Hyperbolic Discounting in Multi-Agent Reinforcement Learning

Raja Farrukh Ali
Kansas State University
rfali@ksu.edu

John Woods
Kansas State University
jwoods03@ksu.edu

Esmaeil Seraj
Georgia Institute of Technology
eseraj3@gatech.edu

Kevin Duong
Kansas State University
kevduong@ksu.edu

Vahid Behzadan
University of New Haven
vbehzadan@newhaven.edu

William Hsu
Kansas State University
bhsu@ksu.edu

Abstract

Decisions often require balancing immediate gratification against long-term benefits. In Reinforcement Learning (RL), this balancing act is influenced by temporal discounting, which quantifies the devaluation of future rewards. Prior research indicates that human decision-making aligns more closely with hyperbolic discounting than the conventional exponential discounting used in RL. As artificial agents become more advanced and pervasive, particularly in multi-agent settings alongside humans, the need for appropriate discounting models becomes critical. Although hyperbolic discounting has been proposed for single-agent learning, its potential in multi-agent reinforcement learning (MARL) remains unexplored. We introduce and formulate hyperbolic discounting in MARL, establishing theoretical and practical foundations across various frameworks, including independent learning, centralized policy gradient, and value decomposition methods. We evaluate hyperbolic discounting on diverse cooperative tasks, comparing it to the exponential discounting baseline. Our results show that hyperbolic discounting achieves higher returns in 60% of scenarios and performs on par with exponential discounting in 95% of tasks, with significant improvements in sparse reward and coordination-intensive environments. This work opens new avenues for robust decision-making processes in the development of advanced multi-agent systems.

1 Introduction

As AI systems evolve, their ability to interact autonomously with the world, collaborate with other agents, and engage with humans becomes increasingly crucial (Islam et al., 2023; Huang et al., 2024). Autonomous AI systems are designed to perform complex tasks without human intervention while aligning with human values, making them essential in environments requiring rapid decision-making and adaptability (Klingefjord et al., 2024; Spitzer et al., 2024). When these systems interact with other agents, they must navigate intricate social dynamics and coordinate actions to achieve shared goals (Oesterheld et al., 2024). This capability is vital in multi-agent settings such as multi-robot teams, autonomous vehicles, or AI assistants in smart homes (Durante et al., 2024; Vats et al., 2024).

In reinforcement learning (RL), the goal of maximizing rewards is central to learning intelligent behavior (Silver et al., 2021). This involves prioritizing reward maximization to generate complex behaviors without specialized problem formulations. The treatment of the reward signal is crucial in developing intelligent agents. Human and animal behavior often shows a preference for immediate rewards over delayed ones (O'Donoghue & Rabin, 2000), rooted in temporal discounting, where the value of rewards diminishes over time. In RL, discounting influences the time-preference for rewards, enforces shortest path strategies, and represents the probability of termination (Puterman, 2014).

Discounting plays a pivotal role, particularly in infinite horizon objectives, to ensure well-defined long-term reward goals (Sutton & Barto, 2018).

The standard approach in RL uses a discount factor $\gamma \in [0, 1)$ to exponentially reduce the present value of future rewards (Bellman, 1957b; Samuelson, 1937). Exponential discounting ensures time-consistent preferences, maintaining proportional reward values regardless of delay (Strotz, 1955). However, human and animal behaviors often deviate from this model, exhibiting hyperbolic discounting instead (Ainslie, 1975; Mazur, 1985; Green & Myerson, 2004), which better captures time-inconsistent preferences (Green et al., 1994) and is optimal under uncertain risks (Sozou, 1998). Hyperbolic discounting is expressed as $V = \frac{r_t}{1+kt}$, where $k > 0$ is the hyperbolic discount rate.

In this work, we posit that incorporating hyperbolic discounting into MARL can enhance agents' adaptability to diverse partners by aligning their decision-making with human temporal preferences. This can improve the robustness and flexibility of MARL systems and foster more effective human-AI collaboration by making agents' behavior more predictable and intuitive. We explore hyperbolic discounting for multi-agent learning, focusing on its impact on agent interactions, particularly in human-AI collaboration scenarios. We compare hyperbolic discounting against exponential discounting as a baseline, noting our agent-centric perspective. Our findings reveal that hyperbolic discounting consistently outperforms exponential discounting, yielding higher returns, especially in environments with sparse rewards and the need for intricate coordination. These improvements are evident across various learning modalities and are particularly pronounced in the Centralized Training Decentralized Execution (CTDE) framework. The main contributions of this work are:

- We establish theoretical and empirical foundations for incorporating hyperbolic discounting across six MARL algorithms, covering independent learning, centralized policy gradient, and value decomposition methods.

- We propose and conduct a comprehensive comparative analysis of two hyperbolic discounting schemes against the traditional exponential model: one computes a hyperbolic value estimate, and the other averages multiple value estimates using normally distributed exponential discount factors. We perform extensive empirical evaluations across four cooperative multi-agent tasks, demonstrating the advantages of hyperbolic discounting in various settings.

2 Preliminaries

2.1 Survival and Hazard Rate

We start by motivating against the use of a single, fixed discount factor. In survival analysis (Cox, 1972), the primary focus is on analyzing and modeling the time until specific events occur, such as death. Sozou (1998) extend this by formalizing time preferences, showing that future rewards should be discounted according to the probability that an agent will not survive to collect them due to encountered risks or hazards. This survival probability is defined as $s(t) = P(\text{agent is alive}|\text{at time } t)$. The present value of a future reward r_t is discounted by $s(t)$, i.e., $v(r_t) = s(t)r_t$. If $s(t) = 1$, the reward is not discounted. The hazard rate, $h(t)$, is defined as the negative rate of change of the log-survival probability, $h(t) = -\frac{d}{dt} \ln s(t)$. For a constant hazard rate λ, the survival rate is $s(t) = e^{-\lambda t}$, leading to an exponential discount function $s(t) = \gamma^t$ with $\gamma = e^{-\lambda}$. Increasing hazard leads to myopic behavior (as $\lambda \to \infty$, $\gamma \to 0$), and decreasing hazard leads to strategic behavior (as $\lambda \to 0$, $\gamma \to 1$). When the hazard rate is uncertain, the survival rate is computed by integrating over a prior distribution $p(\lambda)$, $s(t) = \int_0^\infty p(\lambda)e^{-\lambda t}d\lambda$. For an exponential prior $p(\lambda) = \frac{1}{k}\exp(-\lambda/k)$, the expected survival rate becomes hyperbolic, $s(t) = \frac{1}{1+kt} \equiv \Gamma_k(t)$, where $\Gamma_k(t)$ is the hyperbolic discount function. Different priors over the hazard rate yield different discount functions (Sozou, 1998).

2.2 Hazardous Markov Games

For the multi-agent setting, we formalize our problem based on the Markov Game (Littman, 1994), generalized to include partial observability. Moreover, to consider distributions over the hazard rate, and use non-exponential discounting functions, we remove the discount factor γ and introduce two additions; a hazard distribution, and a general discount function. Concretely, the Hazardous Markov Game (HMG) for N agents is defined by the tuple $\mathcal{G} = \langle \mathcal{N}, \mathcal{S}, \{\mathcal{O}_i\}_{i\in\mathcal{N}}, \{\mathcal{A}_i\}_{i\in\mathcal{N}}, \Omega, \mathcal{P}, r\rangle$, with agents $i \in \mathcal{N} = \{1, \ldots, N\}$, state space $\mathcal{S}$, joint observation space $\mathcal{O} = O_1 \times \ldots \times O_N$, and joint action space $\mathcal{A} = A_1 \times \ldots \times A_N$. Each agent i only perceives local observations $o_i \in \mathcal{O}_i$, which depend on the state and joint action via the observation function $\Omega : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{O})$. The transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ returns a distribution over states given a state and a joint action $A = (a_1, a_2, \ldots, a_N)$. $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the shared reward function, with $r(s, a_1, a_2, \ldots, a_N)$ representing the reward received by all agents after taking actions $a_1, a_2, \ldots, a_N$ in state s. $\mathcal{H}$ is the hazard distribution from which a hazard rate $\lambda \in [0, \infty)$ is sampled at the beginning of each episode. Finally, instead of γ, we consider $d(t)$, which is a general discount function, of which exponential and hyperbolic will be special cases. The objective is to jointly optimize the discounted cumulative reward $G = \mathbb{E}_\lambda \mathbb{E}_{s_t, A_t} \left[\sum_t^\infty d(t) r_t\right]$ where A_t is the joint action at timestep t and $\lambda \sim \mathcal{H}$.

3 Hyperbolic Discounting in MARL

We now discuss the theoretical foundations that can allow us to derive temporal-difference learning solutions while using hyperbolic discounting. The fundamental concepts on value learning apply to single and multi-agent settings alike, with the only difference in multi-agent centralized training paradigms requiring the existence of a central value learner.

3.1 Value-Based Methods

We first show how exponentially discounted Q-values can be used to derive hyperbolic discounted Q-values, building on prior work (Fedus et al., 2019). The Bellman equation (Bellman, 1957a) is written as:

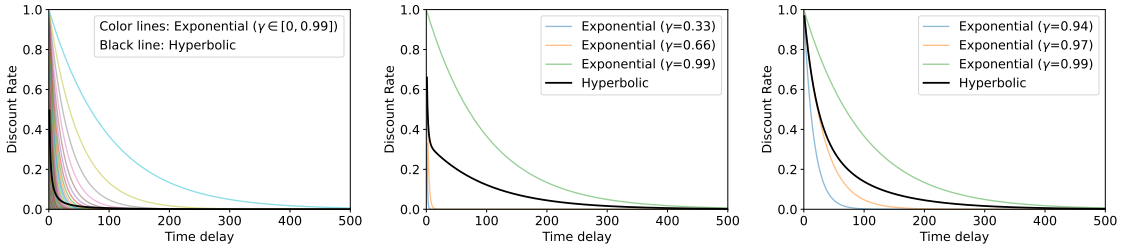
$$Q_\pi^{\gamma^t}(s, a) = \mathbb{E}_{\pi,P}[R(s, a) + \gamma Q_\pi(s', a')] \tag{1}$$

We establish a connection between hyperbolic $Q_\pi^{\Gamma_k}$-values and values obtained through standard Q-learning. The hyperbolic discount Γ_k can be represented as the integral of a specific function $f(\gamma, t)$ for $\gamma = [0, 1)$:

$$\int_0^1 \gamma^{kt} d\gamma = \frac{1}{1 + kt} = \Gamma_k(t) \tag{2}$$

The integration of the function $f(\gamma, t) = \gamma^{kt}$ across the domain $\gamma \in [0, 1)$ results in the hyperbolic discount factor $\Gamma_k(t)$. This integration, incorporating an infinite set of exponential discount factors γ, reveals that γ^{kt} functions as the standard exponential discount factor, linking the concept to traditional Q-learning. This approach suggests that by aggregating an infinite collection of γ values, hyperbolic discounts can be derived for each respective time step t. For a hyperbolic discount function $\Gamma_k(t)$, the hyperbolic Q-values can be written as:

$$\begin{aligned} Q^{\Gamma_\pi(s,a)} =& \mathbb{E}_\pi \left[\sum_t \Gamma_k(t) R(s_t, a_t) \middle| s, a\right] = \mathbb{E}_\pi \left[\sum_t \left(\int_{\gamma=0}^1 \gamma^{k^t} d\gamma\right) R(s_t, a_t) \middle| s, a\right] \\ =& \int_{\gamma=0}^1 \mathbb{E}_\pi \left[\sum_t R(s_t, a_t)(\gamma^{kt}) \middle| s, a\right] d\gamma = \int_{\gamma=0}^1 Q_\pi^{(\gamma^{kt})}(s, a) d\gamma \end{aligned} \tag{3}$$



(a) Exponential vs Hyperbolic (b) Averaged Horizon (c) Hyperbolic

Figure 1: (a) Exponential vs. Hyperbolic discounting. The colored lines are hundred different exponentially discounted curves with $\gamma \in [0, 0.99]$ whereas the blank curve is the mean of all of them. (b) The curves associated with discount factors used in the Averaged Horizon method (number of γ=3) (c) The curves associated with discount factors used in the Hyperbolic discounting method (tail-heavy distribution of discount factors).

3.2 Policy Gradient Methods

In policy gradient methods, particularly Actor-Critic variants utilizing the Advantage function for value estimation, we use a simple method to extend the hyperbolic value estimation described in the previous section. Advantage is defined as $A(s_t, a_t) = Q(s_t, a_t) - V(s_t)$. It follows from Eq. 3 that the hyperbolic advantage can be written as:

$$\begin{aligned} A^{\Gamma}_{\pi}(s_t, a_t) &= \int_0^1 A^{(\gamma^k)^t}_{\pi}(s_t, a_t) d\gamma = \int_0^1 \left[Q^{(\gamma^k)^t}_{\pi}(s_t, a_t) - V^{(\gamma^k)^t}_{\pi}(s_t) \right] d\gamma \\ &= \int_0^1 \left[Q^{(\gamma^k)^t}_{\pi}(s_t, a_t) \right] d\gamma - \int_0^1 \left[V^{(\gamma^k)^t}_{\pi}(s_t) \right] d\gamma \\ &= \int_0^1 \left[r_t + \gamma^k V^{(\gamma^k)^t}_{\pi}(s_{t+1}) \right] d\gamma - \int_0^1 \left[V^{(\gamma^k)^t}_{\pi}(s_t) \right] d\gamma \end{aligned} \tag{4}$$

where we compute $V^{(\gamma^{kt})}$ in a similar manner as $Q^{(\gamma^{kt})}$ from Eq. 3 for each of the n_γ heads by using a multi-head architecture where each head corresponds to the value function for each γ^k. By computing the value function over all γ^k where $0 \leq \gamma < 1$, we estimate hyperbolically discounted advantage. We consider a finite set of γ^k to approximate the advantage through a Riemann sum:

$$A^{\Gamma}_{\pi}(s, a) \approx \sum_{\gamma_i} (\gamma_{i+1} - \gamma_i) w(\gamma) A^{(\gamma^k)_i}_{\pi}(s, a) \tag{5}$$

where $w(\gamma)$ is the exponential weighting condition, that is, there exists a function $w : [0, 1] \to \mathbb{R}$ such that $d(t) = \int_0^1 w(\gamma)\gamma^t d\gamma$,. The advantage estimation is done in the Critic part of the Actor-Critic network, which then supplies the Advantage to the Actor for optimizing the objective function. For the Critic's value estimation learning, we minimize the average of the losses calculated for these multiple γ^k such that the loss function corresponding to each γ^k is defined as $L^{\gamma^k}_v(\theta) = \hat{\mathbb{E}}_t \left[\left(V^{\gamma^k}_{\theta}(s_t) - \hat{V}^{\gamma^k}_{targ} \right)^2 \right]$:

4 Experiments

4.1 Methods

We introduce two novel discounting methods: hyperbolic discounting and averaged horizon discounting. The latter is a special case of the former, where the agent learns over multiple discount factors γ

and averages the resulting value estimates for action selection and learning (network architectures provided in Appendix B). Figure 1 shows the discount functions and their associated curves.

Hyperbolic Discounting Following Fedus et al. (2019), we implement hyperbolic discounting in MARL using a multi-headed value output structure, where each head corresponds to a distinct discount factor. We approximate the hyperbolic value function by integrating multiple value estimates via a Riemann sum:

$$Q^{\Gamma}(s,a) \approx \sum_{(\gamma_{i+1})\subseteq\mathcal{G}} (\gamma_{i+1}-\gamma_i) w(\gamma_i) \cdot Q^{\gamma_i}(s,a) \tag{6}$$

Here, $\mathcal{G} = [\gamma_0, \gamma_1, \ldots, \gamma_n]$ is the set of discount factors, with Q^{γ_i} denoting the Q-values for each γ_i.

Averaged Horizon Discounting Inspired by model averaging techniques (Arpit et al., 2022; Churchill et al., 2023), we propose Averaged Horizon Discounting, where the value predictions from multiple heads are averaged:

$$Q^{\Gamma}(s,a) \approx \frac{1}{|\mathcal{G}|} \sum_{\gamma_i\subseteq\mathcal{G}} Q^{\gamma_i}(s,a) \tag{7}$$

This approach leverages the strengths of ensemble methods by averaging Q-values across different discount factors, integrating diverse discounting horizons into the overall value estimation.

4.2 Setup

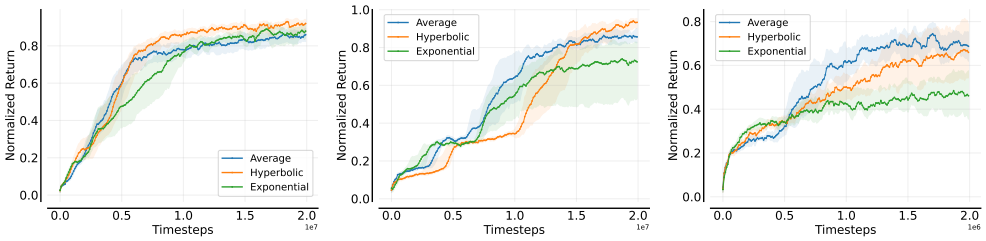
We evaluate the effectiveness of the proposed hyperbolic and average-horizon discounting methods across six MARL algorithms: three from independent learning (IQL, IPPO, IA2C) and three from centralized training with decentralized execution (CTDE) frameworks (QMIX, MAPPO, MAA2C). Please see Appendix C for details of each method. These methods are tested in four distinct MARL environments: Level Based Foraging (LBF) (Albrecht & Ramamoorthy, 2013), Multi-Robot Warehouse (RWARE) (Papoudakis et al., 2021), Multi Particle Environment (MPE) (Mordatch & Abbeel, 2017), and StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019). Each environment presents unique challenges to assess the scalability, generalization, and coordination capabilities of the algorithms. Implementation setup is given at Appendix D, with details about environments at Appendix K.

4.3 Results

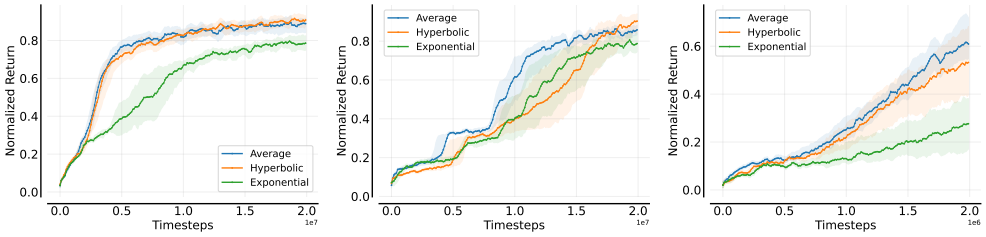
We present results for the two proposed discounting methods and the baseline for each of the six MARL algorithms across four benchmarks, comparing their performance. The results are aggregated across tasks using the marl-eval library, following recommendations in Gorsane et al. (2022). We show results for LBF and RWARE in this section (see Appendix E for MPE and SMAC). Figure 2 and Figure 3 show the comparison of hyperbolic and averaged-horizon discounting against exponential discounting for the six methods across LBF and RWARE environments. Generally, performance differences are noticeable in LBF and RWARE (and for QMIX in SMAC), with one of the proposed variants performing better, while performance differences in MPE are minimal. Tabular results for exponential, averaged-horizon, and hyperbolic discounting are provided in Appendix J.

5 Discussion

Please see Appendix A for related works. We introduce hyperbolic discounting for MARL settings. Our experiments revealed improvements in performance, stability, and sample efficiency with non-exponential discounting methods, which outperformed traditional exponential discounting on more

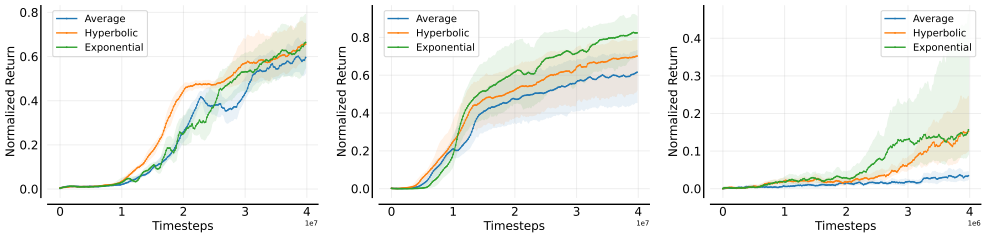


(a) IA2C (b) IPPO (c) IQL

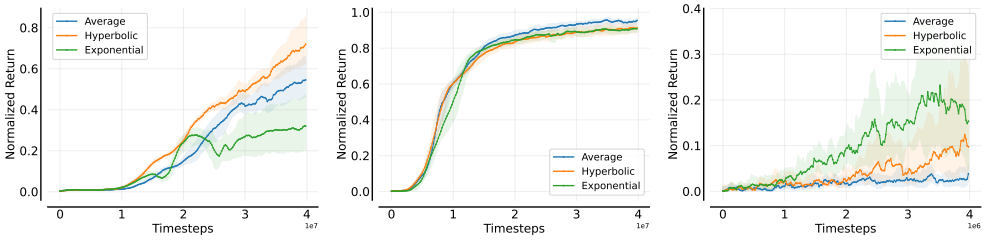


(d) MAA2C (e) MAPPO (f) QMIX

Figure 2: **LBF Results**. A comparison of the proposed hyperbolic discounting policies and the exponential discounting baseline across 6 methods in the LBF benchmark. The hyperbolic variants (blue and orange) perform better than exponential discounting (green) in most of the methods.



(a) IA2C (b) IPPO (c) IQL



(d) MAA2C (e) MAPPO (f) QMIX

Figure 3: **RWARE Results**. A comparison of the proposed hyperbolic discounting policies and the exponential discounting baseline across 6 methods in the RWARE benchmark.

than 60% of the tasks (Appendix J). Hyperbolic discounting emerged as the most reliable method, showing smaller standard deviations and enhanced performance across various algorithms. The structural differences in algorithms influenced the impact of non-exponential discounting, with some benefiting more than others. Future research could explore ensemble methods to further improve non-exponential discounting functions. These findings highlight the potential of non-exponential discounting in reinforcement learning, promoting more efficient and effective decision-making in real-world applications.

There are avenues of future work. The results of Bowling et al. (2023) address scenarios with a constant exponential discount factor, not considering hyperbolic discounting. Since hyperbolic discounting, as approximated by Fedus et al. (2019) and extended here, uses multiple constant exponential discount factors, further theoretical analysis would be beneficial, such as Pitis (2023). Moreover, it would be interesting to study effects of reward discounting in human-AI teams where long-term decision trade-offs are involved.

References

George Ainslie. Specious reward: A behavioral theory of impulsiveness and impulse control. *Psychological Bulletin*, 82(4):463–496, 1975. doi: 10.1037/h0076860.

Stefano V Albrecht and Subramanian Ramamoorthy. A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. In *AAMAS'13 Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pp. 1155–1156. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Raja Farrukh Ali, Nasik Muhammad Nafi, Kevin Duong, and William Hsu. Efficient multi-horizon learning for off-policy reinforcement learning. In *NeurIPS Deep Reinforcement Learning Workshop*, 2022.

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. Ensemble of averages: Improving model selection and boosting performance in domain generalization. *Advances in Neural Information Processing Systems*, 35:8265–8277, 2022.

Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1957a.

Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pp. 679–684, 1957b.

Michael T Bixter and Christian C Luhmann. The social contagion of temporal discounting in small social networks. *Cognitive Research: Principles and Implications*, 6(1):13, 2021.

Michael Bowling, John D Martin, David Abel, and Will Dabney. Settling the reward hypothesis. In *International Conference on Machine Learning*, pp. 3003–3020. PMLR, 2023.

Victor Churchill, Steve Manns, Zhen Chen, and Dongbin Xiu. Robust modeling of unknown dynamical systems via ensemble averaged learning. *Journal of Computational Physics*, 474:111842, 2023.

David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

Christian Schroeder De Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

Zane Durante, Bidipta Sarkar, Ran Gong, Rohan Taori, Yusuke Noda, Paul Tang, Ehsan Adeli, Shrinidhi Kowshika Lakshmikanth, Kevin Schulman, Arnold Milstein, et al. An interactive agent foundation model. *arXiv preprint arXiv:2402.05929*, 2024.

William Fedus, Carles Gelada, Yoshua Bengio, Marc G Bellemare, and Hugo Larochelle. Hyperbolic discounting and learning over multiple horizons. *arXiv preprint arXiv:1902.06865*, 2019.

Rihab Gorsane, Omayma Mahjoub, Ruan John de Kock, Roland Dubb, Siddarth Singh, and Arnu Pretorius. Towards a standardised performance evaluation protocol for cooperative marl. *Advances in Neural Information Processing Systems*, 35:5510–5521, 2022.

Leonard Green and Joel Myerson. A discounting framework for choice with delayed and probabilistic rewards. *Psychological bulletin*, 130(5):769, 2004.

Leonard Green, Nathanael Fristoe, and Joel Myerson. Temporal discounting and preference reversals in choice between delayed outcomes. *Psychonomic Bulletin & Review*, 1(3):383–389, 1994.

Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *Autonomous Agents and Multiagent Systems: AAMAS 2017 Workshops*, pp. 66–83. Springer, 2017.

Qiuyuan Huang, Naoki Wake, Bidipta Sarkar, Zane Durante, Ran Gong, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Noboru Kuno, Ade Famoti, et al. Agent ai towards a holistic intelligence. *arXiv preprint arXiv:2403.00833*, 2024.

Md Saiful Islam, Srijita Das, Sai Krishna Gottipati, William Duguay, Clodéric Mars, Jalal Arabneydi, Antoine Fagette, Matthew Guzdial, and Matthew Taylor. Human-ai collaboration in real-world complex environment with reinforcement learning, 2023.

Oliver Klingefjord, Ryan Lowe, and Joe Edelman. What are human values, and how do we align ai to them? *arXiv preprint arXiv:2404.10636*, 2024.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Neural Information Processing Systems (NIPS)*, 2017.

Xueguang Lyu, Yuchen Xiao, Brett Daley, and Christopher Amato. Contrasting centralized and decentralized critics in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.04402*, 2021.

James E Mazur. Probability and delay of reinforcement as factors in discrete-trial choice. *Journal of the Experimental Analysis of Behavior*, 43(3):341–351, 1985.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *NeurIPS Deep Learning Workshop*, 2013.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. *arXiv preprint arXiv:1703.04908*, 2017.

Ted O'Donoghue and Matthew Rabin. The economics of immediate gratification. *Journal of behavioral decision making*, 13(2):233–250, 2000.

Caspar Oesterheld, Johannes Treutlein, Roger B Grosse, Vincent Conitzer, and Jakob Foerster. Similarity-based cooperative equilibrium. *Advances in Neural Information Processing Systems*, 36, 2024.

Georgios Papoudakis, Filippos Christianos, Arrasy Rahman, and Stefano V Albrecht. Dealing with non-stationarity in multi-agent deep reinforcement learning. *arXiv preprint arXiv:1906.04737*, 2019.

Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS)*, 2021.

Silviu Pitis. Consistent aggregation of objectives with diverse time preferences requires non-markovian rewards. *Advances in Neural Information Processing Systems*, 36, 2023.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4295–4304. PMLR, 2018.

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.

Paul A Samuelson. A note on measurement of utility. *The review of economic studies*, 4(2):155–161, 1937.

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

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

David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. Reward is enough. *Artificial Intelligence*, 299:103535, 2021.

Peter D Sozou. On hyperbolic discounting and uncertain hazard rates. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 265(1409):2015–2020, 1998.

Philipp Spitzer, Joshua Holstein, Patrick Hemmer, Michael Vössing, Niklas Kühl, Dominik Martin, and Gerhard Satzger. On the effect of contextual information on human delegation behavior in human-ai collaboration, 2024.

Robert Henry Strotz. Myopia and inconsistency in dynamic utility maximization. *The review of economic studies*, 23(3):165–180, 1955.

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction, Second Edition*. MIT press Cambridge, 2018.

Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pp. 330–337, 1993.

Vanshika Vats, Marzia Binta Nizam, Minghao Liu, Ziyuan Wang, Richard Ho, Mohnish Sai Prasad, Vincent Titterton, Sai Venkat Malreddy, Riya Aggarwal, Yanwen Xu, Lei Ding, Jay Mehta, Nathan Grinnell, Li Liu, Sijia Zhong, Devanathan Nallur Gandamani, Xinyi Tang, Rohan Ghosalkar, Celeste Shen, Rachel Shen, Nafisa Hussain, Kesav Ravichandran, and James Davis. A survey on human-ai teaming with large pre-trained models, 2024.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

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.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pp. 321–384, 2021.

Appendix

A Related Work

We focus on the aspect of discounting preferences in social settings which involves more than one individual (we refer the reader to Fedus et al. (2019) for an in depth review of discounting in individual human preferences). In controlled studies, discounting future rewards has been mostly studied as a personal preference parameter, where each individual is given a questionnaire to evaluate their valuation of future rewards. These studies show how decisions involving immediate versus long-term benefits are influenced by temporal discounting—where individuals place less value on delayed rewards. Recent studies have expanded this concept to decisions made in group settings, like dyads or small groups, revealing that direct interactions can lead to aligned preferences among participants, making them more similar in patience level over time. Bixter & Luhmann (2021) study whether such social influences could also be indirect, such as through mutual acquaintances within a group. Focusing on hypothetical monetary rewards, the research involved groups of three where one member's decision preferences before collaboration were linked to another's preferences after collaborating with an intermediary. Findings highlighted that decision-making tendencies regarding time can spread through a social network's connections, showing the presence of indirect social influence in a controlled setting.

B Network Architectures

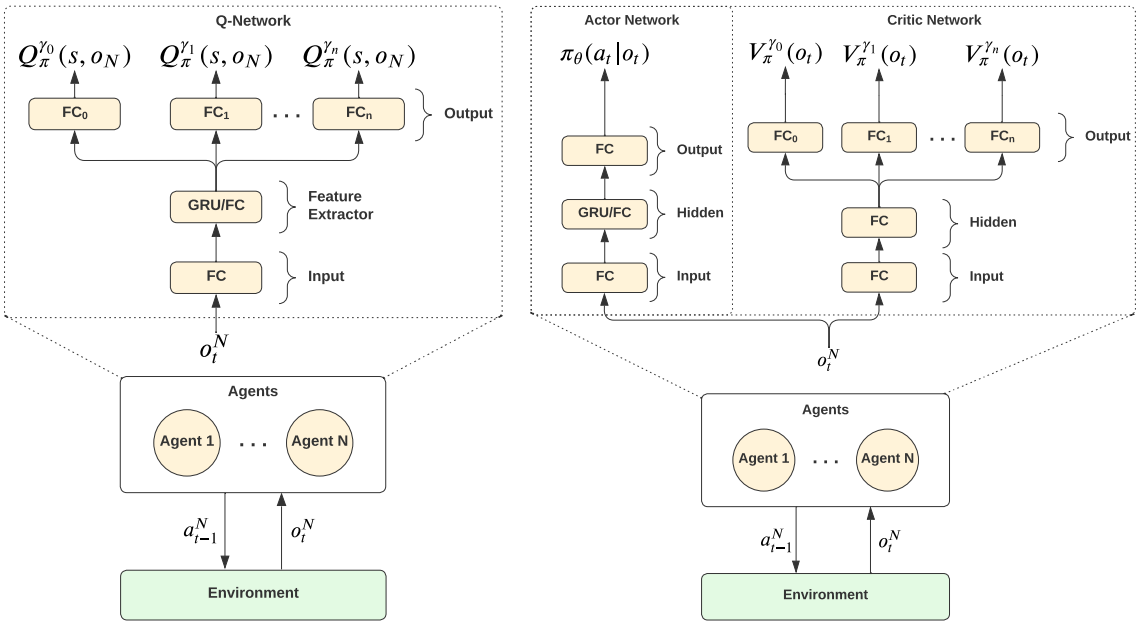
We present a practical way of implementing hyperbolic discounting in multi-agent RL methods. Integrating non-exponential discounting methods into multi-agent reinforcement learning algorithms requires careful consideration of the neural network design. We explore the three main types of neural network architectures found in MARL algorithms, and discuss changes necessary to incorporate hyperbolic discounting in each.

B.1 Value-Based Q-Networks

Value-based networks approximate the value function or the action-value function of a state-action pair, representing the expected cumulative reward an agent can obtain by following a certain policy from a given state, and form the basis of temporal difference learning methods such as Q-learning (Watkins & Dayan, 1992). In deep multi-agent reinforcement learning (MARL), Q-networks were first used to study the interaction and learning of multiple agents. We use a Multi-Layer Perceptron (MLP) network with fully connected (FC) layers for agent observations, representation learning, and Q-value generation, utilizing ReLU activation for non-linearity. Hidden layers can be FC or Gated Recurrent Units (GRU) layers to capture temporal dependencies. In MARL, agent behavior and learning are modeled with or without parameter sharing. Without parameter sharing, each agent has unique network parameters, while with parameter sharing, all agents use a common set of parameters, receiving additional inputs identifying each agent through a one-hot vector (Gupta et al., 2017). This shared approach enables distinct behaviors, optimized by the collective loss from all agents. To accommodate non-exponential discounting, our modified Q-network outputs a set of Q-values for each agent, represented as $Q_\pi^{\gamma_i}$, with N fully connected layers corresponding to the number of discount factors, each producing Q-values for a unique discount factor γ_i. These multi-headed Q-values allow each agent to learn its own set, although no joint Q-value is learned, a concept discussed in methods that decompose a joint Q-value into individual agent Q-values. Figure 4a shows the proposed Q-value network for hyperbolic discounting in multi-agent Q-learning methods.

B.2 Policy Gradient Actor-Critic Networks

Policy-based reinforcement learning methods optimize an action's probability distribution instead of a value function, facing challenges like high variance and slow convergence. Actor-critic methods



(a) (b)

Figure 4: (a) Network architecture for hyperbolic discounting in IQL. (b) Network architecture for hyperbolic discounting variants for Actor-Critic methods used in IA2C, IPPO, MAA2C, MAPPO.

combine the benefits of value-based and policy-based approaches by incorporating a critic network for value function estimation and an actor network for action selection based on the critic's assessments. Notable actor-critic frameworks include Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Advantage Actor-Critic (A2C) (Mnih et al., 2016). In this framework, the actor network learns a policy that maps the current state to a probability distribution over actions, while the critic network approximates the value function to evaluate the quality of a given state. The actor network's adjustments are informed by the policy gradient, calculated using the critic's value estimates, and the critic network is refined through temporal-difference learning. To accommodate multiple discount factors, a multi-headed critic network can be introduced, outputting separate value functions for each discount factor, which are then used to compute the advantage function for optimizing the actor network. Figure 4b shows the proposed actor-critic network for hyperbolic discounting in centralized multi-agent policy gradient methods.

B.3 Value Factorization Networks

Value factorization techniques in multi-agent reinforcement learning (MARL) facilitate cooperation among agents by decomposing the collective value function into individual components corresponding to each agent. This allows for the optimization of joint actions while maintaining the autonomy of individual agent decision-making processes. The Value Decomposition Network (VDN) (Sunehag et al., 2017) and QMIX (Rashid et al., 2018) are two notable methods that employ this technique.

For this class of methods, we use QMIX as a template for incorporating non-exponential discounting within the framework of value decomposition methods. Within QMIX, individual agents generate Q-values using the Deep Q-Network architecture (Mnih et al., 2013). These Q-values are then forwarded to a mixing network, constructed from hypernetworks, which nonlinearly combines the individual Q-values into a collective value, Q_{total}. This collective value guides the action selection process for each agent, fostering collaboration while preserving agent autonomy. To implement variable discounting within QMIX, the methodology introduced in Section 3.1 is adopted, establishing several output layers for distinct discount factors, γ. The individual agent Q-values, Q_π^n, are fed

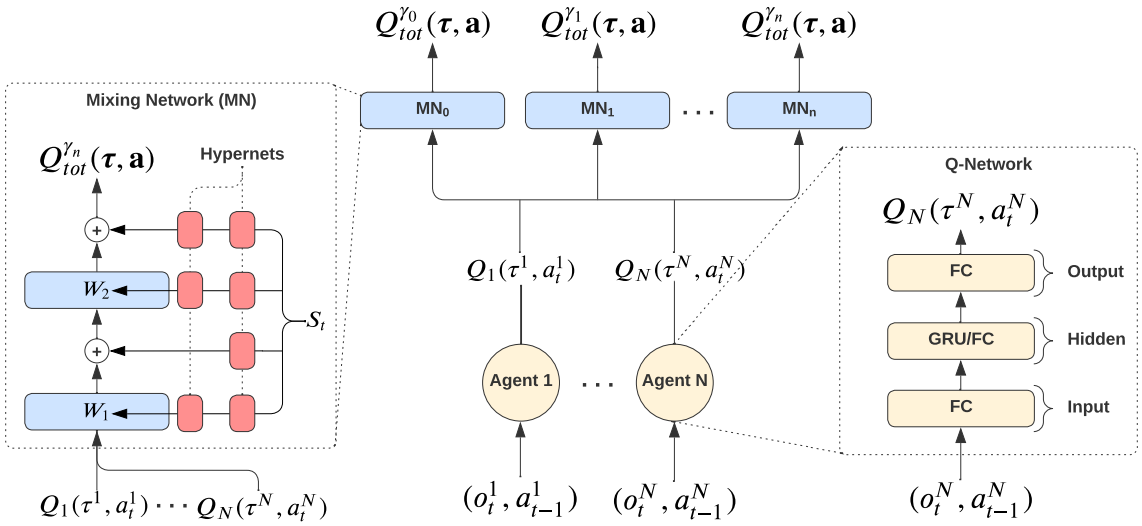


Figure 5: Network architecture for hyperbolic discounting in QMIX. The mixing network uses hypernetworks to determine the weights and biases for its layers. The agent network can have a recurrent layer depending upon the environment characteristics (see Table 2).

into their respective mixing networks, producing individual Q_{total} values, $Q_{\text{tot}}^{\gamma_i}$, for each discount factor γ_i. This approach preserves the foundational network design by (Rashid et al., 2018) while integrating a distinct mixing network for every different discount factor, enabling the incorporation of non-exponential discounting within the value decomposition framework. Figure 5 shows the proposed network for hyperbolic discounting in QMIX like value factorization methods.

C Multi-Agent Algorithms

To assess a fair evaluation of non-exponential discounting, a wide range of algorithms were selected to cover multiple approaches in multi-agent reinforcement learning. Using the network modifications and mathematical formulations presented in the previous sections, our algorithms are now suited for non-exponential discounting strategies. In order to evaluate the effects of non-exponential discounting in multi-agent reinforcement learning, a diverse range of algorithms was selected for this work. These algorithms can be grouped into two primary categories: Independent Learning and Centralized Learning Decentralized Execution (CTDE). The IL category contains IQL, IA2C, and IPPO, while the CTDE category contains QMIX, MAA2C, and MAPPO, in which QMIX is a value factorization/decomposition method, and MAA2C and MAAPO are centralized policy gradient methods. By modifying the network structures and using the proposed mathematical formulations mentioned in the previous discussion, these algorithms have been adapted to incorporate non-exponential discounting functions. In this discussion, we emphasize the key differences and similarities between the algorithms and demonstrate how non-exponential discounting strategies are integrated into their learning and optimization processes.

C.1 Independent Learning (IL)

In independent learning, single-agent reinforcement learning algorithms are applied to individual agents without the consideration of a multi-agent structure. Each agent perceives other agents as part of the environment, treating them as dynamic and unpredictable elements rather than explicitly acknowledging their presence as separate learning entities (Papoudakis et al., 2021). This leads to a lack of communication and coordination among agents during both the training and execution phases (Zhang et al., 2021). While independent learning allows for simpler and more efficient training processes, this approach may result in suboptimal and less coordinated behaviors, especially in

tasks where agent collaboration is crucial for success. Moreover, the agents may not adapt well to changes in the strategies of their counterparts or be able to handle emergent behaviors that arise from interactions within the multi-agent system. Despite these limitations, independent learning serves as a baseline for evaluating more advanced multi-agent reinforcement learning methods. For this work, We have selected and documented implementation details for these three independent learning algorithms:

Independent Q-Learning (IQL). IQL utilizes a decentralized value-based Q-network to approximate Q-values for the individual observations of each agent (Tan, 1993). Each agent then updates the Q-network by minimizing the Q-learning loss through the calculation of a temporal difference (TD) error. In traditional methods of discounting, a single discount rate γ is used during the TD calculation. With our modification, the Q-network outputs a set of Q-values for each unique discount rate. These sets of values are then used to calculate separate TD targets, each using its corresponding discount rate (Ali et al., 2022; Fedus et al., 2019). During action selection, we use the discounting functions, as presented earlier, to combine a set of Q-values values for different discount factors into a single set of Q-values, using either hyperbolic or averaged horizon discounting. The final set of Q-values is then used for action selection through an *argmax*.

Independent Advantage Actor-Critic (IA2C) This method uses a decentralized actor-critic network to optimize its policy and estimate state values based on individual observations. The actor network in IA2C maps the agent's state to a probability distribution over actions, while the critic network estimates the state-value function. To update both networks, IA2C calculates the advantage function using the values of a critic network. This advantage function helps in determining how much better a specific action is compared to the average actions taken in the current state. Each agent independently updates its actor and critic networks by minimizing the policy gradient loss and the TD error. Similar to the modifications made to IQL, IA2C utilizes multiple discount rates for value estimations. However, due to the action selection relying on a probability distribution instead of values, the advantage function is used in the optimization objective, as opposed to a state-action function.

Independent Proximal Policy Optimization (IPPO) This method follows a similar decentralized approach as IA2C, with each agent using its own actor-critic network for policy optimization and state value estimation (De Witt et al., 2020). The key difference between IPPO and IA2C lies in the optimization process. IPPO employs a trust region optimization approach, which uses a ratio of new and old policies in the surrogate objective function to ensure small, stable updates to the policy and improve sample efficiency (Schulman et al., 2017). Similar to the modifications made in IA2C, adapting IPPO to utilize a non-exponential discounting function requires the calculations of independently discounted advantages. These advantages are then passed through a non-exponential discounting function and then used for optimization.

C.2 Centralized Learning Decentralized Execution (CTDE)

In contrast to independent learning, centralized training decentralized execution (CTDE) methods address the challenges in multi-agent reinforcement learning by allowing agents to share information during the training phase while maintaining decentralized executions (Papoudakis et al., 2021). This approach enables agents to learn coordinated policies by utilizing a global perspective and accounting for the actions and observations of other agents during training. However, during the execution phase, each agent makes decisions independently based on its own observations and learning policy, without relying on any direct communication or information sharing with other agents. This balance between centralized learning and decentralized execution enables CTDE methods to improve coordination and performance in multi-agent tasks while retaining the scalability and robustness associated with decentralized systems. We have selected and documented implementation details for these three CTDE algorithms:

Q-Value Mixing Network (QMIX) is a cooperative multi-agent reinforcement learning algorithm that combines the advantages of centralized learning with decentralized execution to tackle coordination challenges in multi-agent settings (Rashid et al., 2018; 2020)). In QMIX, the agents employ decentralized Q-networks to approximate individual action-value functions, while a centralized mixing network is utilized to aggregate the agents' Q-values into a joint action-value function. The mixing network is designed to be a monotonic function, ensuring that the optimal joint-action can be derived from the optimal individual actions. Since individual Q-values of each agent are combined to compute the joint-action values, the discounted estimations are based on the mixing network outputs instead of the individual Q-values. In order to implement a non-exponential discounting function, additional mixing networks are introduced to compute separate Q-totals corresponding to individual discount factors. These separate Q-totals are then aggregated through the discounting functions proposed to compute a total loss that is then backpropagated to the individual agents.

Multi-Agent Advantage Actor-Critic (MAA2C) is a cooperative multi-agent reinforcement learning algorithm that builds upon the actor-critic framework by incorporating centralized learning to enhance coordination between agents. Similar to IA2C, MAA2C employs decentralized actor networks for each agent, mapping individual observations to probability distributions over actions. However, unlike IA2C, MAA2C utilizes a centralized critic network, which considers the joint observations of all agents to estimate the state-value function (Lyu et al., 2021). This centralized learning aspect enables MAA2C to capture the interactions between agents, which improves coordination among agents. For incorporating non-exponential discounting, we follow a similar setup to IA2C and utilize multiple discount rates for value estimations. Since MAA2C relies on a probability distribution for action selection, similar to IA2C, the discounting function is applied to the advantage estimations. The difference between MAA2C and IA2C is the inputs to the critic networks. MAA2C aggregates the observations of all agents to combine a joint-observation input, whereas IA2C keeps these observations separated. The actual learning algorithm implementation of non-exponential discounting itself remains the same as IA2C.

Multi-Agent Proximal Policy Optimization (MAPPO) MAPPO (Yu et al., 2022) is a cooperative multi-agent reinforcement learning algorithm that extends the Proximal Policy Optimization (PPO) method (Schulman et al., 2017) to accommodate multiple agents, focusing on enhanced coordination and stable policy updates in complex environments. Similar to MAA2C, MAPPO utilizes a centralized critic network and a decentralized actor network. The key difference between MAPPO and MAA2C lies in the optimization process, similar to the differences between IA2C and IPPO. MAPPO employs a trust region optimization approach, which uses a ratio of new and old policies in the surrogate objective function to ensure small, stable updates to the policy and improve sample efficiency. However, similar to the modifications made in MAA2C, adapting MAPPO to utilize a non-exponential discounting function requires the calculations of independently discounted advantages corresponding to unique discount rates. These advantages are then used in the optimization objective $J(\theta)$.

D Implementation

For each environment, experiments were done for both hyperbolic-weighted and hyperbolic-average discounting methods applied to the six selected MARL algorithms. Agents are trained for steps specified for each method and environment combination described previously, with results averaged over 5 seeds to account for variability. Parameter sharing is employed which allows for each agent to have a different behavior while sharing the same base network, and environment-specific adaptations, such as action selection probability adjustments for invalid actions, are implemented to ensure fairness and comparability across tests. All experiments were run across two Ubuntu-based servers. Each server is equipped with a 16-core Intel Xeon 4215 processor clocked at 3.50GHz, 1 TB of RAM, and 8 NVIDIA A40 GPUs.

D.1 Evaluation Criteria

The performance of each algorithm is assessed based on several criteria, including the efficiency of learning (speed of convergence), the robustness of the learned policies, scalability to larger and more complex scenarios, and the ability to generalize to unseen environments. Additional metrics, specific to each environment, such as resource collection efficiency in LBF and RWARE, task completion time in MPE, and win rate in SMAC, are also considered. To ensure fair comparison and compensate for the higher sample efficiency of off-policy algorithms relative to on-policy ones, we follow the procedure recommended by Papoudakis et al. (2021) and adjust training steps accordingly. For MPE and LBF, on-policy algorithms (IA2C, IPPO, MAA2C, MAPPO) are trained for 20 million timesteps, and off-policy algorithms (IQL, QMIX) are trained for 2 million timesteps. In SMAC and RWARE environments, the on-policy and off-policy training is set to 40 and four million timesteps, respectively. Despite on-policy algorithms not reusing samples via experience replay and hence resulting in lower sample efficiency, they are not considered slower. Evaluations are carried out at every 10k steps for 2M train steps, every 100k steps for 20M train steps, every 20k steps for 4M train steps, and every 200k steps for 40M train steps, maintaining the total number of evaluations to be constant (41 evaluations), are done at regular intervals throughout the training and are done for 100 episodes per evaluation.

D.2 Performance Metrics

To evaluate the performance of the algorithms, we consider two metrics, following the recommendations of Papoudakis et al. (2021): maximum returns and average returns. **Maximum Returns**: For each algorithm, we identify the evaluation timestep during training where the algorithm achieves the highest average evaluation returns across five random seeds. We report the average returns and the 95% confidence interval across these five seeds from this evaluation timestep. This metric represents the peak performance achieved by the algorithm. **Average Returns:** We also report the average returns achieved throughout all evaluations during training. This metric is computed over all evaluations executed during the training process, considering not only the final achieved returns but also the algorithm's learning speed.

D.3 The γ Interval

The number of concurrent horizons (n_γ) and the values of γ that set the minimum and maximum horizons are crucial for determining the agent's reward sensitivity. In the Hyperbolic discounting variants, we follow Fedus et al. (2019) by choosing γ values such that the exponent of the largest value in the interval with respect to the hyperbolic exponent k approximately equals γ_{max} (set to 0.99). Using a power-method, the base b must satisfy $b = \exp\left(\frac{\ln(1-\gamma_{max}^{1/k})}{n_\gamma}\right)$. This ensures γ_{max} remains stable. For the Averaged-horizon variants, we selected equally spaced values of γ between $\in (0, 1)$ corresponding to the number of total discount factors (n_γ).

D.4 Hyperparameters

Table 1 shows the hyperparameters related to discounting and common to all methods and environments, whereas Table 2 shows the hyperparameters for each of the MARL methods in our experiments.

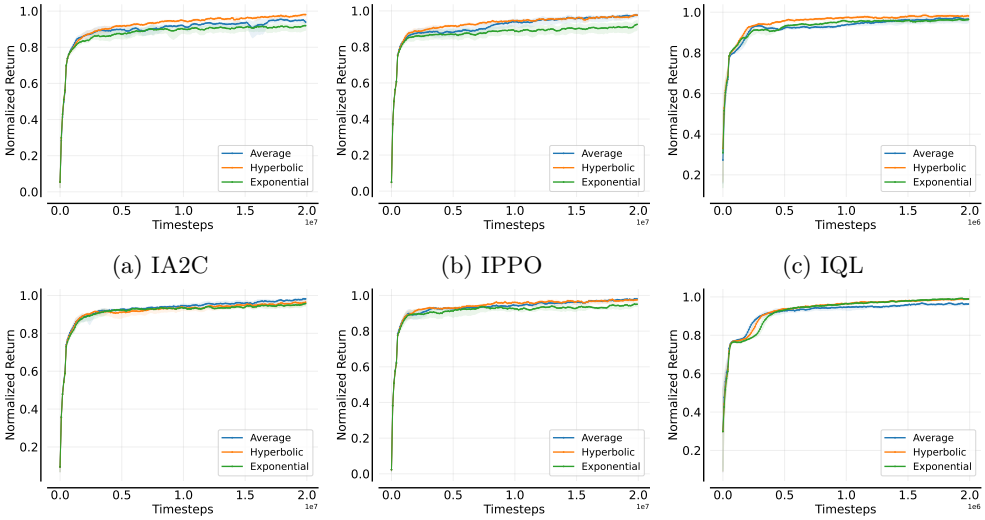
E Additional Results

E.1 MPE

Figure 6 presents a comparison of the proposed hyperbolic discounting and average discounting formulations against the baseline of exponential discounting for each of the 6 methods on environments from the MPE (Lowe et al., 2017) benchmark using marl-eval (Gorsane et al., 2022).

Hyperparameters	Values
Number of γ (hyperbolic)	3
γ_{max}	0.99
Hyperbolic exponent k	0.1
Integral estimate	lower
γ values (Exponential)	0.99
γ values (Hyperbolic)	[0.94, 0.975, 0.99]
γ values (Average)	[0.33, 0.66, 0.99]

Table 1: Hyperparameters related to reward discounting (common to all methods and environments).



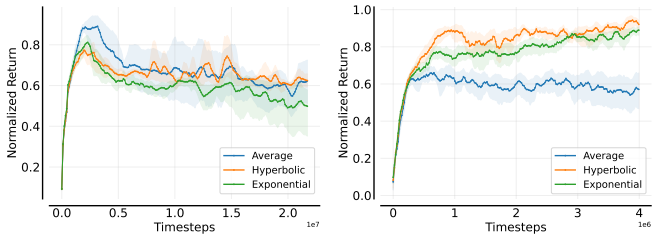
(a) IA2C (b) IPPO (c) IQL

(d) MAA2C (e) MAPPO (f) QMIX

Figure 6: **MPE Results**. A comparison of the proposed hyperbolic discounting policies and the exponential discounting baseline across 6 methods in the MPE benchmark.

E.2 SMAC

Figure 7 presents a comparison of the proposed hyperbolic discounting and average discounting formulations against the baseline of exponential discounting for 2 methods on environments from the SMAC (Samvelyan et al., 2019) benchmark.



(a) MAA2C (b) QMIX

Figure 7: **SMAC Results**. A comparison of the proposed hyperbolic discounting policies and the exponential discounting baseline across 2 methods in the SMAC benchmark.

Table 2: Hyperparameters for MARL algorithms with parameter sharing

Algs	Hyperparameter	MPE	SMAC	LBF	RWARE
IA2C	hidden dimension	64	128	128	64
	learning rate	0.0005	0.0005	0.0005	0.0005
	reward standardization	True	True	True	True
	network type	GRU	FC	GRU	FC
	entropy coefficient	0.01	0.01	0.001	0.01
	target update	0.01 (soft)	0.01 (soft)	0.01 (soft)	0.01 (soft)
	n-step	5	5	5	5
IPPO	hidden dimension	64	128	128	128
	learning rate	0.0003	0.0005	0.0003	0.0005
	reward standardization	True	False	False	False
	network type	GRU	GRU	FC	GRU
	entropy coefficient	0.01	0.001	0.001	0.001
	target update	0.01 (soft)	0.01 (soft)	200 (hard)	0.01 (soft)
	n-step	5	10	5	10
IQL	hidden dimension	128	128	128	64
	learning rate	0.0005	0.0005	0.0003	0.0005
	reward standardization	True	False	True	True
	network type	FC	GRU	GRU	FC
	evaluation epsilon	0.0	0.05	0.05	0.05
	epsilon anneal	200,000	50,000	200,000	50,000
	target update	0.01 (soft)	200 (hard)	200 (hard)	0.01 (soft)
MAA2C	hidden dimension	128	128	128	64
	learning rate	0.0005	0.0005	0.0005	0.0005
	reward standardization	True	True	True	True
	network type	GRU	FC	GRU	FC
	entropy coefficient	0.01	0.01	0.01	0.01
	target update	0.01 (soft)	0.01 (soft)	0.01 (soft)	0.01 (soft)
	n-step	5	5	10	5
MAPPO	hidden dimension	64	64	128	128
	learning rate	0.0005	0.0005	0.0003	0.0005
	reward standardization	True	False	False	False
	network type	FC	GRU	FC	FC
	entropy coefficient	0.01	0.001	0.001	0.001
	target update	0.01 (soft)	0.01 (soft)	0.01 (soft)	0.01 (soft)
	n-step	5	10	5	10
QMIX	hidden dimension	64	128	64	64
	learning rate	0.0005	0.005	0.0003	0.0005
	reward standardization	True	True	True	True
	network type	GRU	GRU	GRU	FC
	evaluation epsilon	0.0	0.05	0.05	0.05
	epsilon anneal	200,000	50,000	200,000	50,000
	target update	0.01 (soft)	200 (hard)	0.01 (soft)	0.01 (soft)

F Individual Results: LBF

Figure 8 presents results on a set of three environments from the LBF (Papoudakis et al., 2021) benchmark.

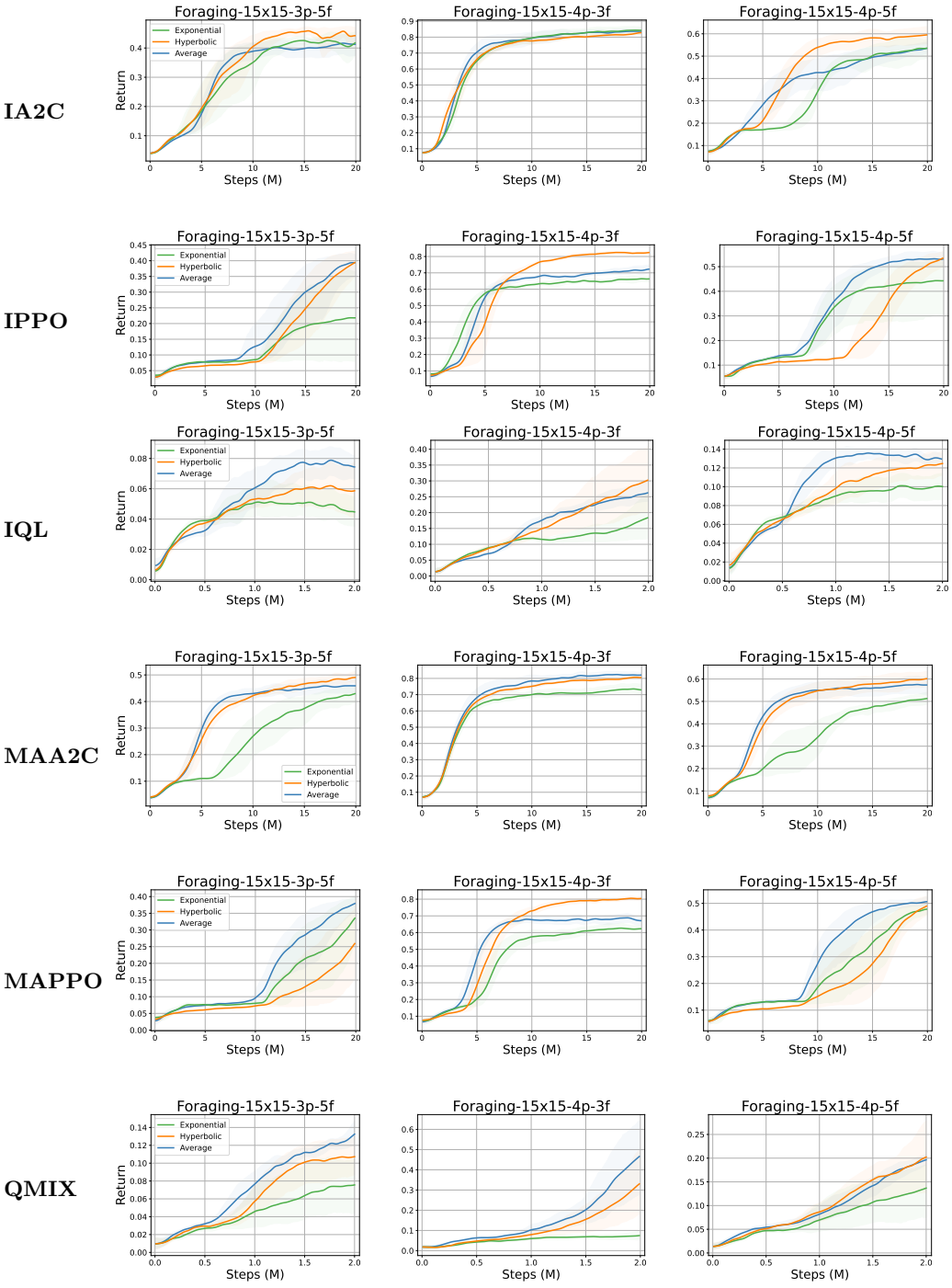


Figure 8: Individual results of each of the 3 discounting policies on all 6 methods in 3 LBF environments. The hyperbolic variants (blue and orange) perform better than exponential discounting (green) in most of the environments.

G Individual Results: RWARE

Figure 9 presents results on a set of three environments from the Robot Warehouse (RWARE) (Papoudakis et al., 2021) benchmark.

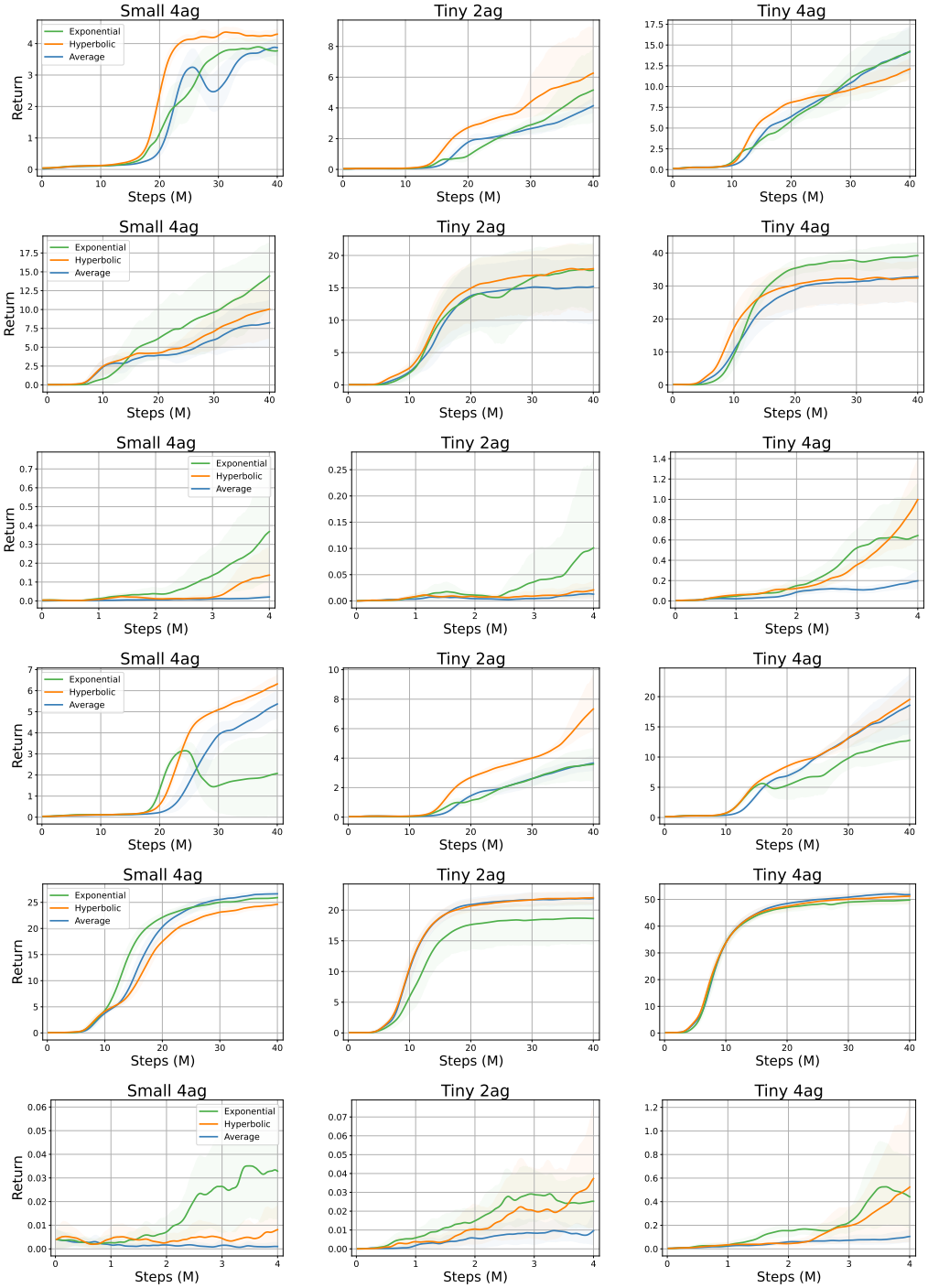


Figure 9: Individual results of each of the 3 discounting policies on all 6 methods (IA2C, IPPO, IQL, MAA2C, MAAPO, QMIX) in 3 RWARE environments.

H Individual Results: MPE

Figure 10 presents results on a set of three environments from the Multi Particle Environment (MPE) (Lowe et al., 2017) benchmark.

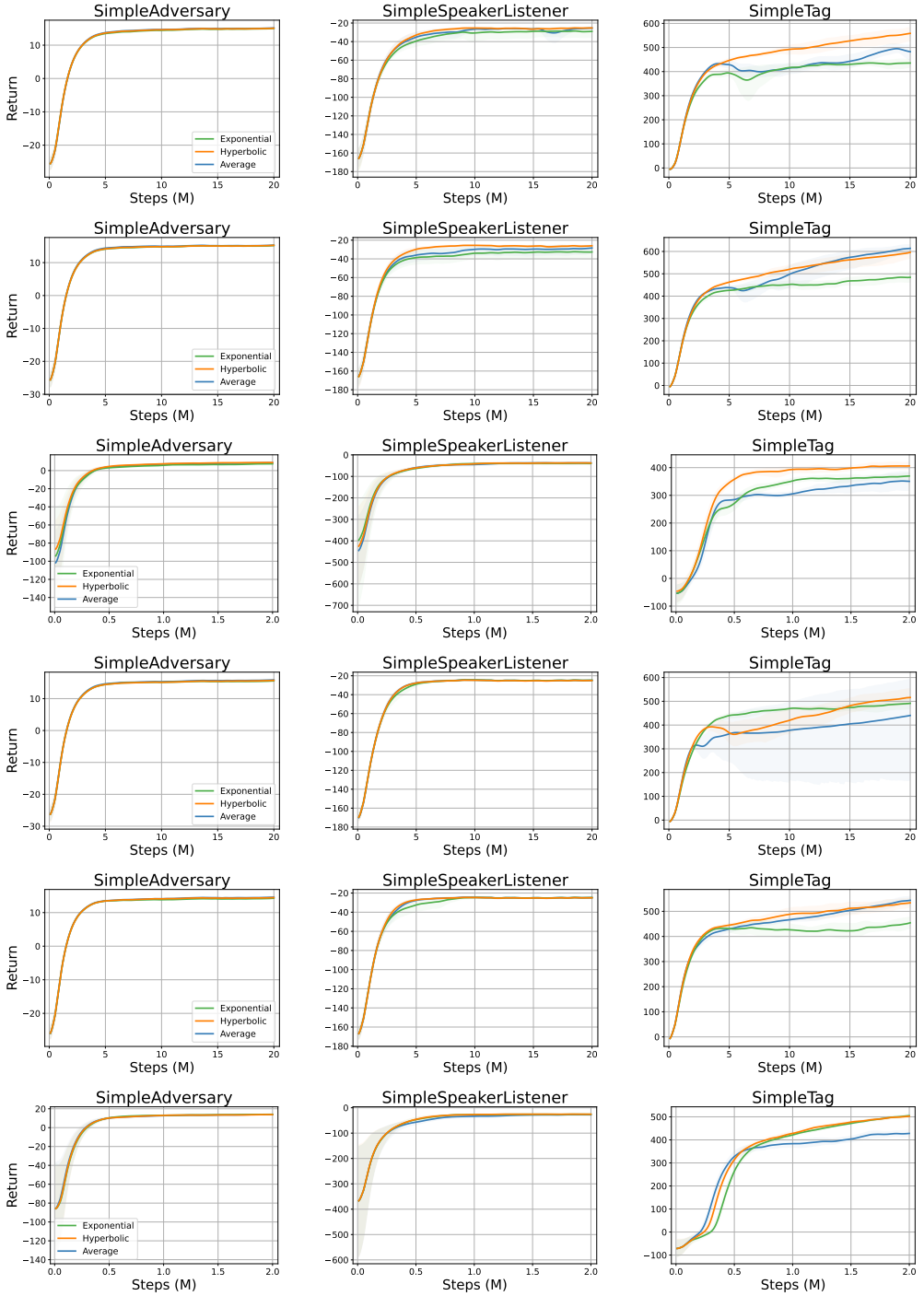


Figure 10: Individual results of each of the 3 discounting policies on 6 methods (IA2C, IPPO, IQL, MAA2C, MAAPO, QMIX) in 3 MPE environments. There is no notable performance difference between discounting methods except in the SimpleTag environment.

I Individual Results: SMAC

Figure 11 presents a comparison of the proposed hyperbolic discounting and average discounting formulations against the baseline of exponential discounting for the QMIX method on a set of three maps from the StarCraft MultiAgent Challenge (Samvelyan et al., 2019) benchmark. Due to the increased runtimes of SMAC, we only present results here for QMIX, which is the most widely studied MARL algorithm, especially in the SMAC environment.

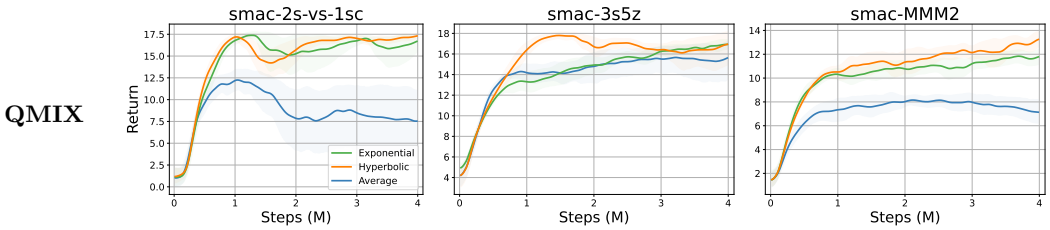


Figure 11: Individual results of each of the 3 discounting policies on only QMIX for 3 SMAC maps.

J Tabular Results

Tables 3 and 4 showcase the maximum returns for each environment and algorithm (independent learning and CTDE methods), while Table 5 and 6 showcase the average returns for each environment and algorithm (independent learning and CTDE methods). Please see definitions for maximum and average returns in Appendix D.2. The highest mean values are highlighted in bold. To determine the statistical significance of the performance differences, two-sided t-tests were conducted with a significance level of 0.05, comparing the highest-performing algorithm against each other algorithm for every task. If an algorithm's performance was not statistically significantly different from the best-performing algorithm, the corresponding value is marked with an asterisk (*). Therefore, the bold values or values with asterisks in the table represent the algorithms that achieved the best performance for each task.

Table 3: Maximum returns and 95% confidence interval over five seeds for the three discounting policies across all tasks for IL Methods.

Algs	Environment	Exponential	Average	Hyperbolic
IA2C	Foraging-15x15-3p-5f-v2	0.44 ± 0.03*	0.43 ± 0.05*	**0.48 ± 0.01**
	Foraging-15x15-4p-3f-v2	**0.85 ± 0.03**	0.84 ± 0.02*	0.84 ± 0.01*
	Foraging-15x15-4p-5f-v2	0.55 ± 0.04*	0.55 ± 0.03*	**0.60 ± 0.05**
	Simpleadversary-v0	15.23 ± 0.16*	**15.36 ± 0.22**	15.25 ± 0.24*
	Simplespeakerlistener-v0	-27.04 ± 6.01*	**-23.51 ± 1.41**	-23.80 ± 1.38*
	Simpletag-v0	443.64 ± 17.92	502.55 ± 18.65	**567.29 ± 34.67**
	Rware-small-4ag-v1	3.98 ± 0.44	4.06 ± 0.42*	**4.58 ± 0.24**
	Rware-tiny-2ag-v1	5.53 ± 3.14*	4.70 ± 1.16*	**6.64 ± 2.99**
	Rware-tiny-4ag-v1	10.33 ± 7.40*	11.56 ± 6.46*	**13.02 ± 1.67**
IPPO	Foraging-15x15-3p-5f-v2	0.23 ± 0.17*	0.41 ± 0.05*	**0.41 ± 0.03**
	Foraging-15x15-4p-3f-v2	0.68 ± 0.02	0.74 ± 0.03	**0.83 ± 0.01**
	Foraging-15x15-4p-5f-v2	0.45 ± 0.13*	0.54 ± 0.04*	**0.56 ± 0.03**
	Simpleadversary-v0	15.41 ± 0.25*	**15.50 ± 0.19**	15.47 ± 0.26*
	Simplespeakerlistener-v0	-30.80 ± 7.73*	-26.51 ± 3.10*	**-23.90 ± 1.46**
	Simpletag-v0	495.76 ± 34.47	**624.45 ± 25.76**	611.43 ± 24.86*
	Rware-small-4ag-v1	**16.04 ± 4.52**	8.90 ± 3.73	10.60 ± 4.55*
	Rware-tiny-2ag-v1	**18.36 ± 5.88**	15.53 ± 6.18*	18.31 ± 6.89*
	Rware-tiny-4ag-v1	28.98 ± 18.92*	**33.22 ± 8.71**	32.90 ± 7.67*
IQL	Foraging-15x15-3p-5f-v2	0.05 ± 0.02	**0.08 ± 0.01**	0.06 ± 0.02*
	Foraging-15x15-4p-3f-v2	0.20 ± 0.09*	0.28 ± 0.05*	**0.32 ± 0.12**
	Foraging-15x15-4p-5f-v2	0.11 ± 0.01	**0.14 ± 0.01**	0.13 ± 0.02*
	Simpleadversary-v0	7.74 ± 0.48	**9.20 ± 0.22**	9.15 ± 0.49*
	Simplespeakerlistener-v0	-37.07 ± 7.67*	-35.48 ± 6.67*	**-35.45 ± 6.48**
	Simpletag-v0	375.73 ± 14.91	355.77 ± 39.68	**413.09 ± 13.24**
	Rware-small-4ag-v1	**0.42 ± 0.47**	0.03 ± 0.02*	0.15 ± 0.18*
	Rware-tiny-2ag-v1	**0.12 ± 0.22**	0.02 ± 0.01*	0.03 ± 0.02*
	Rware-tiny-4ag-v1	0.75 ± 0.64*	0.25 ± 0.17	**1.20 ± 0.60**

Table 4: Maximum returns and 95% confidence interval over five seeds for the three discounting policies across all tasks for CTDE Methods.

Algs	Environment	Exponential	Average	Hyperbolic
MAA2C	Foraging-15x15-3p-5f-v2	0.44 ± 0.04	0.47 ± 0.02*	**0.50 ± 0.02**
	Foraging-15x15-4p-3f-v2	0.74 ± 0.04	**0.84 ± 0.02**	0.82 ± 0.01*
	Foraging-15x15-4p-5f-v2	0.52 ± 0.02	0.59 ± 0.05*	**0.61 ± 0.04**
	Simpleadversary-v0	15.75 ± 0.24*	**16.09 ± 0.22**	15.91 ± 0.24*
	Simplespeakerlistener-v0	**-23.13 ± 1.30**	-23.21 ± 1.35*	-23.43 ± 1.31*
	Simpletag-v0	502.37 ± 32.86*	453.35 ± 271.68*	**523.43 ± 40.35**
	Rware-small-4ag-v1	3.56 ± 0.17	5.73 ± 0.86*	**6.65 ± 0.50**
	Rware-tiny-2ag-v1	3.78 ± 1.44	3.57 ± 0.48	**8.07 ± 3.03**
	Rware-tiny-4ag-v1	13.46 ± 5.23*	11.17 ± 7.67*	**18.27 ± 9.78**
MAPPO	Foraging-15x15-3p-5f-v2	0.37 ± 0.05*	**0.40 ± 0.02**	0.30 ± 0.12*
	Foraging-15x15-4p-3f-v2	0.64 ± 0.05	0.70 ± 0.03	**0.81 ± 0.01**
	Foraging-15x15-4p-5f-v2	0.50 ± 0.02*	**0.52 ± 0.03**	0.51 ± 0.04*
	Simpleadversary-v0	14.52 ± 0.28*	**14.75 ± 0.23**	14.66 ± 0.22*
	Simplespeakerlistener-v0	**-23.14 ± 1.36**	-23.23 ± 1.31*	-23.35 ± 1.35*
	Simpletag-v0	470.61 ± 24.88	**552.06 ± 19.96**	541.54 ± 14.49*
	Rware-small-4ag-v1	26.20 ± 0.76*	**26.86 ± 0.71**	25.05 ± 0.52
	Rware-tiny-2ag-v1	18.77 ± 4.26*	22.06 ± 1.20*	**22.13 ± 1.37**
	Rware-tiny-4ag-v1	50.71 ± 1.97*	31.28 ± 24.69*	**51.50 ± 2.50**
QMIX	Foraging-15x15-3p-5f-v2	0.08 ± 0.04	**0.14 ± 0.02**	0.11 ± 0.02*
	Foraging-15x15-4p-3f-v2	0.08 ± 0.03	**0.51 ± 0.22**	0.37 ± 0.20*
	Foraging-15x15-4p-5f-v2	0.15 ± 0.06*	0.20 ± 0.01*	**0.21 ± 0.10**
	Simpleadversary-v0	**14.42 ± 0.15**	14.27 ± 0.72*	14.17 ± 0.65*
	Simplespeakerlistener-v0	-24.15 ± 1.47*	-25.51 ± 1.95*	**-24.00 ± 1.59**
	Simpletag-v0	**511.80 ± 13.05**	435.48 ± 6.02	509.49 ± 12.30*
	Rware-small-4ag-v1	**0.05 ± 0.04**	0.00 ± 0.01*	0.01 ± 0.02*
	Rware-tiny-2ag-v1	0.04 ± 0.02*	0.01 ± 0.02*	**0.05 ± 0.05**
	Rware-tiny-4ag-v1	0.62 ± 0.52*	0.12 ± 0.07*	**0.65 ± 0.86**
	Smac-2s-vs-1sc	17.81 ± 0.31*	13.21 ± 1.69	**17.97 ± 0.28**
	Smac-3s5z	15.30 ± 0.86	15.21 ± 1.83	**18.10 ± 0.30**
	Smac-mmm2	11.19 ± 0.34*	8.35 ± 0.32	**11.86 ± 1.29**

Table 5: Average returns and 95% confidence interval over five seeds for the three discounting policies across all tasks for Indepndent Learning methods.

Algs	Environment	Exponential	Average	Hyperbolic
IA2C	Foraging-15x15-3p-5f-v2	0.32 ± 0.06*	0.33 ± 0.03*	**0.35 ± 0.05**
	Foraging-15x15-4p-3f-v2	0.72 ± 0.04*	**0.73 ± 0.04**	0.72 ± 0.03*
	Foraging-15x15-4p-5f-v2	0.36 ± 0.04	0.40 ± 0.06*	**0.45 ± 0.04**
	Simpleadversary-v0	13.68 ± 0.30*	**13.86 ± 0.29**	13.78 ± 0.28*
	Simplespeakerlistener-v0	-34.88 ± 6.71*	-32.27 ± 4.25*	**-30.87 ± 1.92**
	Simpletag-v0	405.60 ± 22.42	429.43 ± 24.97	**485.93 ± 21.78**
	Rware-small-4ag-v1	1.91 ± 0.40	1.76 ± 0.28	**2.45 ± 0.15**
	Rware-tiny-2ag-v1	1.92 ± 0.69*	1.75 ± 0.27*	**2.85 ± 1.02**
	Rware-tiny-4ag-v1	5.11 ± 3.50*	5.68 ± 2.90*	**6.97 ± 0.71**
IPPO	Foraging-15x15-3p-5f-v2	0.13 ± 0.07*	**0.20 ± 0.06**	0.17 ± 0.05*
	Foraging-15x15-4p-3f-v2	0.58 ± 0.03	0.60 ± 0.03*	**0.65 ± 0.04**
	Foraging-15x15-4p-5f-v2	0.30 ± 0.09*	**0.35 ± 0.04**	0.25 ± 0.04
	Simpleadversary-v0	14.16 ± 0.28*	**14.33 ± 0.28**	14.21 ± 0.28*
	Simplespeakerlistener-v0	-37.05 ± 7.86*	-33.78 ± 6.66*	**-29.78 ± 1.98**
	Simpletag-v0	446.13 ± 19.62	510.58 ± 37.86*	**516.53 ± 24.54**
	Rware-small-4ag-v1	**6.77 ± 3.86**	4.43 ± 1.76*	5.17 ± 1.91*
	Rware-tiny-2ag-v1	11.23 ± 4.75*	10.44 ± 4.43*	**12.08 ± 4.66**
	Rware-tiny-4ag-v1	20.96 ± 14.08*	23.71 ± 6.22*	**25.46 ± 6.52**
IQL	Foraging-15x15-3p-5f-v2	0.04 ± 0.01*	**0.06 ± 0.01**	0.05 ± 0.01*
	Foraging-15x15-4p-3f-v2	0.12 ± 0.03*	0.17 ± 0.02*	**0.17 ± 0.06**
	Foraging-15x15-4p-5f-v2	0.08 ± 0.01	**0.11 ± 0.01**	0.10 ± 0.01*
	Simpleadversary-v0	4.14 ± 1.27*	5.11 ± 1.02*	**5.75 ± 1.24**
	Simplespeakerlistener-v0	-50.38 ± 8.32*	-50.23 ± 8.13*	**-49.67 ± 7.58**
	Simpletag-v0	318.72 ± 17.06	296.17 ± 24.84	**362.06 ± 15.69**
	Rware-small-4ag-v1	**0.11 ± 0.11**	0.01 ± 0.01*	0.04 ± 0.04*
	Rware-tiny-2ag-v1	**0.03 ± 0.05**	0.01 ± 0.01*	0.01 ± 0.01*
	Rware-tiny-4ag-v1	0.29 ± 0.17*	0.09 ± 0.04	**0.29 ± 0.10**

Table 6: Average returns and 95% confidence interval over five seeds for the three discounting policies across all tasks for CTDE methods.

Algs	Environment	Exponential	Average	Hyperbolic
MAA2C	Foraging-15x15-3p-5f-v2	0.27 ± 0.05	0.38 ± 0.03*	**0.38 ± 0.03**
	Foraging-15x15-4p-3f-v2	0.65 ± 0.03	**0.72 ± 0.04**	0.70 ± 0.02*
	Foraging-15x15-4p-5f-v2	0.36 ± 0.05	**0.49 ± 0.04**	0.49 ± 0.04*
	Simpleadversary-v0	14.44 ± 0.27*	**14.67 ± 0.28**	14.50 ± 0.28*
	Simplespeakerlistener-v0	-29.02 ± 1.71*	**-28.61 ± 1.17**	-28.82 ± 1.27*
	Simpletag-v0	**452.36 ± 15.49**	385.21 ± 197.41*	434.42 ± 42.02*
	Rware-small-4ag-v1	1.23 ± 0.89	1.95 ± 0.41	**2.71 ± 0.19**
	Rware-tiny-2ag-v1	1.62 ± 0.47	1.56 ± 0.32	**2.89 ± 0.47**
	Rware-tiny-4ag-v1	5.82 ± 3.53*	5.65 ± 3.60*	**7.55 ± 3.74**
MAPPO	Foraging-15x15-3p-5f-v2	0.15 ± 0.06*	**0.19 ± 0.04**	0.11 ± 0.05
	Foraging-15x15-4p-3f-v2	0.48 ± 0.04	0.57 ± 0.04*	**0.61 ± 0.04**
	Foraging-15x15-4p-5f-v2	0.26 ± 0.06*	**0.31 ± 0.04**	0.23 ± 0.06
	Simpleadversary-v0	13.36 ± 0.27*	**13.61 ± 0.26**	13.53 ± 0.27*
	Simplespeakerlistener-v0	-29.43 ± 2.97*	-28.39 ± 1.71*	**-28.24 ± 1.30**
	Simpletag-v0	426.19 ± 24.48	472.18 ± 15.47*	**481.18 ± 20.32**
	Rware-small-4ag-v1	**17.82 ± 1.04**	16.97 ± 1.00*	15.30 ± 0.85
	Rware-tiny-2ag-v1	13.96 ± 3.50*	16.99 ± 1.18*	**17.03 ± 1.17**
	Rware-tiny-4ag-v1	40.41 ± 2.82*	24.94 ± 19.71*	**41.25 ± 2.11**
QMIX	Foraging-15x15-3p-5f-v2	0.05 ± 0.02	**0.08 ± 0.01**	0.07 ± 0.02*
	Foraging-15x15-4p-3f-v2	0.06 ± 0.01	**0.17 ± 0.07**	0.13 ± 0.06*
	Foraging-15x15-4p-5f-v2	0.08 ± 0.04*	0.11 ± 0.02*	**0.11 ± 0.03**
	Simpleadversary-v0	10.52 ± 1.09*	**10.63 ± 1.46**	10.45 ± 1.27*
	Simplespeakerlistener-v0	-37.60 ± 5.48*	-42.29 ± 7.68*	**-37.12 ± 5.12**
	Simpletag-v0	368.15 ± 10.75*	346.31 ± 18.55	**383.42 ± 10.88**
	Rware-small-4ag-v1	**0.02 ± 0.01**	0.00 ± 0.00*	0.00 ± 0.01*
	Rware-tiny-2ag-v1	**0.02 ± 0.01**	0.01 ± 0.01*	0.01 ± 0.02*
	Rware-tiny-4ag-v1	**0.20 ± 0.13**	0.06 ± 0.03*	0.16 ± 0.18*
	Smac-2s-vs-1sc	**14.51 ± 1.58**	10.04 ± 2.36	14.39 ± 1.35*
	Smac-3s5z	13.41 ± 0.94	13.81 ± 1.06	**15.66 ± 0.74**
	Smac-mmm2	9.96 ± 0.69*	7.30 ± 0.86	**10.32 ± 0.83**

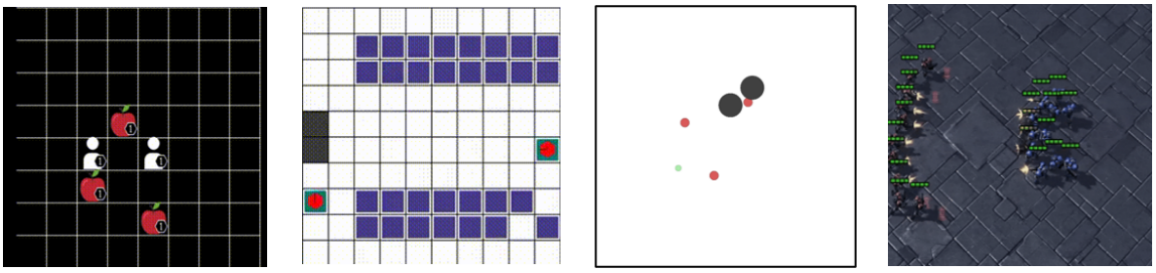


Figure 12: The four MARL benchmarks studied in this work. From left to right: Level Based Foraging (LBF), Multi-Robot Warehouse (RWARE), Multi Particle Environment (MPE) and StarCraft Multi-Agent Challenge (SMAC).

K MARL Environments

The following multi-agent environments were used in this work, an overview of which is given in Table 7 and visualized in Figure 12.

Level-Based Foraging (LBF) (Albrecht & Ramamoorthy, 2013) is a grid-world scenario where agents must collect food items scattered randomly. Agents and items are assigned levels, and a group of one or more agents can collect an item if the sum of their levels is greater than or equal to the item's level. Agents can move in four directions and have an action to attempt loading an adjacent item, which succeeds based on the agents' levels. LBF allows for configuring various tasks, including partial observability or highly cooperative scenarios where all agents must participate simultaneously to collect items. Seven distinct tasks with varying world sizes, agent numbers, observability, and cooperation settings are defined to test the multi-agent reinforcement learning (MARL) algorithms' ability to cooperate in collecting resources within the grid world. We use 3 of the available 7 tasks in this work.

Multi-Robot Warehouse (RWARE) (Papoudakis et al., 2021) simulates a dynamic warehouse scenario where robotic agents must collaborate to transport and sort packages efficiently. It is a cooperative, partially observable scenario with sparse rewards. In this grid-world warehouse simulation, agents (robots) must locate and deliver requested shelves to workstations and return them after delivery. Agents receive rewards only upon completely delivering the requested shelves. The environment features sparse and high-dimensional observations, consisting of a 3×3 grid containing information about surrounding agents and shelves. Agents can move forward, rotate, and load/unload shelves. Three tasks are defined with varying world sizes, agent numbers, and shelf requests. The sparsity of rewards and partial observability make RWARE challenging, as agents must coordinate their actions effectively to complete a series of steps before receiving any reward signal. RWARE serves as a benchmark for evaluating algorithm performance in cooperative, partially observable scenarios with sparse rewards, where effective coordination is crucial. We use all the 3 available tasks in this study.

Multi-Agent Particle Environments (MPE) (Mordatch & Abbeel, 2017; Lowe et al., 2017) consist of several two-dimensional navigation tasks designed to evaluate agent coordination and the ability to handle non-stationarity. In this study, we investigate four tasks from MPE that emphasize coordination: Speaker-Listener, Spread, Adversary, and Predator-Prey. In these tasks, agents receive high-level feature vectors as observations, including relative agent and landmark locations, and agents are required to navigate to fulfill environment-defined tasks. While the Speaker-Listener task requires binary communication and is partially observable, the remaining tasks are fully observable. The MPE tasks serve as a benchmark for assessing agent coordination and their capability to deal with non-stationarity. The rewards in these tasks are highly dependent on the joint actions of the agents, and a lack of effective coordination among individual agents can severely reduce the received rewards (Papoudakis et al., 2019). As mentioned above, we use 4 of the available 6 tasks in MPE.

The StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) simulates battle scenarios in the real-time strategy game StarCraft, where a team of controlled agents must destroy an enemy team using fixed policies. Agents have limited observation radii and can move and attack enemies. SMAC offers many tasks, known commonly as maps, varying in the number and types of controlled units. The key challenges include accurately estimating state values under partial observability, coordinating an increasing number of diverse agent types across tasks, and handling large action spaces as agents can select targets for healing or attacking based on their unit type. This environment serves as a complex benchmark for multi-agent reinforcement learning algorithms, demanding effective coordination, decision-making under partial observability, strategic planning, adaptability, and teamwork in realistic scenarios with large action spaces and diverse agents. We study three maps for SMAC in our work. Due to limited computing, we restrict our analysis of SMAC to QMIX, the most widely used and benchmarked method on SMAC.

Table 7: Overview of MARL environments (reproduced from (Papoudakis et al., 2021)).

Environment	**Observability**	**Reward Sparsity**	**Agents**	**Main Difficulty**
LBF	Partial / Full	Sparse	2-4	Coordination
RWARE	Partial	Sparse	2-4	Sparse reward
MPE	Partial / Full	Dense	2-3	Non-stationarity
SMAC	Partial	Dense	2-10	Large action space