Interpretable Multi-Agent Communication via Information Gating

Stav Belogolovsky [1] **Eran Iceland** [2] **Itay Naeh** [1] **Ariel Barel** [3] **Shie Mannor** [1 4]

Abstract

Multi-Agent Reinforcement Learning (MARL) holds significant promise for complex coordination tasks, where communication is often vital. However, emergent communication in MARL is typically opaque and unintelligible to humans, and forcing human-like language can impede learning or performance. Furthermore, agents struggle when processing high-dimensional unstructured communication. Addressing these critical challenges, we propose a novel framework enabling agents to learn effective communication while ensuring its interpretability. Our core idea is to make the subject of communication transparent, rather than the message content itself. This is achieved by training agents to learn a policy that selectively gates the information flow from their observation to the communication channel. By adopting an object-oriented perspective and using a text-to-mask model that maps terms to observation features, agents learn to select and communicate only relevant information. This approach provides enhanced interpretability by revealing message context, and mitigates information overload through selective masking. We introduce this comprehensive framework, demonstrating its effectiveness and robustness across multi-agent tasks that require communication. We analyze emergent communication protocols and their resulting interpretability and release our code and environments to support further research.

1. Introduction

Multi-Agent Reinforcement Learning (MARL) is a widely studied subject, where multiple agents interact with a shared environment, learning to optimize their return. Although typically each agent operates according to its own experience (i.e., decentralized joint policy), many potential real-life applications allow the agents to communicate. By allowing agents to exchange critical information relevant to their shared task, communication can enhance overall performance and coordination. In fact, recent research has been dedicated to learning in this challenging setting. Although most of the previous research was aimed at improving performance, a recent research trend focuses on human interpretability. In addition to the direct interpretation and analysis of the agent's policies, interpretable communications allow teaming up with human players and, in some cases, learning from humans. When considering interpretable communication, a trivial approach is to assimilate human communication. An approach to solving this problem is to force agents to communicate as humans (Li et al., 2024; Havrylov & Titov, 2017; Lazaridou et al., 2016), however, it has been shown that agents are less likely to learn a human-like communication protocol (Kottur et al., 2017).

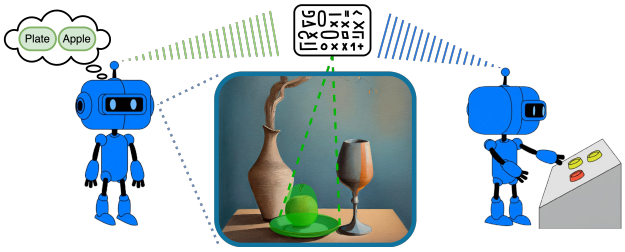


Figure 1. Understanding the context instead of translating the actual message; While the actual message may not be intelligible, the context provides enhanced interpretability.

Artificial agents can be perceived as bees or ants, as they usually share goals and, in many cases, are trained in a centralized manner. Allowing agents to develop their own language may result in a nontranslatable communication protocol. Current approaches for learning interpretable communication rely on various methods; alignment with human language, using a meaningful communication protocol, or receiving guidance from a large language model (LLM),

[1]Department of Electrical and Computer Engineering, Technion—Israel Institute of Technology, Haifa, Israel 3200003 [2]Hebrew University of Jerusalem, Jerusalem, Israel 9190501 [3]The Faculty of Computer Science, Technion—Israel Institute of Technology, Haifa, Israel 3200003 [4]Nvidia Research. Correspondence to: Stav Belogolovsky <stav.belo@gmail.com>.

Accepted at the ICML 2025 Workshop on Collaborative and Federated Agentic Workflows (CFAgentic@ICML'25), Vancouver, Canada. July 19, 2025.

although most of the literature focused on giving direct semantics to the messages. While these approaches are promising, the observation that the agents' language may greatly differ from human's language can harm the ability to translate the resulted messages or impair the agents' performance. Beyond the challenge of making communication interpretable, agents operating in high-dimensional observation spaces face the burden of processing and effectively utilizing all incoming information. Simply transmitting raw observations (e.g., images) or dense, unstructured communication vectors requires the receiving agents to disentangle relevant signals from noise. This "information overload" can impede the learning of communication and degrade overall performance.

To tackle these challenges, we introduce a novel framework designed to encourage agents to learn effective communication while ensuring its interpretability. Our fundamental insight is to step away from attempting to enforce linguistic similarity and instead focus on making the **subject** of the communication transparent to human observers Figure 1. We achieve this by enabling agents to learn a communication policy that gates the information flow from their observation to the communication channel. Our framework draws inspiration from the human tendency to perceive tasks and communicate about relevant entities. A key component is the text-to-mask model Section 4.1, which maps human-understandable textual terms to designated regions or feature sets within the agent's observation space. The agent's communication policy then learns to select which of these textual terms are most relevant. This selection is converted by the text-to-mask model into a mask, applied to the agent's observation. This effectively filters out irrelevant features, controlling the information content of the message.

Our contribution: First, we propose a comprehensive framework for interpretable multi-agent communication that achieves interpretability by gating the information flow from observation to communication. Second, we introduce the text-to-mask model as a crucial component to ground the observation space in textual terms, enabling object-oriented perception. Third, we demonstrate the effectiveness and robustness of our framework in two multi-agent tasks characterized by high partial observability, and release our code and environments to facilitate further research. Finally, we analyze sample trajectories to illustrate how the context of communication becomes human-interpretable.

2. Related Work

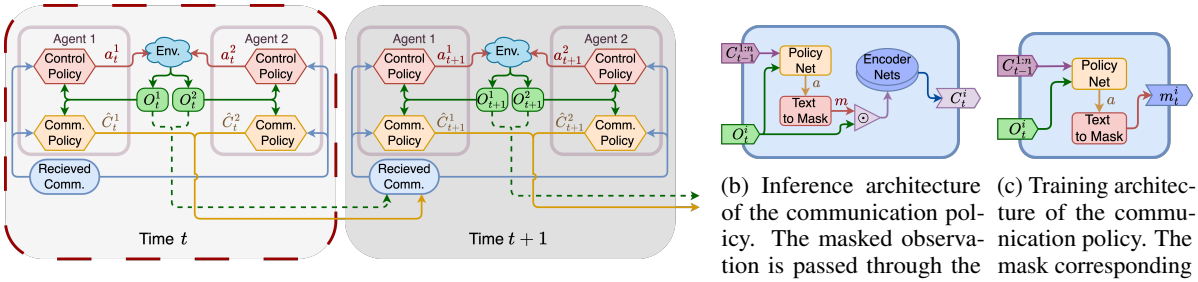
The nonstationary nature of cooperative MARL problems is gaining much attention recently. Many existing approaches embrace the centralized learning paradigm, which enables better performance. In Foerster et al. (2016), the authors present a method for learning between agents by propagating gradients through a communication channel. The paper Sukhbaatar et al. (2016) introduces a multi-agent communication model that uses a continuous vector to represent messages. Lowe et al. (2017) utilizes the policies of the other agents when choosing an action. Jaques et al. (2019) proposes a reward shaping to promote causal influence on other agents. Decentralized learning has also been explored in recent research. In Lin et al. (2021), agents learn an encoding for their observation. Lo et al. (2023) utilizes similar concepts for encoding the joint state of all agents. Eccles et al. (2019) promote communication-dependent policies via reward shaping. Jiang & Lu (2018) proposes an attention-based communication model that allows agents to selectively attend to incoming messages. Finally, Das et al. (2019) uses instance-specific communication.

Interpretability and the emergence of natural language have also been studied in the MARL setting. Lazaridou et al. (2016) uses a discrete set of symbols for communication, which can be translated by matching the emergent symbols with the corresponding human labels. Havrylov & Titov (2017) utilizes a similar discrete protocol, along with grounding, making the encoder assimilate to human language. Karten et al. (2023) proposes a three-phase learning process, in which agents first learn an emerging communication protocol, then the messages are pruned, and the final phase involves teaming up with human players. On the other hand, Kottur et al. (2017) shows that the emerging language is less likely to assimilate to natural language. Information gating has also been previously studied; the dropout mechanism (Srivastava et al., 2014) is utilized in most ML training procedures, improving learning and robustness. In the context of RL, Tomar et al. (2023) shows how to mask irrelevant features by adding dedicated layers and a loss component to aid the robustness and generalization of a policy.

Our work is built upon these previous approaches; by learning across agents (Foerster et al., 2016), using communication as a mapping of the observation (Lin et al., 2021), filtering out irrelevant information (Jiang & Lu, 2018), and shaping the reward (Eccles et al., 2019; Jaques et al., 2019). In addition, our work introduces a novel framework that combines human knowledge with RL and could be applied jointly with (almost) any other method for our setting, to enhance its performance and increase its interpretability.

3. Background and Problem Setting

Decentralized Partially Observable Markov Decision Process (DEC-POMDP), as introduced in Bernstein et al. (2002), describes a framework in which multiple agents need to apply a decentralized policy, based on each agent's observation independently. Here, the reward function and the transition kernel operate over the joint policy of all agents, and partial observability may extend to the central-



(a) Interconnections between agents/policies across time-steps. The dashed lines represent the dependency on the previous observation of all agents, which holds only during the training process. Here $\hat{C}$ denotes the output of the communication-policy, whether it is a mask, or the actual signal, depends on the phase; training or inference.

(b) Inference architecture of the communication policy. The masked observation is passed through the encoder networks of the communication-policy and control policy, then communicated.

(c) Training architecture of the communication policy. The mask corresponding with the chosen action is returned, instead of the communication signal.

Figure 2. Model architecture

ized setting. A popular framework to deal with the challenges arising from this approach is to use a communication channel where agents share information regarding their state and future actions to result in a better overall policy. The formal definition is given in the appendix.

Similarly to Kilinc & Montana (2018), we view the communication as a continuous mapping from one agent's observation to the transmitted message; this allows the agent to choose when to send a message, while the message itself is a continuous vector. In this case, the actual messages are expected to be relatively stationary and lossless in terms of information contained within the original observation. More formally, each agent i observes its own observation at time t $o_t^{(i)}$ and the broadcast communication channel c_t, where c_t is a concatenation of $\{\phi^j(o_{t-1}^j)\}_{j=1}^K)$, where ϕ^j is a mapping from the observation space to some vector field.

Importantly, even without explicit communication, agents could learn a well-coordinated behavior. This phenomenon can be attributed to two root causes; (1) required information to solve the problem is static (i.e., the hidden information that should be communicated is fixed through the episodes or highly correlated with each agent's observation), and (2) implicit communication. We design our environments (Section 5) to avoid these, by changing the hidden information at each episode and omitting direct connections between an agent's actions and the other agents' observations.

4. Framework

In this section, we present our proposed framework, Information Gating Multi-Agent (InGaMA), which aims to improve communication interpretability while maintaining the overall performance of the underlying algorithm. As a baseline algorithm, we use the well-known MAPPO (Yu et al., 2022) which is a multi-agent version of Proximal Policy Optimization (PPO) (Schulman et al., 2017) in which all agent instances share parameters. Although our proposed framework can be used with any underlying MARL algorithm, previous research suggests that the proximal policy search in the on-policy setting outperforms other approaches in many domains. Our method relies on two components that are elaborated in the following subsections; a text-to-mask model (Section 4.1) which we use to ground the information to a semantically meaningful text, and a dedicated architecture (Section 4.2) that allows us to use the text-to-mask model with a communication policy to gate the transmitted information and learn an interpretable communication. The end-to-end training procedure is elaborated in the appendix.

4.1. Text-to-Mask

When faced with a new task, humans often communicate with each other quite effectively. This happens thanks to an already existing form of communication protocol (i.e., language), an agreed terminology, and prior knowledge of the task that allows the players to communicate well. Generally, humans have an object-oriented perception, and a task's terminology usually refers to a textual description of objects and their states. This allows focus on a few relevant objects when communicating information, to avoid misunderstanding. While humans determine the relevancy of an object from the task's description, prior knowledge, and previous biases, artificial agents cannot directly interpret it, and it is unclear how to generally embed them. Similarly to humans, artificial agents can benefit from more focused communication (Tomar et al., 2023), but learning a human communication policy from demonstrations or including feedback in the learning process (RLHF) is likely to be unrealistic; the dimensions of MARL problems are relatively high, which would require either collecting a very large dataset of demonstrations or requesting human feedback over a huge amount of simulations.

To ground the observations, we construct a **text-to-mask** model, which maps from a textual description of an object to a mask $m \in \{0, 1\}$ in the same dimensions of the observation space. The text-to-mask model may use both the textual description and the current observation to calculate

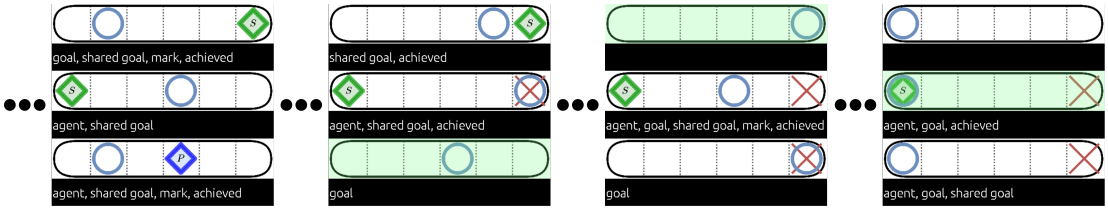


Figure 3. Sample trajectory from Coordinate, it shows the communication at each time-step. The time advances from left to right.

the mask; for example, in the case of image observations, a mask can be a segmentation of the desired object. In the general vector case, it is natural to view the observation as a collection of feature sets, each corresponding to a textual term that represents an object. Formally, we define a text-to-mask model $\mathcal{F} : \mathcal{T} \times O \rightarrow \{0,1\}^{|O|}$, where $\mathcal{T}$ is the textual input space, O is the observation space of each agent, and $\{0,1\}^{|O|}$ is a binary mask space with the same dimensionality as O. Although most environments have a description of the features (e.g., angle, velocity, position, etc.) that could be used to build $\mathcal{T}$, complex settings may require a set of tailored terms. For image observations, it is possible to use existing pre-trained object detection models or LLMs to extract the mask. In our implementation, we use a set of terms, each corresponding to a fixed set of elements of the observation space, which defines a mask.

4.2. Model Architecture

Each agent is implemented by two decoupled policies: the control policy π_θ^{cont} and the communication policy π_ϕ^{comm} (parameterized by θ, ϕ, respectively). While π^{cont} is defined over the original action space A_i, π^{comm} in our setting corresponds to choosing any subset of $\mathcal{T}_i$, which we parametrize with $|\mathcal{T}_i|$ i.i.d. Bernoulli random variables, each one indicates whether a term $\tau \in \mathcal{T}_i$ is included in the chosen subset. This subset is translated into a mask m using the text-to-mask model, and a dedicated encoder G_ψ (parametrized by ψ) calculates the transmitted communication as follows: $C^i = G_\psi(m \odot o)$, where $\odot$ stands for element-wise multiplication and $o \in \Omega_i$ is the current observation of the agent. To allow sparse communication to emerge, we add an additional dimension to the action space, which determines whether to send a message at all or mask out the entire message, resulting in a $|\mathcal{T}_i| + 1$-dimensional communication action space. $C^i \in \mathcal{C}$ is the transmitted communication of agent i, and contains information about the observation of the agent (to be received in the following time step). Since all agents send their own communication, policies π^{cont}, π^{comm} are exposed to current observation o and an additional vector of concatenated incoming communication signals from all agents $C^{1:n} \in \mathcal{C}^n$, where $C^{1:n} = (C^1, \ldots, C^n)$. As depicted in Figure 2b, the number of encoder instances depends on the number of policy networks and, in most cases, can be reduced to two; π^{cont}, π^{comm}. This means that the actual communication bandwidth is doubled, although it allows the incorporation of differentiable communication (Foerster et al., 2016).

We incorporate a similar idea to DIAL (Foerster et al., 2016) to propagate policy gradients from the loss of the receiving agent to the encoder of the transmitting agent. It is likely to aid policies in extracting information from the other agents' observations. $\mathcal{C}$ can be discrete or continuous with a small dimension, the only requirement is that the encoder be differentiable to allow gradient propagation. Propagating gradients through communication requires a change in architecture during (centralized) training and (decentralized) inference. For π^{comm}, its inference architecture is presented in Figure 2b, but for training purposes, we use the architecture described in Figure 2c. When training is performed, π^{comm} outputs the mask from the text-to-mask model, then it is stored along with the observation in the receiving agent's buffer. During the training process, each policy has an encoder which can be trained naturally;since we store its inputs, we can propagate gradients through the encoder. Then in inference, we switch the architecture for transmitting encoded messages only. Note that the encoder operates on the masked observations of each agent separately to allow decentralized execution. The high-level interconnections between agents and policies in consecutive time steps are presented in Figure 2a. Additionally, we use a centralized critic, which is used only during training.

5. Experiments

Our framework was tested in two simulated environments in which the optimal policy is largely based on communication. We compare our method against two baselines that utilize only a control policy: **No comm.**, a pure decentralized policy to show the added value of communication, and a **Dense comm.** setting, in which observations are unmasked when communicated, and expected to perform optimally under sufficient training steps. The dense setting can be seen as a variant of DIAL, with a different underlying RL algorithm and parameterization. The underlying algorithm for all baselines is MAPPO. In the following sections, we describe each environment and present the numerical results. We utilize a centralized critic, which obtains an unmasked central observation. The critic is shared by the control policy

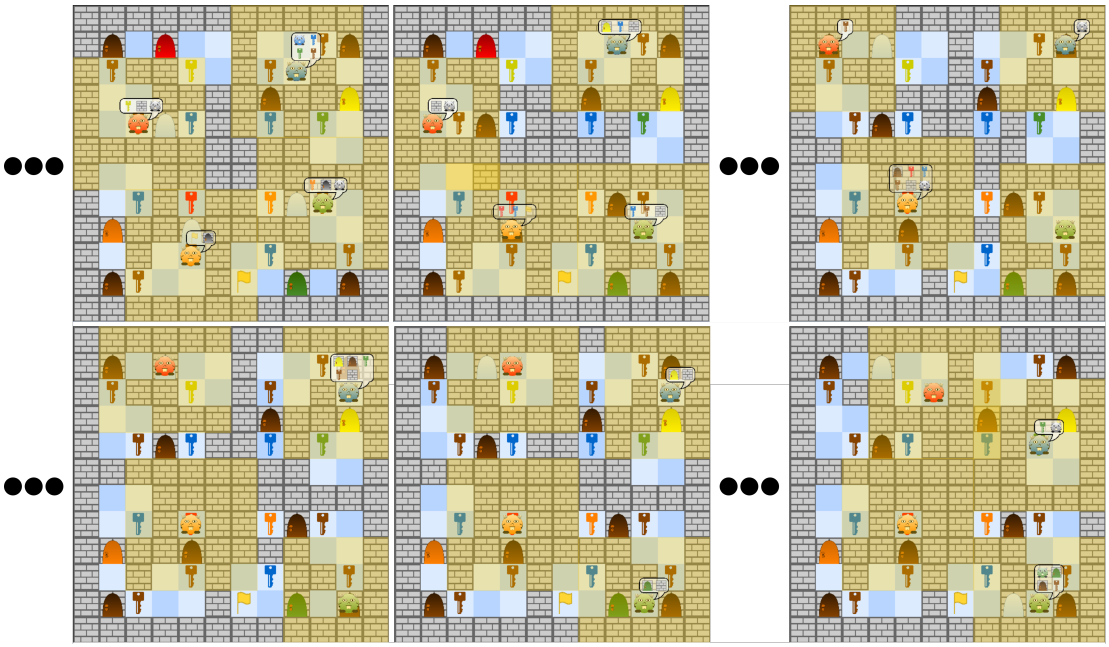


Figure 4. Sample trajectory from Multi-Maze. We visualize the textual communication via symbols, representing the broadcasted objects.

and the communication policy. We share our code[1] which is based on JaxMARL (Rutherford et al., 2024).

Coordinate: Coordinate is a simple game. As depicted in Figure 3, each agent has its own 1-dimensional grid world of size k (in our experiments $k = 6$) that contains a single goal, which is either **shared** or **private**. At each time step, agents can choose to *do nothing*, move one step *left* or *right* (moving towards the edge results in staying put), *claim goal*, or *mark* their location. Claiming a goal is possible when the agent and the goal are in the same location, using the *claim goal* action. A shared goal requires all other agents to mark the location of **this** goal in **their** grid world. After a goal is claimed, it disappears. Marking a new spot makes the previous mark disappear. The game ends when all the goals are claimed or the length of the episode is more than 40.

The reward is shared across agents, although in practice we define it individually, then use its mean as the shared reward signal. The individual reward for any state and action is always $r = -2$ unless: (1) Private goal claimed: the agent who claimed the goal receives $r = 10$ (for a single turn). (2) Shared goal claimed: the agent who claimed the goal receives $r = 100$ (for a single turn). (3) An agent who has already claimed a goal receives $r = 0$.To maximize the cumulative reward, the agents should cooperate to help each other claim the shared goal, while also collecting any private goals available. In our experiments, we use three agents. The definition of the observation space includes five feature sets, correspond with the following words: **agent**, **goal**, **shared goal**, **mark**, and **achieved**. Further description is available in the appendix.

Multi-Maze: This environment implements four 2D mazes, positioned side by side as depicted in Figure 4. The agents have a limited observation: instead of the entire maze, each observes the highlighted areas in Figure 4. At each turn, the agents choose their actions simultaneously, which correspond to moving up, down, left, or right. Agents cannot stand on walls or closed doors. A door is opened as long as an agent is in the same position with the key of the same color. The only exception is brown doors, which can be viewed as one-way paths, as each door can only be opened by the brown key next to it. The objective is to reach the flag (goal), which requires the agents to cooperate; Only one agent can reach the flag, but the other agents are required to help it by navigating to the correct keys. The one-way brown doors make the choices of the agents strict. Each maze has a dedicated "regular" color, which determines the color of the agent that plays it and the door to its locked room. At the beginning of each episode, a yellow door may replace one of the agents' doors, leaving one color of keys redundant, with no effect of stepping on it. In each episode, all agents are needed to solve the maze, forcing each one to choose which path to take, with no option to regret once crossed a brown door, making it a combinatorical challenge that requires information exchange.

The reward is $-n_d$, where n_d is the minimal number of doors that agents must pass before reaching the flag. The text-to-mask model uses five words that describe objects; **wall**, **goal**, **agent**, **key**, **door**, and words that describe colors;

[1] https://github.com/ingama-rl/InGaMa.git

Table 1. Results for both environments. We measure the success rate, along with the mean cummulative reward, for five random seeds. Each entry indicates the average and standard deviation.

	Coordinate		**Multi-Maze**	
	Mean cumulative reward	Success rate	Mean cumulative reward	Success rate
No comm.	19.9 ± 15	0.79 ± 0.21	-154 ± 17.6	0.53 ± 0.16
Dense comm.	29.2 ± 8.2	0.82 ± 0.1	$\mathbf{-42.5 \pm 1.1}$	$\mathbf{0.99 \pm 0.004}$
InGaMA **(ours)**	$\mathbf{33.9 \pm 14.9}$	$\mathbf{0.88 \pm 0.17}$	-53.7 ± 5.5	0.98 ± 0.014

red, **blue**, **green**, **orange**, **yellow**, and **brown**. Each object word corresponds with the features that represent an object, the color words enable additional masking for colored objects (agents, keys, and doors) by passing only the relevant colors. The game ends when the episode length is 48. The observation space is elaborated in the appendix.

Results: We present the results in Table 1, we measure the mean cumulative reward over 5 random seeds. We train each seed over 10^8 environment steps in Coordinate, and over $5 \cdot 10^8$ steps in Multi-Maze. Our method along with the 'Dense comm' learns a close-to-optimal behavior, surprisingly, in Coordinate, InGaMA performs better. Without communication, the 'No comm' method cannot reach the performance of the communicating methods.

6. Interpretability and Emergent Behavior

To illustrate the interpretability of the communication that emerges from our framework, we present examples of sample trajectories from our trained InGaMA agents. These examples highlight how InGaMA facilitates a human-understandable view of agent interactions. Full trajectories for both environments are available in the Appendix.

In the Coordinate environment, the trajectory (Figure 3) reveals interesting coordination dynamics. We observe a that the top agent to be prioritized in the early stages, even when the middle shared goal could be achieved faster. This behavior appears to stem from the top agent's strategy of selectively sharing its position ('agent'). By withholding this information, the top agent seems to influence the other agents to prioritize assisting it. This behavior can be interpreted as a learned mechanism to mitigate a "Buridan's ass" paradox, where agents face difficulty in choosing between equally attractive options. The selective sharing of information provides a subtle way to break symmetry. This insight potentially explains the stronger performance of InGaMA compared to the 'Dense comm' in this environment. Another recurring behavior is that once an agent has marked its location, it shares his mark position, which serves as a signal to other agents that the shared goal can be claimed.

In the Multi-Maze environment (Figure 4), an agent that observes the flag will share its location with the other agents, ensuring that all agents are aware of the target destination. Furthermore, the blue agent is observed to communicate the presence of the blue key. This is useful for the other agents to understand that the blue key is useless. When an agent encounters a closed door, it repeatedly communicates the presence of that door as long as it remains closed. This "nagging" behavior signals to the other agents that their assistance is required. Interestingly, the blue agent also communicates the presence of the yellow door as soon as it observes it, probably because yellow is not its "regular" door color. In addition to these task-specific communications, agents also periodically broadcast information about the walls or their own presence. This behavior can be interpreted as a way of sharing positional information, allowing agents to maintain awareness of each other's location.

7. Discussion and Future Work

This work introduces InGaMA, a novel framework that enhances interpretability and mitigates information overload in MARL. InGaMA achieves this by enabling agents to communicate the subject of their information via learned gating, allowing human observers to understand what information is exchanged. This approach contrasts with methods that focus on giving meaning to the message content itself. Our experiments demonstrate InGaMA's ability to learn effective, interpretable communication protocols. Analyzing emergent communication reveals how InGaMA provides valuable insights into agent decision making.

Future work can build upon similar notion to extend our work. Firstly, the current text-to-mask model could be extended. Integrating advanced pretrained models, such as YOLO (Jocher et al., 2023), would allow InGaMA to handle richer images-based observations. Secondly, exploring alternative learning paradigms for communication policy is promising. Although MAPPO effectively learns the gating mechanism, the use of LLM and in-context learning could offer benefits. LLMs might enable for faster adaptation by leveraging their ability to generalize.

In conclusion, InGaMA offers a promising approach to address the challenges of interpretability and information overload in multi-agent communication. Potential extensions of this work, including enhanced perception and LLM-based communication policies, can pave the way for more transparent and collaborative AI systems.

References

Bernstein, D. S., Givan, R., Immerman, N., and Zilberstein, S. The complexity of decentralized control of markov decision processes. *Mathematics of operations research*, 27(4):819–840, 2002.

Das, A., Gervet, T., Romoff, J., Batra, D., Parikh, D., Rabbat, M., and Pineau, J. Tarmac: Targeted multi-agent communication. In *International Conference on machine learning*, pp. 1538–1546. PMLR, 2019.

Eccles, T., Bachrach, Y., Lever, G., Lazaridou, A., and Graepel, T. Biases for emergent communication in multi-agent reinforcement learning. *Advances in neural information processing systems*, 32, 2019.

Foerster, J., Assael, I. A., De Freitas, N., and Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Havrylov, S. and Titov, I. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. *Advances in neural information processing systems*, 30, 2017.

Jaques, N., Lazaridou, A., Hughes, E., Gulcehre, C., Ortega, P., Strouse, D., Leibo, J. Z., and De Freitas, N. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International conference on machine learning*, pp. 3040–3049. PMLR, 2019.

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

Jocher, G., Chaurasia, A., and Qiu, J. Ultralytics YOLO, January 2023. URL https://github.com/ultralytics/ultralytics.

Karten, S., Tucker, M., Li, H., Kailas, S., Lewis, M., and Sycara, K. Interpretable learned emergent communication for human-agent teams. *IEEE Transactions on Cognitive and Developmental Systems*, 2023.

Kilinc, O. and Montana, G. Multi-agent deep reinforcement learning with extremely noisy observations. *arXiv preprint arXiv:1812.00922*, 2018.

Kottur, S., Moura, J. M., Lee, S., and Batra, D. Natural language does not emerge'naturally'in multi-agent dialog. *arXiv preprint arXiv:1706.08502*, 2017.

Lazaridou, A., Peysakhovich, A., and Baroni, M. Multi-agent cooperation and the emergence of (natural) language. *arXiv preprint arXiv:1612.07182*, 2016.

Li, H., Nourkhiz Mahjoub, H., Chalaki, B., Tadiparthi, V., Lee, K., Moradi Pari, E., Lewis, C., and Sycara, K. Language grounded multi-agent reinforcement learning with human-interpretable communication. *Advances in Neural Information Processing Systems*, 37:87908–87933, 2024.

Lin, T., Huh, J., Stauffer, C., Lim, S. N., and Isola, P. Learning to ground multi-agent communication with autoencoders. *Advances in Neural Information Processing Systems*, 34:15230–15242, 2021.

Lo, Y. L., Sengupta, B., Foerster, J. N., and Noukhovitch, M. Learning multi-agent communication with contrastive learning. In *The Twelfth International Conference on Learning Representations*, 2023.

Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Rutherford, A., Ellis, B., Gallici, M., Cook, J., Lupu, A., Ingvarsson Juto, G., Willi, T., Hammond, R., Khan, A., Schroeder de Witt, C., et al. Jaxmarl: Multi-agent rl environments and algorithms in jax. *Advances in Neural Information Processing Systems*, 37:50925–50951, 2024.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Sukhbaatar, S., Fergus, R., et al. Learning multiagent communication with backpropagation. *Advances in neural information processing systems*, 29, 2016.

Tomar, M., Islam, R., Taylor, M., Levine, S., and Bachman, P. Ignorance is bliss: Robust control via information gating. *Advances in Neural Information Processing Systems*, 36:38624–38641, 2023.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems*, 35:24611–24624, 2022.

A. Problem Setting

Consider a standard DEC-POMDP: $\big(\mathcal{I}, \mathcal{S}, \{A_i\}_{i\in\mathcal{I}}, P, R, \{\Omega_i\}_{i\in\mathcal{I}}, \{O_i\}\big)$, where $\mathcal{I}$ is the set of agents, $\mathcal{S}$ is the state space, A_i is the action space of agent $i \in \mathcal{I}$, P is the global transition dynamics, R is the global reward function, Ω_i, O_i are (respectively) the observation space and the conditional observation probabilities of agent $i \in \mathcal{I}$. Furthermore, let $n = |\mathcal{I}|$ be the number of agents in the environment. Where a few formulations exist, adding communication to this setting can be reduced to an equivalent DEC-POMDP with increased observation spaces (due to the communication signals) and additional action spaces (for the communication protocol). In our setting, communication is allowed under the following realistic condition: At time step t where an agent observes o_t (the current observation) and $C^i_{t-1}, i = 1, \ldots, n$, the communication received (from all agents), it needs to choose both a_t, the action for the environment and C_t the communication signal that would be available to the other agents at the next time step $t + 1$. That means that C_t could only depend on the information the agent has in a time step t, hence the receiver obtains the information in delay of a single time step.

B. InGaMA Training

The main difficulty that accompanies communication-dependent control directly comes from the inconsistency of the communication policy throughout the training phase; it changes between iterations and is initially random. We found that the best performance is achieved by training the communication policy and the control policy interchangeably. Although our framework can potentially be combined with any RL algorithm, Proximal Policy Optimization (PPO) (Schulman et al., 2017) is particularly well suited for this task as it prevents the policies from deviating too drastically during training, helping to cope with the non-stationarity introduced by the decoupled nature of the communication and control policies. Specifically, we utilize MAPPO (Yu et al., 2022), a multi-agent version of PPO in which all agent instances share parameters, so we always have two policies to train: π^{comm}, π^{cont}. We train them sequentially and batch by batch; each batch of trajectories is only stored in the rollout buffer of one of the policies; then, after the policy is updated, we sample another batch for the other policy. In this way, the sampled trajectories remain "on policy", which results in a better estimation of the policy gradient. Both policies share the same reward.

C. Implementation details

To ensure a fair comparison in all baselines, we maintain a consistent architecture for the critic and the control policy (actor), with the primary distinction being the handling of communication signals. Specifically, InGaMA introduces an additional actor dedicated to the communication policy. For the critic, we employ a transformer-based architecture, leveraging its capacity to capture long-range dependencies within the joint state. In contrast, actors, responsible for individual agent actions, utilize a recurrent neural network (RNN) architecture. This choice is motivated by the RNN's suitability for processing sequential observation data and incorporating temporal dynamics.

Each observation is structured into three distinct components to provide a comprehensive representation of the environment.

1. Decentralized observation: The agent's local view of the environment.
2. Centralized observation: The global information about the environment, accessible to the critic during training.
3. Raw communication: The communication signals exchanged between agents. It is termed "raw" because it is a decentralized observation of the previous timestep and undergoes further processing before being used.

The critic's input consists of the centralized observation and the raw communication from all agents. This allows the critic to assess the overall state of the system and guide the agents towards coordinated behavior. The actor's input comprises the decentralized observation of the agent and the raw communication, enabling each agent to make decisions based on its local perception and the messages received from others.

The key difference between the baselines is the raw communication: For No Comm, the raw communication input is a vector of zeros, effectively disabling communication between agents. For Dense Comm, the raw communication consists of the previous observation of each agent, allowing for a direct exchange of information. Finally, InGaMA employs masked previous observations as raw communication. This masking is determined by the communication policy and the text-to-mask model. In addition, raw communication includes a vector that indicates which objects were chosen to be transmitted. For the other baselines, this vector is either all zeros (No Comm) or all ones (Dense Comm). To prevent the communication

policy from exploiting the redundant elements of the object selection vector, we mask out objects that are not present in the observation.

The raw communication is processed by a dedicated neural network module, CommunicationNet. This module applies the same mapping function to each agent's raw communication separately, generating individual message vectors. These individual message vectors are then concatenated to form a single vector of processed communication, representing the collective messages of all agents. CommunicationNet is learned within the receiving agent during training, enabling gradient propagation. At inference time, the learned CommunicationNet can be applied within the sending agent to generate communication signals. In our implementation, the masking of objects within the raw communication is performed within the environment. This is where the text-to-mask model resides, allowing for a simple environment API, that receives actions from both the communication and control policies.

For more details, including specific hyperparameters for each experiment, we refer the reader to our code files and documentation available here.

C.1. Environments

The observation space of Coordinate is composed of the following feature sets:

1. **agent**: The location of the agent on its own grid, a one-hot k-dimensional vector.
2. **goal**: One hot k-dimensional array that represents the location of the private goal.
3. **shared goal**: One hot k-dimensional array that represents the location of the shared goal.
4. **mark**: One-hot k-dimensional array that represents the location of the mark.
5. **achieved**: Boolean, *true* if the goal is already claimed, otherwise *false*.

The observation space of Multi-Maze is composed of the following:

1. Six 10×10 matrices, each one corresponding to a door of a specific color.
2. Six 10×10 matrices, each corresponding to a key of a specific color.
3. Four 10×10 matrices, each corresponding to a different agent.
4. A 10×10 matrix representing the walls.
5. A 10×10 matrix representing the flag.
6. Three 10×10 matrices that correspond to any door, any key, and any agent.

For each matrix, the element at the location of its corresponding objects on the grid is set to 1 and the rest are 0s. These matrices are masked according to the agent's location.

C.2. Computational Resources

The experiments in the Multi-Maze environment were performed on Nvidia GeForce RTX 4090, where each seed takes about 10-12 hours. For the Coordinate environment, we used a Nvidia GeForce RTX 2090ti, where each seed takes four to five hours. Both experiments were performed on machines with 64GB RAM.

C.3. Existing Assets

As mentioned in the main body, we build our code on Rutherford et al. (2024), under the APACHE 2.0 license.

D. Emergent Communication

In this section we present complete trajectories and describe the interesting communication transactions among them. While there are redundant messages, it is possible to track critical information, and provide justifications for its necessity.

D.1. Coordinate

We present three whole trajectories of trained InGaMA agents, sampled from the Coordinate environment. While there are much simpler cases (e.g., only single shared goal or no shared goals), we show cases with two shared goals, as they are more interesting. Figure 5 is the full trajectory of Figure 3. In this environment, each agent transmits messages approximately 80% of the timesteps.

Figure 5. The top agent shares its location after the middle agent is closer to the top shared goal, making the group prioritize the top goal. Note that the communication reflects the **previous** position and affects the **following** action.

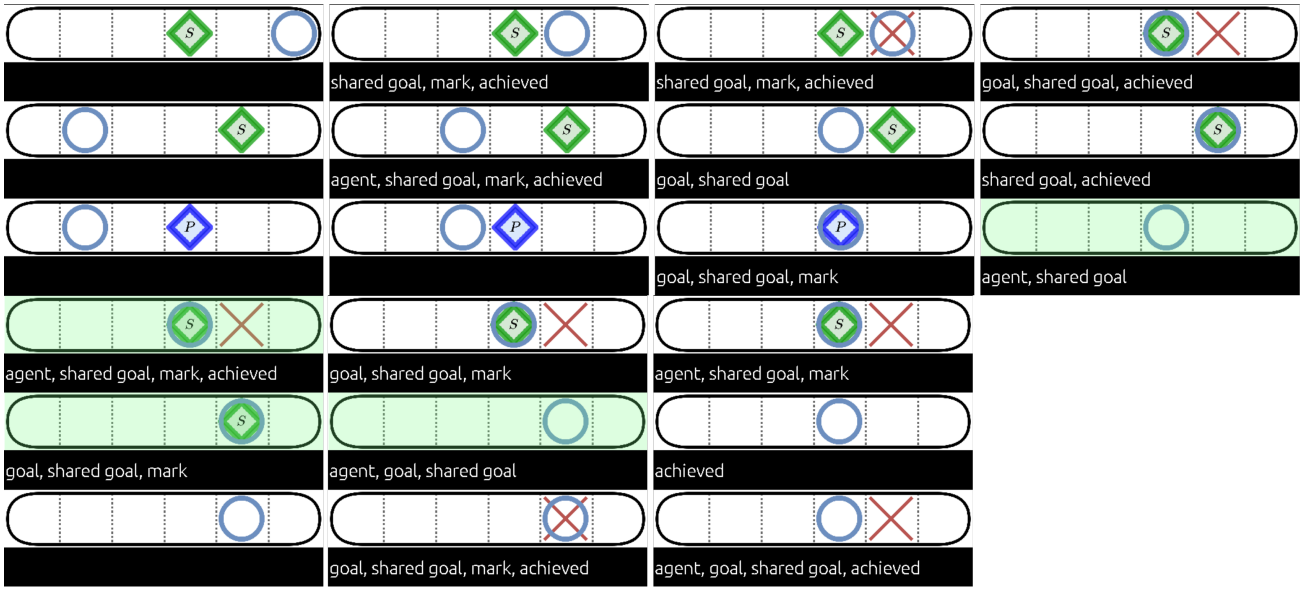


Figure 6. For different training seed we observe different behavior. Here agents are headed towards their own goal first, then help the others. The middle agent is prioritized, now due to the fact it shared its position first.

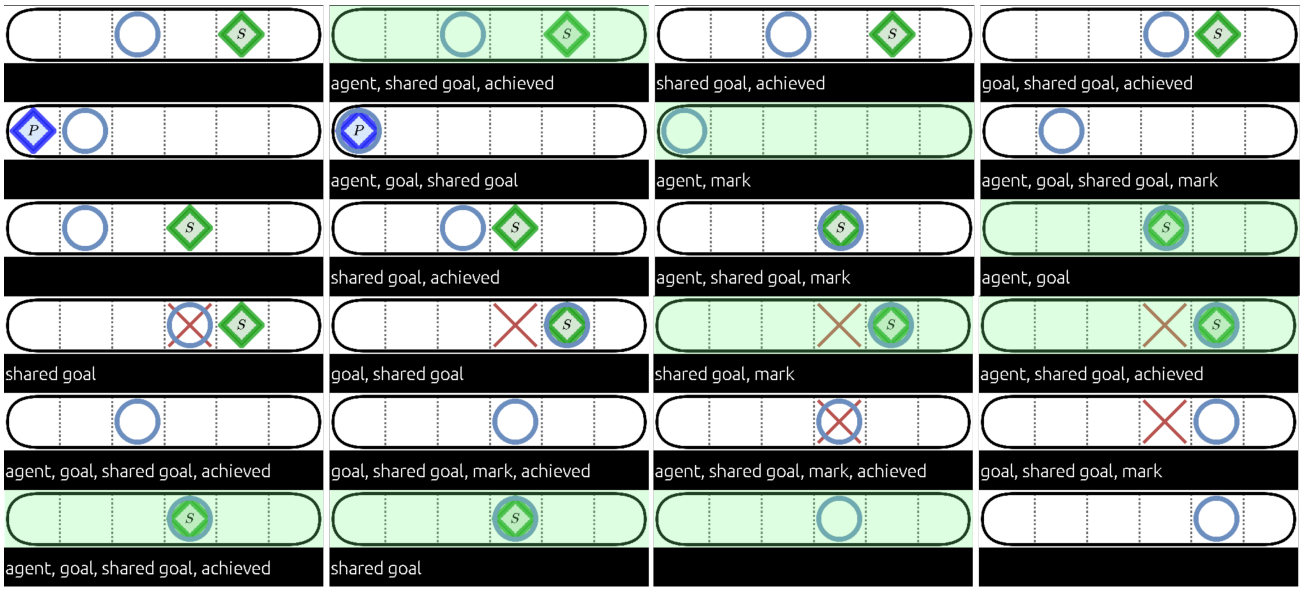


Figure 7. Here, the bottom agent "insisting" on staying put until it claims the goal, and stops communication once it did, which causes the middle agent to move its' mark to aid the top agent.

D.2. Multi-Maze

We present three representing trajectories from the Multi-Maze environment, sampled by InGaMA trained agents, where Figure 8 is the "complete" trajectory of Figure 4. Note that while the length of every episode is 48, we show only part of the path until the goal is reached. Here, each agent transmits messages approximately 60% of the timesteps. One general behavior we observed is that all agents move to a position in which they can observe which keys they can attain. This is their method to learn about the scenario they are facing, then they communicate the important objects to other agents. While agents do communicate static objects that seem not important (e.g., walls, brown keys and doors, etc.), we interpret it as a method of sharing their position with the others.

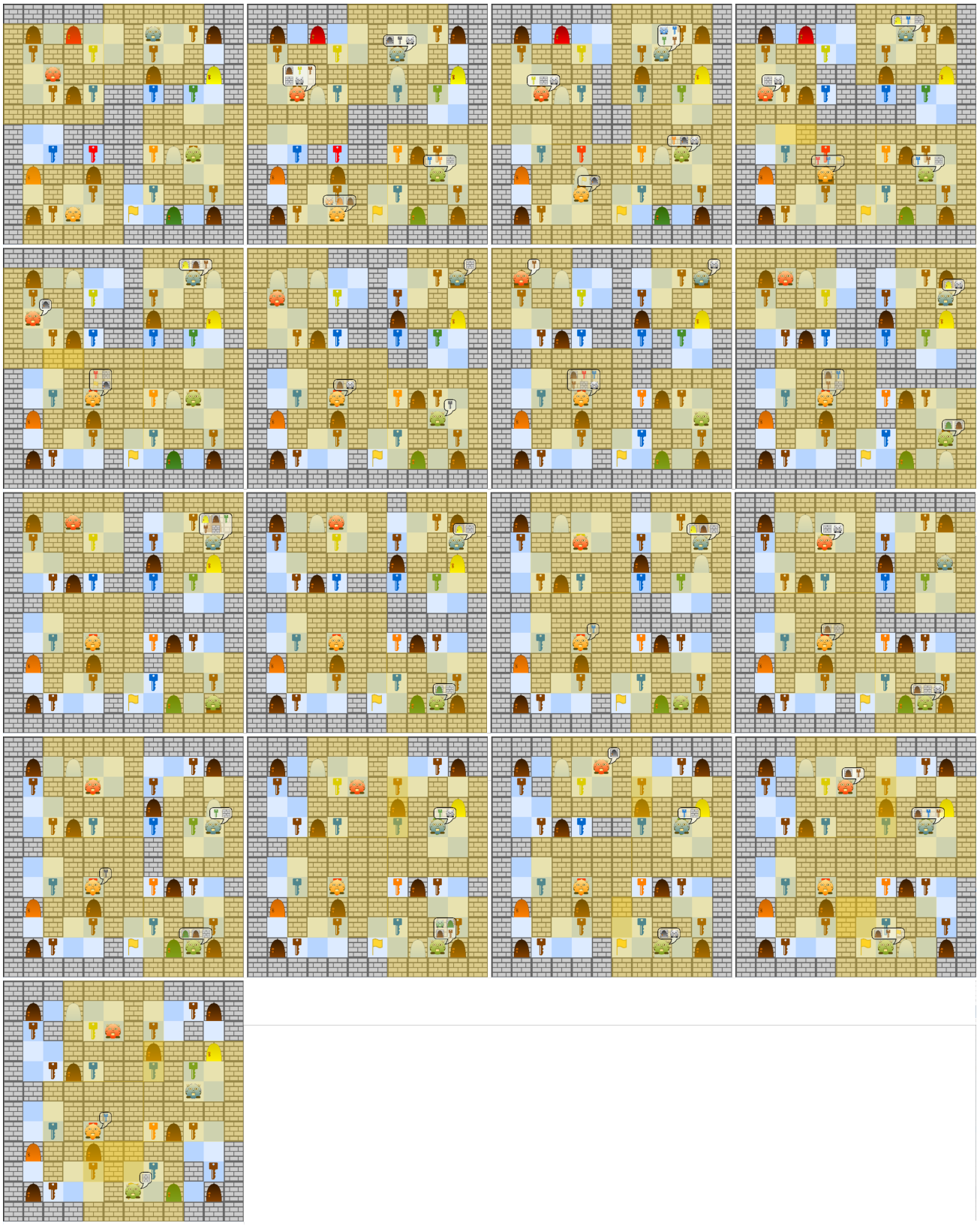


Figure 8. The orange agent communicates the flag position once he observes it. Once the orange agent stands on the red key, it commnicates it, promoting the red agent to go to the room behind the red door. Both the blue and the green agent communicate the doors that block them from reaching their destination.

Figure 9. Here, the agents identify that the green key is redundant, hence no need to go for them, and they are able to solve the game without communicating the flag's location. The orange agent is the only one with no redundant key, but the orange key is locked behind the orange door, making it inaccesible.

Figure 10. In this episodem, the yellow key is redundant, although the goal should be communicated to assure it is not in the same room of a yellow key before passing a one-way brown door. Similarly to other trajectories, agents communicate closed doors they intend to pass.