Introducing Multi-Agent Frozen Lake

Israel Yago Pereira*
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, RS 90040-060
israel.yago@ufrgs.br

Jonathan Choy Rivera†
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, RS 90040-060
jonathan.rivera@ufrgs.br

Abstract

We present a multi-agent extension of Frozen Lake introducing partial observability and discrete communication to investigate cooperative reinforcement learning challenges. Benchmarking with PPO and parameter sharing across seven configurations over 10,000 iterations, we uncover a counterintuitive observability paradox: agents with restricted 5×5 local views achieve mean rewards of 60, while agents with complete state information fail entirely at -0.6. This challenges the conventional MARL assumption that full observability facilitates coordination. Additional findings reveal that LSTM memory provides minimal benefit in deterministic settings, while both discrete communication and stochastic transitions impede learning—communication degrades performance under partial observability, and slippery dynamics prevent any learning regardless of configuration. These results establish our environment as a challenging benchmark exposing fundamental limitations in standard cooperative learning algorithms and motivate investigation into alternative approaches including value decomposition methods and mechanistic analysis of why restricted perception paradoxically enables effective coordination.

1 Introduction

1.1 Motivation

Reinforcement Learning (RL) has achieved substantial progress in complex sequential decision-making problems [21], yet many practical settings require multiple agents to coordinate under uncertainty and partial observability, often formalized through Dec-POMDPs [14]. Classical environments such as Frozen Lake, available in the OpenAI Gym suite [1], provide a simple testbed for navigation and stochastic transitions, but they are traditionally limited to single-agent, fully observable scenarios. To explore cooperative behavior, communication, and coordination challenges in a controlled environment, we extend Frozen Lake into a multi-agent framework with additional complexities such as partial observability and limited communication, as commonly studied in multi-agent RL settings [8]. While multi-agent coordination in grid-world environments has been studied extensively [22], we seek to establish whether our extended environment presents meaningful learning challenges for standard MARL algorithms. This project therefore serves as an exploratory study using PPO with parameter sharing, a widely adopted cooperative learning method, to benchmark the environment's difficulty and identify fundamental coordination challenges that may motivate future algorithmic development.

*https://github.com/israelyago/
†https://github.com/jchoy8890

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

1.2 Contributions

This project introduces a modified multi-agent version of the Frozen Lake environment designed to highlight core challenges in cooperative RL. The key contributions are:

- A multi-agent extension of Frozen Lake with shared objectives and individual dynamics.
- Partial observability that restricts each agent's perception of the environment.
- A discrete communication channel enabling agents to exchange simple messages.
- An empirical evaluation using PPO with parameter sharing [18], revealing significant learning challenges that motivate future algorithmic improvements.
- A systematic ablation study examining how observability, communication, stochasticity, and memory affect training outcomes in cooperative settings.

1.3 Overview of Approach

We train multiple agents using Proximal Policy Optimization (PPO) [18] with a shared policy across agents, implemented through Ray RLlib [11]. The agents operate under partial observations and may optionally utilize simple communication signals to coordinate their behavior, following common approaches in MARL communication studies [5]. Training focuses on maximizing episodic reward, combining individual and shared components to encourage both survival and overall task performance. We conduct systematic ablations over seven environment configurations, varying observability (full versus partial 5×5 views), memory mechanisms (feed-forward versus LSTM-based policies), communication (enabled versus disabled), and transition dynamics (deterministic versus slippery). Each configuration is trained for 10,000 iterations across 5 random seeds to ensure reliable performance estimates. This experimental design allows us to isolate which environmental factors most significantly impact cooperative learning in navigation tasks.

Our results reveal unexpected learning dynamics across configurations, with performance varying dramatically based on observability and transition stochasticity. These findings provide insights into the coordination challenges present in multi-agent navigation and suggest directions for future algorithmic development, including value decomposition methods [16, 20] and exploration strategies tailored to cooperative settings.

Code Availability. All source code for the multi-agent Frozen Lake environment, training pipeline, and experiment configurations is publicly available at: `https://github.com/israelyago/MA-FrozenLake`

2 Related Work

2.1 Multi-Agent Reinforcement Learning

Multi-Agent Reinforcement Learning (MARL) extends single-agent RL to settings where multiple agents interact within a shared environment, introducing non-stationarity as each agent's policy evolves during training [8]. This non-stationarity poses significant challenges for convergence and credit assignment. Early cooperative MARL approaches demonstrated emergent behaviors under shared objectives [22], while more recent methods employ parameter sharing across agents to reduce variance and improve sample efficiency in symmetric environments [9]. Centralized training with decentralized execution has become a standard paradigm for stabilizing learning in cooperative settings [12], enabling agents to leverage global information during training while executing independently at test time. Our work adopts parameter sharing with PPO [18] as a baseline approach to evaluate whether standard policy gradient methods can handle the coordination demands of our extended environment.

2.2 Communication in MARL

Effective coordination in partially observable multi-agent systems often requires explicit communication channels. Foerster et al. pioneered differentiable communication architectures that allow agents

to learn communication protocols end-to-end through backpropagation [4]. Alternative approaches include discrete communication with reinforcement learning [19] and emergent communication through shared objectives [5]. These studies demonstrate that learned communication can significantly improve performance in cooperative tasks requiring information sharing. We incorporate a simple discrete communication channel in our environment to investigate whether symbolic message passing aids coordination in navigation tasks, though our preliminary results suggest that communication alone may be insufficient without appropriate learning mechanisms.

2.3 Grid-World Benchmarks for MARL

Grid-world environments provide interpretable testbeds for studying multi-agent coordination due to their discrete action spaces and controllable complexity. The AI Safety Gridworlds suite introduced variants focused on safety constraints and stochastic transitions [10], while other efforts have developed multi-agent grid-world benchmarks specifically for coordination research [3]. The classic Frozen Lake environment from OpenAI Gym [1] exemplifies navigation under stochasticity with sparse rewards, though it was originally designed for single-agent settings. Our environment builds directly on Frozen Lake by introducing multiple agents with partial observability and optional communication, creating a more complex coordination problem while maintaining the interpretability and simplicity that make grid-worlds valuable for algorithmic analysis.

2.4 Value Decomposition Methods

Recent advances in cooperative MARL have focused on decomposing team value functions into individual agent contributions while maintaining representational capacity. Value-Decomposition Networks (VDN) [20] and QMIX [16] learn factorized representations that enable decentralized execution while leveraging centralized value estimates during training. These methods have shown strong performance on cooperative benchmarks by addressing credit assignment more effectively than independent learners. While we do not implement value decomposition in this work, such methods represent a promising direction for future investigation given the credit assignment challenges inherent in cooperative navigation tasks with varying observability and reward structures.

3 Environment

We implement a multi-agent extension of the classic Frozen Lake environment using the PettingZoo framework [23], which provides standardized APIs for multi-agent RL research. Our environment uses the ParallelEnv interface, where all agents act simultaneously at each timestep, reflecting realistic decentralized control scenarios where coordination must occur without turn-taking.

3.1 Grid Layout and Dynamics

The environment consists of a 7×7 grid containing safe tiles (frozen surface), holes, and a designated goal location. Agents navigate this discrete space using four directional actions (up, down, left, right) and they do not collide with each other. We support both deterministic movement, where actions execute as intended, and stochastic "slippery" transitions, where each action has a probability of instead moving the agent in a perpendicular direction. This stochasticity models realistic uncertainty in navigation and increases the difficulty of coordinated planning. Importantly, agents do not collide, as multiple agents may occupy the same tile simultaneously, eliminating physical interference as a coordination mechanism.

3.2 Multi-Agent Coordination

Multiple agents are initialized at distinct random starting positions (no two agents share the same starting tile), and must navigate to a shared goal location that is also randomly placed at the beginning of each episode. All agents act in parallel at each timestep, creating simultaneous decision-making without turn structure. An individual agent terminates when it falls into a hole, but surviving agents continue to act. The episode ends when all remaining agents reach the goal position or when a step-based timeout is reached. This termination structure, combined with randomized initial configurations,

requires agents to develop general coordination strategies rather than memorizing fixed trajectories, while implicitly coordinating their arrival at the goal and avoiding premature termination from holes.

3.3 Partial Observability

In partial observability configurations, each agent receives an egocentric 5×5 local view centered on its current position rather than the full 7×7 grid state. This limited perception window restricts agents' awareness of distant teammates, holes, and the goal location, creating a partially observable Markov decision process (POMDP) [14]. Agents must therefore navigate using incomplete information, making coordination substantially more challenging. In fully observable configurations, agents receive the complete grid state for comparison. Figure 1 illustrates the difference between the global state and individual agent observations, showing how partial observability restricts each agent's perception to only nearby tiles within their local viewing radius.

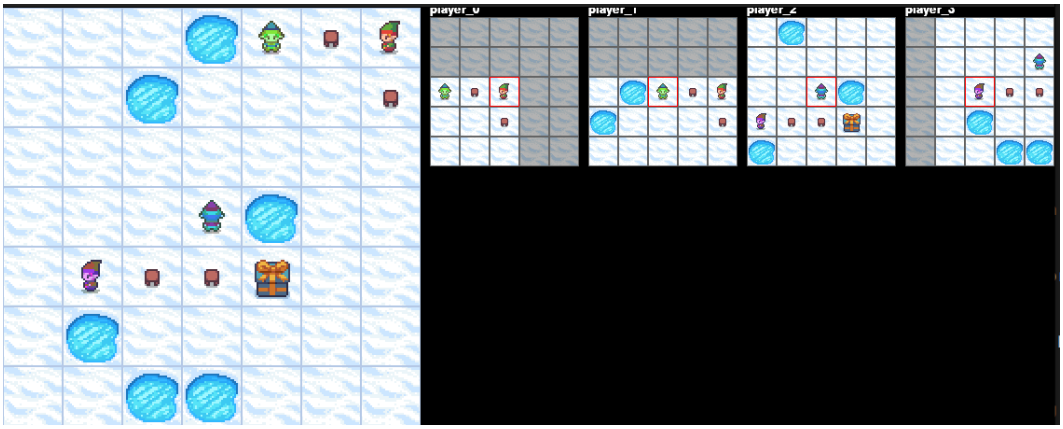

Figure 1: Comparison of global environment state (left) and individual agent observations (right panels) in partial observability mode. The global state shows the full 7×7 grid with three agents at their starting positions (indicated by colored chairs), holes (dark tiles), and the goal location. Each agent observes only a 5×5 egocentric window centered on its position, limiting awareness of distant teammates and environment features. Agent colors aid visual disambiguation.

3.4 Communication Channel

We equip agents with a discrete communication mechanism allowing each agent to broadcast one of five distinct messages to all teammates at each timestep. These messages are symbolic and predefined, meaning agents do not learn a continuous communication protocol but rather select from a fixed vocabulary. Each agent observes the messages broadcast by all other agents in the previous timestep, incorporated directly into their observation vector. When communication is disabled in ablation experiments, all agents receive a default message value of 0, effectively removing information exchange. This simple communication structure tests whether explicit signaling aids coordination in navigation tasks.

3.5 Reward Structure

We design a hybrid reward function that balances individual and collective incentives. At each timestep, all agents receive a small negative step penalty ($r_{\text{step}} = -0.001$) to encourage efficient navigation. Individual agents receive a positive reward ($r_{\text{goal}} = +1.0$) for reaching and remaining at the goal position. Additionally, when all surviving agents simultaneously occupy the goal, a shared team reward ($r_{\text{team}} = +10.0$) is issued to all surviving agents and the episode terminates successfully. Agents that fall into holes receive a substantial negative reward ($r_{\text{hole}} = -1.0$) and terminate individually. Finally, if the episode reaches the step-based timeout, all surviving agents receive a timeout penalty ($r_{\text{timeout}} = -0.1$). This reward structure encourages both personal survival and collective coordination: agents must reach the goal themselves while ensuring teammates also succeed for maximum cumulative reward.

4 Methods

4.1 Problem Formulation

We formalize our multi-agent navigation task as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [14]. A Dec-POMDP is defined by the tuple $\langle \mathcal{I}, \mathcal{S}, \{\mathcal{A}i\}_{i\in\mathcal{A}}, T, R, \{Oi\}_{i\in\mathcal{A}}, \gamma \rangle$, where:

- $\mathcal{I} = \{1, \ldots, n\}$ is the set of n agents
- $\mathcal{S}$ is the global state space representing grid configurations
- $\mathcal{A}_i$ is the action space for agent i (four directional movements + no movement)
- $\mathcal{O}_i$ is the observation space for agent i (local grid view and teammate messages + teammate relative positions to self)
- $T : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \to \Delta(\mathcal{S})$ is the joint transition function
- $R : S \times A_1 \times \cdots \times A_n \to \mathbb{R}$ is the shared reward function
- $O_i : \mathcal{S} \times \mathcal{A}_i \to \Delta(\mathcal{O}_i)$ is the observation function for agent i
- $\gamma \in [0, 1)$ is the discount factor

In our partial observability configurations, each agent i observes only a 5×5 egocentric view $o_i^t \in \mathcal{O}_i$ rather than the full state $s^t \in \mathcal{S}$, along with the discrete messages broadcast by teammates and their relative positions if in the agent i field of view.

The agents must learn a joint policy $\pi = (\pi_1, \ldots, \pi_n)$ that maximizes expected cumulative discounted reward $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^t, a_1^t, \ldots, a_n^t)\right]$, where each agent's policy $\pi_i : \mathcal{O}_i \to \Delta(\mathcal{A}_i)$ maps observations to action distributions.

4.2 PPO with Parameter Sharing

We train agents using Proximal Policy Optimization (PPO) [18], implemented through Ray RLlib [11, 13]. PPO optimizes the clipped surrogate objective:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right] \tag{1}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{\text{old}}}(a_t|o_t)}$ is the probability ratio, $\hat{A}_t$ is the generalized advantage estimate [17], and ϵ controls the clipping range.

All agents share a single set of policy and value network parameters θ, meaning $\pi_1 = \pi_2 = \cdots = \pi_n = \pi_\theta$. This parameter sharing approach reduces variance in cooperative settings by pooling experiences across all agents and is particularly effective in symmetric environments where agents have identical observation and action spaces. Each agent's experience tuple $\langle o_i^t, a_i^t, r_i^t, o_i^{t+1} \rangle$ contributes to gradient estimates for the shared policy.

4.3 Network Architecture

We employ two network architectures depending on the experimental configuration. The base architecture consists of a two-layer fully connected network with hidden dimensions $[64, 64]$, followed by separate policy and value heads. For configurations requiring temporal memory, we augment this with an LSTM layer [6] having cell size 64 and maximum sequence length 32. The LSTM-based architecture processes observation sequences to maintain a hidden state that captures temporal dependencies, enabling implicit belief-state tracking in partially observable settings. The network takes as input the concatenated observation vector including grid features, teammate messages and their relative positions when in view, and outputs action logits and a state value estimate.

4.4 Training Configuration

We train four agents for 10,000 iterations using the following hyperparameters: learning rate $\alpha = 10^{-5}$, training batch size of 512 samples per learner, and 2 epochs of minibatch SGD per iteration.

We use Ray's parallel rollout workers to collect experience across multiple environment instances simultaneously, improving sample efficiency and gradient stability. Each experimental configuration is trained with 5 random seeds to account for stochasticity in initialization and environment dynamics. For configurations with slippery transitions enabled, we set the slip probability to ensure an expected success rate of approximately $1/3$ per intended action, substantially increasing navigation difficulty.

4.5 Experimental Configurations

We evaluate seven distinct environment configurations to isolate the effects of observability, memory, communication, and stochasticity:

- **Baseline**: Full observability, LSTM enabled, no communication, deterministic transitions
- **Partial obs. without LSTM**: Partial observability (5×5 view), no LSTM, no communication, deterministic transitions
- **Partial obs.**: Partial observability, LSTM enabled, no communication, deterministic transitions
- **Slippery**: Full observability, LSTM enabled, no communication, slippery transitions
- **With comm**: Full observability, LSTM enabled, communication enabled, deterministic transitions
- **Comm partial obs.**: Partial observability, LSTM enabled, communication enabled, deterministic transitions
- **Slip comm partial**: Partial observability, LSTM enabled, communication enabled, slippery transitions

These configurations enable systematic ablation of environment features and architectural components to identify which factors most significantly impact learning performance.

5 Experiments

5.1 Evaluation Protocol

We evaluate our multi-agent Frozen Lake environment across seven distinct configurations to systematically ablate the effects of observability, memory mechanisms, communication, and stochastic dynamics. Each configuration is trained for 10,000 iterations with 5 random seeds to account for variability in initialization and environment stochasticity. This experimental design allows us to isolate which environmental factors most significantly impact cooperative learning performance [7].

5.2 Performance Metrics

We measure training progress using mean episodic reward averaged across all agents and episodes within each training iteration. This metric directly captures both individual success (avoiding holes, reaching the goal) and cooperative performance (all agents coordinating to reach the goal simultaneously for the team reward). For each configuration, we report the mean reward trajectory across 5 seeds with standard error bands to visualize learning dynamics and variance. Additionally, we compute the final average reward over the last 100 training iterations to compare converged performance across configurations.

6 Results

6.1 Learning Dynamics

Figure 2 presents learning curves for all seven experimental configurations over 10,000 training iterations. The curves reveal a stark dichotomy in learning outcomes: configurations with partial observability (partial obs., partial obs. without LSTM, and comm. partial obs.) demonstrate clear learning progress with steadily increasing rewards, while configurations with full observability (baseline, with comm., slippery) converge to near-zero rewards and show no improvement throughout

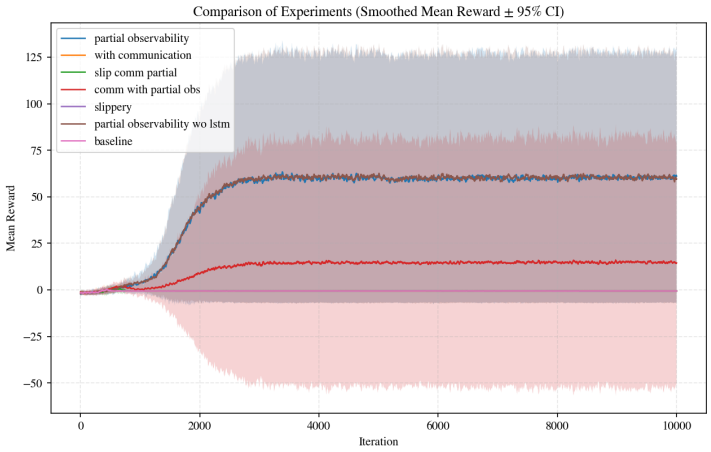

Figure 2: Learning curves showing mean episodic reward over 10,000 training iterations for all experimental configurations. Trajectories show mean across 5 random seeds with shaded regions indicating standard error. Curves are smoothed using a rolling window of 20 iterations for visual clarity. Partial observability configurations exhibit clear learning progress while full observability configurations fail to improve beyond initial performance.

training. The slip comm. partial configuration, which combines partial observability with slippery transitions and communication, also fails to learn, suggesting that stochastic dynamics dominate the learning difficulty when present.

6.2 Final Performance Comparison

Table 1 and Figure 3 summarize converged performance by averaging rewards over the final 100 training iterations. The partial observability configuration achieves the highest mean reward of 60.82 ± 0.19, closely followed by partial obs. without LSTM at 59.81 ± 0.21. The comm. partial obs. configuration achieves intermediate performance at 14.72 ± 0.10, suggesting that communication provides some benefit under partial observability but does not reach the same level as non-communicating variants. All full observability configurations (baseline, with comm., slippery, slip comm. partial) converge to approximately -0.60 reward with negligible variance, indicating complete failure to learn cooperative navigation despite having access to complete state information.

Table 1: Final performance metrics across experimental configurations. Values show mean ± standard error of episodic reward averaged over the last 100 training iterations (5 seeds per configuration).

Configuration	Full Obs.	LSTM	Comm.	Slip.	Mean Reward	Std. Error
Baseline	✓	✓			-0.60	0.00
Partial obs. without LSTM					59.81	0.21
Partial obs.		✓			60.82	0.19
Slip. comm. partial obs.		✓	✓	✓	-0.60	0.00
Slippery	✓	✓		✓	-0.60	0.00
Comm. partial obs.		✓	✓		14.72	0.10
With comm.	✓	✓	✓		-0.60	0.00

6.3 Ablation Analysis

6.3.1 Effect of Observability

Comparing baseline (full observability) to partial obs. configurations reveals the most striking finding: partial observability dramatically improves learning outcomes. While the baseline achieves -0.60 reward, the partial obs. configuration reaches 60.82, a difference of over 61 reward units. This

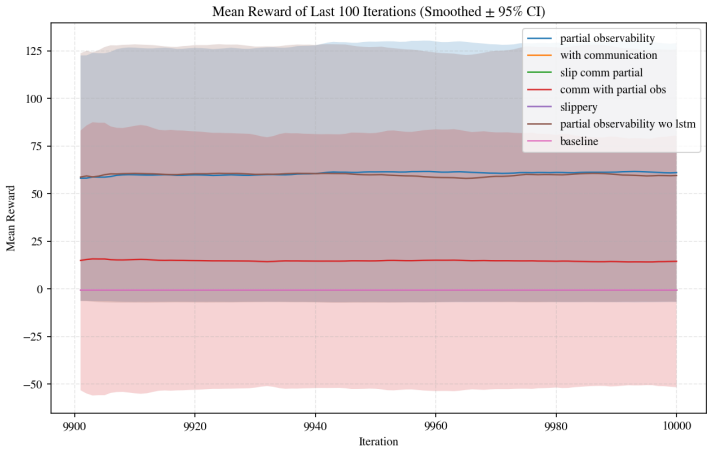

Figure 3: Last 100 mean rewards from training with different configurations.

counterintuitive result challenges conventional assumptions in cooperative MARL, where complete state information is typically expected to facilitate rather than hinder coordination. The restricted 5×5 local view appears to fundamentally reshape the learning dynamics, potentially by simplifying the observation space, improving credit assignment, or constraining the policy space to more learnable behaviors.

6.3.2 Effect of Memory (LSTM)

Comparing partial obs. without LSTM (59.81 ± 0.21) to partial obs. with LSTM (60.82 ± 0.19) shows minimal performance difference, with overlapping standard errors. This suggests that in deterministic partial observability settings, temporal memory provides little advantage. Agents can navigate effectively using only current local observations without needing to integrate information over time. The lack of benefit from recurrent architectures stands in contrast to typical POMDP settings [6], possibly because the 5×5 observation window provides sufficient local information for immediate decision-making in our navigation task.

6.3.3 Effect of Communication

Communication shows detrimental effects across both observability conditions. Under full observability, communication provides no benefit (with comm. achieves -0.60, identical to baseline). Under partial observability, enabling communication substantially degrades performance (comm. partial obs. achieves 14.72 compared to 60.82 for partial obs. without communication). This suggests that the discrete communication channel introduces additional complexity to the action and observation spaces without providing actionable coordination benefits. The five-symbol vocabulary may be insufficient for meaningful information exchange, or agents may struggle to learn effective communication protocols simultaneously with navigation policies [4].

6.3.4 Effect of Stochasticity

Slippery transitions completely prevent learning across all configurations where they are enabled. Both slippery (full obs.) and slip comm. partial (partial obs. with stochasticity) achieve -0.60 reward, identical to failed deterministic configurations. With a per-action success rate of only 1/3, the stochastic dynamics create severe exploration and credit assignment challenges. The combination of action noise, sparse rewards, and multi-agent coordination appears insurmountable for PPO with parameter sharing within 10,000 iterations, consistent with known difficulties in stochastic multi-agent benchmarks [15].

6.4 Simplified Environment Analysis

To investigate whether the learning failures in full observability configurations stem from environment scale or coordination complexity, we conducted a diagnostic experiment with reduced problem dimensions. We decreased the number of agents from four to two, reduced the grid from 7×7 to 4×4, and adjusted the egocentric view from 5×5 to 3×3. Additionally, we modified the observation representation to use a layered structure where each feature (floor tiles, holes, goal, and individual agents) occupies a separate channel, potentially simplifying credit assignment. Due to computational constraints, we trained these simplified configurations for 3,000 iterations rather than 10,000. Figure 4 shows the learning dynamics for the simplified environment. The results reveal a qualitatively different pattern from the main experiments. While partial observability configurations still outperform full observability settings, the performance gap narrows substantially. The partial obs., comm. partial obs., and partial obs. without LSTM configurations converge to mean rewards around 26.70, representing successful but less dramatic learning compared to the 60+ rewards in the main experiment. More significantly, full observability configurations (baseline, with comm.) now achieve moderate success at 12.5 mean reward rather than complete failure at -0.60. Even the slippery configuration shows learning progress, reaching 4.7 mean reward, though performance remains substantially degraded compared to deterministic settings.

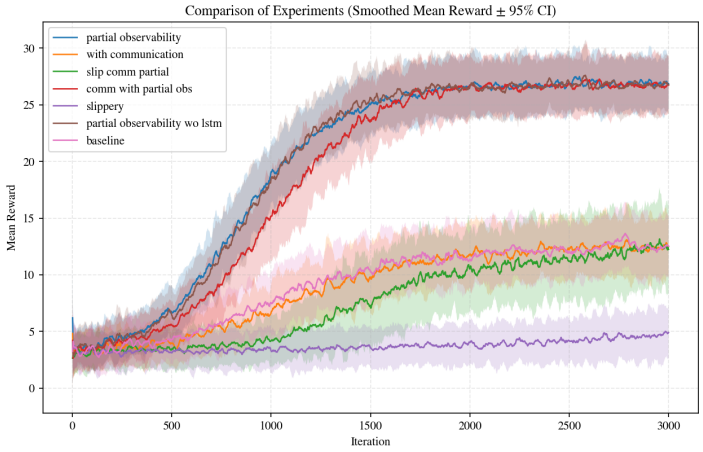

Figure 4: Learning curves for simplified environment (2 agents, 4×4 grid, 3×3 view, layered observations) over 3,000 training iterations. Trajectories show mean across 5 random seeds with shaded regions indicating standard error. Unlike the main experiments, full observability configurations achieve moderate learning, though partial observability still outperforms.

These findings suggest that the catastrophic failure of full observability in the main experiments stems from a combination of environment scale and coordination complexity rather than a fundamental algorithmic limitation. With fewer agents and a smaller state space, PPO with parameter sharing can learn even under full observability. The layered observation representation may also contribute by providing a more structured input that aids credit assignment. However, partial observability maintains its advantage even in the simplified setting, indicating that observation space reduction consistently benefits learning across different problem scales. The persistent performance gap suggests that the mechanisms underlying the observability paradox, whether related to exploration, policy space constraints, or implicit regularization, operate across multiple environment scales, though their severity increases with problem complexity.

7 Conclusion

7.1 Summary of Key Findings

Our experiments reveal three primary findings that challenge conventional expectations in cooperative MARL. First, partial observability dramatically outperforms full observability in enabling cooperative

learning. In our main experiments with four agents on a 7×7 grid, agents with restricted 5×5 local views achieve mean rewards around 60, while agents with complete state access fail entirely with rewards near -0.6. Diagnostic experiments with a simplified environment (two agents, 4×4 grid, layered observations) show that full observability configurations can achieve moderate learning (mean reward 12.5), but partial observability maintains its advantage (mean reward 26.7), suggesting the observability paradox operates across multiple problem scales with severity increasing with coordination complexity. Second, LSTM memory mechanisms provide minimal benefit in deterministic settings, with nearly identical performance between feed-forward and recurrent architectures under partial observability. Third, both communication and stochasticity impede rather than facilitate coordination under our experimental conditions, with communication degrading performance and slippery transitions preventing any learning in the main experiments, though simplified settings show slippery configurations can achieve limited learning (mean reward 4.7). These results establish our multi-agent Frozen Lake extension as a non-trivial benchmark that exposes unexpected learning dynamics in PPO with parameter sharing.

7.2 Limitations

Several limitations constrain the scope and generalizability of our findings. First, we evaluate only a single algorithm, PPO with parameter sharing, leaving open whether the observability paradox persists with other MARL methods such as value decomposition approaches [16, 20] or actor-critic variants [12]. Second, our environment uses a relatively small 7×7 grid with simple discrete actions, which may not reflect coordination challenges in larger or continuous control settings. While our simplified experiments demonstrate that problem scale affects learning difficulty, the mechanisms underlying this relationship remain unclear. Third, we do not analyze the learned policies or agent behaviors in detail, making it difficult to determine the mechanisms underlying the performance differences between observability conditions. Fourth, our discrete five-symbol communication channel is fixed and symbolic rather than learned and differentiable [4], potentially limiting its utility for coordination. Fifth, the simplified diagnostic experiments used layered observation representations that differ structurally from the main experiments, making direct comparison challenging. Finally, hyperparameter tuning was minimal, and the simplified experiments used only 3,000 iterations compared to 10,000 in the main study. Different learning rates, network architectures, training durations, or reward structures might alter the relative performance of configurations.

7.3 Future Work

Several promising directions could extend this work and clarify the unexpected findings. First, manipulating the minimum distance between agents' initial positions and the goal location would test whether successful partial observability agents are genuinely learning exploration and navigation or merely exploiting fortuitous starting configurations where the goal falls within their initial observation radius. Second, an asymmetric setup with one blind agent (zero observation radius) paired with one omniscient agent (full observability) would create a pure cooperation scenario requiring explicit information sharing, providing a cleaner test of communication mechanisms. Third, more thorough analysis of communication actions, including entropy metrics, message diversity, and temporal patterns, could reveal whether agents develop meaningful communication protocols or treat messages as noise. Adding a small penalty for communication usage would further test whether the coordination benefits justify the action space complexity. Fourth, investigating intrinsic motivation techniques [2] or curriculum learning approaches could address the exploration failures in full observability and stochastic settings. Fifth, systematically varying environment scale (grid size, agent count) with consistent observation representations would isolate how coordination complexity contributes to the observability paradox independently of other factors. Sixth, comparing PPO against value decomposition methods, independent learners, and centralized critics would establish whether the observability paradox is algorithm-specific or a fundamental property of the environment. Finally, visualizing learned policies through trajectory heatmaps and attention mechanisms could provide mechanistic insights into why partial observability facilitates learning in this cooperative navigation task.

References

[1] Gregpom Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[2] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

[3] Maxime Chevalier-Boisvert, Louis Willems, and Suman Pal. Bias-variability tradeoffs in multitask representation learning. In *NeurIPS Deep RL Workshop*, 2018.

[4] Jakob Foerster, Yiannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 2016.

[5] Jakob N Foerster, Yiannis M Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in neural information processing systems*, 2016.

[6] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI Fall Symposium Series*, 2015.

[7] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[8] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey of multi-agent reinforcement learning. *Communications of the ACM*, 2019.

[9] Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

[10] Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A. Ortega, Tom Everitt, Antoine Lefrancq, Laurent Orseau, and Shane Legg. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

[11] Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph Gonzalez, Michael Jordan, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning. *International Conference on Machine Learning*, pages 3053–3062, 2018.

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

[13] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging ai applications. *13th USENIX Symposium on Operating Systems Design and Implementation*, pages 561–577, 2018.

[14] Frans A Oliehoek and Christopher Amato. A concise introduction to decentralized pomdps. *Springer*, 2016.

[15] Georgios Papoudakis, Filippos Christianos, Michael Schaarschmidt, and Stefan V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021.

[16] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for decentralised multi-agent reinforcement learning. *International Conference on Machine Learning*, 2018.

[17] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

[19] Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in neural information processing systems*, pages 2244–2252, 2016.

[20] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

[21] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[22] Ardi Tampuu, Tambet Matiisen, Igor Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PLoS One*, 2017.

[23] J Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043, 2021.