearson Education India, 2010.

[30] 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.

[31] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

[32] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[33] Esmaeil Seraj, Zheyuan Wang, Rohan Paleja, Daniel Martin, Matthew Sklar, Anirudh Patel, and Matthew Gombolay. Learning efficient diverse communication for cooperative heterogeneous teaming. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 1173–1182, 2022.

[34] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[35] 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.

[36] Yang Su, Yali Du, and Yansha Deng. Goal-oriented semantic communication in bandwidth-constrained marl. In *2025 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1274–1279. IEEE, 2025.

[37] Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. *Advances in neural information processing systems*, 29, 2016.

[38] Qingshuang Sun, Denis Steckelmacher, Yuan Yao, Ann Nowe, and Raphael Avalos. Dynamic size message scheduling for multi-agent communication under limited bandwidth. *IEEE Transactions on Mobile Computing*, 2024.

[39] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on Machine Learning*, pages 330–337, 1993.

[40] 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.

[41] Rundong Wang, Xu He, Runsheng Yu, Wei Qiu, Bo An, and Zinovi Rabinovich. Learning efficient multi-agent communication: An information bottleneck approach. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9908–9918. PMLR, 13–18 Jul 2020.

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

[43] Yuanfei Wang, Fangwei Zhong, Jing Xu, and Yizhou Wang. Tom2c: Target-oriented multi-agent communication and cooperation with theory of mind. *arXiv preprint arXiv:2111.09189*, 2021.

[44] 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.

[45] Lebin Yu, Qiexiang Wang, Yunbo Qiu, Jian Wang, Xudong Zhang, and Zhu Han. Effective multi-agent communication under limited bandwidth. *IEEE Transactions on Mobile Computing*, 23(7):7771–7784, 2024.

[46] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[47] Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Efficient communication in multi-agent reinforcement learning via variance based control. *Advances in neural information processing systems*, 32, 2019.

A Experiment Details

Table 2: **Hyperparameter configuration for the SLIM architecture across all benchmarks.** These values were optimised using a fixed message dimensionality of $d = 64$ (2^6) to ensure that the architecture remains robust across varying bandwidths without overfitting to specific constraints.

	Predator-Prey		Traffic Junction		Navigation	SHAPES
	Easy	Medium	Easy	Medium	/	/
Clip param. ε	0.2	0.2	0.2	0.2	0.2	0.2
Entropy coeff. σ	0.02	0.02	0.02	0.02	0.02	0.02
Episodes/epoch	500	500	500	500	100	100
Number of epochs	2000	2000	600	4000	3200	2000
PPO Epochs	5	5	1	1	1	1
Hidden size	128	128	128	128	128	128
Message dim.	2^6	2^6	2^6	2^6	2^6	2^6
Using cache	`True`	`True`	`False`	`False`	`False`	`True`
Discount γ	0.99	0.99	0.99	0.99	0.99	0.99
Learning rate	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Replay buffer (# ep.)	1000	1000	1000	1000	250	5000
GAE λ	0.95	0.95	0.95	0.95	0.95	0.95

A.1 Detailed Communication Parameters

Table 3: **Communication parameters for the normalized agent bandwidth configurations.** For each bandwidth budget β, we report the graph density σ, communication rounds k, and largest feasible message dimension d satisfying $\sigma \times k \times d \leq \beta$. TarMAC is evaluated in the dense one-pass setting used in our sweeps. Dense CommFormer cannot satisfy $\beta = 2^0$ with integer $d \geq 1$. En-dashes (–) denote inaccessible configurations.

	$\beta = 2^0$			$\beta = 2^1$			$\beta = 2^2$			$\beta = 2^3$			$\beta = 2^4$			$\beta = 2^5$			$\beta = 2^6$		
Model	σ	k	d	σ	k	d	σ	k	d	σ	k	d	σ	k	d	σ	k	d	σ	k	d
CommFormer	–	–	–	1.0	2	2^0	1.0	2	2^1	1.0	2	2^2	1.0	2	2^3	1.0	2	2^4	1.0	2	2^5
CommFormer (sparse)	0.5	2	2^0	0.5	2	2^1	0.5	2	2^2	0.5	2	2^3	0.5	2	2^4	0.5	2	2^5	0.5	2	2^6
CommNet	1.0	1	2^0	1.0	1	2^1	1.0	1	2^2	1.0	1	2^3	1.0	1	2^4	1.0	1	2^5	1.0	1	2^6
IC3Net	1.0	1	2^0	1.0	1	2^1	1.0	1	2^2	1.0	1	2^3	1.0	1	2^4	1.0	1	2^5	1.0	1	2^6
TarMAC	1.0	1	2^0	1.0	1	2^1	1.0	1	2^2	1.0	1	2^3	1.0	1	2^4	1.0	1	2^5	1.0	1	2^6
SLIM	1.0	1	2^0	1.0	1	2^1	1.0	1	2^2	1.0	1	2^3	1.0	1	2^4	1.0	1	2^5	1.0	1	2^6

B Additional Results

Navigation In order to illustrate that the method could scale to more agents, we also illustrate SLIM using 4, 20, 24, 32 and 42 agents (anonymised links). These illustrations show agents successfully reaching their individual goals while avoiding collisions despite not seeing each other.

Due to computational constraints, we have only benchmarked SLIM against TarMAC (the strongest baseline when using 4 agents) in the 20 agents setting. In this setting, TarMAC gets a reward of 0.02 ± 0.01 while SLIM gets 0.95 ± 0.003.

Reproducibility

We publicly released the code to reproduce all our experiments on GitHub. All experiments can be run using a simple script. Environment files are available with all the libraries version used to be able to exactly reproduce our setting.

Table 4: **Results of the ablation study on the impact of the cache in non-jointly observable environments.** We report mean episode length across varying communication dimensions for Predator-Prey and rewards for SHAPES. Lower values are better for episode length; higher values are better for rewards.

Setting	Cache	β						
		1	2	4	8	16	32	64
Pred.-Prey Easy	w/	**6.61**$\pm$.29	7.10$\pm$1.2	**5.19**$\pm$.18	**5.01**$\pm$.03	**4.99**$\pm$.03	**4.99**$\pm$.04	**4.97**$\pm$.04
	w/o	**6.65**$\pm$.34	**6.49**$\pm$.1	**5.16**$\pm$.06	5.22$\pm$.04	5.21$\pm$.05	5.12$\pm$.02	5.26$\pm$.04
Pred.-Prey Medium	w/	27.2$\pm$0.6	**23.3**$\pm$1.5	17.9$\pm$2.5	**13.1**$\pm$0.23	**12.8**$\pm$0.1	**12.6**$\pm$0.2	**12.6**$\pm$0.2
	w/o	**26.8**$\pm$0.34	27.2$\pm$2.9	**15.8**$\pm$4.5	14.3$\pm$1.5	13.3$\pm$0.4	13.1$\pm$0.2	13.3$\pm$0.4
SHAPES $\times 10^2$	w/	**-5.3**$\pm$.20	**-4.8**$\pm$.82	**-4.7**$\pm$.30	**-3.9**$\pm$.64	**-3.1**$\pm$.18	**-3.0**$\pm$.32	**-4.6**$\pm$.22
	w/o	-7.0$\pm$.40	-7.2$\pm$.55	-6.8$\pm$.79	-5.9$\pm$.76	-6.4$\pm$.44	-7.0$\pm$.43	-7.2$\pm$.44

Licenses

We release our implementation under the MIT licence. We do not redistribute the code of the baselines. CommFormer is available through the Apache License 2.0, IC3Net and TarMAC through the MIT License, and CommNet through the BSD license.

The Predator-Prey and Traffic Junction environments are available using the MIT License, VMAS through the GPL3 license and we reimplemented the SHAPES environment based on the description provided in the original paper, thus we release it under the MIT License.

Environments Details

Predator-Prey. The Predator-Prey environment is a multi-agent grid world where agents (predators) must cooperate to locate a prey with limited vision (local observability) centred on them. Each agent gets a negative reward until it reaches the prey. Agents can move in four directions (up, down, left, right) or stay in place. They benefit from a strong communication since knowing where the prey is not gives information about its location. Since the prey cannot move, the environment is not jointly fully observable: information about previously seen locations is information on the state, that could be missing from the current joint observation. To address this, we enable the message history cache for this environment and provide an ablation study in Section 4.7 to assess its impact.

Traffic Junction. The Traffic Junction environment simulates intersections that agents *without any vision* need to cross. Agents enter the environment stochastically and navigate along randomly assigned routes. The lack of vision makes communication essential to avoid collisions, making this environment a suitable benchmark for communication MARL. In this environment, agents do not have to navigate: they can either go forward along their route or wait. The reward strongly incentivise collision avoidance while lightly encouraging efficiency: agents receive an increasingly negative reward for each time step they are active in the environment, and a large negative reward if they collide with another car. Since the concatenation of all agents' local observations (cars position) fully characterises the global configuration of the system, the environment is categorised as a decentralised Markov decision process (Dec-MDP). Consequently, the message history cache is deactivated for this environment.

Navigation The navigation environment [2] implements a bounded continuous world where n randomly spawned agents must navigate to individual randomly spawned goals without colliding with other agents, without seeing each other. Each agent only sees its position, speed and relative distance to its goal. Actions are acceleration in different directions and the environment simulates momentum, making fine control complicated. Agents receive a reward proportional to the distance gained to their goal at each time step, and a large negative reward if they collide with another agent.

SHAPES. In order to verify the effectiveness of the message history cache in Section 4.7, we introduced a variation of the simple SHAPES environment [1, 4]. Images containing coloured shapes

are generated. Agents spawn at random positions on a random image and must find a (possibly different) random colour each whilst only being able to observe their surroundings. This environment is not partially jointly fully observable. Since agents have different goals, information on what other agents have seen in the past can contain information about another agent's goal. Each agent receives a reward of 10^{-2} until they reach their target, at which point they receive a reward of 0.