MULTI-AGENT REINFORCEMENT LEARNING WITH ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Centralized training with decentralized execution (CTDE) exploits privileged information during training, yet decentralized multi-agent reinforcement learning policies often remain driven by sparse environmental rewards. Online policy distillation can provide dense supervision, but applying it to MARL introduces observation mismatch between a globally informed teacher and local students, as well as policy learning mismatch between their continuously evolving policies. We propose **MAOPD**, an online policy distillation framework that jointly trains an autoregressive centralized teacher and decentralized parallel students. Agent distillation bridges observation mismatch by using a local prefix predictor to construct locally realizable decision representations that match privileged teacher features. Action distillation addresses the structural policy mismatch by factorizing the teacher's joint policy into agent-wise conditional targets on student-generated trajectories. The teacher and students further share most policy parameters and co-adapt online through separate reinforcement-learning objectives, alleviating evolving policy mismatch without separate teacher pre-training. Across 26 tasks from multiple MARL benchmarks (i.e., RWARE, SMAX, and MPE), MAOPD achieves the highest aggregate performance under matched environment-interaction budgets and ranks first or tied for first on 22 tasks. Ablations confirm the complementary benefits of both distillation objectives and the online co-adaptation mechanisms.

1 INTRODUCTION

Multi-agent Reinforcement Learning (MARL) has been used in autonomous driving (Chen et al., 2023; Zhang et al., 2023), robot coordination (Agrawal et al., 2023; Ma et al., 2021), communication networks (Du et al., 2023; Guo et al., 2022), and large-scale resource management (Gao et al., 2023; 2024). In MARL, the centralized training with decentralized execution (CTDE) paradigm is the most popular adopted paradigm, exploiting privileged information such as global states during training while retaining policies that individual agents can execute independently. Existing CTDE methods instantiate this principle through value factorization, centralized critics, or auto-regressive policy modeling, as exemplified by QMIX (Rashid et al., 2018), MAPPO (Yu et al., 2022), and MAT (Wen et al., 2022).

In most CTDE methods (Rashid et al., 2018; Wen et al., 2022; Qin et al., 2025), privileged information is not continuously transformed into dense decision-level supervision. Instead, policy improvement remains predominantly driven by the reward signals from the environment. Agents still rely heavily on repeated trial-and-error interaction to discover effective coordinated behaviors that contribute to environmental feedback. This dependence on rewards can substantially limit learning efficiency in tasks with sparse rewards, long-horizon, or extensive exploration (Li et al., 2024a).

Researchers (Hong et al., 2022; Zhao et al., 2024; Li et al., 2026) have explored policy distillation with privileged information in MARL. However, these are off-policy methods that train a teacher policy and later distill it into student policies. Off-policy distillation can lead to initialization gap with poor student performance, as shown in Figure 1 (a). Even a well-trained centralized teacher (16M-step trained) may yield a poor student (the second lowest performance) when their initial policy distributions are substantially mismatched. This initialization gap is not resolved by simply updating the teacher during distillation; as the teacher and students continue to evolve, their policy distributions can remain persistently mismatched, leading to a policy learning mismatch.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

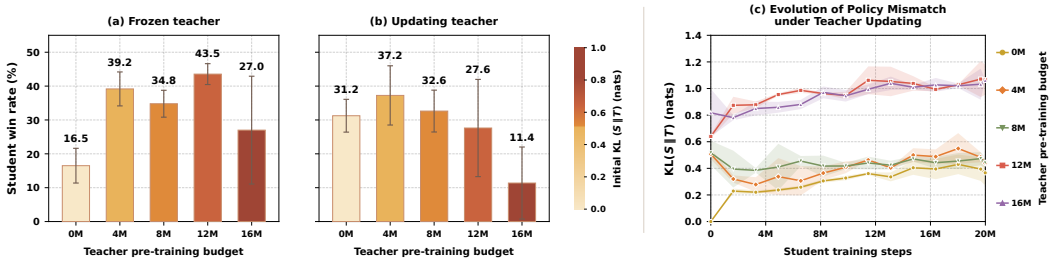


Figure 1: Why directly applying OPD to MARL is challenging: initialization and policy learning mismatches in the `10m_vs_11m` task. (a) With a frozen pre-trained teacher, stronger teacher pre-training does not consistently yield a better student, exposing an initialization gap between centralized teachers and decentralized students. (b) Naively updating the teacher during distillation also fails to consistently improve student performance, particularly with larger pre-training budgets. Bar colors in (a–b) denote the initial reverse KL divergence $D_{\mathrm{KL}}(\pi_{S_0} \| \pi_T)$. (c) With an updating teacher, $D_{\mathrm{KL}}(\pi_S \| \pi_T)$ remains non-negligible throughout training and increases substantially for heavily pre-trained teachers, revealing a persistent and evolving teacher–student policy mismatch.

Recently, on-policy distillation (OPD) has emerged as an effective approach for improving the capabilities of large language models (LLMs), particularly by enabling models to continuously refine their policies using dense supervision during training (Agarwal et al., 2024; Penaloza et al., 2026). OPD can complement sparse environment rewards and substantially accelerate policy learning (Zhao et al., 2026). In this work, we therefore explore introducing OPD into MARL to improve learning efficiency, using a centralized teacher to transform privileged information into dense supervision for decentralized student policies.

Directly incorporating OPD into MARL poses two key challenges: observation mismatch and policy learning mismatch. The observation mismatch arises because the centralized teacher conditions on privileged global information unavailable to decentralized students, making its decisions difficult to reproduce under local execution. The policy learning mismatch arises because the teacher and students evolve simultaneously under different information conditions and rollout distributions, causing their policies to drift apart throughout training. Figure 1(b,c) shows that simply updating the teacher does not eliminate this mismatch. Moreover, the teacher models an autoregressive joint policy over the combinatorial action space, whereas each student acts independently, making direct action distillation difficult.

To address these challenges, we propose **MAOPD**, an OPD framework tailored to MARL. MAOPD jointly trains an autoregressive centralized teacher and decentralized student policies, with each student optimized by both environment rewards and dense teacher supervision. To mitigate observation mismatch, MAOPD introduces agent distillation, which uses a local prefix predictor to transfer privileged teacher representations into locally realizable student representations. To address policy learning mismatch, action distillation factorizes the teacher's joint autoregressive policy into agent-wise conditional targets and applies them to student-generated trajectories. The teacher and students further share most backbone parameters and are optimized concurrently with separate reinforcement-learning objectives, allowing them to co-adapt online and reducing policy drift.

We evaluate MAOPD on 26 environments across RWARE, SMAX, and MPE. MAOPD consistently improves learning efficiency and aggregate performance, while ablations validate the complementary roles of agent distillation and action distillation in this work.

2 Background and Related Work

Dec-POMDPs Cooperative MARL is commonly formulated as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) (Oliehoek & Amato, 2016), represented as $\langle \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, P, r, \{\mathcal{O}_i\}_{i=1}^N, \{\Omega_i\}_{i=1}^N, N \rangle$. Here, $\mathcal{S}$ denotes the underlying environment state space, and $\mathcal{A}_i$ and $\mathcal{O}_i$ represent the action and observation spaces of agent i, respectively. At timestep t, agents execute the joint action $\mathbf{a}_t = (a_t^1, \ldots, a_t^N)$, resulting in a transition $s_{t+1} \sim P(\cdot \mid s_t, \mathbf{a}_t)$ and a shared reward r_t. Due to partial observability, agent i receives only a local observation $o_t^i \in \mathcal{O}_i$ and acts from its local action–observation history $\tau_t^i = (o_0^i, a_0^i, \ldots, a_{t-1}^i, o_t^i)$ according to its decen-

tralized policy $\pi_{\mathrm{S}}^i(a_t^i \mid \tau_t^i)$. For a memoryless policy, τ_t^i reduces to o_t^i. We write $\boldsymbol{\tau}_t = (\tau_t^1, \ldots, \tau_t^N)$ for the joint local histories. During centralized training, privileged context available to a centralized policy is denoted by ξ_t, which may include the environment state, joint observations, or other global interaction information.

On-Policy Distillation On-policy distillation (OPD) applies teacher supervision to student-generated trajectories (Agarwal et al., 2024; Gu et al., 2024; Song & Zheng, 2026), while privileged-information self-distillation additionally gives a shared teacher pathway access to training-only context (Penaloza et al., 2026). In MARL, the teacher may condition on such context ξ_t, while agent i acts only from its local history τ_t^i. A generic agent-wise OPD objective is

$$\mathcal{L}_{\mathrm{OPD}} = \mathbb{E}_{(\xi_t, \boldsymbol{\tau}_t, \mathbf{a}_t) \sim \mathcal{D}_{\mathrm{S}}} \left[\frac{1}{N} \sum_{i=1}^{N} D_{\mathrm{KL}} \left(\pi_{\mathrm{S}}^i(\cdot \mid \tau_t^i) \,\middle\|\, \mathrm{sg}[\pi_{\mathrm{T}}(\cdot \mid \boldsymbol{\tau}_t, \xi_t)] \right) \right], \tag{1}$$

where $\mathcal{D}_{\mathrm{S}}$ contains trajectories generated by the decentralized students and records the privileged context needed only for centralized training. $\pi_{\mathrm{T}}(\cdot \mid \boldsymbol{\tau}_t, \xi_t)$ is the policy of the centralized teacher. The students determine the action distribution on which distillation occurs, whereas the teacher supplies denser targets using information unavailable during decentralized execution. $\mathrm{sg}(\cdot)$ is the stop-gradient operator.

MARL Most MARL methods adopt the CTDE paradigm, which uses centralized information during training while retaining decentralized policies. CTDE methods can be categorized into value-based methods and policy-based methods. Value-based methods (Li et al., 2024b; Qin et al., 2025) learn a global value function that is conditioned on global observations and privileged information (such as state), such as VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), QTran (Son et al., 2019), QPlex (Wang et al., 2021), etc. Policy-based methods used centralized information to guide decentralized policies, such as COMA (Foerster et al., 2018), FOP (Zhang et al., 2021), MAPPO (Yu et al., 2022), MAAC (Iqbal & Sha, 2019), and HAPPO (Kuba et al., 2022).

Some Centralized Training with Centralized Execution (CTCE) methods are proposed. The most notable ones are MAT (Wen et al., 2022) and SABLE (Mahjoub et al., 2025), which are autoregressive methods. Although they achieve state-of-the-art performance, they fall short in settings where agents can access only local observations. Some methods study agent communications, such as (Das et al., 2019; Jiang & Lu, 2018; Li et al., 2025). In our work, agents rely on local observations only.

MARL Distillation The Centralized Teacher with Decentralized Students (CTDS) method (Zhao et al., 2024) and PTDE (Chen et al., 2024) learn a centralized value function that relies on joint observation and priviledged information, and then distill it into a decentralized value function that relies on local observations. MAGPO (Li et al., 2026) adopts an autoregressive policy structure and periodically distill the teacher policy into the student policy. To reduce the policy gap, the teacher policy is periodically copied from the student policy. These methods are off-policy methods that guide the student with the trajectories of teachers, which suffer from the initialization gap and the policy learning challenge.

3 MAOPD: Multi-Agent On-Policy Distillation

3.1 Centralized Teacher and Decentralized Students

Figure 2 summarizes the MAOPD framework. The centralized teacher in Figure 2(b) uses an autoregressive policy $\boldsymbol{\pi}_{\mathrm{T}}$ to represent coordinated joint actions. Given privileged context ξ_t, the teacher factorizes its joint policy according to the decoding order:

$$\boldsymbol{\pi}_{\mathrm{T}}\left(\mathbf{a}_t \mid \boldsymbol{\tau}_t, \xi_t\right) = \prod_{i=1}^{N} \pi_{\mathrm{T}}\left(a_t^i \mid \tau_t^i, \xi_t, \mathbf{a}_t^{<i}\right), \tag{2}$$

where a_t^i is the action of agent i, $\mathbf{a}_t = (a_t^1, \ldots, a_t^N)$ is the joint action, and $\mathbf{a}_t^{<i} = (a_t^1, \ldots, a_t^{i-1})$ contains the actions of preceding agents. Moreover, τ_t^i denotes agent i's local action–observation history, and $\boldsymbol{\tau}_t = (\tau_t^1, \ldots, \tau_t^N)$ collects all agents' histories. Each autoregressive action therefore conditions on the agent's local history, privileged context, and preceding actions, allowing the teacher to capture cross-agent dependencies. We denote its on-policy rollout batch by $\mathcal{D}_{\mathrm{T}} \sim \boldsymbol{\pi}_{\mathrm{T}}$.

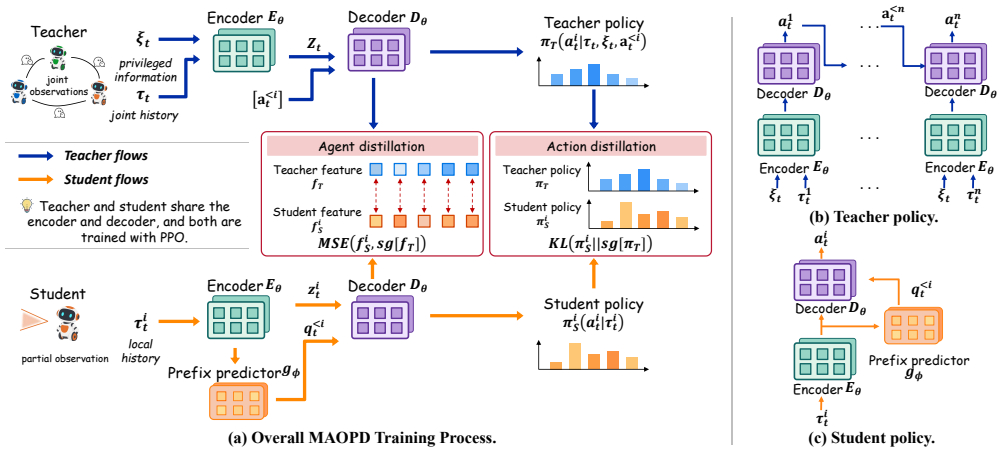


Figure 2: Overview of MAOPD. Agent and action distillation transfers knowledge from a centralized autoregressive teacher to parallel decentralized students that act only from local histories.

By contrast, the students in Figure 2(c) must execute independently and in parallel under partial observability. Their joint policy therefore factorizes as

$$\boldsymbol{\pi}_{\mathrm{S}}\left(\mathbf{a}_t \mid \boldsymbol{\tau}_t\right)=\prod_{i=1}^{N} \pi_{\mathrm{S}}^i\left(a_t^i \mid \tau_t^i\right). \tag{3}$$

All student agents share policy parameters, but each agent selects its action independently and simultaneously using only τ_t^i. We denote the resulting on-policy rollout batch by $\mathcal{D}_{\mathrm{S}} \sim \boldsymbol{\pi}_{\mathrm{S}}$; it is used for the students' reinforcement-learning update and both distillation objectives.

Equations 2 and 3 clarify why directly applying OPD is insufficient. The teacher can use the joint histories $\boldsymbol{\tau}_t$, privileged context ξ_t, and preceding actions $\mathbf{a}_t^{<i}$, whereas each student observes only τ_t^i, creating an *observation mismatch*. In addition, the teacher represents a sequential joint policy, while the students form a parallel product policy and evolve under a different rollout distribution, creating a *policy learning mismatch*. As illustrated in Figure 2(a), MAOPD addresses the former through agent distillation and the latter through action distillation and online Teacher–Student training/co-adaptation.

3.2 Agent Distillation

Matching teacher and student actions alone cannot resolve observation mismatch: the same local history may correspond to different teacher decisions because the teacher can distinguish privileged global contexts and preceding teammate actions. Requiring a student to reconstruct this unavailable context would also violate decentralized execution. MAOPD therefore transfers only its decision-relevant effect. Specifically, each student predicts a compact token q_t^i from its local history and uses it as a locally realizable surrogate for the teacher's autoregressive context:

$$\mathbf{z}_t^i=E_\theta(\tau_t^i), \qquad q_t^i=\mathrm{Softmax}\big(g_\phi(\mathbf{z}_t^i)\big), \tag{4}$$

where E_θ is the encoder shared by the teacher and student paths, and g_ϕ is the prefix predictor shared across student agents. Because q_t^i is a deterministic function of τ_t^i, it introduces no nonlocal information and can be computed independently by every agent.

Let $f_{\mathrm{T}}(\xi_t, \mathbf{a}_t^{<i})$ denote the teacher's decision feature for agent i, and let $f_{\mathrm{S}}^i(\tau_t^i, q_t^i)$ be the corresponding feature produced by the student. Both features are extracted at the same location in the shared decision decoder. For transitions from the student rollout data $\mathcal{D}_{\mathrm{S}}$, MAOPD minimizes

$$\mathcal{L}_{\text{agent}}=\widehat{\mathbb{E}}_{(\xi_t,\boldsymbol{\tau}_t,\mathbf{a}_t)\sim\mathcal{D}_{\mathrm{S}}}\left[\frac{1}{N}\sum_{i=1}^{N}\frac{1}{d}\left\|f_{\mathrm{S}}^i(\tau_t^i,q_t^i)-\mathrm{sg}\big[f_{\mathrm{T}}\big(\boldsymbol{\tau}_t,\xi_t,\mathbf{a}_t^{<i}\big)\big]\right\|_2^2\right], \tag{5}$$

where d is the feature dimension and $\mathrm{sg}(\cdot)$ stops gradients through the teacher. The student is not required to reconstruct the privileged context ξ_t or the preceding teammate actions $\mathbf{a}_t^{<i}$. Instead, the

loss optimizes q_t^i so that the locally computed student feature $f_{\mathrm{S}}^i(\tau_t^i, q_t^i)$ matches the teacher feature $f_{\mathrm{T}}(\boldsymbol{\tau}_t, \xi_t, \mathbf{a}_t^{<i})$. Thus, the prefix predictor learns the decision-relevant coordination information conveyed by the teacher's additional inputs, while the student remains executable from τ_t^i alone.

3.3 Action Distillation and Online Co-adaptation

Action Distillation We first construct agent-wise autoregressive targets. Direct joint-policy distillation scales over the product action space $\mathcal{A}_1 \times \cdots \times \mathcal{A}_N$ and does not provide a separate target for each decentralized agent. MAOPD instead pairs each student policy with the teacher's agent-wise policy at the corresponding decoding position:

$$\pi_{\mathrm{S}}^i\big(a_t^i \mid \tau_t^i\big) \quad \longrightarrow \quad \pi_{\mathrm{T}}\big(a_t^i \mid \boldsymbol{\tau}_t, \xi_t, \mathbf{a}_t^{<i}\big), \qquad i = 1, \ldots, N. \tag{6}$$

This pairing reduces joint-policy distillation to N individual action-space alignments while preserving the coordination encoded by privileged context and preceding agents' decisions.

Although agent-wise autoregressive target resolves the action-space incompatibility, learning on teacher-generated trajectories/rollout would introduce an off-policy gap for the decentralized students. For each transition in $\mathcal{D}_{\mathrm{S}}$, we instead query the teacher with the recorded privileged context ξ_t and the factual prefix $\mathbf{a}_t^{<i}$ composed of actions actually taken by preceding student agents. Thus, the student policy determines the decision points being supervised, while the teacher provides a privileged action distribution at those same points. We minimize

$$\mathcal{L}_{\mathrm{action}} = \widehat{\mathbb{E}}_{(\xi_t, \boldsymbol{\tau}_t, \mathbf{a}_t) \sim \mathcal{D}_{\mathrm{S}}} \left[\frac{1}{N} \sum_{i=1}^{N} w_i D_{\mathrm{KL}} \left(\pi_{\mathrm{S}}^i\big(\cdot \mid \tau_t^i\big) \;\middle\|\; \mathrm{sg}\big[\pi_{\mathrm{T}}\big(\cdot \mid \boldsymbol{\tau}_t, \xi_t, \mathbf{a}_t^{<i}\big)\big] \right) \right], \tag{7}$$

where $w_i \geq 0$ is a nonnegative weight for agent i. The reverse KL divergence aligns complete action distributions at student-induced decisions.

Co-adaptation The teacher and students collect separate on-policy batches, $\mathcal{D}_{\mathrm{T}} \sim \boldsymbol{\pi}_{\mathrm{T}}$ and $\mathcal{D}_{\mathrm{S}} \sim \boldsymbol{\pi}_{\mathrm{S}}$, and optimize their own PPO objectives (Schulman et al., 2017). The teacher learns centralized coordination, while the students learn locally executable behavior and receive both distillation signals on $\mathcal{D}_{\mathrm{S}}$. To limit policy drift without forcing identical policies, the teacher and students share most of the policy backbone, including E_θ and the decision decoder, while retaining distinct inputs and decision paths. This shared representation space allows coordination knowledge to propagate implicitly and couples their online evolution. Detailed PPO objectives are provided in Appendix A.2.

Together, action distillation and co-adaptation address the policy learning mismatch. Action distillation provides compatible agent-wise targets on student-generated trajectories, while co-adaptation couples the evolving teacher and students through concurrent learning and shared representations, thereby reducing policy drift during online transfer.

3.4 Training Overall Objective

MAOPD combines reward-driven teacher and student learning with the two complementary distillation objectives:

$$\mathcal{L}_{\mathrm{MAOPD}} = \mathcal{L}_{\mathrm{PPO,T}} + \mathcal{L}_{\mathrm{PPO,S}} + \alpha \mathcal{L}_{\mathrm{agent}} + \beta \mathcal{L}_{\mathrm{action}}, \tag{8}$$

where α and β control agent and action distillation, respectively.

A fixed teacher order could make an agent repeatedly receive supervision from the same decoding position and preceding teammates. We therefore randomize the agent order during training; the main text omits the permutation for clarity, while the permutation-aware objectives are provided in Appendix A.1. This randomization affects only teacher queries and distillation pairings and introduces no ordering requirement at execution. The complete training procedure is summarized in Algorithm 1 in Appendix A.

At execution, only the parameter-shared student policies $\{\pi_{\mathrm{S}}^i\}_{i=1}^N$ and their prefix predictor are retained, allowing all agents to compute q_t^i and select actions in parallel from local histories τ_t^i alone.

4 EXPERIMENTS

We evaluate MAOPD against strong MARL baselines on 26 tasks from three benchmarks. Section 4.1 describes the environments, baselines, and evaluation protocol. The main results in Section 4.2 show that MAOPD performs strongly under matched environment-interaction budgets, with particularly clear gains in sparse-reward settings. In Section 4.3, we verify the complementary contributions of agent and action distillation and show that they effectively address the observation and policy learning mismatches. We also compare alternative implementations and design choices for both distillation components, for which our proposed designs achieve consistently strong results. Finally, the sensitivity analysis shows stable performance across a range of distillation weights, indicating that MAOPD does not require precise hyperparameter tuning.

4.1 EXPERIMENTAL SETUP

Environments. We evaluate MAOPD on 26 tasks from three cooperative MARL benchmark suites. RobotWarehouse (RWARE) (Papoudakis et al., 2021) comprises 13 partially observable warehouse-logistics tasks in which robots must coordinate to deliver requested shelves. Rewards are obtained only for successful deliveries, creating a hard-exploration setting with sparse feedback, particularly in larger and harder layouts. The JAX-based StarCraft Multi-Agent Challenge (SMAX) (Rutherford et al., 2024) contains 10 combat tasks that require decentralized agents to coordinate heterogeneous units against an opposing team. Finally, we consider three MPE `simple_spread` variants with 3, 5, and 10 agents (Lowe et al., 2017), where agents must cooperatively cover landmarks while avoiding collisions. Together, these suites test coordination under sparse rewards, partial observability, diverse interaction dynamics, and increasing team sizes.

Baselines. We group the baselines by how they use centralized information. Standard MARL baselines include QMIX (Rashid et al., 2018), which learns decentralized utilities through centralized value factorization; MAPPO (Yu et al., 2022) and HAPPO (Kuba et al., 2022), which optimize decentralized policies under centralized training; and MAT (Wen et al., 2022), an autoregressive MARL method. We further compare with two policy distillation methods. CTDS (Zhao et al., 2024) directly distills the behavior of a centralized teacher into decentralized students, whereas MAGPO (Li et al., 2026) constrains a centralized guide to remain compatible with decentralized learners.

Evaluation protocol. Unless otherwise specified, MAOPD uses SABLE (Mahjoub et al., 2025) as the autoregressive teacher backbone. All methods use the same total environment-interaction budget. MAOPD divides this budget equally between teacher and student rollouts and counts both on the horizontal axis. Each algorithm is trained with 10 independent seeds per task. Training is conducted for 20 million environment steps, with 122 evenly spaced evaluation checkpoints. At each checkpoint, we record the task-specific metrics (e.g., mean episode return and win rate) over 32 evaluation episodes. For results, we report the mean and 95% confidence intervals. The hyperparameters are detailed in Appendix C.

4.2 MAIN RESULTS

Table 1 summarizes final performance across all 26 tasks. MAOPD ranks first or tied for first on 22 tasks, including 12 of 13 RWARE tasks, 7 of 10 SMAX tasks, and all 3 MPE tasks. These results show that its gains are broad across benchmark suites rather than driven by a small subset of tasks.

The gains are particularly pronounced on RWARE, where rewards are obtained only after successful deliveries. Figure 3 complements the final-score comparison with learning curves on three representative tasks. Under the same total interaction budget, MAOPD achieves the highest final return on all three. On `tiny-4ag` and `medium-4ag-hard`, it continues improving after several baselines begin to plateau, while on `tiny-4ag-hard`, it ultimately surpasses the initially stronger baselines. We further analyze sample efficiency in Appendix Table 2. MAOPD consistently requires fewer environment interactions to reach the target and attains a higher target within the logged budget than all competing methods. Together, these results provide strong evidence that dense teacher supervision improves both exploration efficiency and final decentralized performance under sparse environmental feedback.

Table 1: Performance across RWARE, SMAX, and MPE over the final five regular evaluations. RWARE and MPE report episode return; SMAX reports test win rate (%). Higher is better for all metrics. Values are means with sample standard deviations shown beneath the means.

Task	MAPPO	HAPPO	MAGPO	CTDS	QMIX	MAT	Ours
RWARE Episode return (↑)							
tiny-2ag	12.18 ±6.17	10.80 ±2.39	17.98 ±1.24	11.37 ±1.95	0.00 ±0.00	16.48 ±0.88	**20.93** ±0.41
tiny-2ag-hard	13.53 ±0.73	9.18 ±3.23	16.42 ±0.92	8.28 ±1.29	0.00 ±0.00	6.25 ±5.92	**16.92** ±0.44
tiny-4ag	26.36 ±3.20	18.05 ±1.92	30.57 ±3.07	13.82 ±1.49	2.29 ±0.71	25.78 ±1.79	**38.53** ±3.86
tiny-4ag-hard	19.48 ±1.68	16.65 ±2.69	21.04 ±1.55	10.85 ±1.46	1.08 ±0.76	21.21 ±0.73	**24.07** ±2.40
small-4ag	10.73 ±1.00	9.66 ±2.52	14.73 ±1.32	10.82 ±0.75	0.19 ±0.43	11.97 ±1.61	**14.88** ±1.14
small-4ag-hard	9.39 ±0.48	6.49 ±1.14	11.51 ±0.65	7.50 ±0.84	0.07 ±0.15	3.34 ±4.36	**11.63** ±0.69
medium-4ag	7.59 ±3.33	4.03 ±0.59	11.65 ±1.24	8.09 ±0.48	0.04 ±0.12	1.17 ±2.67	**13.04** ±1.61
medium-4ag-hard	2.86 ±3.11	3.73 ±0.53	6.03 ±2.92	4.86 ±0.87	0.00 ±0.00	0.00 ±0.00	**9.26** ±0.46
medium-6ag	12.33 ±0.65	4.96 ±0.79	8.24 ±8.84	8.39 ±1.18	0.29 ±0.61	10.67 ±2.67	**13.56** ±2.28
large-4ag	2.99 ±2.33	0.59 ±0.56	**7.95** ±1.39	4.90 ±0.59	0.00 ±0.00	0.00 ±0.00	7.88 ±0.43
large-4ag-hard	0.00 ±0.00	0.00 ±0.00	3.85 ±2.18	2.25 ±1.57	0.00 ±0.00	0.00 ±0.00	**4.14** ±0.29
large-8ag	8.41 ±0.89	0.00 ±0.01	10.63 ±0.70	7.98 ±0.63	0.00 ±0.00	0.00 ±0.01	**11.62** ±0.61
large-8ag-hard	3.49 ±3.40	0.00 ±0.00	7.78 ±3.01	4.26 ±2.98	0.06 ±0.18	0.00 ±0.00	**8.64** ±0.19
SMAX Test win rate (%) (↑)							
2s3z	99.56 ±0.42	98.75 ±1.14	**100.00** ±0.00	99.31 ±0.62	99.14 ±0.79	99.75 ±0.60	99.50 ±0.52
3s5z	99.69 ±0.44	97.50 ±1.59	**100.00** ±0.00	98.88 ±1.31	99.30 ±0.73	99.56 ±0.72	**100.00** ±0.00
3s5z_vs_3s6z	92.94 ±4.44	54.50 ±12.85	98.50 ±1.75	86.75 ±8.14	74.89 ±8.10	94.19 ±1.96	**99.38** ±0.44
3s_vs_5z	98.38 ±0.67	94.81 ±1.22	98.69 ±1.12	95.88 ±1.07	94.44 ±1.85	96.62 ±1.80	**98.75** ±0.77
6h_vs_8z	99.38 ±1.18	99.38 ±0.42	**99.94** ±0.20	99.50 ±0.49	71.87 ±30.40	99.38 ±0.72	99.62 ±0.56
10m_vs_11m	41.38 ±7.05	16.12 ±4.46	91.25 ±14.72	76.75 ±25.09	95.63 ±3.34	95.88 ±8.71	**98.38** ±1.37
5m_vs_6m	28.75 ±38.72	4.06 ±1.96	11.31 ±9.25	16.94 ±23.59	79.60 ±16.19	**88.25** ±20.52	54.38 ±34.94
smacv2_5_units	76.75 ±2.53	70.25 ±3.05	84.38 ±3.58	80.19 ±3.24	78.31 ±2.90	80.31 ±2.09	**88.62** ±3.17
smacv2_10_units	78.50 ±2.67	48.69 ±4.25	76.56 ±2.29	64.06 ±3.91	70.88 ±3.09	69.19 ±2.41	**82.62** ±2.70
27m_vs_30m	86.56 ±7.86	79.88 ±2.99	87.81 ±4.93	**98.50** ±0.94	57.98 ±24.79	90.31 ±5.36	**98.50** ±1.22
MPE Episode return (↑)							
simple_spread_3ag	−6.75 ±0.23	−6.72 ±0.53	−6.04 ±0.28	−6.34 ±0.36	−9.33 ±1.28	−11.26 ±1.06	**−5.13** ±0.18
simple_spread_5ag	−22.49 ±0.40	−23.29 ±0.64	−18.32 ±0.36	−20.08 ±0.50	−30.59 ±1.93	−31.08 ±1.03	**−16.68** ±0.14
simple_spread_10ag	−41.56 ±0.40	−43.27 ±0.54	−40.19 ±0.54	−39.97 ±0.46	−55.93 ±3.16	−55.40 ±0.90	**−38.81** ±0.62

Appendix Table 3 further shows that, beyond the SABLE-based implementation used in our main experiments, MAOPD can also be instantiated with MAT and improves over vanilla MAT on most RWARE tasks. This demonstrates that MAOPD is applicable to different autoregressive policy architectures rather than being specific to SABLE.

Beyond final performance and architectural generality, we directly examine whether MAOPD resolves the policy learning mismatch that motivates our method. Appendix Section B.1.3 closes the loop

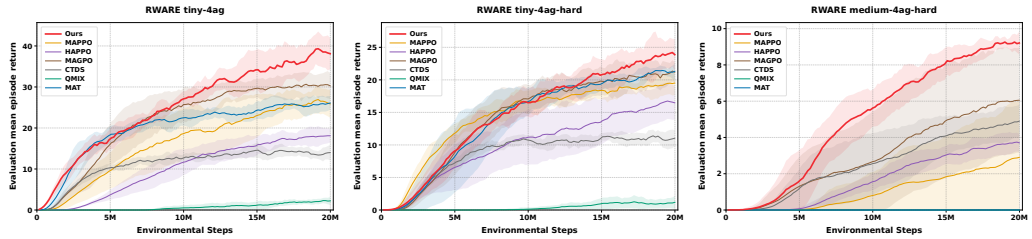


Figure 3: Learning curves on three representative RWARE tasks under matched total environment-interaction budgets: `tiny-4ag` (left), `tiny-4ag-hard` (middle), and `medium-4ag-hard` (right).

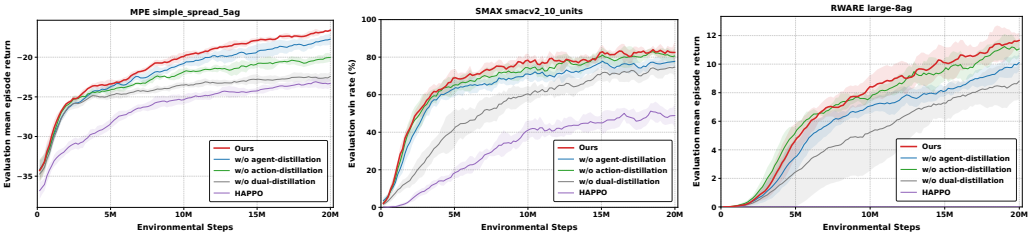


Figure 4: Ablations of the two distillation modules on MPE (left), SMAX (middle), and RWARE (right). We remove agent distillation, action distillation, or both (w/o dual-distillation) while retaining the remaining training framework. Higher is better.

with Figure 1, showing that MAOPD achieves the lowest Teacher–Student KL divergence among the compared methods while learning faster and attaining higher performance than MAGPO and CTDS.

4.3 Ablation Studies

We first ablate the two objectives corresponding to the method components in Secs. 3.2 and 3.3, and then examine the design choices used to implement effective online transfer.

Agent/Action Distillation Ablation. Figure 4 shows that both distillation objectives contribute to the final decentralized policy. Removing agent distillation degrades performance across all three suites, especially SMAX and RWARE, supporting its role in transferring privileged features to local decision representations and mitigating observation mismatch. Removing action distillation also hurts performance, most clearly on MPE, confirming the benefit of direct policy supervision under the student's information constraint. Removing both produces the largest degradation, demonstrating the complementary roles of the two objectives.

Implementation for Agent/Action Distillation. Figure 5 validates the mechanisms developed for the two distillation methods. For agent-distillation, as shown in the left part of Figure 5, removing the prefix predictor hurts performance, especially with more agents, confirming that locally inferred context mitigates observation mismatch. For action-distillation, removing either teacher- or student-side PPO learning, or the parameter-sharing of encoder/decoder, also degrades performance: online co-learning maintains useful supervision, while parameter-sharing limits teacher–student drift. To further examine whether these mechanisms are necessary, we consider naive OPSD, which shares teacher–student parameters but removes teacher-side training and the prefix predictor, retaining only action-level KL distillation. Figure 6 shows that naive OPSD underperforms MAOPD and can fall below HAPPO, demonstrating that these multi-agent-specific mechanisms are essential for effective transfer.

Distillation choices and hyperparameter sensitivity. Appendix Figure. 11 further shows that distillation on student-generated trajectories consistently outperforms distillation on teacher rollouts, while the reverse-KL objective used by MAOPD outperforms forward KL. These results support grounding teacher guidance in the student's behavioral distribution and aligning the student toward the teacher under its partial observation constraint. The middle and right columns of Figure. 6 vary the

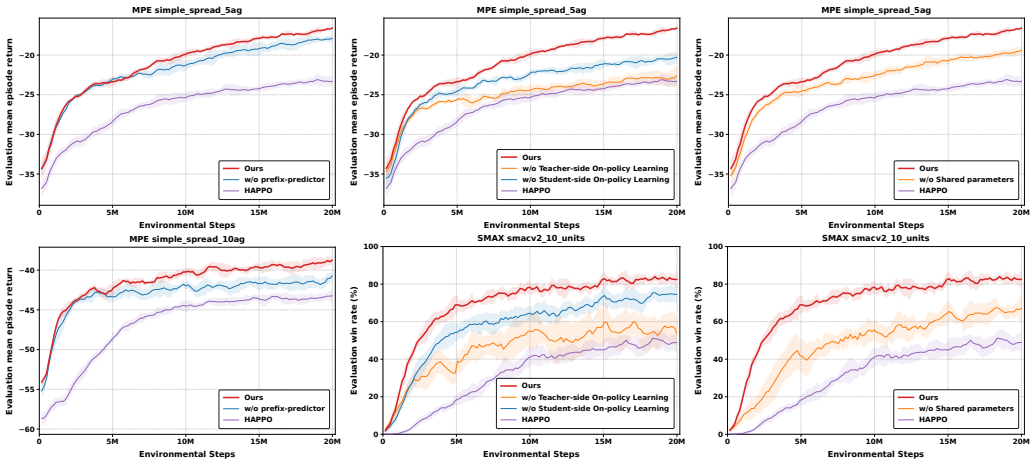


Figure 5: Ablations of the mechanisms used to address the observation and policy learning mismatches. The left column removes the local prefix predictor; the middle column removes either Teacher-side or Student-side on-policy learning; and the right column removes Teacher–Student backbone parameter sharing. Higher is better.

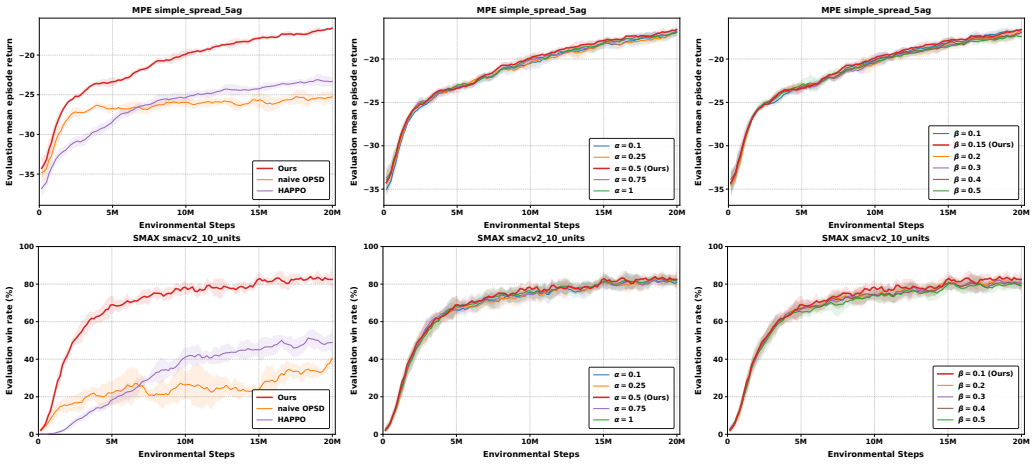


Figure 6: Comparison with naive OPSD and sensitivity to the distillation weights. Here, naive OPSD shares teacher–student parameters but removes Teacher-side training and the prefix predictor, retaining only action-level KL distillation. The left column compares MAOPD with a naive OPSD; the middle and right columns vary the agent-distillation weight α and action-distillation weight β, respectively. The top and bottom rows report results on MPE `simple_spread_5ag` and SMAX `smacv2_10_units`. Higher is better.

agent-distillation weight α and action-distillation weight β, respectively. Performance remains stable across the tested ranges, with the default settings consistently among the strongest configurations on both tasks. Thus, MAOPD's gains arise from its distillation design rather than sensitivity to finely tuned distillation weights.

5 Conclusion

We introduced MAOPD, an online policy self-distillation framework that transfers coordination knowledge from a centralized autoregressive teacher to decentralized parallel students. MAOPD decomposes joint teacher guidance into agent-specific conditional targets, transfers privileged decision features through agent distillation, and combines student-grounded action distillation with shared-parameter online learning to mitigate observation and policy distribution mismatches. Across 26 tasks from RWARE, SMAX, and MPE, MAOPD achieves strong aggregate performance and improved learning efficiency under matched environment-interaction budgets, while ablations support the complementary contributions of its components. At execution, only the student policy is retained, enabling fully decentralized and parallel decision-making without privileged information.

AI USE STATEMENT

In this work, we used generative AI tools to polish the language of the manuscript, assist with literature retrieval and discovery, and support code development and the execution of computational experiments. All AI-assisted text, code, and experimental outputs were reviewed and verified by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide the complete MAOPD training procedure, including the permutation-aware distillation objectives, concurrent teacher–student PPO learning, and backbone-specific implementation details, in Appendix A. All experiments are conducted on publicly available benchmark environments from RWARE, SMAX, and MPE. Default training configurations, task-wise hyperparameter search spaces, and baseline settings are reported in Appendix C.

ETHICS STATEMENT

This work focuses on developing and evaluating multi-agent reinforcement learning methods in simulated environments. Our experiments use publicly available benchmark environments and do not involve human subjects, personally identifiable information, or sensitive personal data. The proposed method is evaluated solely on simulated warehouse coordination, multi-agent combat, and cooperative navigation tasks. We do not identify specific ethical concerns related to privacy, fairness, or human-subject research arising from the experiments conducted in this work. Any future deployment in real-world multi-agent or robotic systems would require independent safety validation.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *ICLR*, 2024.

Aakriti Agrawal, Amrit Singh Bedi, and Dinesh Manocha. RTAW: an attention inspired reinforcement learning method for multi-robot task allocation in warehouse environments. In *ICRA*, 2023.

Dong Chen, Mohammad R. Hajidavalloo, Zhaojian Li, Kaian Chen, Yongqiang Wang, Longsheng Jiang, and Yue Wang. Deep multi-agent reinforcement learning for highway on-ramp merging in mixed traffic. *IEEE Trans. Intell. Transp. Syst.*, 24(11):11623–11638, 2023.

Yiqun Chen, Hangyu Mao, Jiaxin Mao, Shiguang Wu, Tianle Zhang, Bin Zhang, Wei Yang, and Hongxing Chang. Ptde: Personalized training with distilled execution for multi-agent reinforcement learning. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pp. 31–39, 2024. doi: 10.24963/ijcai.2024/4. URL https://doi.org/10.24963/ijcai.2024/4.

Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. Tarmac: Targeted multi-agent communication. In *ICML*, volume 97, pp. 1538–1546, 2019.

Xiao Du, Ting Wang, Qiang Feng, Chenhui Ye, Tao Tao, Lu Wang, Yuanming Shi, and Mingsong Chen. Multi-agent reinforcement learning for dynamic resource management in 6g in-x subnetworks. *IEEE Trans. Wirel. Commun.*, 22(3):1900–1914, 2023.

Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *AAAI*, pp. 2974–2982, 2018.

Zhen Gao, Lei Yang, and Yu Dai. Large-scale computation offloading using a multi-agent reinforcement learning in heterogeneous multi-access edge computing. *IEEE Trans. Mob. Comput.*, 22(6): 3425–3443, 2023.

Zhen Gao, Lei Yang, and Yu Dai. Large-scale cooperative task offloading and resource allocation in heterogeneous MEC systems via multiagent reinforcement learning. *IEEE Internet Things J.*, 11 (2):2303–2321, 2024.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *ICLR*, 2024.

Ziyang Guo, Zhenyu Chen, Peng Liu, Jianjun Luo, Xun Yang, and Xinghua Sun. Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks. *IEEE J. Sel. Areas Commun.*, 40(5):1587–1599, 2022.

Yitian Hong, Yaochu Jin, and Yang Tang. Rethinking individual global max in cooperative multi-agent reinforcement learning. In *NeurIPS*, 2022.

Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *ICML*, volume 97, pp. 2961–2970, 2019.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. In *NeurIPS*, pp. 7265–7275, 2018.

Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *ICLR*, 2022.

Xinran Li, Zifan Liu, Shibo Chen, and Jun Zhang. Individual contributions as intrinsic exploration scaffolds for multi-agent reinforcement learning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28387–28402. PMLR / OpenReview.net, 2024a. URL https://proceedings.mlr.press/v235/li24aw.html.

Xinran Li, Ling Pan, and Jun Zhang. Kaleidoscope: Learnable masks for heterogeneous multi-agent reinforcement learning. In *NeurIPS*, 2024b.

Xinran Li, Xiaolu Wang, Chenjia Bai, and Jun Zhang. Exponential topology-enabled scalable communication in multi-agent reinforcement learning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=CL3U0GxFRD.

Yueheng Li, Guangming Xie, and Zongqing Lu. Multi-agent guided policy optimization. In *ICLR*, 2026.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NeurIPS*, 2017.

Ziyuan Ma, Yudong Luo, and Hang Ma. Distributed heuristic multi-agent path finding with communication. In *ICRA*, 2021.

Omayma Mahjoub, Sasha Abramowitz, Ruan John De Kock, Wiem Khlifi, Simon Verster Du Toit, Jemma Daniel, Louay Ben Nessir, Louise Beyers, Juan Claude Formanek, Liam Clark, and Arnu Pretorius. Sable: a performant, efficient and scalable sequence model for MARL. In *ICML*, 2025.

Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. Springer Briefs in Intelligent Systems. Springer, 2016.

Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *NeurIPS*, 2021.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. In *ICML*, 2026.

Haoyuan Qin, Zhengzhu Liu, Chenxing Lin, Chennan Ma, Songzhu Mei, Siqi Shen, and Cheng Wang. Gradps: Resolving futile neurons in parameter sharing network for multi-agent reinforcement learning. In *ICML*, 2025.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*, 2018.

Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Garðar Ingvarsson, Timon Willi, Ravi Hammond, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Lange, Shimon Whiteson, Bruno Lacerda, Nick Hawes, Tim Rocktäschel, Chris Lu, and Jakob Foerster. JaxMARL: Multi-agent RL environments and algorithms in JAX. In *NeurIPS*, 2024.

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

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Hostallero, and Yung Yi. QTRAN: learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *ICML*, pp. 5887–5896, 2019.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. In *arXiv*, 2026.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *AAMAS*, pp. 2085–2087, 2018.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. In *ICLR*, 2021.

Muning Wen, Jakub Grudzien Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. In *NeurIPS*, 2022.

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. In *NeurIPS*, 2022.

Jiawei Zhang, Cheng Chang, Xianlin Zeng, and Li Li. Multi-agent drl-based lane change with right-of-way collaboration awareness. *IEEE Trans. Intell. Transp. Syst.*, 24(1):854–869, 2023.

Tianhao Zhang, Yueheng Li, Chen Wang, Guangming Xie, and Zongqing Lu. FOP: factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12491–12500. PMLR, 2021. URL http://proceedings.mlr.press/v139/zhang21m.html.

Jian Zhao, Xunhan Hu, Mingyu Yang, Wengang Zhou, Jiangcheng Zhu, and Houqiang Li. CTDS: Centralized teacher with decentralized student for multiagent reinforcement learning. *IEEE Transactions on Games*, 16(1):140–150, 2024.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *ICML*, 2026.

Appendix

A MAOPD Details

A.1 Randomized Agent Ordering during Distillation

The main text omits randomized agent ordering from the notation for readability. When computing the two distillation objectives on a student transition, a permutation $\sigma = (\sigma(1), \ldots, \sigma(N))$ is sampled uniformly from the set of all agent permutations. The agent at teacher decoding position i is $\sigma(i)$, and its factual student-action prefix is

$$\mathbf{a}_t^{\sigma(<i)} = \left(a_t^{\sigma(1)}, \ldots, a_t^{\sigma(i-1)}\right). \tag{A.1}$$

Let ξ_t^σ denote the privileged context after all of its agent-indexed components have been reordered according to the same permutation; global components without an agent axis remain unchanged. Agent-specific action masks are reordered in the same way. The permutation-aware agent-distillation objective is

$$\mathcal{L}_{\text{agent}} = \widehat{\mathbb{E}}_{\substack{(\xi_t, \boldsymbol{\tau}_t, \mathbf{a}_t) \sim \mathcal{D}_{\text{S}} \\ \sigma \sim \text{Unif}(\mathfrak{S}_N)}} \left[\frac{1}{N} \sum_{i=1}^{N} \frac{1}{d} \left\| f_{\text{S}}^{\sigma(i)}\left(\tau_t^{\sigma(i)}, q_t^{\sigma(i)}\right) - \text{sg}\left[f_{\text{T}}\left(\xi_t^\sigma, \mathbf{a}_t^{\sigma(<i)}\right)\right] \right\|_2^2 \right]. \tag{A.2}$$

Similarly, the permutation-aware action-distillation objective is

$$\mathcal{L}_{\text{action}} = \widehat{\mathbb{E}}_{\substack{(\xi_t, \boldsymbol{\tau}_t, \mathbf{a}_t) \sim \mathcal{D}_{\text{S}} \\ \sigma \sim \text{Unif}(\mathfrak{S}_N)}} \left[\frac{1}{N} \sum_{i=1}^{N} w_i D_{\text{KL}}\left(\pi_{\text{S}}^{\sigma(i)}\left(\cdot \mid \tau_t^{\sigma(i)}, q_t^{\sigma(i)}\right) \,\middle\|\, \text{sg}\left[\pi_{\text{T}}\left(\cdot \mid \xi_t^\sigma, \mathbf{a}_t^{\sigma(<i)}\right)\right] \right) \right]. \tag{A.3}$$

The prefix in Eqs. A.2 and A.3 contains actions recorded in the student trajectory rather than actions resampled from the teacher. Neither σ nor $\mathbf{a}_t^{\sigma(<i)}$ is provided to the student predictor; they only determine the teacher's autoregressive context and the corresponding teacher–student pairing. Randomizing the order prevents a fixed agent from always occupying the same decoding position or observing supervision based on the same set of preceding teammates. Consequently, each agent is trained against teacher targets formed from different positions and prefix lengths. This randomization is used only during training: decentralized execution does not require an agent order, privileged context, or teammate actions.

A.2 Concurrent Teacher–Student PPO Learning

The centralized teacher and decentralized students are optimized concurrently from separate on-policy trajectory batches. Let $X \in \{\text{T}, \text{S}\}$ index the teacher or student path, with $\mathcal{D}_X$ denoting its corresponding rollout batch. Their PPO objectives are written uniformly as

$$\mathcal{L}_{\text{PPO},X} = -\widehat{\mathbb{E}}_{\mathcal{D}_X} \left[\frac{1}{N} \sum_{i=1}^{N} \min\left(\rho_{t,i}^X \widetilde{A}_{t,i}^X, \text{clip}\left(\rho_{t,i}^X, 1-\epsilon, 1+\epsilon\right) \widetilde{A}_{t,i}^X \right) \right] + c_V \mathcal{L}_{V,X} - c_H \mathcal{H}_X, \tag{A.4}$$

where $\rho_{t,i}^X$ and $\widetilde{A}_{t,i}^X$ are the per-agent probability ratio and advantage, respectively; ϵ is the clipping threshold, $\mathcal{L}_{V,X}$ is the value-regression loss, and $\mathcal{H}_X$ is the policy entropy. Advantages and return targets are computed separately on $\mathcal{D}_{\text{T}}$ and $\mathcal{D}_{\text{S}}$. Thus, the teacher learns from centralized rollouts with privileged information, while the students learn from decentralized rollouts under local information constraints. The two learning processes interact through the shared backbone and the distillation objectives rather than by exchanging off-policy PPO targets.

A.3 Training Algorithm

Algorithm 1 summarizes the joint on-policy learning and distillation procedure of MAOPD. The shared actor–critic parameters are denoted by θ, while the agent predictor g_ϕ is specific to the student path. Teacher targets used by both distillation objectives are recomputed with the current parameters and detached before backpropagation.

Algorithm 1 MAOPD Joint On-Policy Learning and Distillation

Require: Shared actor–critic parameters θ; student predictor parameters ϕ; distillation weights α, β; position weights $\{w_i\}_{i=1}^{N}$; training iterations K; PPO update epochs E
Ensure: Decentralized student policy $\boldsymbol{\pi}_{\mathrm{S}}$

for $k = 0, \ldots, K-1$ **do**
  Snapshot the behavior policies $\boldsymbol{\pi}_{\mathrm{T},k}$ and $\boldsymbol{\pi}_{\mathrm{S},k}$
  Collect teacher trajectories $\mathcal{D}_{\mathrm{T}}^{k} \sim \boldsymbol{\pi}_{\mathrm{T},k}$ using privileged context ξ_t
  Collect student trajectories $\mathcal{D}_{\mathrm{S}}^{k} \sim \boldsymbol{\pi}_{\mathrm{S},k}$ using only local histories $\boldsymbol{\tau}_t$
  Record ξ_t and $\mathbf{a}_t$ along student trajectories for training-only teacher queries
  Compute returns and advantages separately for $\mathcal{D}_{\mathrm{T}}^{k}$ and $\mathcal{D}_{\mathrm{S}}^{k}$
  for $e = 1, \ldots, E$ **do**
    Sample minibatches $\mathcal{B}_{\mathrm{T}} \subset \mathcal{D}_{\mathrm{T}}^{k}$ and $\mathcal{B}_{\mathrm{S}} \subset \mathcal{D}_{\mathrm{S}}^{k}$
    Compute $\mathcal{L}_{\mathrm{PPO},T}^{k}$ on $\mathcal{B}_{\mathrm{T}}$ using $\boldsymbol{\pi}_{\mathrm{T},k}$
    Compute $\mathcal{L}_{\mathrm{PPO},S}^{k}$ on $\mathcal{B}_{\mathrm{S}}$ using $\boldsymbol{\pi}_{\mathrm{S},k}$
    Initialize $\mathcal{L}_{\mathrm{agent}}^{k} \leftarrow 0$ and $\mathcal{L}_{\mathrm{action}}^{k} \leftarrow 0$
    for all $(\xi_t, \boldsymbol{\tau}_t, \mathbf{a}_t) \in \mathcal{B}_{\mathrm{S}}$ **do**
      Sample a uniform agent permutation $\sigma = (\sigma(1), \ldots, \sigma(N))$
      for $i = 1, \ldots, N$ **do**
        $\mathbf{a}_t^{\sigma(<i)} \leftarrow (a_t^{\sigma(1)}, \ldots, a_t^{\sigma(i-1)})$
        $\mathbf{z}_t^{\sigma(i)} \leftarrow E_\theta(\tau_t^{\sigma(i)})$; compute $q_t^{\sigma(i)}$ with the student predictor g_ϕ
        Evaluate $f_{\mathrm{S}}^{\sigma(i)}(\tau_t^{\sigma(i)}, q_t^{\sigma(i)})$ and $\pi_{\mathrm{S}}^{\sigma(i)}(\cdot \mid \tau_t^{\sigma(i)}, q_t^{\sigma(i)})$
        Query the teacher with ξ_t and the *factual student-action prefix* $\mathbf{a}_t^{\sigma(<i)}$
        Obtain $f_{\mathrm{T}}(\xi_t, \mathbf{a}_t^{\sigma(<i)})$ and $\pi_{\mathrm{T}}(\cdot \mid \xi_t, \mathbf{a}_t^{\sigma(<i)})$
        Accumulate $\mathcal{L}_{\mathrm{agent}}^{k}$ using Eq. A.2 with the teacher feature detached
        Accumulate $\mathcal{L}_{\mathrm{action}}^{k}$ using Eq. A.3 with the teacher distribution detached
      end for
    end for
    Normalize both distillation losses over $\mathcal{B}_{\mathrm{S}}$ and the N decoding positions
    $\mathcal{L}^{k} \leftarrow \mathcal{L}_{\mathrm{PPO},T}^{k} + \mathcal{L}_{\mathrm{PPO},S}^{k} + \alpha\mathcal{L}_{\mathrm{agent}}^{k} + \beta\mathcal{L}_{\mathrm{action}}^{k}$
    Update (θ, ϕ) by minimizing $\mathcal{L}^{k}$
  end for
end for
return $\boldsymbol{\pi}_{\mathrm{S}}$; discard the teacher and all training-only privileged inputs

The teacher query uses actions recorded in the student trajectory, not actions newly sampled from the teacher. Thus, both distillation objectives supervise decisions induced by the decentralized student. Uniform position weights, $w_i = 1$, are used unless stated otherwise. For compactness, ξ_t in Algorithm 1 implicitly includes consistent reordering of any agent-indexed teacher inputs under σ; the explicit notation ξ_t^{σ} is defined above.

A.4 Parallel Execution of the Decentralized Student

Although the teacher and student share backbone parameters during training, they use separate computation paths. At execution, the teacher path, centralized critic outputs, and privileged context ξ_t are discarded. The N agents evaluate the same student parameters independently, each using only its local history τ_t^i.

Local student computation. Following the notation of Sec. 3.2, agent i computes $\mathbf{z}_t^i = E_\theta(\tau_t^i)$ and the single predicted token q_t^i according to Eq. 4. For continuous-action domains, the corresponding implementation uses $q_t^i = \tanh(g_\phi(\mathbf{z}_t^i))$. Thus, q_t^i is a deterministic function of τ_t^i and introduces no nonlocal information; it is suppressed in the policy notation $\pi_{\mathrm{S}}^i(\cdot \mid \tau_t^i)$.

For each agent, the student constructs an action-token stream and an observation-conditioning stream of fixed length:

$$X_t^i = [b, \iota(q_t^i)], \qquad U_t^i = [\mathbf{0}, \mathbf{z}_t^i], \tag{A.5}$$

where b is the beginning-of-sequence token and ι maps q_t^i to the backbone's action-token format. The shared causal decoder evaluates

$$(Y_t^i, s_t^i) = D_\theta(X_t^i, U_t^i; s_{t-1}^i), \tag{A.6}$$

where s^i_{t-1} is optional local decoder memory. The shared action head uses $Y^i_{t,2}$ to produce $\pi^i_{\mathrm{S}}(\cdot \mid \tau^i_t)$, with a local action mask when required. Both decoder inputs are available before action sampling, so this path contains no dependence on another agent's current action.

Backbone-specific implementations. MAOPD is not tied to a particular autoregressive architecture and can be instantiated with either SABLE or MAT without changing its distillation objectives. In the SABLE-based implementation, each student agent maintains its own encoder, decoder self-retention, and decoder cross-retention states. The decoder processes the BOS token and q^i_t in two successive recurrent updates, each batched across all agents. In the MAT-based implementation, the student encoder processes each agent's local input separately, with a local GRU maintaining its history. The two decoder tokens are evaluated in a single causal-attention forward pass, and no recurrent decoder state s^i_t is required.

In both implementations, the agent axis is folded into the batch axis. For a batch of B environments, the encoder input therefore has shape $[BN, 1, d_o]$, enabling all student agents to be evaluated in parallel.

Parallelism and complexity. Because $\mathbf{z}^i_t$, q^i_t, and s^i_t depend only on τ^i_t, the student realizes the decentralized factorization in Eq. 3. All agents can therefore sample their actions simultaneously. In contrast to the teacher's N successive conditional decisions, the student's sequential decoding depth with respect to the number of agents is $O(1)$. For fixed architecture, observation and action dimensions, and per-agent memory size, total actor computation and state storage scale as $O(N)$. Actual latency additionally depends on batching and hardware parallelism; temporal state updates remain causal.

B Further experimental results

This section provides additional empirical evidence from three perspectives. Section B.1 presents task-level learning curves, sample-efficiency results, Teacher–Student policy alignment, and a MAT-based implementation. Together, these results show that MAOPD improves decentralized learning efficiency, reduces policy mismatch, and generalizes beyond its default SABLE backbone. Section B.2 examines teacher information and alternative distillation designs. It further shows that MAOPD does not substantially compromise teacher capability: the jointly trained teacher remains close to the fully centralized SABLE policy and stronger than the MAGPO teacher/guider (Section B.2.2). Moreover, the decentralized students closely approach their strong teacher despite acting in parallel from local information alone (Section B.2.3). Finally, Section B.3 reports the training-cost comparison with MAGPO, showing that MAOPD achieves a lower recorded training time on most tasks.

B.1 Additional Performance Analyses

B.1.1 Task-level learning curves.

Figure 7 provides additional task-level learning curves across RWARE, MPE, and SMAX under matched total interaction budgets. MAOPD evaluates only the decentralized parallel Student, and its horizontal axis counts both Teacher and Student interactions.

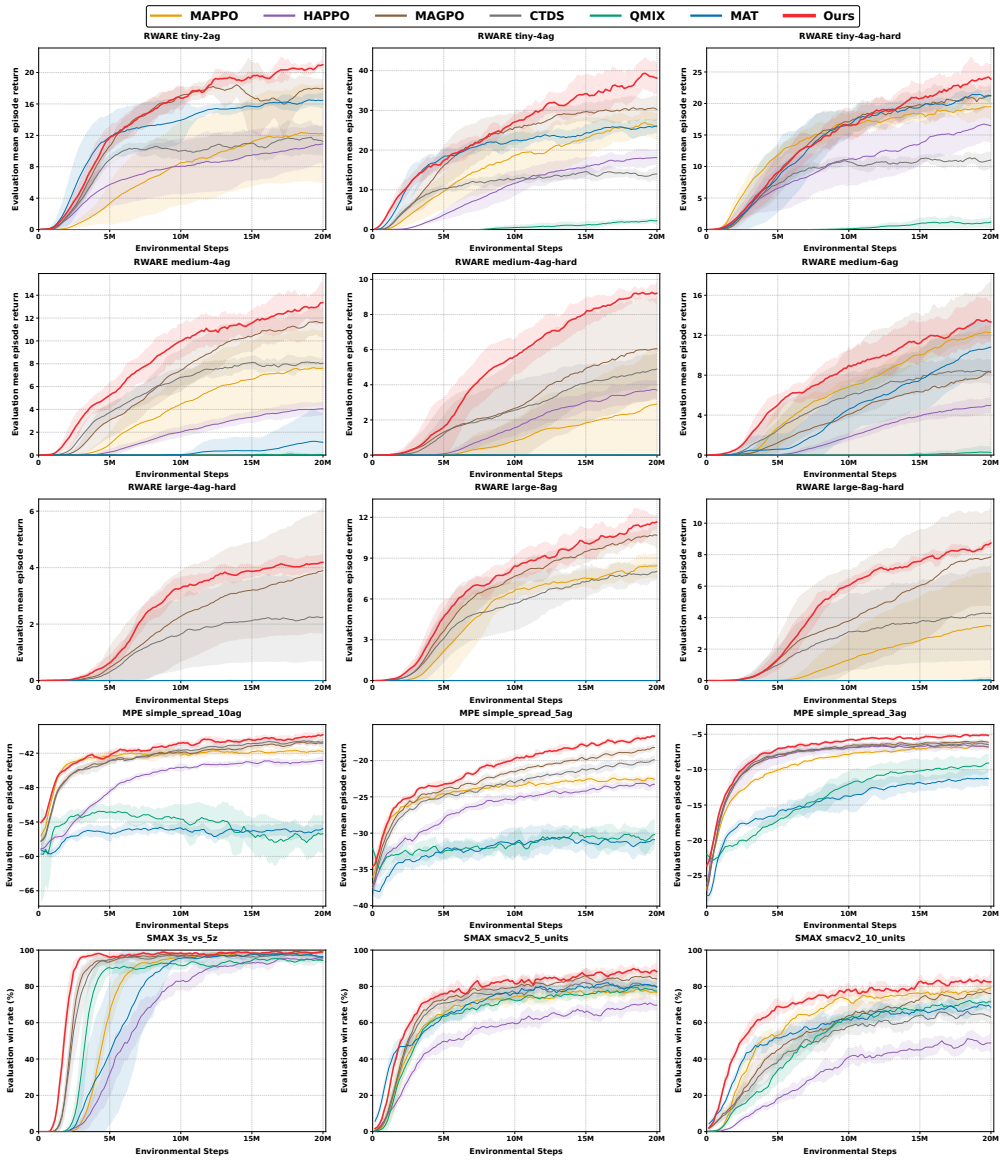


Figure 7: Learning curves across RWARE (first three rows), MPE (fourth row), and SMAX (last three rows) under matched total environment-interaction budgets. For MAOPD, we evaluate only the decentralized parallel Student, and the horizontal axis counts both Teacher and Student interactions. The legend applies to all panels.

B.1.2 NORMALIZED SAMPLE EFFICIENCY.

To compare tasks with different metric scales, task scores are min–max normalized within each benchmark suite and aggregated using the interquartile mean (IQM). Table 2 reports To make scores from different tasks directly comparable, we first min–max normalize each task within its benchmark suite to place all tasks on a common scale. We then summarize performance across tasks using the interquartile mean (IQM), which averages the middle 50% of task scores and is therefore less affected by unusually high or low results. We evaluate sample efficiency by measuring how many environment interactions each method requires to reach a set of normalized-IQM performance targets. Specifically, Table 2 reports the earliest un-smoothed evaluation checkpoint at which each method reaches each target, so the comparison directly reflects the observed training progress rather than a smoothed learning curve. Across MPE, RWARE, and SMAX, MAOPD consistently reaches attainable IQM targets with fewer environment interactions than the compared methods. The advantage is particularly clear on RWARE, where MAOPD is the only method that reaches a normalized IQM of 0.7 within the logged interaction budget. These results show that MAOPD not only achieves strong final performance, but also learns more efficiently by reaching comparable performance levels with fewer environment interactions.

Table 2: Sample efficiency from unsmoothed IQM evaluations (sensitivity analysis). Environment steps ($\times 10^6$, $\downarrow$) required to reach a target normalized IQM.

Target τ	**MAPPO**	**HAPPO**	**MAGPO**	**CTDS**	**QMIX**	**MAT**	**Ours**
MPE							
0.5	1.15	2.46	1.15	1.31	7.38	13.11	**0.98**
0.6	1.80	4.42	1.64	1.80	NR	NR	**1.31**
0.7	3.44	7.37	3.11	3.11	NR	NR	**2.29**
0.8	17.69	NR	8.36	9.50	NR	NR	**6.72**
RWARE							
0.5	19.33	NR	10.49	NR	NR	NR	**9.01**
0.6	NR	NR	13.93	NR	NR	NR	**11.96**
0.7	NR	NR	NR	NR	NR	NR	**16.55**
SMAX							
0.5	4.26	7.37	2.29	2.29	4.18	2.29	**1.64**
0.6	4.92	10.65	2.62	2.46	4.51	2.62	**1.80**
0.7	5.73	15.40	3.28	3.28	5.66	3.60	**2.13**
0.8	9.01	NR	4.26	4.59	8.46	5.24	**2.46**

First recorded checkpoint at or above the target in the UNSMOOTHED normalized IQM series; no interpolation. NR indicates that the target is never reached within the logged interaction budget. For example, NR for HAPPO at $\tau = 0.8$ on MPE means that HAPPO never attains an IQM of 0.8 during the recorded training run.

B.1.3 CLOSING THE POLICY MISMATCH.

Figure 1 diagnoses policy mismatch on SMAX `10m_vs_11m` using five centralized Teachers pre-trained for 0, 4, 8, 12, or 16 million environment steps. In Figure 1(a), each Teacher is frozen, while the Student is trained from scratch with its standard reinforcement-learning objective and additionally receives action-level KL supervision from the Teacher. In Figure 1(b,c), the same Student training and action distillation are used, but the Teacher also continues to learn alongside the Student. The frozen setting reveals an initialization gap, while the updating setting shows that jointly updating the Teacher alone does not prevent the two policies from remaining mismatched.

Figure 8 closes this loop by revisiting the updating-Teacher setting without pre-training (0M) and adding naive OPSD, which retains only parameter sharing and action-level KL distillation. Although naive OPSD substantially reduces the Teacher–Student KL divergence relative to the updating-Teacher baselines, its Student win rate reaches only about 45%, showing that policy alignment alone does not guarantee effective transfer when Teacher-side learning and agent-level distillation are absent. MAGPO achieves an even smaller KL divergence and a stronger Student, but still remains

below MAOPD; as analyzed in Section B.2.2, repeatedly pulling its centralized guider toward the decentralized Student can limit the guider's ability to exploit centralized information. MAOPD instead combines the tightest Teacher–Student alignment with a strong centralized Teacher, yielding substantially faster learning and higher final performance than naive OPSD, MAGPO, and CTDS.

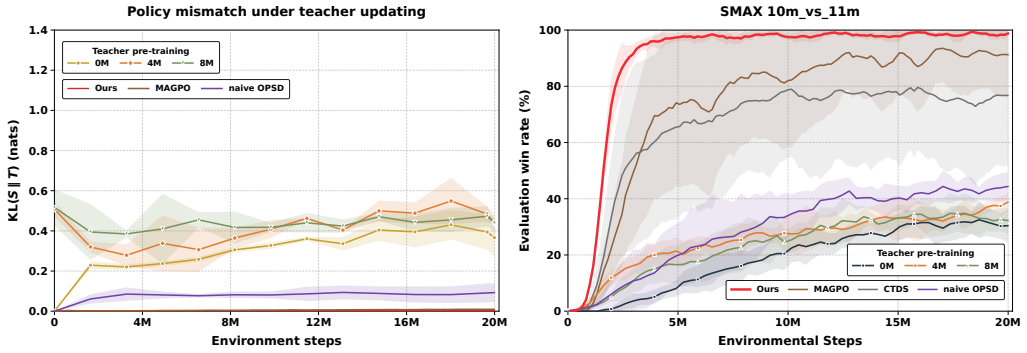


Figure 8: Teacher–Student policy alignment and decentralized performance on SMAX `10m_vs_11m`. The left panel compares the Teacher–Student KL divergence of MAOPD, MAGPO, and naive OPSD with the updating-Teacher baselines initialized after 0, 4, or 8 million pre-training steps. The right panel reports the corresponding Student test win rates together with CTDS. MAOPD achieves the tightest alignment and the highest win rate.

B.1.4 Backbone generality.

Our main implementation uses SABLE as the autoregressive backbone. To examine whether MAOPD generalizes beyond this architecture, we further instantiate it with the centralized version of MAT. Table 3 compares the improvements achieved by different MAT-based transfer methods over centralized version of MAT. MAT is the vanilla decentralized version of MAT. It performs poorly compare to its centralized version . MAT-based MAOPD outperforms MAT-based MAGPO on 11 RWARE tasks. These results show that the proposed distillation framework is not specific to SABLE and remains effective when applied to a different autoregressive backbone.

Table 3: LAST_5 improvement over centralized version of MAT. Each entry is the method's mean score minus MAT's mean score. RWARE reports return differences.

Task	MAT	MAT_MAGPO	MAT_Ours
RWARE Return difference (↑)			
`tiny-2ag`	−0.29	−1.28	**+0.98**
`tiny-2ag-hard`	−7.18	−1.43	**+1.12**
`tiny-4ag-hard`	+0.52	−9.24	**+4.23**
`small-4ag`	−6.12	−6.40	**−3.27**
`small-4ag-hard`	−6.17	+0.40	**+1.71**
`medium-4ag`	−6.31	−2.47	**+4.07**
`medium-4ag-hard`	−4.58	−2.98	**+2.79**
`medium-6ag`	−2.68	−1.90	**+0.19**
`large-4ag`	−4.42	**+3.33**	+3.06
`large-4ag-hard`	−2.30	−0.93	**+0.84**
`large-8ag`	−14.92	−4.89	**+0.53**
`large-8ag-hard`	−8.91	−4.01	**−0.70**

B.2 Additional Ablation Studies

Beyond the component ablations reported in the main text, this subsection examines how teacher information, centralized teacher capability, and distillation design affect MAOPD. These analyses test whether MAOPD depends on a particular teacher configuration and whether its student-grounded online distillation mechanisms are necessary for effective knowledge transfer.

B.2.1 PRIVILEGED TEACHER INFORMATION.

Figure 9 examines teachers operating under different information conditions. The complete method performs best when the teacher has access to both the privileged context ξ_t and preceding actions $\mathbf{a}_t^{<i}$. Its advantage remains when the teacher receives only one of these information sources, although the performance gap narrows when global privileged context is unavailable. Across all three settings, removing both distillation objectives produces the largest degradation. These results show that MAOPD can exploit different forms of teacher information, while centralized autoregressive context provides the most informative supervision.

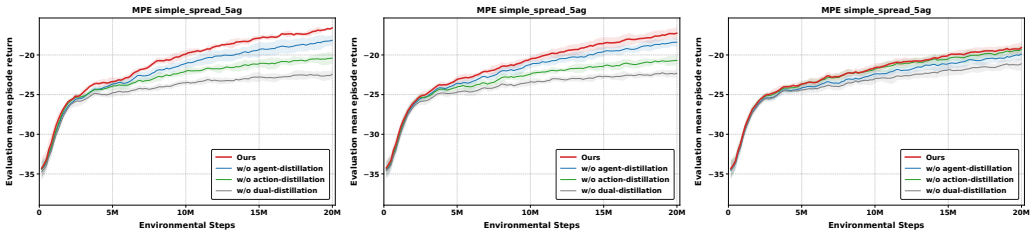


Figure 9: MAOPD with teachers conditioned on different sources of privileged information. From left to right, the teacher conditions on both the privileged context ξ_t and preceding actions $\mathbf{a}_t^{<i}$, on the privileged context ξ_t only, and on preceding actions $\mathbf{a}_t^{<i}$ only. In all variants, the teacher guides the same decentralized student through the proposed MAOPD framework. Higher is better.

B.2.2 DOES MAOPD HURT TEACHER PERFORMANCE

Figure 10 examines whether online distillation limits the capability of the centralized teacher. We compare the teacher learned by MAOPD with standalone SABLE, a fully CTCE autoregressive policy that can also be viewed as an unconstrained teacher, and the centralized guider of MAGPO. The MAOPD teacher performs close to SABLE across the three representative tasks, showing that concurrent teacher learning preserves a strong centralized policy. In contrast, the MAGPO guider is consistently weaker. MAGPO performs distillation off-policy and periodically copies the student policy to the teacher to reduce policy mismatch; although this coupling facilitates transfer, it repeatedly pulls the teacher toward the decentralized student and can limit the teacher's ability to exploit centralized information. **MAOPD therefore maintains effective knowledge transfer without substantially sacrificing teacher capability.**

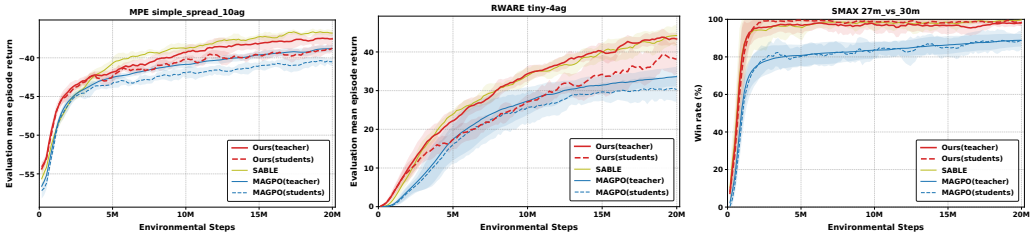


Figure 10: Performance of the centralized policies during training. The jointly learned teacher in MAOPD is compared with SABLE and the centralized guider in MAGPO on MPE `simple_spread_10ag` (left), RWARE `tiny-4ag` (middle), and SMAX `27m_vs_30m` (right). MPE and RWARE report episode return, while SMAX reports test win rate.

B.2.3 CLOSING THE GAP WITHOUT WEAKENING THE TEACHER

Figure 10 shows that the decentralized MAOPD Students can close the performance gap even when the centralized Teacher retains its capability. While the MAOPD Teacher remains close to standalone SABLE, the Students progressively approach the Teacher, nearly matching it on MPE and SMAX. On RWARE, a gap remains, but the MAOPD Student still surpasses the centralized MAGPO Teacher. These results show that MAOPD preserves high-quality centralized supervision throughout training and effectively transfers the capability of a strong Teacher into decentralized policies executable from local information alone.

B.2.4 DISTILLATION DESIGN.

Figure 11 evaluates three additional distillation choices. Naive OPSD shares teacher–student parameters but removes Teacher-side training and the prefix predictor, retaining only action-level KL distillation of MADOP. We find that Directly applying OPSD directly into the SABLE method substantially underperforms MAOPD on both tasks, as it is shown in Figure 11 (left). Replacing student-grounded distillation with supervision on teacher-generated trajectories also reduces performance, confirming the importance of providing guidance on the student's behavioral distribution, as it is shown in Figure 11 (middle). Finally, the reverse KL used by MAOPD consistently outperforms the forward KL, as it is shown in Figure 11 (right). Together, these results validate the student-grounded reverse-KL design used by MAOPD.

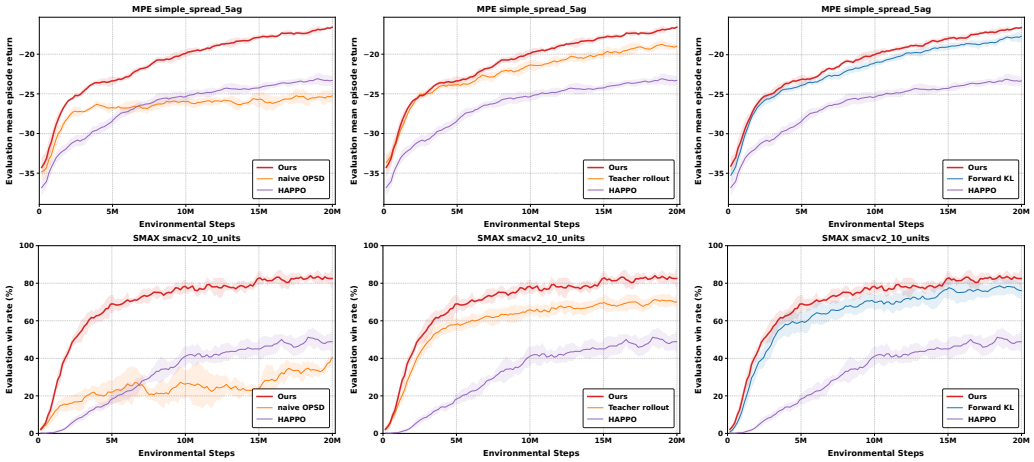


Figure 11: Additional ablations of the distillation design. The left column compares MAOPD with a direct application of OPSD; the middle column replaces student-grounded distillation with distillation on teacher-generated rollouts; and the right column replaces the reverse KL $D_{\mathrm{KL}}(\pi_{\mathrm{S}}\|\pi_{\mathrm{T}})$ with the forward KL $D_{\mathrm{KL}}(\pi_{\mathrm{T}}\|\pi_{\mathrm{S}})$.

B.3 TRAINING COST

Table 4 reports the recorded wall-clock training time of MAOPD and MAGPO under the same total interaction budget of $19{,}988{,}480$ environment steps per run. For MAOPD, this budget includes both teacher and student interactions. The measurements aggregate 122 training intervals covering rollout collection, optimization, and compilation, while excluding evaluation, checkpoint I/O, and startup. Each entry reports the mean and sample standard deviation over 10 complete seeds.

MAOPD records a lower mean training time on 24 of the 26 tasks. Its average training time is 19.96 minutes on RWARE, 17.32 minutes on SMAX, and 17.74 minutes on MPE. Across all three benchmark suites, the recorded average decreases from 40.80 to 18.34 minutes, corresponding to an approximately 55% reduction.

Table 4: Training cost of archived MAOPD and self-run MAGPO experiments. Lower is better. Entries report means with sample standard deviations shown as subscripts; aggregation levels are defined below. Average rows summarize each benchmark and all three benchmarks. Bold identifies the smaller displayed mean. Implementation and hyperparameter settings differ, so these values describe historical runs rather than controlled algorithmic speedups.

Task	Ours	MAGPO
RWARE Training time (min) (↓)		
tiny-2ag	$\mathbf{10.35}_{\pm 2.15}$	$12.73_{\pm 0.20}$
tiny-2ag-hard	$\mathbf{13.70}_{\pm 2.04}$	$18.55_{\pm 2.41}$
tiny-4ag	$\mathbf{22.46}_{\pm 5.53}$	$47.49_{\pm 0.56}$
tiny-4ag-hard	$\mathbf{21.59}_{\pm 0.86}$	$42.11_{\pm 0.51}$
small-4ag	$\mathbf{24.99}_{\pm 3.02}$	$35.30_{\pm 0.26}$
small-4ag-hard	$\mathbf{12.63}_{\pm 2.15}$	$33.91_{\pm 0.37}$
medium-4ag	$\mathbf{19.13}_{\pm 4.03}$	$24.45_{\pm 0.25}$
medium-4ag-hard	$47.24_{\pm 23.10}$	$\mathbf{20.65}_{\pm 5.71}$
medium-6ag	$\mathbf{16.53}_{\pm 2.60}$	$30.70_{\pm 9.07}$
large-4ag	$\mathbf{15.49}_{\pm 1.21}$	$25.67_{\pm 6.59}$
large-4ag-hard	$\mathbf{16.10}_{\pm 0.59}$	$26.21_{\pm 6.96}$
large-8ag	$\mathbf{19.53}_{\pm 1.15}$	$48.30_{\pm 0.47}$
large-8ag-hard	$\mathbf{19.76}_{\pm 0.79}$	$69.54_{\pm 0.64}$
RWARE average	$\mathbf{19.96}_{\pm 9.18}$	$33.51_{\pm 15.32}$
SMAX Training time (min) (↓)		
2s3z	$\mathbf{9.01}_{\pm 1.33}$	$17.40_{\pm 0.19}$
3s5z	$\mathbf{13.18}_{\pm 2.30}$	$25.09_{\pm 0.11}$
3s5z_vs_3s6z	$\mathbf{13.34}_{\pm 2.20}$	$26.01_{\pm 0.14}$
3s_vs_5z	$\mathbf{8.47}_{\pm 2.12}$	$15.60_{\pm 0.21}$
6h_vs_8z	$\mathbf{19.30}_{\pm 1.88}$	$32.97_{\pm 0.14}$
10m_vs_11m	$\mathbf{12.81}_{\pm 3.67}$	$32.33_{\pm 0.16}$
5m_vs_6m	$\mathbf{11.01}_{\pm 2.17}$	$11.08_{\pm 0.13}$
smacv2_5_units	$19.62_{\pm 2.42}$	$\mathbf{14.49}_{\pm 0.07}$
smacv2_10_units	$\mathbf{22.35}_{\pm 2.61}$	$48.02_{\pm 0.31}$
27m_vs_30m	$\mathbf{44.10}_{\pm 3.65}$	$86.82_{\pm 0.63}$
SMAX average	$\mathbf{17.32}_{\pm 10.50}$	$30.98_{\pm 22.52}$
MPE Training time (min) (↓)		
simple_spread_3ag	$\mathbf{16.63}_{\pm 1.04}$	$46.86_{\pm 0.34}$
simple_spread_5ag	$\mathbf{22.97}_{\pm 1.21}$	$65.88_{\pm 0.70}$
simple_spread_10ag	$\mathbf{13.62}_{\pm 0.89}$	$61.01_{\pm 0.46}$
MPE average	$\mathbf{17.74}_{\pm 4.77}$	$57.92_{\pm 9.88}$
Overall average	$\mathbf{18.34}_{\pm 1.42}$	$40.80_{\pm 14.88}$

Budget: 19,988,480 actual environment steps per run; MAOPD counts both teacher and student streams. Time sums 122 timed training intervals covering rollout and optimization, including compilation within those intervals. Evaluation, checkpoint I/O, and startup are excluded. Entries are means with sample standard deviations across 10 complete seeds per method. Both methods used RTX 4090 GPUs.

C HYPERPARAMETERS

To ensure a fair comparison, we tune every evaluated method separately on each task rather than relying on a single default configuration across all environments. For each method–task pair, we search a predefined set of candidate hyperparameters under the same evaluation protocol and report the performance of the best configuration identified for that pair. Consequently, the comparisons throughout the paper reflect the strongest performance we obtained from each method on each task. This task-wise tuning protocol is intended to compare the achievable performance of the algorithms and to avoid disadvantaging any baseline through unsuitable default hyperparameters.

For transparency and reproducibility, Tables 5, 6, and 7 report the default architecture and optimization settings for MAT and SABLE, HAPPO and MAPPO, and QMIX, respectively; the MAT settings correspond to its decentralized implementation. The complete candidate values used in the task-wise searches are listed in Tables 8–13. Parameters outside the stated search spaces retain their corresponding default values. MAOPD inherits the backbone and PPO settings of SABLE and additionally tunes the agent- and action-distillation coefficients α and β. MAGPO additionally tunes its double-clipping coefficient δ and RL auxiliary-loss coefficient λ.

Table 5: Default Hyperparameters for MAT and SABLE.

Parameter	**Value**
Activation function	GeLU
Normalize Advantage	True
Value function coefficient	0.1
Discount	0.99
GAE	0.9
Rollout length	128
Add one-hot agent ID	True

Table 6: Default Hyperparameters for HAPPO and MAPPO.

Parameter	**Value**
Value network layer sizes	[128,128]
Policy network layer sizes	[128,128]
Number of recurrent layers	1
Size of recurrent layer	128
Activation function	ReLU
Normalize Advantage	True
Value function coefficient	0.1
Discount	0.99
GAE	0.9
Rollout length	128
Add one-hot agent ID	True

Table 7: Default Hyperparameters for QMIX.

Parameter	**Value**
Q-network layer sizes	[128, 128]
Number of recurrent layers*	1
Size of recurrent layer*	256
Activation function	ReLU
Maximum gradient norm	10
Add one-hot agent ID	True
Sample sequence length	20
Hard target update	False
Polyak averaging coefficient τ	0.01
Minimum exploration value ϵ	0.05
Exploration value decay rate	0.00001
Rollout length	2
Epochs	2
Add one-hot agent ID	True

Table 8: Hyperparameter Search Space for MAT.

Parameter	Value
PPO epochs	$\{2, 5, 10, 15\}$
Number of minibatches	$\{1, 2, 4, 8\}$
Entropy coefficient	$\{0.1, 0.01, 0.001, 1\}$
PPO clip ϵ	$\{0.05, 0.1, 0.2\}$
Maximum gradient norm	$\{0.5, 5, 10\}$
Learning rate	{1e-3, 5e-4, 2.5e-4, 1e-4, 1e-5}
Model embedding dimension	$\{32, 64, 128\}$
Number of transformer heads	$\{1, 2, 4\}$
Number of transformer blocks	$\{1, 2, 3\}$

Table 9: Hyperparameter Search Space for SABLE.

Parameter	Value
PPO epochs	$\{2, 5, 10, 15\}$
Number of minibatches	$\{1, 2, 4, 8\}$
Entropy coefficient	$\{0.1, 0.01, 0.001, 1\}$
PPO clip ϵ	$\{0.05, 0.1, 0.2\}$
Maximum gradient norm	$\{0.5, 5, 10\}$
Learning rate	{1e-3, 5e-4, 2.5e-4, 1e-4, 1e-5}
Model embedding dimension	$\{32, 64, 128\}$
Number retention heads	$\{1, 2, 4\}$
Number retention blocks	$\{1, 2, 3\}$
Retention heads scaling parameter	$\{0.3, 0.5, 0.8, 1\}$

Table 10: Hyperparameter Search Space for HAPPO and MAPPO.

Parameter	Value
PPO epochs	$\{2, 4, 8\}$
Number of minibatches	$\{2, 4, 8\}$
Entropy coefficient	$\{0, 0.01, 0.00001\}$
PPO clip ϵ	$\{0.05, 0.1, 0.2\}$
Maximum gradient norm	$\{0.5, 5, 10\}$
Value Learning rate	{1e-4, 2.5e-4, 5e-4}
Policy Learning rate	{1e-4, 2.5e-4, 5e-4}
Recurrent chunk size	$\{8, 16, 32, 64, 128\}$

Table 11: Hyperparameter Search Space for QMIX.

Parameter	Value
Batch size	$\{16, 32, 64, 128\}$
Q-network learning rate	$\{3e{-}3, 3e{-}4, 3e{-}5, 3e{-}6\}$
Replay buffer size	$\{2000, 4000, 8000\}$
Target network update period	$\{100, 200, 400, 800\}$
Mixer network embedding dimension	$\{32, 64\}$

Table 12: Hyperparameter Search Space for MAGPO.

Parameter	Value
Double clip δ	$\{1.1, 1.2, 1.3, 1.5, 2, 3\}$
RL auxiliary loss λ	$\{0, 1, 2, 4, 8\}$

Table 13: Hyperparameter Search Space for MAOPD.

Parameter	**Value**
Action distillation loss coefficient β	$\{0.1, 0.15, 0.3\}$
Agent distillation loss coefficient α	$\{0.1, 0.5, 0.75, 1.0\}$