

000 SKARL: PROVABLY SCALABLE KERNEL MEAN FIELD 001 REINFORCEMENT LEARNING FOR VARIABLE-SIZE 002 003 MULTI-AGENT SYSTEMS 004

006 **Anonymous authors**

007 Paper under double-blind review

011 ABSTRACT

013 Scaling multi-agent reinforcement learning (MARL) requires both scalability to
014 large swarms and flexibility across varying population sizes. A promising ap-
015 proach is mean-field reinforcement learning (MFRL), which approximates agent
016 interactions via population averages to mitigate state-action explosion. However,
017 this approximation has limited representational capacity, restricting its effective-
018 ness in truly large-scale settings. In this work, we introduce Scalable Kernel
019 MeAn-Field Multi-Agent Reinforcement Learning (SKARL), which lifts this bot-
020 tleneck by embedding agent interactions into a reproducing kernel Hilbert space
021 (RKHS). This kernel mean embedding provides a richer, size-agnostic repres-
022 entation that enables scaling across swarm sizes without retraining or architectural
023 changes. Furthermore, a cylindrical kernel function is introduced to ensure univer-
024 sal approximation over functional space. For efficiency, we design an implemen-
025 tation based on functional gradient updates with Nyström approximations, which
026 makes kernelized mean-field learning computationally tractable. From the theore-
027 tical side, we establish convergence guarantees for both the kernel functionals and
028 the overall SKARL algorithm. Empirically, SKARL trained with 64 agents gen-
029 eralizes seamlessly to deployments ranging from 4 to 256 agents, outperforming
030 MARL baselines.

031 1 INTRODUCTION

033 Multi-agent reinforcement learning (MARL) has achieved remarkable progress in domains such as
034 multi-robot coordination (Vinyals et al., 2019; Berner et al., 2019). However, scaling MARL to
035 large populations remains a fundamental challenge (Du et al., 2023). As the number of agents in-
036 creases, the joint state–action space grows exponentially, and interaction dynamics become increas-
037 ingly complex. This induces a curse of dimensionality that makes conventional learning unstable
038 and inefficient (Tan, 1993; Tampuu et al., 2015). Moreover, most existing MARL methods lack
039 population scalability: policies trained with one swarm size often fail to generalize to other scales in
040 zero-shot. These limitations naturally raise the question: How can we design MARL algorithms that
041 scale efficiently to hundreds of agents while generalizing seamlessly to unseen population sizes?

042 A promising direction is the use of mean-field approximations (Caines et al., 2006; Lasry & Lions,
043 2007). By summarizing agent interactions through a population distribution, mean-field MARL
044 (MFRL) (Yang et al., 2018) avoids exponential complexity growth and exploits the permutation
045 invariance of homogeneous swarms. Prior work has demonstrated the feasibility of mean-field
046 methods in large-scale settings (Angiuli et al., 2021; Gu et al., 2025). However, how to design a
047 universally effective way to represent the population distribution remain a bottleneck. Traditional
048 distribution representation paradigms in the field of mean-field mainly fall into two categories. The
049 first method employs spatial discretization techniques such as histograms (Carmona et al., 2019)
050 and ϵ -net (Gu et al., 2021), which preserve distributional information with theoretical guarantee but
051 suffer from the curse of dimensionality in high-dimensional spaces, as the number of discrete units
052 grows exponentially with state dimensions. The second relies on statistical moments, ranging from
053 first-order means (Yang et al., 2018) to higher-order statistics (Pham & Warin, 2023). This paradigm
adapts well to high-dimensional state spaces as moment calculations avoid explicit space partition-
ing, but the representational capacity is limited, as the conflation of distinct distributions and lack of

critical structural details like multi-modality. As a result, current mean-field approaches still struggle to achieve scalability when applied to sufficiently large populations.

In recent years, kernel-based methods have emerged as the third direction for distribution representation (Wang et al., 2020b; Liu et al., 2020; Cui et al., 2023; Fiedler et al., 2023; 2025), aiming to combine the scalability of moment-based methods with the expressiveness of discretization-based approaches. In the context of mean-field systems, these methods leverage kernel functions (e.g. radial basis function kernel) to embed population distributions into high-dimensional feature spaces, transforming distribution-level interactions into tractable feature operations (Wang et al., 2020b; Liu et al., 2020; Cui et al., 2023). Nevertheless, existing kernel-based methods rely on fixed kernel functions or constrained feature structures, failing to guarantee that their representation space can fully span all possible population distributions, especially when agent number tends to larger. This incompleteness in representational coverage may lead to missed critical distributional characteristics, ultimately restricting the scalability performance.

To address this problem, we introduce Scalable Kernel MeAn-Field Multi-Agent Reinforcement Learning (SKARL), a novel approach that integrates mean-field learning with reproducing kernel Hilbert space (RKHS) representations to achieve both scalability and flexibility. Unlike traditional kernel methods constrained by fixed structures, SKARL employs kernel mean embeddings to map the entire population distribution into the RKHS, capturing intrinsic structural details (e.g., multi-modality) in a size-agnostic manner. Furthermore, to ensure the global approximation, we model the Q -function for individual agent as a cylindrical kernel functional, inspired by Guo et al. (2023), and derive functional gradient updates under a dual time-scale learning scheme. To ensure computational efficiency in large populations, we employ Nyström approximations to project functional updates onto low-dimensional subspaces (Williams & Seeger, 2000). Our framework offers both theoretical and empirical benefits. We prove that cylindrical kernel functionals form a universal approximator over distribution spaces, ensuring expressiveness, and establish that the resulting value functions are Wasserstein-Lipschitz continuous, providing robustness to distributional shifts. Crucially, by representing the swarm as a distribution rather than a fixed-size set, our method naturally supports population flexibility **to 4 times larger agent size in deployment compared with training phase**. Empirically, SKARL achieves superior performance on large-scale cooperative tasks, consistently outperforming **MARL baselines with and without mean-field techniques** in cumulative reward and training stability.

In summary, our contributions are as follows:

- We propose the SKARL, a novel MARL framework that integrates RKHS distribution embedding with mean-field multi-agent reinforcement learning, providing a size-agnostic, distribution-level representation beyond moments and fixed kernel embedding representations.
- We model individual Q-functions as cylindrical kernel functionals over the embedded population distribution, significantly enhancing expressive capacity compared with traditional parametric critics.
- We develop a functional gradient algorithm for cylindrical kernel functionals, along with a dual time-scale learning rule and Nyström approximations for efficiency. Theoretically, we prove universal approximation and establish Wasserstein-Lipschitz continuity of the value functions.
- Through extensive experiments on large-scale benchmarks, we demonstrate that SKARL generalizes seamlessly across population sizes and achieves significant improvements over MARL baselines in both performance and stability.

2 PRELIMINARIES

2.1 MULTI-AGENT STOCHASTIC GAME

We consider an episodic mean-field reinforcement learning game with a fixed number of agents $N \in \mathbb{N}$. Such a game is defined by the tuple $\langle \mathcal{S}^N, \mathcal{A}^N, P, (r^i)_{i=1}^N, \gamma \rangle$, where $\mathcal{S}^N = \mathcal{S}_1 \times \dots \times \mathcal{S}_N$ denotes the joint state space: a vector $s = (s^1, \dots, s^N)$ collects the local state $s^i \in \mathcal{S}_i$ of each agent. Similarly, the joint action space is $\mathcal{A}^N = \mathcal{A}_1 \times \dots \times \mathcal{A}_N$, where a joint action $a = (a^1, \dots, a^N)$

108 consists of local actions $a^i \in \mathcal{A}_i$. In the homogeneous setting, agents share the same state and action
 109 spaces, i.e., $\mathcal{S} = \mathcal{S}_1 = \dots = \mathcal{S}_N$ and $\mathcal{A} = \mathcal{A}_1 = \dots = \mathcal{A}_N$. System dynamics are governed by a
 110 stochastic kernel $P : \mathcal{S}^N \times \mathcal{A}^N \rightarrow \mathcal{P}(\mathcal{S}^N)$, where $\mathcal{P}(\mathcal{S}^N)$ denotes the set of probability measures
 111 over \mathcal{S}^N . Each agent receives an instantaneous reward $r^i(s, a) = r(s^i, a^i)$, which couples individual
 112 behavior with the global population. Finally, $0 < \gamma < 1$ is the discount factor weighting future
 113 returns. The objective is to learn a joint policy $\pi = (\pi^1, \dots, \pi^N)$, where each $\pi^i : \mathcal{S} \rightarrow \mathcal{P}(\mathcal{A})$, that
 114 maximizes for every agent i the expected discounted return

$$115 \quad J^i(\pi) = \mathbb{E}_{s_0 \sim d_0, P, \pi} \left[\sum_{t=0}^{T-1} \gamma^t r^i(s_t, a_t) \right],$$

118 with the expectation taken over the initial state distribution d_0 , the transition kernel P , and the
 119 stochastic choices of the joint policy π .

121 2.2 MEAN FIELD REINFORCEMENT LEARNING

123 In multi-agent reinforcement learning with N agents, the Q-function of agent i depends on the
 124 joint action $\mathbf{a} = (a^1, \dots, a^N)$, where each a^j is represented by a one-hot vector. This leads to
 125 an exponential blow-up of the action space, a manifestation of the curse of dimensionality. Mean-
 126 field reinforcement learning (MFRL) (Yang et al., 2018) addresses this by approximating pairwise
 127 interactions through a mean-field term. Specifically, the Q-function of agent i is written as

$$128 \quad Q^i(s, \mathbf{a}) = \frac{1}{N_i} \sum_{j \in \mathcal{N}^i} Q^i(s, a^i, a^j) \approx Q^i(s, a^i, \bar{a}^{-i}),$$

131 where $\bar{a}^{-i} := \frac{1}{N_i} \sum_{j \in \mathcal{N}^i} a^j$ denotes the empirical mean action of agent i 's neighbors \mathcal{N}^i with size
 132 N_i . This induces a dynamical system in which each agent responds to the mean-field action via a
 133 softmax policy as $\pi_t^i(\cdot | s) = \text{softmax}(-\beta Q_t^i(s, \cdot, \bar{a}_t^{-i}))$, where the softmax is taken over all $a \in \mathcal{A}$.

134 For continuous action spaces, the mean-field action is modeled as a distribution on the 2-Wasserstein
 135 space $\mathcal{P}_2(\mathcal{A})$ (Guo & Xu, 2019):

$$137 \quad \nu^{-i} = \frac{1}{N_i} \sum_{j=1}^{N_i} \delta_{a^j},$$

139 where δ_{a^j} is the Dirac measure at action a^j . If the pairwise Q-function is twice Lions-differentiable
 140 with respect to the mean-field action μ_{a^j} , the Lions–Taylor expansion yields (Tang et al., 2024)

$$142 \quad Q^i(s, \mathbf{a}) \approx \bar{Q}^i(s, a^i, \mu^{-i}) + \frac{1}{N_i} \sum_{j=1}^{N_i} \partial_\nu \bar{Q}^i(s, a^i, \mu^{-i})[a^j] \cdot (\bar{a}^i - a^j), \quad (1)$$

145 where $\bar{Q}^i(s, a^i, \delta_{a^j})$ is the Q-function lifted to the Wasserstein space, $\bar{a}^i = \frac{1}{N_i} \sum_j a^j$ is the mean
 146 neighbor action, and $\partial_\nu \bar{Q}^i(s, a^i, \mu^{-i})[\cdot] : \mathcal{A} \rightarrow \mathcal{A}$ is the Lions derivative such that for any sequence
 147 $\{\nu_n\} \subset \mathcal{P}(\mathcal{A})$ with the norm-2 Wasserstein distance converges to 0, when $n \rightarrow 0$, it always hold
 148 that

$$149 \quad \frac{Q(s, a^i, \nu_n) - Q(s, a^i, \nu) - \int_{\mathcal{A}^2} \partial_\nu Q(s, a^i, \nu)(x) \cdot (y - x) \pi(dx, dy)}{\mathcal{W}_2^2(\nu_n, \nu)} \rightarrow 0,$$

151 where $\pi_n \subset \mathcal{P}_2(\mathcal{A} \times \mathcal{A})$ denotes the optimal plan between ν_n and ν .

153 2.3 RKHS AND KERNEL MEAN EMBEDDING IN MFRL

155 In recent years, kernel-based methods have been successfully integrated into MFRL to enhance the
 156 expressiveness and scalability of value functions and policies. A key tool for this integration is the
 157 RKHS, which provides a nonparametric framework for representing distributions over agent states
 158 (). An RKHS (\mathcal{H}_k) over the domain (\mathcal{X}) is a Hilbert space of functions ($g : \mathcal{X} \rightarrow \mathbb{R}$) associated with
 159 a symmetric positive-definite kernel ($k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$). The defining property of an RKHS is the
 160 reproducing identity (Muandet et al., 2017):

$$161 \quad g(x) = \langle g, k(x, \cdot) \rangle_{\mathcal{H}_k} = \int_{\mathcal{X}} g(x') k(x, x') dx'. \quad (2)$$

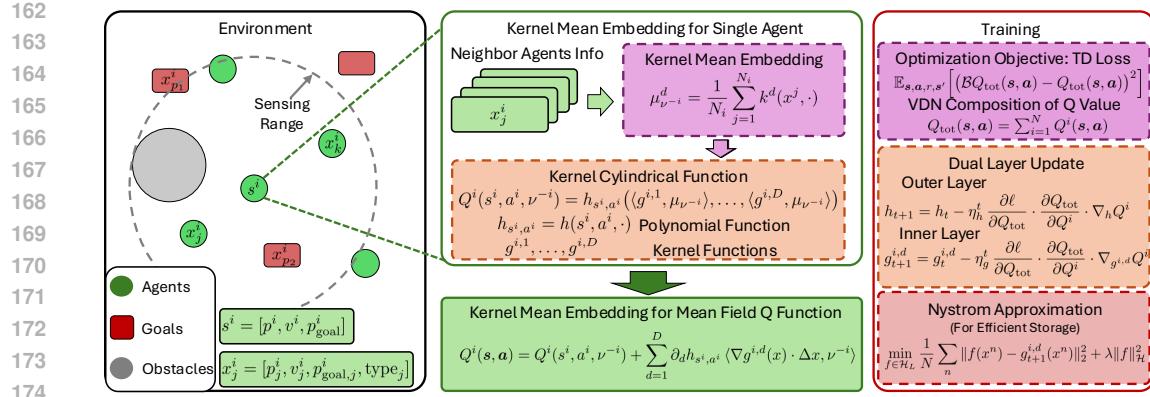


Figure 1: Overview of the SKARL framework. Agent interactions are embedded into RKHS via kernel mean embeddings and evaluated through kernel cylindrical functions to approximate mean-field Q-values. Updates are performed with temporal-difference learning and Nyström projection for scalability and efficiency.

In the context of MFRL, the kernel mean embedding (KME) method is used to represent the distribution of agents in a high-dimensional feature space. For any distribution \mathbb{P} over (\mathcal{X}) , its embedding is defined as:

$$\mu_{\mathbb{P}}(\cdot) := \mathbb{E}_{X \sim \mathbb{P}}[k(\cdot, X)] = \int_{\mathcal{X}} k(\cdot, x) d\mathbb{P}(x), \quad (3)$$

which is a mapping from the distribution to an element of the RKHS. This embedding preserves expectations, such that for any function $g \in \mathcal{H}_k$, the expected value of $g(X)$ under the distribution \mathbb{P} is given by the inner product in \mathcal{H}_k as $\mathbb{E}_{X \sim \mathbb{P}}[g(X)] = \langle g, \mu_{\mathbb{P}} \rangle_{\mathcal{H}_k}$. This allows the representation of the entire mean-field distribution of agents as a single element in the RKHS, facilitating efficient computation and flexible modeling of the agent population's behavior.

3 THE SKARL FRAMEWORK

This section presents the derivation of SKARL within the Reproducing Kernel Hilbert Space (RKHS), as is shown in Figure. 1. In this work, we aim to design a population representation for MFRL that is both expressive and scalable. Classical spatial discretization methods provide theoretical guarantees but suffer from the curse of dimensionality, as the number of cells explodes with the state dimension, making them unsuitable for large-scale RL (Carmona et al., 2019; Gu et al., 2021). Moment-based approaches alleviate this by summarizing populations via first-order means (Yang et al., 2018) or a few higher-order statistics Pham & Warin (2023), yet they fundamentally discard fine-grained distributional structure and struggle to capture complex agent interactions. More recent kernel-based methods strike a middle ground between these two extremes, but typically rely on fixed kernels or rigid feature parameterizations, which do not guarantee that all relevant population distributions can be well approximated (Cui et al., 2023). Motivated by these limitations, we build our framework on KME to represent populations in a high-dimensional RKHS, and further introduce a cylindrical kernel functional that endows the representation with global approximation capabilities over the space of mean-field distributions.

3.1 KERNEL MEAN EMBEDDING REPRESENTATION OF MEAN-FIELD Q-FUNCTIONS

Mean-Field Embedding via KME The mean-field measure is embedded via empirical KME:

$$\mu_{\nu^{-i}}^d = \frac{1}{N_i} \sum_{j=1}^{N_i} k^d(x_j^i, \cdot),$$

where x^i is the latent embedding of neighbor (s^i, a^i) .

216 **Kernel Cylindrical Representation of Pairwise Interactions.** Mean-field Q-functions are functionals
 217 of probability measures over neighbor actions. To approximate such distributional functionals
 218 in a principled and expressive way, we introduce *kernel cylindrical functions*, inspired by work
 219 of Guo et al. (2023), which provide universal approximations within RKHS. Formally, for any con-
 220 tinuous functional $f : \mathcal{P}(\mathcal{M}) \rightarrow \mathbb{R}$ with bounded Lions derivatives, we approximate it by $h(\nu)$ with
 221 definition as

$$222 \quad h(\nu) := h(\langle g^1, \mu_\nu^1 \rangle_{\mathcal{H}_k}, \dots, \langle g^D, \mu_\nu^D \rangle_{\mathcal{H}_k}), \quad (4)$$

223 where each $g^d(\cdot) = k(x^d, \cdot)$ is a kernel anchored at $x^d \in \mathcal{M}$, μ_ν^d denotes the empirical KME, and
 224 $h : \mathbb{R}^D \rightarrow \mathbb{R}$ is a polynomial function with parameters θ_h , with D denotes the number of used ker-
 225 nels and d denotes the index of kernel. The inner products $\langle g^d, \mu_\nu^d \rangle_{\mathcal{H}_k} = \int_{\mathcal{M}} g^d(x) d\nu(x)$ serve as
 226 kernel-based summaries of ν . Base on this function type, we develop the following theorem, implying
 227 that any smooth mean-field Q-function can be approximated arbitrarily well by such cylindrical
 228 representations.

229 **Theorem 3.1** (Density of Kernel Cylindrical Functions). *Let $\mathcal{P}(\mathcal{M})$ be the space of Borel probabili-
 230 ty measures over a compact manifold $\mathcal{M} \subset \mathbb{R}^d$. Define*

$$232 \quad \mathcal{G}_D(\mathcal{M}) := \{h(\mu) = h(\langle g^1, \mu \rangle_{\mathcal{H}_k}, \dots, \langle g^D, \mu \rangle_{\mathcal{H}_k}) \mid h \text{ (polynomial), } \{g^d\}_{d=1}^D \text{ kernels}\}. \quad (5)$$

234 Let $\mathcal{C}^{1,1}(\mathcal{M})$ denote the space of Fréchet differentiable functions with Lipschitz derivatives. Then,
 235 for any $f \in \mathcal{C}^{1,1}(\mathcal{M})$ and any $\epsilon > 0$, there exists $h \in \mathcal{G}_D(\mathcal{M})$ such that $|f(\mu) - h(\mu)| < \epsilon$ for all
 236 $\mu \in \mathcal{P}(\mathcal{M})$, provided D is sufficiently large.

237 This directly yields a representation of the pairwise interaction in agent i 's Q-function:

$$239 \quad Q^i(s^i, a^i, \nu^{-i}) = h_{s^i, a^i}(\langle g^{i,1}, \mu_{\nu^{-i}} \rangle, \dots, \langle g^{i,D}, \mu_{\nu^{-i}} \rangle),$$

241 where $h_{s^i, a^i}(\cdot) = h(s^i, a^i, \cdot) : \mathbb{R}^D \rightarrow \mathbb{R}$ is differentiable with its parameters, and $g^{i,d} =$
 242 $\sum_{m=1}^M \alpha_m^d k^d(x^m, \cdot)$, with anchor points $\{x^m\}_{m=1}^M$ in latent space \mathcal{X} and learnable weights $\{\alpha_m^d\}$.
 243 The gradient of $g^{i,d}$ is $\nabla g^{i,d}(x) = \sum_m \alpha_m^d \partial_x k^d(x^m, x)$. To guaranty continuity, we assume Lips-
 244 chitz continuity and boundedness of kernels.

245 **Assumption 3.1** (Lipschitz Continuity and Boundedness). *Each kernel $g^{i,d}$ is L_g -Lipschitz:*

$$247 \quad |g^d(x) - g^d(y)| \leq L_g \|x - y\|_2, \quad \forall x, y \in \mathcal{X},$$

248 and uniformly bounded: $|k(x, y)| < \infty$, $\forall x, y \in \mathcal{X}$. Without loss of generality, assume
 249 $\sup_{x \in \mathcal{X}} |k(x, x)| \leq 1$.

251 The Lions derivative of a cylindrical function $h(\nu)$ is (Guo et al., 2023):

$$253 \quad \partial_\nu h(\nu)(x) = \sum_{d=1}^D \partial_d h(\nu) \nabla g^d(x),$$

256 where $\partial_d h$ denotes the derivative with respect to the d -th argument.

258 **Local Value Function Approximation.** Combining state-action embeddings, cylindrical functionals,
 259 and mean-field embeddings yields a computational representation of the local Q-function.
 260 Analogous to Eq. (1), we approximate
 261

$$262 \quad Q^i(s, a) = h_{s^i, a^i}(\langle g^{i,1}, \mu_{\nu^{-i}} \rangle, \dots, \langle g^{i,D}, \mu_{\nu^{-i}} \rangle) + \sum_{d=1}^D \partial_d h_{s^i, a^i} \langle \nabla g^{i,d}(x) \cdot \Delta x, \nu^{-i} \rangle, \quad (6)$$

265 where $\Delta x := \bar{x}^i - x$ and $\bar{x}^i = \frac{1}{N_i} \sum_j x^j$. The first term captures mean-field interactions, while the
 266 second encodes gradient corrections.

268 This representation integrates seamlessly with standard multi-agent value-decomposition methods
 269 such as VDN (Sunehag et al., 2017), QMIX (Rashid et al., 2018), and QPLEX (Wang et al., 2020a).
 270 Analogous constructions apply to the state-value function $V^i(s)$ and advantage function $A^i(s, a)$.

270 3.2 VALUE FUNCTION UPDATE WITH STORAGE EFFICIENCY
271272 **Updating Cylindrical Kernel Functions.** The total value function Q_{tot} is decomposed into agent-
273 wise functions Q^i under the Individual Global Max (IGM) principle (Rashid et al., 2018) (See
274 Appendix E). To update Q^i , we optimize the temporal-difference (TD) loss (Sutton, 1988)

275
$$\ell(\mathcal{B}Q_{\text{tot}}, Q_{\text{tot}}) = \mathbb{E}_{\mathbf{s}, \mathbf{a}, r, \mathbf{s}'} \left[(\mathcal{B}Q_{\text{tot}}(\mathbf{s}, \mathbf{a}) - Q_{\text{tot}}(\mathbf{s}, \mathbf{a}))^2 \right],$$

276

277 where \mathcal{B} denotes the Bellman operator (Puterman, 1994), i.e.,
278

279
$$(\mathcal{B}Q_{\text{tot}})(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{\mathbf{s}'} \left[r(\mathbf{s}, \mathbf{a}) + \gamma \max_{\mathbf{a}'} Q_{\text{tot}}(\mathbf{s}', \mathbf{a}') \right].$$

280

281 Parameters are updated by gradient descent in two spaces. For the outer function h and RKHS
282 components $\{g^{i,d}\}$, with learning rate η_h^t, η_g^t .
283

284
$$\theta_h^{t+1} = \theta_h^t - \eta_h^t \frac{\partial \ell}{\partial Q_{\text{tot}}} \cdot \frac{\partial Q_{\text{tot}}}{\partial Q^i} \cdot \nabla_h Q^i, \quad g_{t+1}^{i,d} = g_t^{i,d} - \eta_g^t \frac{\partial \ell}{\partial Q_{\text{tot}}} \cdot \frac{\partial Q_{\text{tot}}}{\partial Q^i} \cdot \nabla_{g^{i,d}} Q^i, \quad (7)$$

285

286 where $\{g^{i,d}\}$ are updated via the Fréchet derivative.
287288 **Proposition 3.2** (Fréchet Derivative Form). *The Fréchet derivative of Q^i with respect to $g^{i,d}$ de-
289 composes as*

290
$$\nabla_{g^{i,d}} Q^i = \underbrace{\left(\partial_d h + \sum_{d'} \partial_{dd'}^2 h \langle \nabla g^{d'} \cdot \Delta x, \nu^{-i} \rangle \right) \mu_{\nu^{-i}}}_{\text{Mean interaction term}} - \underbrace{\partial_d h \nabla \cdot (\nu^{-i} \Delta x)}_{\text{Divergence term}}, \quad (8)$$

291

292 where $\Delta x := \bar{x}^i - x$. See Remark D.3 in the Appendix for the explicit form with N_i neighbors.
293294 **Nyström Approximation for Efficient Storage.** The direct updates in Eq. (7) face two key
295 challenges: (i) the divergence term lies outside the RKHS (Remark D.3), and (ii) naive imple-
296 mentation requires storing $O(N_i T)$ kernels per agent after T iterations, which is infeasible for
297 large swarms and long horizons. To address this, we apply the Nyström approximation, project-
298 ing updated functions onto a low-dimensional kernel subspace. Let the anchor set for $g_{t+1}^{i,d}$ be
299 $\{x^n\}_{n=1}^{N_i+M} := \{x^j\}_{j=1}^{N_i} \cup \{x^m\}_{m=1}^M$, where $\{x^m\}$ are anchor points from $g_t^{i,d}$ and $\{x^j\}$ are in-
300 puts from ν . We select a subset of landmark points $\{z^l\}_{l=1}^L \subset \{x^n\}$, spanning an L -dimensional
301 subspace $\mathcal{H}_L \subset \mathcal{H}$. The projection of $g_{t+1}^{i,d}$ onto \mathcal{H}_L via Tikhonov regularization is:
302

303
$$\tilde{g}_{t+1}^{i,d} = \arg \min_{f \in \mathcal{H}_L} \frac{1}{N_i + M} \sum_{n=1}^{N_i+M} \|f(x^n) - g_{t+1}^{i,d}(x^n)\|_2^2 + \lambda \|f\|_{\mathcal{H}}^2. \quad (9)$$

304

305 By the representer theorem (Schölkopf & Smola, 2002), the solution takes the form
306 $\tilde{g}_{t+1}^{i,d} = \sum_{l=1}^L \alpha_l^d k^d(z^l, \cdot)$. Let $\mathbf{K}_{LL}^d := [k^d(z^l, z^{l'})]_{1 \leq l, l' \leq L}$ and $\mathbf{K}_{N_i+M, L}^d :=$
307 $[k^d(x^n, z^l)]_{1 \leq n \leq N_i+M, 1 \leq l \leq L}$. Then coefficients $\boldsymbol{\alpha}^d = [\alpha_1^d, \dots, \alpha_L^d]^\top$ admit the closed-form so-
308 lution (Rudi et al., 2015):
309

310
$$\boldsymbol{\alpha}^d = (\mathbf{K}_{N_i+M, L}^\top \mathbf{K}_{N_i+M, L} + \lambda(N_i + M) \mathbf{K}_{LL}^d)^\dagger \mathbf{K}_{N_i+M, L}^\top \mathbf{b},$$

311

312 where $\mathbf{b} \in \mathbb{R}^{N_i+M}$ with entries $\mathbf{b}_n = \langle k(x^n, \cdot), g_{t+1}^{i,d} \rangle_{\mathcal{H}_k}$. Here \dagger denotes the Moore–Penrose
313 pseudoinverse. This reduces kernel storage from $O(N_i T)$ to $O(L)$ with $L \ll N_i T$. In our experi-
314 ments we use uniform sampling for landmark points $\{z^l\}$; other selection strategies are discussed in
315 Remark D.5.321 3.3 PROPOSED ALGORITHM
322

323 With the components mentioned above, the final proposed algorithm is summarized in Algorithm 1.

324 **Algorithm 1** Mean-Field Cylindrical Kernel Method

325 **Input:** Agent swarm size N , number of iterations M , trajectory batch size B , anchor points number

326 L , learning rate (η_h, η_g)

327 1: Initialize local Q function Q^i with kernel functions $\{g_t^{i,d}\}_{d=1}^D \leftarrow 0$ and outer function h^i for

328 each agent; initialize trajectory set \mathcal{T} .

329 2: **for** $m = 1, \dots, M$ **do**

330 3: **while** Sampling phase **do**

331 4: Sample trajectories using the current policy $\{\pi^i\}_{i=1}^N$ with environment, store in \mathcal{T} .

332 5: **end while**

333 6: Sample B trajectories from \mathcal{T} with length T for each trajectory.

334 7: Update the outer function h and $\{g_t^{i,d}\}$ with Eq. (7).

335 8: Select new anchor points $\{x_t\}_{t=1}^L$ via methods in Remarks D.5.

336 9: Projection updated $\{g_t^{i,d}\}$ to $\{\tilde{g}_t^{i,d}\}$ via Eq. (9) and update Q^i with $\{\tilde{g}_t^{i,d}\}$.

337 10: **end for**

338 11: **return** final local Q function Q^i .

4 ANALYSIS OF PROPOSED SKARL

4.1 COMPUTATIONAL COMPLEXITY, SCALABILITY, AND FLEXIBILITY

We compare the computational complexity of SKARL with value decomposition methods (e.g., QMIX (Rashid et al., 2018)) and mean-field reinforcement learning (MFRL) (Yang et al., 2018)). Table 1 summarizes the results.

Table 1: Comparison of computational complexity and key metrics. B : batch size; N : number of agents; L : landmark points; D : number of kernel features.

Metric	SKARL	QMIX	MFRL
Q Function Input Size	$O(\mathcal{S} + \mathcal{A} + D)$	$O(N \mathcal{S} + N \mathcal{A})$	$O(\mathcal{S} + \mathcal{A})$
Computation Complexity	$O(B(L^2N + L^3)D)$	$O(BN^2)$	$O(B)$
Memory Usage	$O(DL)$	$O(N)$	$O(1)$
Scalability in N	Linear	Exponential	Linear

Q Function Input size. SKARL avoids the $N|\mathcal{A}|$ blow-up in QMIX by using kernel-based embeddings (Eq. 6), with $L \ll N$ and $D \ll N$. MFRL is even simpler, but lacks multi-scale coordination.

Computation. Complexity is dominated by kernel projections (Eq. 9), scaling with B , N , and L . QMIX suffers $O(N^2)$ due to its mixing network, while MFRL requires only $O(1)$ per agent. When L grows with N (e.g., $L \approx \sqrt{N}$), SKARL’s complexity approaches QMIX—this is the main computational drawback.

Scalability. SKARL maintains linear dependence on N , unlike QMIX’s exponential scaling.

Flexibility. SKARL generalizes across swarm sizes. If trained with N agents and deployed with M , the approximation error is bounded by $O(N^{-1/d} + M^{-1/d})$, where d is the dimension of the state-action space.

Theorem 4.1 (Flexibility of Kernel Cylindrical Functions). *Let ν_N, ν_M denote the empirical mean-field distributions of swarms with N and M agents, sampled from the same distribution ν . Under Assumption 3.1, for a cylindrical function h there exist constants $C_1, C_2 > 0$ such that*

$$\mathbb{E}[|h(\nu_N) - h(\nu_M)|] \leq C_1 N^{-1/d} + C_2 M^{-1/d}.$$

4.2 CONVERGENCE AND SUBOPTIMALITY

Convergence of Cylindrical Functions. The density result in Theorem 3.1 implies approximation power. We now establish convergence rate with respect to the kernel number D .

378 **Theorem 4.2** (Convergence Rate). *Under Assumption 3.1, let $\tilde{f}(\mu_\nu) = f(\nu)$ be a functional de-
379 pending on the KME μ_ν (Eq. 3). Then with probability at least $1 - \delta$,*

$$381 \quad |h(\nu) - f(\nu)| \leq \sup_\nu \left\| \frac{\delta \tilde{f}}{\delta \mu_\nu} \right\| \left(\sqrt{\frac{1}{D}} + \sqrt{\frac{2 \log(1/\delta)}{D}} \right).$$

384 *Thus h converges to f at rate $O(D^{-1/2})$.*

386 **Convergence of Updates.** For the update rules in Eq. equation 7, convergence follows under Rob-
387 bins–Monro step-size conditions and two-time-scale separation (Borkar, 2008).

388 **Assumption 4.1** (Robbins–Monro). *Step sizes η_h and η_g satisfy $\sum_t \eta = \infty$, $\sum_t \eta^2 < \infty$, and
389 $\lim_{t \rightarrow \infty} \eta_g / \eta_h = 0$.*

390 **Theorem 4.3** (Convergence). *Under Assumptions 3.1 and 4.1, the updates converge to $(h^*, \{g^{i,d,*}\})$
391 minimizing the Bellman TD loss.*

393 4.3 ERROR OF NYSTRÖM APPROXIMATION

395 Although the Nyström method substantially reduces storage and computational cost, this method in-
396 evitably introduces approximation error. To ensure the reliability of SKARL, it is therefore essential
397 to quantify error of Nyström approximation. We measure the error of projection as

$$398 \quad \mathcal{E}(f) = \|f - g_{t+1}^{i,d}\|_{L^2},$$

400 for $f \in \mathcal{H}$, where $L_k f(x) = \langle f, k(x, \cdot) \rangle_{\mathcal{H}_k}$ is the kernel integral operator (Eq. 2). Intuitively, $\mathcal{E}(f)$
401 captures the deviation between the projected function and the ideal update.

402 To analyze this error, we introduce two standard conditions from statistical learning theory:

404 **Assumption 4.2.** *Define the effective dimension $\mathcal{N}(\lambda) = \text{tr}((\lambda I + L_k)^{-1} L_k)$. Assume there exists
405 a constant $C_0 > 0$ independent of λ such that for any $\lambda > 0$, $\mathcal{N}(\lambda) \leq C_0 \lambda^{-\gamma}$, for some $0 < \gamma \leq 1$.*

406 **Assumption 4.3.** *There exists $s \geq 0$, $1 \leq R < \infty$, such that $\|L_k^{-s} f_{\mathcal{H}}\|_{\mathcal{H}} < R$, where $f_{\mathcal{H}} :=$
407 $\arg \min_f \mathcal{E}(f)$.*

408 Combining the Lipschitz continuity of kernel cylindrical functions (Assumption 3.1) with the above
409 spectral assumptions, we obtain the following finite-sample error bound.

411 **Theorem 4.4** (Nyström Error Bound). *Under Assumptions 3.1, 4.2, and 4.3, let $\delta \in (0, 1)$ and suffi-
412 ciently large $N_i + M$. With probability at least $1 - \delta$, the excess error of the Nyström approximation
413 satisfies*

$$414 \quad \mathcal{E}(\tilde{g}_{t+1}^{i,d}) - \min_{f \in \mathcal{H}} \mathcal{E}(f) \leq C_{k,\gamma} \left(\log \frac{6}{\delta} \right)^2 (N_i + M)^{-\frac{2v+1}{2v+\gamma+1}},$$

416 where $v = \min(s, 1/2)$, $\lambda = \|L_k\| (N_i + M)^{-\frac{1}{2v+\gamma+1}}$, and $L \geq C_\lambda \log \frac{12}{\lambda \delta}$. Constants $C_{k,\gamma}, C_\lambda$
417 depend only on the kernel family.

419 Theorem 4.4 shows that the Nyström approximation converges to the optimal RKHS projection
420 at a rate depending on both the eigenvalue decay γ and the smoothness parameter s . In practice,
421 this means that as the number of anchor points $(N_i + M)$ grows, the approximation error shrinks
422 polynomially fast, and only a logarithmic number of landmark points L (relative to the effective
423 dimension) is needed to achieve near-optimal accuracy. This justifies the use of Nyström projection
424 in SKARL.

426 5 EXPERIMENTS AND RESULTS

428 5.1 EXPERIMENTAL SETUP

430 We evaluate our method following the work of Nayak et al. (2023), with four environments: (i)
431 **Move**: Each agent tries to move as fast as possible and avoid collisions. (ii) **Target**: Each agent tries
to reach the assigned goal and avoid collisions. (iii) **Coverage**: Each agent tries to go to a goal and

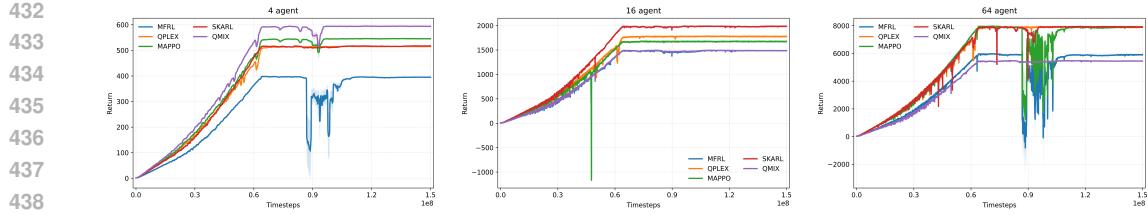


Figure 2: Training results of SKARL and baselines across three environments (5 random seeds).

avoid collisions, and ensure no more than one agent reaching the same goal. (iv) **Line**: There are two landmarks, and the agents try to position themselves equally spread out in a line between the two. For detailed observation, reward and action design, please refer to the Appendix Environments. We compare SKARL against several standard MARL algorithms: (i) **QMIX** (Rashid et al., 2018), (ii) **QPLEX** (Wang et al., 2020a), (iii) **MAPPO** (Yu et al., 2022), (iv) **MFRL** (Yang et al., 2018), and **Fixed-Kernel MFRL** (Cui et al., 2023). For detailed implementation of SKARL and baselines, please refer to Appendix E. We report the test results with 100 max steps.¹

Table 2: Performance Comparison between SKARL and Baselines in Move Environment

Algorithm	N = 4			N = 16			N = 64		
	R(\uparrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	# col(\downarrow)	S(\uparrow)
MAPPO	947.6	0.56 \pm 0.174	0.16 \pm 0.0079	3360.2	2.6 \pm 1.12	0.14 \pm 0.0562	14284.8	9.6 \pm 6.98	0.15 \pm 0.0459
QMIX	835.4	4.94 \pm 4.94	0.15 \pm 0.0431	2845.4	21.9 \pm 12.7	0.13 \pm 0.08517	10446.2	2.8 \pm 1.00	0.11 \pm 0.0732
QPLEX	911.4	0.56 \pm 0.194	0.14 \pm 0.0213	3625.8	20.5 \pm 10.2	0.17 \pm 0.0622	14073.8	22.5 \pm 9.55	0.15 \pm 0.0404
MFRL	734.6	0 \pm 0	0.12 \pm 0.0247	3083.69	38.4 \pm 6.99	0.12 \pm 0.0440	11411.1	204.2 \pm 22.8	0.14 \pm 0.0451
Fixed-Kernel	838.3	0 \pm 0	0.14 \pm 0.0143	3578.13	26.5 \pm 3.52	0.15 \pm 0.0172	13021.5	23.1 \pm 10.8	0.15 \pm 0.0722
SKARL	902.8	0 \pm 0	0.15 \pm 0.0192	3755.9	12.32 \pm 5.847	0.17 \pm 0.0500	14423.8	7.9 \pm 5.37	0.15 \pm 0.034

Table 3: Performance Comparison between SKARL and Baselines in Target Environment

Algorithm	N = 4				N = 16				N = 64			
	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)
MAPPO	327.3	0.14	2.6 \pm 1.45	100	12	0.56	8.93 \pm 5.87	40.6	0	1.00	0	0
QPLEX	330.3	0.18	1.3 \pm 0.982	100	-3.1e4	1.00	4.92 \pm 2.55	0	-1.9e5	1.00	19.8 \pm 12.42	0
QMIX	337.0	0.14	0.67 \pm 0.35	100	-5.1e4	1.00	6.6 \pm 4.28	0	-6.4e5	1.00	41.6 \pm 18.1	0
MFRL	330.8	0.14	6.4 \pm 3.38	100	1.1	0.81	14.375 \pm 10.28	31.2	-5.7e5	1.00	35.8 \pm 15.1	0
Fixed-Kernel	350.8	0.14	5.6 \pm 2.54	100	4.2	0.18	11.23 \pm 8.12	80.2	10.2	0.97	26.7 \pm 8.2	1.2
SKARL	329.3	0.18	7.2 \pm 3.15	100	5.6	0.96	23.2 \pm 20.5	100	44.75	0.98	44.3 \pm 10.6	3.1

5.2 MAIN RESULTS

We report the experiment of main experiments on Move and Target environment with 5 random seeds. For other experiments and ablation study, please refer to Appendix F.

Scale up to large-scale swarms Figure 2, Table 2 and Table 3 demonstrates SKARL’s effectiveness across swarm sizes $N = 4, 16, 64$. We select three metrics: (i) R: global reward. (ii) # col: total collisions. (iii) S: average speed of each agent. For small swarms, SKARL achieves near-optimal reward while entirely eliminating collisions. As the swarm scales to large scale, SKARL outperforms all baselines, achieving the highest reward and fastest speed, with low reduction rate of collision. Notably, SKARL balances safety and efficiency, collisions decrease without sacrificing speed, matching top baselines. These results highlight SKARL’s scalability, particularly excelling in mid-to-large swarms where coordination complexity increases.

Generalize to different swarm sizes Table 4 and Table 5 reveals SKARL’s zero-shot flexibility when tested on varying swarm sizes M . When trained on small swarm size, SKARL fails to maintain reasonable performance up to $M = 256$. However, training on larger swarms ($N = 16/64$) enables robust generalization. Most impressively, $N = 64$ -trained SKARL achieves near-optimal reward

¹Code at <https://anonymous.4open.science/r/SKARL-050D>, based on JaxMARL (Rutherford et al., 2023)

486 Table 4: Zero-Shot Flexibility Performance of SKARL in Move Environment
487

488 Training	489 Metric	490 $M = 4$	491 $M = 8$	492 $M = 16$	493 $M = 32$	494 $M = 64$	495 $M = 128$	496 $M = 256$
$N = 4$	R/N	225.7	168.5	177.8	155.2	166.9	168.5	173.9
	(# col)/N	0 ± 0	2.22 ± 1.18	1.36 ± 0.794	0.62 ± 0.419	0.25 ± 0.146	0.12 ± 0.0745	0.22 ± 0.0762
	S	0.15 ± 0.0190	0.12 ± 0.0842	0.13 ± 0.085	0.12 ± 0.065	0.11 ± 0.0657	0.11 ± 0.0698	0.12 ± 0.0680
$N = 16$	R/N	236.9	235.2	234.7	235.2	225.4	205.6	202.6
	(# col)/N	0 ± 0	0.98 ± 0.437	0.77 ± 0.365	0.57 ± 0.207	0.17 ± 0.115	0.12 ± 0.0652	0.04 ± 0.0221
	S	0.16 ± 0.00434	0.17 ± 0.0612	0.17 ± 0.0500	0.17 ± 0.0469	0.15 ± 0.0323	0.14 ± 0.0521	0.14 ± 0.0542
$N = 64$	R/N	231.5	221.3	227.3	224	223.2	221.6	218.7
	(# col)/N	0 ± 0	0.45 ± 0.408	0.28 ± 0.257	0.44 ± 0.275	0.15 ± 0.109	0.11 ± 0.0866	0.09 ± 0.0476
	S	0.15 ± 0.0126	0.15 ± 0.0406	0.15 ± 0.0237	0.16 ± 0.0591	0.15 ± 0.0459	0.15 ± 0.0401	0.15 ± 0.0436
$N = 256$	R/N	279.1	278.2	263.2	261.5	252.4	237.4	220.8
	(# col)/N	0 ± 0	0 ± 0	0.14 ± 0.235	0.28 ± 0.254	0.17 ± 0.315	0.13 ± 0.312	0.10 ± 0.451
	S	0.17 ± 0.0723	0.17 ± 0.109	0.16 ± 0.124	0.16 ± 0.273	0.16 ± 0.301	0.16 ± 0.334	0.15 ± 0.356

500 Table 5: Flexibility Performance of SKARL in Target Environment
501

502 Training	503 Metric	504 $M = 4$	505 $M = 8$	506 $M = 16$	507 $M = 32$	508 $M = 64$	509 $M = 128$	510 $M = 256$
$N = 4$	R/N	82.3	-36.25	-444.0	-2.8e3	-9.0e3	-1.8e4	-3.7e4
	T (step)	18	95	96.5	100	100	100	100
	(# col)/N	0.5 ± 0.42	23 ± 14.0	37.6 ± 26.9	34.875 ± 11.34	46 ± 15.1	138 ± 18.9	342 ± 32.8
$N = 16$	R/N	85.3	7.5	0.35	-2.4e3	-8.3e3	-1.7e3	-2.6e4
	T (step)	17.4	13.8	96.3	98.5	99.4	100	100
	(# col)/N	0.4 ± 0.13	19.25 ± 13.0	23.2 ± 20.5	34.875 ± 11.34	46 ± 15.1	75.5 ± 14.6	116 ± 21.1
$N = 64$	R/N	84.0	77.3	69.8	10.8	0.70	-0.25	-10.5
	T (step)	18.7	27.8	30.6	67.2	98.1	100	100
	(# col)/N	0.5 ± 0.342	3 ± 2.35	6.7 ± 6.45	16.1 ± 5.83	44.3 ± 10.6	66.3 ± 15.2	96.8 ± 17
$N = 256$	R/N	87.2	80.5	78.6	22.9	20.8	15.4	12.1
	T (step)	18.6	22.2	28.5	30.4	32.8	45.1	60.2
	(# col)/N	0 ± 0	0.5 ± 0.412	1.4 ± 0.24	5.7 ± 1.32	11.2 ± 5.2	12.1 ± 9.4	14.2 ± 10.5
	S%	100	100	100	100	100	98.71	95.21

511 per agent at $M = 256$, while collisions remain the lowest. This flexibility stems from SKARL’s
512 distribution-driven policy as is in Theorem 4.1, enabling deployment in real-world scenarios where
513 swarm sizes are dynamic.

524 6 CONCLUSION

525 We propose **SKARL**, a scalable framework for large-scale multi-agent reinforcement learning.
526 **SKARL** resolves the scalability and flexibility bottlenecks of multi-agent reinforcement learning by
527 enabling linear complexity in swarm size and zero-shot transfer across populations. It ensures con-
528 vergence with efficient updates and drastically reduces training overhead, allowing effective learning
529 in large swarms. Experiments confirm that SKARL outperforms state-of-the-art baselines in both
530 performance and generalization. While our methods offers valuable insights into the representation
531 of mean-field, there are several limitations to consider. Our methods relies on the homogeneous
532 assumption, which limits the application to heterogeneous groups. In the future, we aim to improve
533 the design and extend to heterogeneous MARL problems.

535 7 ETHICS STATEMENT

536 This work introduces **SKARL**, a scalable kernel mean-field reinforcement learning framework for
537 large-scale multi-agent systems. Our contributions are primarily theoretical and methodological,
538 with empirical validation performed in simulated multi-agent environments such as swarm naviga-

540
 541
 542
 543
 544
 545
 546
 547
 548
 549
 550
 551
 552
 553
 554
 555
 556
 557
 558
 559
 560
 561
 562
 563
 564
 565
 566
 567
 568
 569
 570
 571
 572
 573
 574
 575
 576
 577
 578
 579
 580
 581
 582
 583
 584
 585
 586
 587
 588
 589
 590
 591
 592
 593
 tion, coordination, and collision avoidance benchmarks. These environments are widely used in the MARL community and do not involve human subjects, sensitive personal data, or proprietary datasets.

We acknowledge that advances in multi-agent reinforcement learning (MARL) may have dual-use implications. While our experiments are limited to academic and open-source benchmarks, similar techniques could be applied in high-stakes domains such as autonomous vehicle fleets, aerial drone swarms, or defense systems. In such settings, ethical concerns may include safety, accountability, and fairness. To mitigate potential risks, our work remains focused on theoretical scalability and generalization, and we refrain from proposing or testing direct real-world deployment scenarios.

From a fairness perspective, the algorithms studied here are agnostic to sensitive human attributes and do not incorporate demographic information. From a privacy and security perspective, no personal or confidential information is processed. From a research integrity perspective, we strictly adhere to reproducible and transparent reporting, with proofs, assumptions, and algorithms explicitly documented. Finally, we affirm that we have read and adhered to the ICLR Code of Ethics, and have conducted this research in alignment with its principles.

8 REPRODUCIBILITY STATEMENT

We have undertaken comprehensive steps to ensure that the theoretical and empirical results reported in this paper are reproducible. For the theoretical contributions, all assumptions are explicitly stated, and full mathematical proofs are provided either in the main text or in the appendix. These proofs establish the universal approximation property of kernel cylindrical functions and the convergence of the dual time-scale learning rule.

For the empirical results, all experiments are conducted on widely used benchmark environments for multi-agent reinforcement learning, such as large-scale swarm coordination tasks. We describe the experimental setup, training protocols, and hyperparameter configurations in detail within the paper and provide additional clarifications in the appendix. Random seeds are fixed across runs, and ablation studies are reported to verify stability.

To further facilitate reproducibility, we release anonymous source code, including implementations of SKARL, training scripts, and environment configuration files, as part of the supplementary materials. This enables other researchers to directly reproduce the results presented in this paper, adapt the framework to new environments, or verify the theoretical guarantees with empirical evidence. Together, these measures ensure that the community can reliably replicate and build upon our contributions.

REFERENCES

Andrea Angiuli, Jean-Pierre Fouque, and Mathieu Lauriere. Reinforcement learning for mean field games, with applications to economics. *arXiv preprint arXiv:2106.13755*, 2021.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Christopher Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning. *CorR*, abs/1912.06680, 2019. URL <http://arxiv.org/abs/1912.06680>.

Patrick Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2013.

Vivek S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Springer, 2008.

Peter E Caines, Minyi Huang, and Roland P Malhamé. Large population stochastic dynamic games: closed-loop mckean-vlasov systems and the nash certainty equivalence principle. *Communications in Information and Systems*, 6(3):221–252, 2006.

René Carmona, Mathieu Laurière, and Zongjun Tan. Linear-quadratic mean-field reinforcement learning: convergence of policy gradient methods. *arXiv preprint arXiv:1910.04295*, 2019.

594 Kai Cui, Sascha Hauck, Christian Fabian, and Heinz Koepll. Learning decentralized partially ob-
 595 servable mean field control for artificial collective behavior. *arXiv preprint arXiv:2307.06175*,
 596 2023.

597 Ruan de Kock, Omayma Mahjoub, Sasha Abramowitz, Wiem Khelifi, Callum Rhys Tilbury, Claude
 598 Formanek, Andries P. Smit, and Arnu Pretorius. Mava: a research library for distributed multi-
 599 agent reinforcement learning in jax. *arXiv preprint arXiv:2107.01460*, 2023. URL <https://arxiv.org/pdf/2107.01460.pdf>.

600 602 Yali Du, Joel Z Leibo, Usman Islam, Richard Willis, and Peter Sunehag. A review of cooperation
 603 in multi-agent learning. *arXiv preprint arXiv:2312.05162*, 2023.

604 605 Richard Mansfield Dudley. The speed of mean glivenko-cantelli convergence. *The Annals of Math-
 606 ematical Statistics*, 40(1):40–50, 1969.

607 608 Christian Fiedler, Michael Herty, and Sebastian Trimpe. On kernel-based statistical learning theory
 609 in the mean field limit. *Advances in Neural Information Processing Systems*, 36:20441–20468,
 2023.

610 611 Christian Fiedler, Michael Herty, Chiara Segala, and Sebastian Trimpe. Recent kernel methods for
 612 interacting particle systems: first numerical results. *European Journal of Applied Mathematics*,
 613 36(2):464–489, 2025.

614 615 Haotian Gu, Xin Guo, Xiaoli Wei, and Renyuan Xu. Mean-field controls with q-learning for cooper-
 616 ative marl: convergence and complexity analysis. *SIAM Journal on Mathematics of Data Science*,
 3(4):1168–1196, 2021.

617 618 Haotian Gu, Xin Guo, Xiaoli Wei, and Renyuan Xu. Mean-field multiagent reinforcement learning:
 619 A decentralized network approach. *Mathematics of Operations Research*, 50(1):506–536, 2025.

620 621 Xin Guo and Renyuan Xu. Stochastic games for fuel follower problem: N versus mean field game.
 622 *SIAM Journal on Control and Optimization*, 57(1):659–692, 2019.

623 624 Xin Guo, Huyê Pham, and Xiaoli Wei. Itô’s formula for flows of measures on semimartingales.
 625 *Stochastic Processes and their applications*, 159:350–390, 2023.

626 627 David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

628 629 Jean-Michel Lasry and Pierre-Louis Lions. Mean field games. *Japanese journal of mathematics*, 2
 630 (1):229–260, 2007.

631 632 Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. *ICML*,
 633 1994.

634 635 Siting Liu, Matthew Jacobs, Wuchen Li, Levon Nurbekyan, and Stanley J Osher. Computational
 636 methods for nonlocal mean field games with applications. *arXiv preprint arXiv:2004.12210*,
 637 2020.

638 639 Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel
 640 mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine
 641 Learning*, 10(1-2):1–141, 2017.

642 643 Siddharth Nayak, Kenneth Choi, Wenqi Ding, Sydney Dolan, Karthik Gopalakrishnan, and Hamsa
 644 Balakrishnan. Scalable multi-agent reinforcement learning through intelligent information aggre-
 645 gation. In *International Conference on Machine Learning*, pp. 25817–25833. PMLR, 2023.

646 647 Victor M Panaretos and Yoav Zemel. Statistical aspects of wasserstein distances. *Annual review of
 648 statistics and its application*, 6(1):405–431, 2019.

649 650 Huyê Pham and Xavier Warin. Mean-field neural networks: learning mappings on wasserstein
 651 space. *Neural Networks*, 168:380–393, 2023.

652 653 Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John
 654 Wiley & Sons, 1994.

648 Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and
 649 Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforce-
 650 ment learning. *ICML*, 2018.

651 Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational
 652 regularization. *Advances in neural information processing systems*, 28, 2015.

653 Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Gardar Ing-
 654 varsson, Timon Willi, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi
 655 Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Tjarko Lange, Shimon Whiteson,
 656 Bruno Lacerda, Nick Hawes, Tim Rocktaschel, Chris Lu, and Jakob Nicolaus Foerster. Jaxmarl:
 657 Multi-agent rl environments in jax. *arXiv preprint arXiv:2311.10090*, 2023.

658 Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines,
 659 regularization, optimization, and beyond*. MIT press, 2002.

660 Friedhelm Schwenker, Hans A Kestler, and Günther Palm. Three learning phases for radial-basis-
 661 function networks. *Neural networks*, 14(4-5):439–458, 2001.

662 Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zam-
 663 baldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Grae-
 664 pel. Value-decomposition networks for cooperative multi-agent learning. *CoRR*, abs/1706.05296,
 665 2017. URL <http://arxiv.org/abs/1706.05296>.

666 Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*,
 667 3(1):9–44, 1988.

668 Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan
 669 Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning.
 670 *CoRR*, abs/1511.08779, 2015. URL <http://arxiv.org/abs/1511.08779>.

671 Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *In Proceed-
 672 ings of the Tenth International Conference on Machine Learning*, pp. 330–337. Morgan Kauf-
 673 mann, 1993.

674 Huaze Tang, Yuanquan Hu, Fanfan Zhao, Junji Yan, Ting Dong, and Wenbo Ding. M³arl:
 675 Moment-embedded mean-field multi-agent reinforcement learning for continuous action space.
 676 In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Process-
 677 ing (ICASSP)*, pp. 7250–7254. IEEE, 2024.

678 Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-
 679 learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

680 Daniele Venturi and Alec Dektor. Spectral methods for nonlinear functionals and functional differ-
 681 ential equations. *Research in the Mathematical Sciences*, 8(2):27, 2021.

682 Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Juny-
 683 oung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan
 684 Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, John P. Agapiou, Max
 685 Jaderberg, Alexander Sasha Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David
 686 Budden, Yury Sulsky, James Molloy, Tom Le Paine, Caglar Gulcehre, Ziyun Wang, Tobias Pfaff,
 687 Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom
 688 Schaul, Timothy P. Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver.
 689 Grandmaster level in starcraft II using multi-agent reinforcement learning. *Nature*, pp. 1–5, 2019.

690 Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling
 691 multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020a.

692 Lingxiao Wang, Zhuoran Yang, and Zhaoran Wang. Breaking the curse of many agents: Provable
 693 mean embedding q-iteration for mean-field reinforcement learning. In *International conference
 694 on machine learning*, pp. 10092–10103. PMLR, 2020b.

695 Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines.
 696 *Advances in neural information processing systems*, 13, 2000.

702 Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean field multi-
 703 agent reinforcement learning. *ICML*, 2018.

704
 705 Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The
 706 surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information
 707 processing systems*, 35:24611–24624, 2022.

708
 709 **A THE USE OF LLM**

710
 711 In the preparation of this paper, we employed large language models (LLMs) strictly as assistive
 712 tools. Their role was confined to three aspects: (i) improving the clarity and readability of the
 713 manuscript by suggesting stylistic refinements and alternative phrasings; (ii) assisting with the orga-
 714 nization and presentation of mathematical proofs, including the checking of algebraic manipulations
 715 and the polishing of logical exposition; and (iii) serving as a coding assistant for routine program-
 716 ming tasks such as code completion, debugging, and documentation generation.

717 Importantly, LLMs were not involved in the generation of research ideas, the design of the SKARL
 718 framework, or the conceptual development of the theoretical results. All scientific insights, algo-
 719 rithmic designs, and experimental implementations originate from the authors. The LLM usage
 720 did not extend to generating novel theorems, creating data, or drawing conclusions. Instead, the
 721 models functioned in a supportive capacity, comparable to grammar-checking or code editor auto-
 722 completion, with the final responsibility for correctness, originality, and integrity resting solely on
 723 the authors.

724 We disclose this usage in alignment with ICLR policy. By transparently reporting the scope of
 725 assistance, we affirm that the LLMs were used responsibly and ethically, and that the intellectual
 726 contributions of this work are entirely attributable to the authors.

727
 728 **B PROOFS OF THEOREMS, LEMMAS AND PROPOSITIONS**

729
 730 **B.1 PROOF OF THEOREM 3.1**

731
 732 *Proof.* We establish the density of proposed $\mathcal{G}_D(\mathcal{M})$. To this end, we first need:

733
 734 **Lemma B.1** (Stone–Weierstrass). *Take a compact Hausdorff space H , and let $\mathcal{C}(H)$ be the algebra
 735 of real-valued continuous functions on H , with the topology of uniform convergence. Let \mathcal{A} be a
 736 subalgebra of $\mathcal{C}(H)$. If \mathcal{A} separates points on H and vanishes at no point on H , then \mathcal{A} is dense in
 $\mathcal{C}(H)$.*

737 Then, following the proof of Lemma 3.12 in Guo et al. (2023), we prove that with appropriate
 738 choices of norms, $\mathcal{G}_D(\mathcal{M})$ is dense in $\mathcal{C}^{1,1}(\mathcal{M})$.

739
 740 **Lemma B.2.** $\mathcal{G}_D(\mathcal{M})$ is dense in $\mathcal{C}^{1,1}(\mathcal{M})$ with the supremum norm of derivatives of all orders: for
 $\Phi \in \mathcal{C}^{1,1}(\mathcal{M})$,

$$741 \|\Phi\|_{\mathcal{M}} := \sup_{(\nu, x) \in \mathcal{P}(\mathcal{M}) \times \mathcal{M}} \left(|\Phi(\nu)| + |\partial_\mu \Phi(\mu)(x)| + \|\partial_x \partial_\mu \Phi(\mu)(x)\| \right)$$

742
 743 We prove this with two steps:

744
 745 **Step 1:** take $\Phi \in \mathcal{C}^{1,1}(\mathcal{M})$, then $\partial_{xx} \frac{\delta \Phi}{\delta \mu}(\mu, x)$ is a continuous function on $\mathcal{P}(\mathcal{M}) \times \mathcal{M}$ by definition,
 746 namely, $\partial_{xx} \frac{\delta \Phi}{\delta \mu}(\mu, x) \in \mathcal{C}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$. Define the algebraic space that contains $\mathcal{G}_D(\mathcal{M})$ for some
 747 $n \in \mathbb{N}$ as

$$748 \mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M}) := \left\{ \Phi(\mu, x) = \sum_{k=1}^n f^k(\langle g^k, \mu \rangle) h^k(x), \right.$$

$$749 \left. \text{monomials } f^k, h^k : \mathbb{R}^D \rightarrow \mathbb{R}, \text{ kernels } g^k : \mathcal{M} \rightarrow \mathcal{M} \right\}.$$

750 We can see the $\mathcal{G}_D(\mathcal{M})$ can be viewed as a subalgebra of $\mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$. Additionally, we can
 751 also see that

756 • $\mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$ separates points on $\mathcal{P}(\mathcal{M}) \times \mathcal{M}$. To check this, take $(\mu, x) \neq (\mu', x') \in$
 757 $\mathcal{P}(\mathcal{M}) \times \mathcal{M}$, with either $\mu \neq \mu'$ or $x \neq x'$. If $\mu' \neq \mu$, from Theorem 30.1 by Billingsley
 758 (2013), there exists a kernel function $k(x_0, \cdot)$ such that $\int_{\mathcal{M}} k(y, x)(\mu - \mu')(dx) \neq 0$,
 759 otherwise, $\mu = \mu'$. In this case, define $p(\mu, x) = \langle k(x_0, x) \rangle \in \mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$. If $\mu' = \mu$,
 760 $x' \neq x$, let $p(\mu, x) = x$, then $p(\mu, x) \neq p(\mu', x')$. In either case, $\mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$ separates
 761 points on $\mathcal{P}(\mathcal{M}) \times \mathcal{M}$.

762 • $\mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$ vanishes at no point on $\mathcal{P}(\mathcal{M}) \times \mathcal{M}$. It can be checked to choose a nonzero
 763 constant function as f_k and h_k .

765 Therefore, it follows from the Stone-Weierstrass lemma that $\mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$ is dense in $\mathcal{C}(\mathcal{P}(\mathcal{M}) \times$
 766 $\mathcal{M})$ with the topology of uniform convergence. Hence, there exists a sequence of functions $p_n, \tilde{p}_n \in$
 767 $\mathcal{H}(\mathcal{P}(\mathcal{M}) \times \mathcal{M})$ such that for any $\epsilon > 0$, there exists $N \in \mathbb{N}$ that for $n \geq N$,

$$769 \quad \sup_{(\mu, x) \in \mathcal{P}(\mathcal{M}) \times \mathcal{M}} \left| p_n(\mu, x) - \partial_{xx} \frac{\delta \Phi}{\delta \mu}(\mu, x) \right| \leq \epsilon, \quad (10)$$

771 and

$$773 \quad \sup_{\mu \in \mathcal{P}(\mathcal{M})} \left| \tilde{p}_n(\mu) - \frac{\delta \Phi}{\delta \mu}(\mu, 0) \right| \leq \epsilon. \quad (11)$$

775 **Step 2:** Let

$$777 \quad P_n(\mu, x) := \tilde{p}_n(\mu) + \int_0^x \int_0^y p_n(\mu, z) dz dy,$$

779 and

$$780 \quad \Phi_n(\mu) := \Phi(\delta_0) + \int_0^1 \int_{\mathcal{M}} P_n(\lambda \mu + (1 - \lambda) \delta_0, x)(\mu - \delta_0)(dx) d\lambda.$$

783 It can be checked that $\Phi_n \in \mathcal{G}_D(\mathcal{M})$ with polynomial kernels. Now we have

$$\begin{aligned} 785 \quad & P_n(\mu, x) - \frac{\delta \Phi}{\delta \mu}(\mu, x) \\ 786 \quad &= \tilde{p}_n(\mu) + \int_0^x \int_0^y p_n(\mu, z) dz dy - \\ 787 \quad & \left(\frac{\delta \Phi}{\delta \mu}(\mu, 0) + \int_0^x \int_0^y \partial_{xx} \frac{\delta \Phi}{\delta \mu}(\mu, z) dz dy \right) \\ 789 \quad &= \tilde{p}_n(\mu) - \frac{\delta \Phi}{\delta \mu}(\mu, 0) + \int_0^x \int_0^y \left(p_n(\mu, z) dz - \partial_{xx} \frac{\delta \Phi}{\delta \mu}(\mu, z) \right) dz dy. \end{aligned}$$

794 Thus, by Eq. (10),

$$795 \quad \sup_{\mathcal{P}(\mathcal{M}) \times \mathcal{M}} |\partial_x P_n(\mu, x) - \partial_\mu \Phi(\mu, x)| \leq K\epsilon,$$

$$797 \quad \sup_{\mathcal{P}(\mathcal{M}) \times \mathcal{M}} \left| P_n(\mu, x) - \frac{\delta \Phi}{\delta \mu}(\mu, x) \right| \leq (1 + K^2)\epsilon.$$

800 Moreover,

$$802 \quad \Phi_n(\mu) - \Phi(\mu)$$

$$\begin{aligned} 803 \quad &= \left(\Phi(\delta_0) + \int_0^1 \int_{\mathcal{M}} P_n(\lambda \mu + (1 - \lambda) \delta_0, x)(\mu - \delta_0)(dx) d\lambda \right) \\ 804 \quad &- \left(\Phi(\delta_0) + \int_0^1 \int_{\mathcal{M}} \frac{\delta \Phi}{\delta \mu}(\lambda \mu + (1 - \lambda) \delta_0, x)(\mu - \delta_0)(dx) d\lambda \right) \\ 805 \quad &= \int_0^1 \int_{\mathcal{M}} \left(P_n(\lambda \mu + (1 - \lambda) \delta_0, x) - \frac{\delta \Phi}{\delta \mu}(\lambda \mu + (1 - \lambda) \delta_0, x) \right) (\mu - \delta_0)(dx) d\lambda. \end{aligned}$$

810 Hence,

$$811 \sup_{\mathcal{P}(\mathcal{M})} |\Phi_n(\mu) - \Phi(\mu)| \leq 2(1 + K^2)\epsilon.$$

812 Therefore,

$$813 \|\Phi_n - \Phi\|_{\mathcal{M}} \leq (1 + K + 2(1 + K^2))\epsilon,$$

814 with $\Phi_n \in \mathcal{G}_D(\mathcal{M})$, which is shown to be dense in $C^{1,1}(\mathcal{M})$.

815 \square

816 B.2 STATEMENT AND PROOF OF WASSERSTEIN LIPSCHITZ CONTINUOUS

817 **Lemma B.3** (Wasserstein Lipschitz Continuous). *If Assumption 3.1 holds, then cylindrical function*
 818 *$h(\mu) \in \mathcal{G}_D(\mathcal{M})$ is C -Lipschitz continuous according to $\mu \in \mathcal{P}(\mathcal{M})$, i.e., for any measure $\mu, \nu \in$*
 819 *$\mathcal{P}_2(\mathcal{M})$, there holds*

$$820 |h(\nu_0) - h(\nu_1)| \leq C\mathcal{W}_2(\nu_0, \nu_1), \quad (12)$$

821 where C is a constant.

822 *Proof.* Since the kernels g^d are uniformly bounded, the input space for outer function h are actually is compact. Therefore, outer function $h : \mathbb{R}^D \rightarrow \mathbb{R}$ (a polynomial function) is L_h -Lipschitz continuous:

$$823 |h(z_1) - h(z_2)| \leq L_h \|z_1 - z_2\|_2, \quad \forall z_1, z_2 \in \mathcal{G}, \quad (13)$$

824 where $\mathcal{G} \subset \mathbb{R}^D$ is a compact subspace. Let π be the optimal coupling between ν_0 and ν_1 . Then:

$$825 |h(\nu_0) - h(\nu_1)| \leq L_h \left(\sum_{d=1}^D |\langle g^d, \mu_{\nu_0} - \mu_{\nu_1} \rangle_{\mathcal{H}_k}|^2 \right)^{1/2}$$

$$830 \leq L_h \sqrt{D} \max_{1 \leq d \leq D} |\langle g^d, \mu_{\nu_0} - \mu_{\nu_1} \rangle_{\mathcal{H}_k}|.$$

831 Therefore, we have that

$$832 |h(\nu_0) - h(\nu_1)|^2 \leq L_h^2 D \max_{1 \leq d \leq D} |\langle g^d, \mu_{\nu_0} - \mu_{\nu_1} \rangle_{\mathcal{H}_k}|^2$$

$$833 \leq L_h^2 D \max_d \left| \int_{\mathcal{X}} (g^d(x))^2 (\mathrm{d}\nu_0 - \mathrm{d}\nu_1)(x) \right|$$

$$834 \leq L_h^2 D \inf_{\pi} \max_d \int_{\mathcal{X} \times \mathcal{X}} (g^d(x))^2 \mathrm{d}\pi(x, y)$$

$$835 \leq L_h^2 D L_g^2 \inf_{\pi} \int_{\mathcal{X} \times \mathcal{X}} \|x - y\|_2^2 \mathrm{d}\pi(x, y)$$

$$836 = C\mathcal{W}_2(\mu, \nu)^2,$$

837 where the last inequality follows from the Kantorovich-Rubinstein duality. Therefore, we have that

$$838 |h(\nu_0) - h(\nu_1)| \leq L_d \sqrt{D} L_g \mathcal{W}_2(\mu, \nu). \quad (14)$$

839 \square

840 B.3 PROOF OF PROPOSITION 3.2

841 *Proof.* We provide derivation of Proposition 3.2. From Eq. (6), we have the form of Q^i . Then, the
 842 functional gradient in the form of Fréchet derivative is

$$843 \nabla_{g^{i,d}} Q^i = \frac{\delta h_{s^i, a^i}}{\delta g} + \sum_{d'=1}^D \frac{\delta(\partial_{d'} h_{s^i, a^i} \langle \nabla g^{i, d'}(x) \cdot \Delta x, \nu^{-i} \rangle)}{\delta g}$$

$$844 = \partial_d h_{s^i, a^i} \mu_{\nu^{-i}} + \sum_{d'=1}^D \frac{\delta(\partial_{d'} h_{s^i, a^i})}{\delta g} \langle \nabla g^{i, d'}(x) \cdot \Delta x, \nu^{-i} \rangle$$

$$845 + \partial_d h_{s^i, a^i} \frac{\delta \langle \nabla g^{i, d}(x) \cdot (\bar{x}^i - x), \nu^{-i}(x) \rangle}{\delta g}.$$

To calculate the last term in $\nabla_{g^{i,d}} Q^i$, we apply the fundamental lemma of calculus of variations. Define function $f(x, g, \nabla g) = g^{i,d}(x) \cdot (\bar{x}^i - x) \nu^{-i}(x)$, then, $\langle \nabla g^{i,d}(x) \cdot (\bar{x}^i - x), \nu^{-i}(x) \rangle$ can be written as

$$\begin{aligned} & \langle \nabla g^{i,d}(x) \cdot (\bar{x}^i - x), \nu^{-i}(x) \rangle \\ &= \int_{\mathcal{M}} \nabla g^{i,d}(x) \cdot (\bar{x}^i - x) \nu^{-i}(x) dx \\ &= \int_{\mathcal{M}} f(x, g, \nabla g) dx. \end{aligned}$$

Therefore, we have that

$$\frac{\delta \langle \nabla g^{i,d}(x) \cdot (\bar{x}^i - x), \nu^{-i}(x) \rangle}{\delta g} = \frac{\partial f}{\partial g} - \nabla \cdot \frac{\partial f}{\partial \nabla g} = -\nabla \cdot ((\bar{x}^i - x) \nu^{-i}(x)).$$

Hence, we have the form in Proposition 3.2.

$$\begin{aligned} \nabla_{g^{i,d}} Q^i &= \partial_d h_{s^i, a^i} \mu_{\nu^{-i}} + \sum_{d'=1}^D \frac{\partial_{dd'}^2 h_{s^i, a^i}}{N_i} \sum_{j=1}^{N_i} \nabla g^{i,d'}(x^j) \cdot (\bar{x}^i - x^j) \mu_{\nu^{-i}} \\ &+ \partial_d h_{s^i, a^i} \nabla \cdot (\nu^{-i}(x)(x - \bar{x}^i)) \end{aligned}$$

□

B.4 PROOF OF THEOREM 4.1

Proof. Under Assumption 3.1, we know that the cylindrical function $h(\mu)$ is Wasserstein continuous by Lemma B.3. Therefore, we have that

$$|h(\nu_n) - h(\nu_M)| \leq C \mathcal{W}_2(\nu_N, \nu_M).$$

Since Wasserstein distance meets the triangle inequality (Panaretos & Zemel, 2019), we have that

$$\mathcal{W}_2(\nu_N, \nu_M) \leq \mathcal{W}_2(\nu_N, \nu) + \mathcal{W}_2(\nu_M, \nu).$$

Since the convergence rate of empirical distribution ν_N to ν under measure of Wasserstein distance is $O(N^{-1/d})$ (Dudley, 1969), namely,

$$\mathbb{E}[\mathcal{W}_2(\nu_N, \nu)] \leq CN^{-1/d}.$$

Therefore, we have that

$$\begin{aligned} \mathbb{E}[|h(\nu_n) - h(\nu_M)|] &\leq C \mathbb{E}[\mathcal{W}_2(\nu_N, \nu)] + C \mathbb{E}[\mathcal{W}_2(\nu_M, \nu)] \\ &\leq C_1 N^{-1/d} + C_2 M^{-1/d}. \end{aligned}$$

□

B.5 PROOF OF THEOREM 4.2

Proof. First, we prove that the convergence rate of cylindrical function is controlled by the convergence rate of empirical kernel mean embedding.

Lemma B.4 (Convergence Rate Bound of Kernel Cylindrical Functions (Lemma 5.2, (Venturi & Dektor, 2021))). *Denote the projection of measure ν on RKHS embedding space \mathcal{H}_M as $\mathcal{P}_D \nu = \sum_d c_d k(x^d, \cdot)$, where $[c_1, \dots, c_D]^\top =: \mathbf{c} = (\mathbf{K}_{DD})^{-1} \mathbf{b}$ and $b_d = \langle k(x^d, \cdot), \nu \rangle$. We have that h defined in Eq. (4) with one type of kernel converges to f for all $\nu \in \mathcal{P}_2(\mathcal{M})$ with the same rate as $\mathcal{P}_D \nu$ converges to the kernel mean embedding μ_ν . Formally, with $\tilde{f} : \mu_\nu \mapsto f(\nu)$, it can be expressed as*

$$|h(\nu) - f(\nu)| \leq \sup_{\nu} \left\| \frac{\delta \tilde{f}}{\delta \mu_\nu} \right\| \|\mu_\nu - \mathcal{P}_D \nu\|_{\mathcal{H}}, \quad (15)$$

where $\delta \tilde{f} / \delta \mu_\nu$ is the Fréchet derivative of function \tilde{f} and μ_ν is the kernel mean embedding defined in Eq. (3).

918 From Lemma B.4, the convergence rate of the cylindrical function is controlled by the convergence
 919 rate of the empirical kernel mean embedding.
 920

921 **Lemma B.5** (Convergence Rate of Empirical Kernel Mean Embedding (Theorem 3.4, (Muandet
 922 et al., 2017))). *Assume the boundedness for kernel k in Assumption 3.1 holds. Then for any $\delta \in$
 923 $(0, 1)$ with probability at least $1 - \delta$,*

$$924 \|\mu_\nu - \mathcal{P}_D \nu\|_{\mathcal{H}} \leq \sqrt{\frac{1}{D}} + \sqrt{\frac{2 \log(1/\delta)}{D}}. \quad (16)$$

926 Combining the results from Lemma B.5, we have that the convergence rate of h to f is the multiple
 927 of Fréchet derivative and $O(D^{-1/2})$, which proves our results. \square
 928

929 **B.6 PROOF OF THEOREM 4.3**

931 *Proof.* First, we introduce the non-linear two-time-scale stochastic approximation.

932 **Lemma B.6** (Nonlinear Two-Time-Scale Stochastic Approximation (Borkar, 2008)). *Consider two
 933 coupled stochastic approximation processes:*

$$935 x_{n+1} = x_n + a(n) \left[f(x_n, y_n) + M_n^{(1)} \right], \quad (17)$$

$$936 y_{n+1} = y_n + b(n) \left[g(x_n, y_n) + M_n^{(2)} \right], \quad (18)$$

938 where $x_n \in \mathbb{R}^d$ (slow process) and $y_n \in \mathbb{R}^k$ (fast process), with step sizes $a(n), b(n) > 0$.

939 Assume that

941 (i) $f : \mathbb{R}^d \times \mathbb{R}^k \rightarrow \mathbb{R}^d$ and $g : \mathbb{R}^d \times \mathbb{R}^k \rightarrow \mathbb{R}^k$ are Lipschitz continuous,

942 (ii) For each fixed x , the ODE $\dot{y}(t) = g(x, y(t))$ has a globally asymptotically stable equilibrium $y^*(x)$.
 943 The ODE $\dot{x}(t) = f(x(t), y^*(x(t)))$ has a globally asymptotically stable equilibrium x^* ,

944 (iii) the sequences $\{a(n)\}$ and $\{b(n)\}$ satisfy Robbins-Monro conditions in Assumption 4.1, and

945 (iv) $\{M_n^{(1)}\}, \{M_n^{(2)}\}$ are martingale differences w.r.t. $\mathcal{F}_n = \sigma(x_m, y_m, M_m^{(1)}, M_m^{(2)}, m \leq n)$, with

$$946 \mathbb{E} \left[\|M_n^{(i)}\|^2 \mid \mathcal{F}_n \right] \leq C(1 + \|x_n\|^2 + \|y_n\|^2), \quad i = 1, 2.$$

947 Then, the iterates (x_n, y_n) converge almost surely to (x^*, y^*) , where $y^* = y^*(x^*)$.

948 Base on the Lemma B.6, we rewrite updates of Eq. 7 as stochastic approximation processes:

$$949 h_{t+1} = h_t + \eta_h \left(F_h(h_t, g_t) + M_h^{t+1} \right), \quad (19a)$$

$$950 g_{t+1} = g_t + \eta_g \left(F_g(h_t, g_t) + M_g^{t+1} \right), \quad (19b)$$

951 where $F_h = -\mathbb{E} \left[\frac{\partial \ell}{\partial Q_{\text{tot}}} \cdot \frac{\partial Q_{\text{tot}}}{\partial Q^i} \nabla_h Q^i \right]$ and F_g is defined analogously. M_h, M_g are martingale difference
 952 noise terms.

953 By the SA theory (Borkar, 2008), the updates approximate:

$$954 \text{(Fast)} \quad \dot{g} = F_g(h, g), \quad (20a)$$

$$955 \text{(Slow)} \quad \dot{h} = F_h(h, g^*(h)), \quad (20b)$$

956 where $g^*(h)$ is the equilibrium of Eq. (20a) for fixed h .

957 Since the Bellman operator is a contraction mapping (Littman, 1994), we have that there exists a
 958 globally asymptotically stable equilibrium g^* and h^* to minimize ℓ . Therefore, by the Lemma B.6, we
 959 have that:

- 960 • The fast process Eq. (19b) tracks Eq. (20a), converging to $g^*(h_t)$ for any slow h_t .
- 961 • The slow process Eq. (19a) converges to h^* , which induces $g^* = g^*(h^*)$.

962 Thus, $(h_t, g_t) \rightarrow (h^*, g^*)$ almost surely. \square

972 B.7 PROOF OF THEOREM 4.4
973974 *Proof.* Theorem 4.4 is the same with Theorem 1 in (Rudi et al., 2015). Define the integral operator
975 L_k for kernel function k by

976
$$977 L_k f(x) = \int_{\mathcal{X}} f(s)k(x, s)ds.$$

978

979 For $\lambda > 0$, define the random variable $\mathcal{N}_x(\lambda) = \langle K_x, (L_k + \lambda I)^{-1} K_x \rangle$ with $x \in \mathcal{X}$. The efficient
980 dimension is

981
$$\mathcal{N}(\lambda) = \mathbb{E} \mathcal{N}_x(\lambda), \quad \mathcal{N}_\infty(\lambda) = \sup_{x \in \mathcal{X}} \mathcal{N}_x(\lambda).$$

982

983 **Theorem B.7** (Error Analysis of Nyström Approximation, Theorem 1 (Rudi et al., 2015)). *Under
984 Assumption 3.1, 4.2 and 4.3, let $\delta \in (0, 1)$, $v = \min(s, 1/2)$, $p = 1 + 1/(2v + \gamma)$ and assume*

985
$$986 N_i + M \geq 1655 + 223 \log \frac{6}{\delta} + \left(\frac{38p}{\|L_k\|} \log \frac{114p}{\|L_k\|\delta} \right)^p \quad (21)$$

987

988 *Then, the following inequality holds with probability at least $1 - \delta$ for,*

989
$$990 \mathcal{E}(\tilde{g}_{t+1}^{i,d}) \leq \min_{f \in \mathcal{H}} \mathcal{E}(f) + q^2 (N_i + M)^{-\frac{2v+1}{2v+\gamma+1}}, \quad (22)$$

991

992 *with*

993
$$994 q = 6R \left(2\|L_k\| + \frac{C_1}{\sqrt{\|L_k\|}} + \sqrt{\frac{C_2}{\|L_k\|^\gamma}} \right) \log \frac{6}{\delta},$$

995

996 *C_1, C_2 are constants, and $\lambda = \|L_k\|(N_i + M)^{-\frac{1}{2v+\gamma+1}}$ and $L \geq \max(67, 5\mathcal{N}_\infty(\lambda)) \log \frac{12}{\lambda\delta}$.*997 \square
998999
1000 In our scenario, for a large swarm with batch size, the $N_i + M$ will easily meet the assumption in
1001 Theorem B.7. For example, if a swarm of $N = 32$ with batch size $B = 128$, along with kernel
1002 number $M = 64$, $N_i + M = B \cdot N + M$ will be 4160, which may satisfy the assumption with
1003 certain δ .1004 C APPENDED REMARKS
10051006 C.1 REMARKS ON KERNEL CYLINDRICAL FUNCTIONS AND MEAN FIELD EMBEDDING
10071008 **Remarks C.1** (Requirements on kernel by Lipschitz continuity). *The Lipschitz continuity requirement
1009 limits the choice of kernel functions. Such as*1010
1011

- **Polynomial kernels:** $k(y, x) = (\alpha x \cdot y + c)^d$ violates the condition when input space \mathcal{X} is
1012 unbounded, as the gradients grow polynomially with $\|x\|_2$.
- **Sigmoid kernels:** $k(y, x) = \tanh(\alpha x \cdot y + c)$ could fail to satisfy global Lipschitz continuity
1013 due to saturation effects in nonlinear regions.
- **Gaussian kernels:** $k(y, x) = \exp(-\gamma\|x - y\|_2^2)$ generally meet the requirement with $L_g =$
1014 $\gamma \sup_x \|x\|_2/2$.

1015 **Remarks C.2** (Inner Product between mean-field measure and component functions). *The inner
1016 product between mean field measure and component function $g^{i,d}$ evaluates to:*

1017
$$1018 \langle g^{i,d}, \mu_{\nu^{-i}}^d \rangle = \frac{1}{N_i} \sum_{m=1}^M \sum_{j=1}^{N_i} \alpha_m^d k^d(x^m, x^j) = \frac{\mathbf{1}^\top \mathbf{K}^d \boldsymbol{\alpha}^d}{N_i}, \quad (23)$$

1019

1020 where $\mathbf{K}^d \in \mathbb{R}^{N_i \times M}$ is the Gram matrix with $\mathbf{K}_{jm}^d = k^d(x^j, x^m)$ and $\mathbf{1} \in \mathbb{R}^{N_i}$ is an all-ones
1021 vector.

Table 6: Kernel Functions and Corresponding Partial Derivative

Kernel Type	Kernel $k(y, x)$	Gradient of kernel $\partial_x k(y, x)$
Linear	$x \cdot y + c$	y
Polynomial	$(\alpha x \cdot y + c)^d$	$\alpha d(\alpha x \cdot y + c)^{d-1} y$
Gaussian	$\exp(-\gamma \ x - y\ ^2)$	$-2\gamma(x - y) \exp(-\gamma \ x - y\ ^2)$
Laplacian	$\exp(-\gamma \ x - y\ _1)$	$-\gamma \text{sign}(x - y) \exp(-\gamma \ x - y\ _1)$
Sigmoid	$\tanh(\alpha x \cdot y + c)$	$\alpha y(1 - \tanh^2(\alpha x \cdot y + c))$

D REMARKS ON KERNEL FUNCTIONS

We list several kernels frequently appearing in the literature.

In our work, in consideration of Lipschitz continuity, representation capability and easy to calculate, we adopt polynomial and Gaussian kernels.

D.1 REMARKS ON MEAN-FIELD REPRESENTATION OF VALUE FUNCTIONS

Remarks D.1 (Expansion of Eq. (6)). *Eq. 6 is expanded as:*

$$Q^i(\mathbf{s}, \mathbf{a}) = h_{s^i, a^i} \left(\frac{\mathbf{1}^\top \mathbf{K}^1 \boldsymbol{\alpha}^1}{N_i}, \dots, \frac{\mathbf{1}^\top \mathbf{K}^D \boldsymbol{\alpha}^D}{N_i} \right) + \frac{1}{N_i} \sum_{d=1}^D \partial_d h_{s^i, a^i} \sum_{m=1}^M \sum_{j=1}^{N_i} \alpha_m^d \partial_x k^d(x^m, x^j) \cdot (\bar{x}^i - x^j).$$

Remarks D.2 (Mean field representation of state value function and advantage function). *Similarly, we can present the state value function $V^i(\mathbf{s})$ and advantage function $A^i(\mathbf{s}, \mathbf{a})$ with the mean field representation in Eq. (6) as*

$$V^i(\mathbf{s}) = h_{s^i}^v (\langle g_v^{i,1}, \mu_{\nu^{-i}} \rangle, \dots, \langle g_v^{i,D}, \mu_{\nu^{-i}} \rangle) + \sum_{d=1}^D \partial_d h_{s^i}^v \langle \nabla g_v^{i,d}(x) \cdot \Delta x, \nu^{-i} \rangle,$$

and

$$A^i(\mathbf{s}) = h_{s^i, a^i}^{adv} (\langle g_{adv}^{i,1}, \mu_{\nu^{-i}} \rangle, \dots, \langle g_{adv}^{i,D}, \mu_{\nu^{-i}} \rangle) + \frac{1}{N_i} \sum_{d=1}^D \partial_d h_{s^i, a^i}^{adv} \langle \nabla g_{adv}^{i,d}(x) \cdot \Delta x, \nu^{-i} \rangle,$$

where $h_{s^i}^v$ and h_{s^i, a^i}^{adv} are the cylindrical kernel functions, with kernel functions $\{g_v^{i,d}\}$ and $\{g_{adv}^{i,d}\}$ for value function V and advantage function A , respectively. In this paper, we focus on the Q function, while we think it is also interesting to expand our conclusions to value and advantage functions.

Remarks D.3 (Explicit form of Fréchet derivative). *In discrete particle approximation with N_i neighbors, Eq. (8) is:*

$$\nabla_{g^{i,d}} Q^i = \sum_{j=1}^{N_i} \left[\frac{\partial_d h}{N_i} + \sum_{d'} \frac{\partial_{dd'}^2 h}{N_i^2} \sum_{j'} \nabla g^{d'}(x^{j'}) \Delta x^{j'} \right] k^d(x^j, \cdot) + \frac{\partial_d h}{N_i} \sum_{j=1}^{N_i} [\delta_{x^j} - \nabla \delta_{x^j} \cdot \Delta x^j].$$

D.2 REMARKS ON NYSTRÖM APPROXIMATION

Remarks D.4. *The gradient inner product admits explicit computation:*

$$\begin{aligned} \langle k(x^n, \cdot), \nabla_{g^{i,d}} Q^i \rangle &= \sum_{j=1}^{N_i} \left[\frac{2\partial_d h}{N_i} + \sum_{d'} \frac{\partial_{dd'}^2 h}{N_i^2} \sum_{j'} \nabla g^{d'}(x^{j'}) \cdot \Delta x^{j'} \right] k^d(x^n, x^j) \\ &\quad - \frac{\partial_d h}{N_i} \sum_{j=1}^{N_i} \nabla_x k^d(x^n, x^j) \cdot \Delta x^j \end{aligned} \tag{24}$$

1080
1081 **Remarks D.5** (Anchor Point Selection). *There are several principled ways to choose anchor points*
1082 $\{z^l\}_{l=1}^L$:

1083 • **Random Subsampling:** Select L points uniformly from RKHS anchor points $\{x^n\}_{n=1}^{N_i+M}$ in
1084 $g_{t+1}^{i,d}$.

$$z^l \sim \text{Uniform}(\{x^n\}_{n=1}^{N_i+M}), \quad l = 1, \dots, L.$$

1085 *Pros:* $O(1)$ computational cost. *Cons:* May miss important regions.

1086 • **k -means Centers:** Solve

$$\{z^l\} = \arg \min_{\{c_l\}} \sum_{x \in \{x_m\}} \min_{1 \leq l \leq L} \|x - c_l\|^2.$$

1087 *Pros:* Captures data geometry. *Cons:* $O(N_i LT)$ computation complexity for T iterations.

1088 • **Kernel Herding:** Select points maximizing the minimum kernel similarity:

$$z_{l+1} = \arg \max_{x \in \{x_m\}} \sum_{l'=1}^l k(x, z_{l'}) - \frac{2}{N_i} \sum_{j=1}^{N_i} k(x, x^j).$$

1089 *Pros:* Constructs maximally representative points. *Cons:* $O(N_i LT)$ computation complexity for T iterations.

1090 • **Leverage Score Sampling:** Sample with probability proportional to diagonal entries of the
1091 kernel matrix:

$$p_j = \frac{(K_{MM})_{jj}}{\text{tr}(K_{MM})}, \quad z^l \sim p_j.$$

1092 *Pros:* Preserves spectral structure of the RKHS.

1093 *In this paper, we apply the random subsampling method for simplicity.*

1108 E IMPLEMENTATION DETAILS OF SKARL AND BASELINES

1111 E.1 IMPLEMENTATION DETAILS OF SKARL

1112 **Base Algorithm of Credit Assignment for SKARL** We apply VDN (Sunehag et al., 2017) as the
1113 basic credit assignment algorithm for SKARL. Namely, the total Q_{tot} value is calculated by

$$1115 Q_{\text{tot}}(\mathbf{s}, \mathbf{a}) = \sum_{i=1}^N Q^i(\mathbf{s}, \mathbf{a}).$$

1116 **Kernel Cylindrical Function Implementation** We adopt a hypernetwork (Ha et al., 2016) for
1117 kernel cylindrical function network. Namely, the ego state and action (s^i, a^i) are used to generate
1118 the parameters of a network for processing $\mu_{\nu_{N_i}}$.

1119 **Tricks** We apply several tricks to help stabilize and fasten training.

1120 • **Dual Network Update:** To avoid over-estimation of Q value, we apply double Q learning
1121 framework (Van Hasselt et al., 2016).

1122 • **Entropy Regularization:** To avoid the performance drops in the last epochs during training,
1123 we apply entropy regularization on the actor policy.

1124 **Codebase** We apply SKARL and baselines with Jax. We organize the code in JaxMARL (Rutherford
1125 et al., 2023) for better organization and class inheritance. We plan to release full codes afterwards.
1126 For now, the code for important implementation can be found via anonymous Github link:
1127 <https://anonymous.4open.science/r/SKARL-050D>.

1128

1129 **Hyperparameters** In this paragraph, we list the hyperparameters in 7 and 8.

1134
1135
1136 Table 7: Environment & Training Configuration
1137
1138

Environment		Training		Optimizer	
Hyperparameter	Value	Hyperparameter	Value	Hyperparameter	Value
Agent Number	4 / 16 / 64	Total Time Steps	2M	Learning Rate	7e-4
Environments Number	128	Update Steps Number	50	Max Grad Norm	10
Test Environment Number	8	Target Update Interval	8	Optimizer	ADAM
Max Train Env Timesteps	50	Test Interval	50k	EPS	1e-5
Max Test Env Timesteps	100			Weight Decay	0
Buffer		Exploration		Learning rate Decay	
Hyperparameter	Value	Hyperparameter	Value	Hyperparameter	Value
Buffer Size	8192	Epsilon	1.0 → 0.05	η_h	$1/t^{0.6}$
Buffer Batch Size	32	Epsilon Anneal Time	50k	η_g	$1/t^{0.8}$
Buffer Sample	Uniform	Anneal Method	Linear	Basic LR	7e-5

1142
1143
1144
1145
1146 Table 8: Network & Algorithm Architecture
1147
1148

Network		Algorithm	
Hyperparameter	Value	Hyperparameter	Value
Embedding Net Layer	3	TD Lambda	0.95
Agent Hidden Dim	16	Gamma	0.99
Mixer Embedding Dim	256	Entropy Rate	0.5
Mixer Hypernet Hidden Dim	256	Anchor Points Number	$L = 64$
Attention Dim	64	Tikhonov Coefficient	0.5
Activation	ReLU	Polynomial Kernel	$(\alpha, d, c) = (1, 2, 1), (1, 3, 1)$
FC Init Scale	2.0	Gaussian Kernel	$\gamma = 0.5, 1.0$

1161 E.2 COMMON SETTINGS FOR ENVIRONMENT
11621163 For learning stability and environment consistency, we conduct following tricks:
11641165 **Re-scale of Environment** To make environment scalable, we conduct re-scale of world size of
1166 environment according to the agents as below:
1167

1168
$$\text{world size} = 2 * \min(\sqrt{N} - 1, 1),$$

1169

1170 where world size serves as the boundary value of environment as $[-\text{world size}, \text{world size}] \times$
1171 $[-\text{world size}, \text{world size}]$ and N denotes the number of agents.
11721173 **Reset of Agents and Landmarks** We generate the new agents and landmark uniformly in the
1174 world of environment, namely, $p^i \sim \text{Uniform}([-\text{world size}, \text{world size}] \times [-\text{world size}, \text{world size}])$
1175 for $i \in \{1, \dots, N\}$. In some implementations, a reject sampling is adopted to avoid collision
1176 between generated agents and landmarks (such as codebase of InforMARL (Nayak et al., 2023),
1177 JaxMARL (Rutherford et al., 2023), Mava (de Kock et al., 2023) and so on). However, we do
1178 not adopt such rejection, due to the consideration of time consumption. Instead, we separate the
1179 environment world into grids and sample among grids to avoid collision.
11801181 During both training and evaluation phases in the **Target** and **Coverage** environments, the episode
1182 terminates and resets automatically once all agents successfully reach their assigned goals (or all
1183 landmarks are uniquely covered for the Coverage task). This design ensures episodic training and
1184 prevents infinite loops. However, since agents are able to receive one-time rewards for several
1185 times, the total episodic reward may temporarily exceed the theoretical maximum (e.g., $N \times 10$ for
1186 N agents) during resets due to reward accumulation in the final timestep.
11871188 **Size and Velocity Settings of Agents and Landmarks** The settings for agents and landmarks are
1189 listed as below in 9.
1190

1188
1189
1190 Table 9: Environment Setup
1191
1192
1193
1194
1195
1196
1197

Hyperparameter	Value
Agent Size	0.15
Landmark Size	0.225
Agent Maximum Speed	0.65 (Move) N/A (Target/Coverage)
Agent Acceleration	5 (Move) 2 (Target/Coverage)

1198
1199 F ADDITIONAL EXPERIMENTS
12001201 F.1 EXPERIMENTS IN TARGET ENVIRONMENT
12021203 In this section, we provide the analysis of results for task **Target**. The experimental results in the
1204 Target environment demonstrate SKARL’s ability to maintain task performance while balancing
1205 safety and scalability across different swarm sizes.
12061207 For small swarms ($N = 4$), SKARL achieves near-optimal performance with a global reward of
1208 329.3, comparable to QMIX (337.0) and QPLEX (330.3), while ensuring a 100% success rate.
1209 However, it exhibits a higher collision count (7.2 ± 3.15) compared to QMIX (0.67 ± 0.35) and
1210 QPLEX (1.3 ± 0.982), suggesting a trade-off between task completion and collision avoidance in
1211 simpler settings.
12121213 As the swarm scales to $N = 16$, SKARL significantly outperforms value-based methods (QMIX,
1214 QPLEX, MFRL), which suffer from catastrophic reward degradation (e.g., QPLEX: -3.1×10^4).
1215 Although MAPPO achieves a higher reward (12.0), its success rate drops to 40.6%, whereas SKARL
1216 maintains a 100% success rate despite increased collisions (23.2 ± 20.5). Additionally, SKARL
1217 reduces collisions by 32% compared to MFRL, indicating its robustness in mid-scale coordination,
1218 which aligns with findings from the Move environment in 2.
12191220 In large-scale swarms ($N = 64$), SKARL demonstrates superior scalability, achieving a positive
1221 reward (44.75) while all baselines fail (rewards ≤ 0). Notably, while the collision count remains
1222 high (44.3 ± 10.6), the drastic improvement in reward over MFRL (-5.7×10^5) and QMIX ($-6.4 \times$
1223 10^5) suggests that SKARL effectively prevents catastrophic failures in complex scenarios. The low
1224 success rate (3.1%) implies that further optimization is needed for very large swarms, but the results
1225 highlight SKARL’s ability to maintain functional performance where other methods collapse.
12261227 Overall, SKARL exhibits strong scalability in the Target environment, particularly excelling in main-
1228 taining task success and reward stability as swarm size increases, with a trade-off in collision avoid-
1229 ance at larger scales. This aligns with its performance in the Move environment, where it achieves
1230 a 96% collision reduction at $N = 64$, reinforcing its effectiveness in large-scale multi-agent coor-
1231 dination. However, the problem of scaling up in Target environment remains to be solved, which
1232 require further works.
12331234 F.2 EXPERIMENTS IN COVERAGE ENVIRONMENT
12351236 Table 10: Performance Comparison between SKARL and Baselines in Coverage Environment
1237

Algorithm	$N = 4$				$N = 16$				$N = 64$			
	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)
MAPPO	339.6	0.40	0.26 ± 0.561	1.00 ± 0.0	167.6	0.57	5.3 ± 2.72	0.13 ± 0.562	97.3	0.87	18.4 ± 8.35	0.05 ± 0.009
MFRL	396.6	0.52	0.03 ± 0.0	1.00 ± 0.0	187.0	0.62	2.5 ± 1.65	0.12 ± 0.456	216.2	0.86	15.1 ± 2.32	0.04 ± 0.871
QMIX	275.4	0.39	4.94 ± 2.46	1.00 ± 0.0	259.5	0.52	19.5 ± 5.3	0.19 ± 0.76	324.2	0.92	11.8 ± 4.13	0.10 ± 0.526
QPLEX	318.5	0.38	0.56 ± 0.194	1.00 ± 0.0	298.7	0.61	7.3 ± 6.22	0.21 ± 0.512	834.5	0.85	21.5 ± 3.65	0.14 ± 0.290
SKARL	387.2	0.51	0.15 ± 0.870	1.00 ± 0.0	320.8	0.61	2.42 ± 1.67	0.22 ± 0.342	907.3	0.76	15.3 ± 5.37	0.17 ± 0.0832

Table 11: Flexibility Performance of SKARL in Coverage Environment

Training	Metric	$M = 4$	$M = 16$	$M = 64$	$M = 128$	$M = 256$
$N = 4$	R/N	96.8	23.7	0.3	-1.2	-9.2
	T (step)	51	74	92	100	100
	(# col)/N	0.0375	0.76	6.932	32.4	78.9
	S%	100	72	4	0	0
$N = 16$	R/N	97.5	24.05	22.3	4.3	0.82
	T (step)	43	61	67	94	100
	(# col)/N	0.0457	0.19	0.203	2.54	5.21
	S%	100	79	6	6.25	0
$N = 64$	R/N	96.2	25.8	14.2	9.3	3.52
	T (step)	41	56	76	89	92
	(# col)/N	0.0557	0.285	0.239	0	9.68
	S%	100	84	13	75	5

Table 12: Performance Comparison between SKARL and Baselines in Line Environment

Algorithm	$N = 4$				$N = 16$				$N = 64$			
	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	T(\downarrow)	# col(\downarrow)	S(\uparrow)
MAPPO	422.3	0.31	0.10 ± 0.20	1.00 ± 0.00	563.4	0.43	1.50 ± 0.90	0.30 ± 0.20	1462.7	0.72	8.00 ± 3.00	0.22 ± 0.08
MFRL	444.8	0.25	0.05 ± 0.10	1.00 ± 0.00	591.2	0.43	0.90 ± 0.60	0.36 ± 0.18	1604.3	0.68	6.00 ± 2.50	0.27 ± 0.09
QMIX	421.6	0.25	0.12 ± 0.25	1.00 ± 0.00	572.1	0.49	1.80 ± 1.10	0.32 ± 0.19	1510.4	0.64	7.20 ± 2.80	0.24 ± 0.09
QPLEX	449.7	0.27	0.07 ± 0.15	1.00 ± 0.00	608.0	0.42	1.20 ± 0.70	0.38 ± 0.17	1624.9	0.67	6.50 ± 2.60	0.26 ± 0.09
SKARL	418.9	0.23	0.03 ± 0.08	1.00 ± 0.00	615.6	0.41	0.70 ± 0.50	0.40 ± 0.16	1765.8	0.66	5.50 ± 2.20	0.30 ± 0.10

F.3 EXPERIMENTS IN LINE ENVIRONMENT

F.4 ABLATION STUDY

Is it necessary to apply gradient in RKHS? There is another way to conduct gradient for cylindrical function: directly update in the Euclidean space (Schwenker et al., 2001). Here we provide a comparison with this method with $N = 4$ and kernel number is 64 in Move environment in Figure 3. The result indicates that with RKHS gradient, both the training stability and final performance are improved.

How number of anchors affect the result? We compare the performance of different anchor points number $L = 1, 2, 8, 32$ under Move task with agent number $N = 4$. As is demonstrated in Figure 4, more anchor points only help to stabilize the training process (as the performance of $L = 32$ achieves the most stale training curve), while the convergence speed and final performance is scarcely affected. Furthermore, since full performance can be achieved with anchor points number 1, it is indicated that SKARL can apply at least one kernel number L with $L \leq \sqrt{N}$ to achieve lower computation complexity compared with value decomposition algorithms e.g. QMIX (as discussed in Section 4).

How types of kernels affect the result? We compare specific choices of different kernels under Move task with agent number $N = 4$. Specifically, we compare the choice of Gaussian kernel and polynomial kernel. For the Gaussian kernel, we adopt γ as $(0.5, 1.0, 2.0)$ and for polynomial kernel, we set parameters as $(\alpha, d, c) = (1, 2, 1), (1, 3, 1), (1, 4, 1)$. The results are demonstrated in Figure 5. We conclude that the choice of kernels may not affect the final performance, as long as the representation capability of this kernel is strong enough.

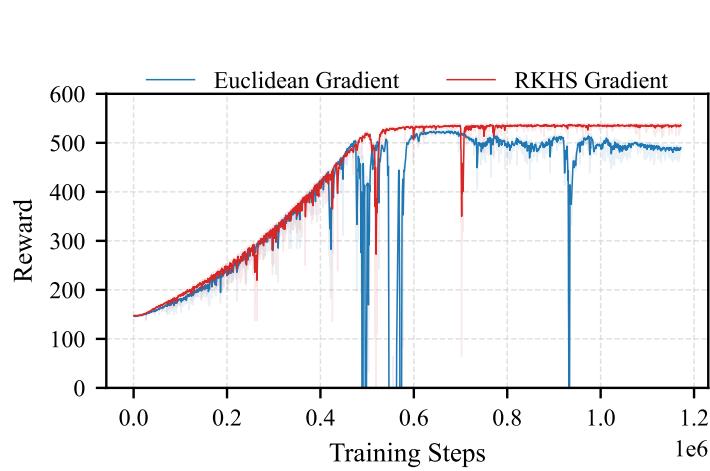


Figure 3: Comparison between gradient in RKHS space and Euclidean space.

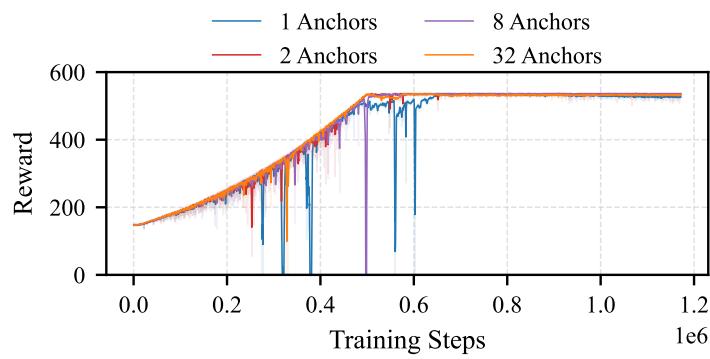


Figure 4: Comparison between different number of anchor points.

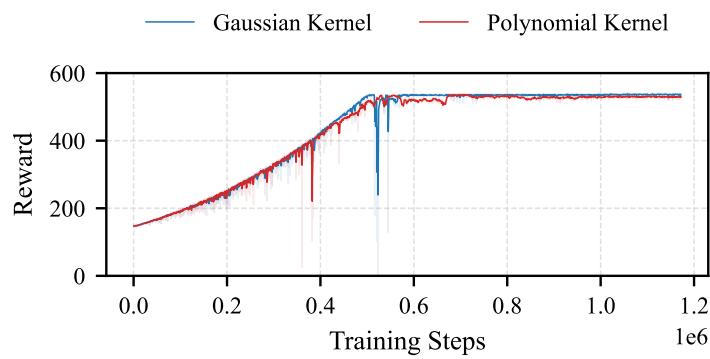


Figure 5: Comparison between different kernel types.

1349

Table 13: Flexibility Performance of SKARL in Line Environment

Training	Metric	$M = 4$	$M = 16$	$M = 64$	$M = 128$	$M = 256$
$N = 4$	R/N	104.7	32.2	-4.3	-10.5	-36.4
	T (step)	23	54	87	100	100
	(# col)/N	0.0075	0.076	0.950	4.1	12.1
	S%	100	72	8	0	0
$N = 16$	R/N	117.5	38.5	20.4	6.3	0.72
	T (step)	32	41	84	91	100
	(# col)/N	0.0005	0.044	0.103	0.874	1.54
	S%	100	40	24	3.25	0
$N = 64$	R/N	123.2	53.4	27.58	18.9	2.31
	T (step)	21	31	66	77	82
	(# col)/N	0.0002	0.029	0.085	0.376	0.985
	S%	100	84	30	27	18

Table 14: Performance Comparison between different value decomposition methods.

Algorithm	$N = 4$			$N = 16$			$N = 64$		
	R(\uparrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	# col(\downarrow)	S(\uparrow)	R(\uparrow)	# col(\downarrow)	S(\uparrow)
SKARL	902.8	0 ± 0	0.15 ± 0.0192	3755.9	12.32 ± 5.847	0.17 ± 0.0500	14423.8	7.9 ± 5.37	0.15 ± 0.0334
SKARL-QMIX	921.2	0 ± 0	0.15 ± 0.0102	3857.2	10.23 ± 8.421	0.18 ± 0.0431	14512.3	6.2 ± 4.32	0.15 ± 0.781
SKARL-QPLEX	922.7	0 ± 0	0.15 ± 0.0021	3920.1	9.42 ± 3.412	0.18 ± 0.0622	14589.1	7.9 ± 2.98	0.16 ± 0.676

How does anchor points distribute? We plot the distribution of anchor points with UMAP in Figure 6 with $N = 4$. We can see the anchor points of Gaussian kernel follows nearly a uniform distribution, while anchor points of polynomial kernel follows certain pattern.

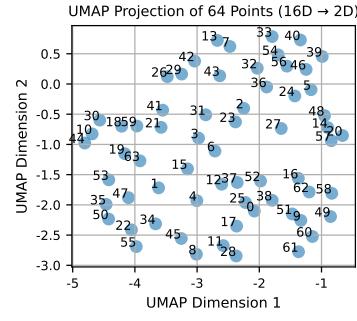
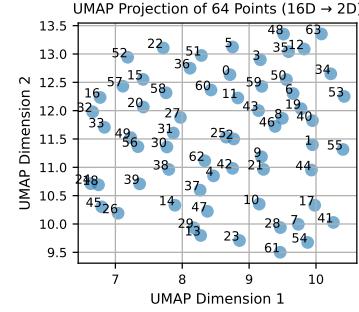
(a) Gaussian Kernel $\gamma = 0.5$ (b) Gaussian Kernel $\gamma = 1.0$

Figure 6: Gaussian kernel anchor points distribution.

1404
 1405
 1406
 1407
 1408
 1409
 1410
 1411
 1412
 1413
 1414
 1415
 1416
 1417
 1418
 1419
 1420
 1421
 1422
 1423
 1424
 1425
 1426
 1427
 1428
 1429
 1430
 1431
 1432
 1433
 1434

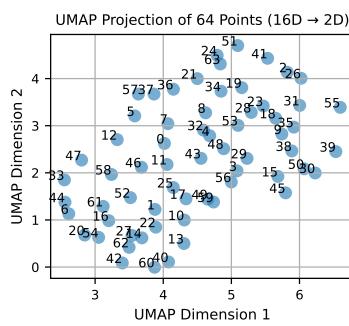
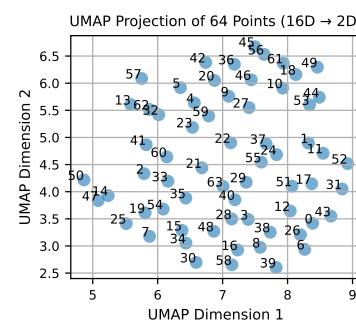
(a) Polynomial Kernel $(\alpha, d, c) = (1, 2, 1)$ (b) Polynomial Kernel $(\alpha, d, c) = (1, 3, 1)$

Figure 7: Polynomial kernel anchor points distribution.

1435
 1436
 1437
 1438
 1439
 1440
 1441
 1442
 1443
 1444
 1445
 1446
 1447
 1448
 1449
 1450
 1451
 1452
 1453
 1454
 1455
 1456
 1457