Theorem 5. *Given true Q-values, the difference in variances between the decentralised and CT-DE estimators admits the following bound:*

$$Var(\boldsymbol{g}_i^C) - Var(\boldsymbol{g}_i^D) \leq \frac{B_i^2 C^2}{1-\gamma^2}.$$

Therefore, we can see that with $k = 1$ the bound on excess variance of the PERLA estimator is the same as for the CT-DE estimator, but as $k \to \infty$, the variance of PERLA estimator matches the one of the fully decentralised estimator. However, this is done while still maintaining a centralised critic, unlike in the fully decentralised case. It turns out that the presence of centralised critic is critical, as it allows us to guarantee the converge of an algorithm to a local optimum with probability 1. The result is stated by the next Theorem.

Theorem 6. *Under the standard assumptions of stochastic approximation theory (Konda and Tsitsiklis 1999), an Actor-Critic algorithm using $\boldsymbol{g}_i^P$ or $\boldsymbol{g}_i^C$ as a policy gradient estimator, converges to a local optimum with probability 1, i.e.*

$$P\left(\lim_{k\to\infty} \|\nabla_{\boldsymbol{\theta}_i} \mathcal{J}(\boldsymbol{\theta}^k)\| = 0\right) = 1,$$

where $\boldsymbol{\theta}^k$ is the value of vector $\boldsymbol{\theta}$ obtained after the kth update following the policy gradient.

Proof sketch. We present a proof sketch here and defer the full proof to Appendix I.
The proof consists of showing that a multi-agent Actor-Critic algorithm using policy gradient estimate $\boldsymbol{g}_i^P$ or $\boldsymbol{g}_i^C$ is essentially a special case of single-agent Actor-Critic. □

Note that because the decentralised critic does not allow us to query the state-action-value for joint action of all agents, a decentralised actor-critic using $\boldsymbol{g}_i^D$ as policy gradient estimate is not equivalent to the single-agent version and we cannot establish convergence for it. Therefore, our PERLA estimator enjoys both the low variance property of the decentralised estimator and the convergence guarantee of the CT-DE one. Additionally, having a centralised critic yields better performance in practice in environments with strong interactions between agents (Kok and Vlassis 2004).

5 Experiments

We ran a series of experiments in *Large-scale Matrix Games* (Son et al. 2019), *Level-based Foraging* (LBF) (Christianos, Schäfer, and Albrecht 2020), *Multi-agent Mujoco* (de Witt et al. 2020b) and the *StarCraft II Multi-agent Challenge* (SMAC) (Samvelyan et al. 2019)[2] to test if PERLA: **1.** Improves overall performance of MARL base learners. **2.** Enables efficient scaling in the number of agents. **3.** Reduces variance of value function estimates. In all tasks, we compared the performance of PERLA against MAPPO (Yu et al. 2021a) and IPPO (de Witt et al. 2020a). Here, we report average training results across multiple maps for LBF and SMAC. Further performance comparisons across these maps are deferred to the Appendix. Lastly, we ran a suite of ablation studies which we deferred to the Appendix.

[2]The specific maps/variants used of each of these environments in given in Section B of the Appendix)

	b_1	b_2	b_3
a_1	**8**	-12	-12
a_2	-12	0	0
a_3	-12	0	0

Table 1: Payoff matrix for 2 agents and 3 actions each.

We used the codebase accompanying the MARL benchmark study in Papoudakis et al. (2021) to implement PERLA on top IPPO (Schulman et al. 2017) and MAPPO (Yu et al. 2021a). IPPO is a decentralised learning algorithm, while MAPPO follows the CT-DE paradigm. Implementing PERLA on both widely used training paradigms enables us to examine the impact of PERLA in each case. Hyperparameters were tuned using simple grid-search, and the values over which we tuned them are presented in Table 3 in the Appendix. All results are statistics of the metric being measured over 3 random seeds unless otherwise stated. In our plots dark lines represent the mean across the seeds, while shaded areas represent 95% confidence intervals.

5.1 Performance Analysis

We conducted experiments to ascertain if PERLA yields performance improvements to base MARL algorithms to which it is added as claimed.

Large-Scale Matrix Games. To demonstrate PERLA's ability to handle various reward structure and scale efficiently we first tested its performance in a set of variants of the hard matrix game proposed in (Son et al. 2019) given in Fig. 1. The game contains multiple stable points and a strongly attracting equilibrium (a_1, b_1).

To ensure sufficient data collection in the joint action space, we adopted uniform data distribution in batch learning techniques. With this fixed dataset, we can compare the optimality of PERLA and baselines from an optimisation perspective. We use 500 training iterations and average the results of $[0-10]$ random seeds in each method. As shown in Table 1, policy-based methods (MAPPO and IPPO) and leading value-based methods (MAIC, QPLEX (Wang et al. 2020) and WQMIX (Rashid et al. 2020)) achieve optimal performance in the initial settings (2 agents with 3 available actions), while other algorithms achieve suboptimal outcomes. When scaling with more agents and larger action space, only PERLA can maintain the optimal performance nearly across all action sizes. This is because PERLA enables agents to maintain estimates that account for other agents' actions. Other policy-based methods (MAPPO and IPPO) only consider states as inputs in critic network, without considering agents' actions. Most of value-based baselines (except for QTRAN) are strictly restricted by IGM consistency, which may not be suitable for complex domains.

Level-based Foraging. Figure 2 shows learning curves averaged across all LBF maps that we ran (the full list of maps is given in the Appendix E). As shown in the plots, PERLA enhances the base algorithm. Compared to standard IPPO, PERLA_IPPO learns significantly faster: the PERLA_IPPO achieves evaluation of return of 0.6