

Coordinated Q-Functionals

Yasin Findik and Reza Azadeh

Abstract—Learning efficiently in continuous-action multi-agent environments remains a central challenge in reinforcement learning. While *value-based* algorithms offer strong sample efficiency in discrete domains, extending them to continuous spaces is non-trivial due to the difficulty of evaluating infinite action sets. *Policy-based* methods address this challenge by using critic networks to guide and stabilize the learning process, yet they often struggle with high variance in gradient estimates and convergence to local optima. Departing from the typical reliance on critics, we propose Coordinated Q-Functionals (CoQF) — a novel *value-based* algorithm for learning in continuous multi-agent domains. CoQF represents each agent’s state using basis functions, enabling efficient evaluation of densely sampled actions and coordinated mixing of value estimates across agents. This architecture supports both effective coordination and improved sample efficiency. Empirical results across six cooperative tasks show that CoQF consistently outperforms four Deep Deterministic Policy Gradient (DDPG) variants in both learning speed and final performance.

I. INTRODUCTION

Multi-agent reinforcement learning (MARL) seeks to equip multiple agents with the ability to learn policies that maximize collective or individual returns within a shared environment. Broadly, MARL methods fall into two paradigms: *value-based* [1] approaches, which estimate action-value functions to inform decision-making, and *policy gradient* [2] approaches, which optimize parameterized policies through gradients of expected returns. While *value-based* methods have shown superior sample efficiency, requiring fewer interactions with the environment to learn effective behavior, they are inherently difficult to apply to continuous action spaces, where enumerating or comparing Q-values across infinitely many actions becomes intractable.

In practice, most state-of-the-art MARL algorithms for continuous control rely on policy-based methods, such as Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [3] and Multi-Agent Proximal Policy Optimization (MAPPO) [4]. These methods extend their single-agent counterparts (DDPG [5], PPO [6]) to multi-agent scenarios using centralized critics and decentralized execution. However, these methods are often sample-inefficient, suffering from high-variance gradient estimates, approximation errors in the critic, and difficulty converging to globally optimal solutions—especially in multi-agent settings where coordination is critical [7].

Recent developments in single-agent reinforcement learning have reopened the conversation around value-based meth-

ods for continuous control. Notably, Q-functionals [8]—a generalization of Q-learning that models Q-values as continuous functions over the action space—has demonstrated that value-based approaches can, in fact, scale to continuous domains when designed appropriately. By evaluating many actions in parallel, Q-functionals enable efficient exploration and Q-value estimation without resorting to policy gradients in single-agent continuous settings.

In this paper, we extend these insights to the multi-agent setting and introduce Coordinated Q-Functionals (CoQF)—a value-based algorithm designed for learning in continuous-action multi-agent environments. CoQF equips agents with expressive Q-functionals and facilitates cooperation by mixing their value representations across agents. In this way, CoQF brings the sample efficiency of value-based learning to continuous-action multi-agent environments, without relying on unstable policy gradients. We evaluate CoQF across six cooperative continuous control tasks spanning two distinct environments, comparing its performance against both *value-* and *policy-based* baselines. Our key contributions are as follows:

- (i) We introduce CoQF along with two variants, Independent Q-functionals (Ind-QF) and Centralized Q-functionals (Ctr-QF), as the first *value-based* methods designed specifically for multi-agent reinforcement learning in continuous action spaces.
- (ii) To the best of our knowledge, this is the first work to demonstrate that *value-based* methods can effectively outperform *policy-based* approaches in cooperative multi-agent reinforcement learning with continuous action spaces, challenging the long-standing assumption that *value-based* strategies are unsuitable for such settings.
- (iii) Through comprehensive experiments, we show that CoQF consistently outperforms DDPG-based methods, and in five out of six tasks, it matches or exceeds the performance of FACMAC, an enhanced policy-based variant and state-of-the-art algorithm, while achieving faster convergence and improved sample efficiency.

II. BACKGROUND AND RELATED WORK

In reinforcement learning, the goal is to enable agents to learn behavior that maximizes their expected cumulative reward through repeated interaction with an environment. When extended to multi-agent settings, this framework is often formalized using the Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [9], which captures the challenges of partial observability and decentralized decision-making among multiple agents.

Authors are with the Persistent Autonomy and Robot Learning (PeARL) Lab, Richard Miner School of Computer and Information Sciences, University of Massachusetts Lowell, MA, USA
yasin.findik@student.uml.edu, reza@cs.uml.edu

A Dec-POMDP is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{R}, \mathcal{T}, N, \gamma \rangle$, where:

- \mathcal{S} denotes the set of true environmental states, with $s \in \mathcal{S}$ representing the current state,
- \mathcal{A} , \mathcal{O} , and \mathcal{R} represent the sets of joint actions, observations, and rewards for the agents,
- $\mathcal{T}(s, \mathcal{A}, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ defines the probability of transitioning to state, s' , given the current state, s , and joint action, \mathcal{A} ,
- N is the total number of agents involved, and
- $\gamma \in [0, 1)$ is a discount factor applied to future rewards to reflect their diminishing significance over time.

This formalism provides a foundation for reasoning about multi-agent interaction in partially observable, stochastic tasks, and underpins a wide range of MARL algorithms.

A. Value-Based Learning: Single to Multi-Agent

Value-based reinforcement learning methods focus on estimating action-value functions that guide decision-making. In single-agent settings, Q-learning [1] learns the expected return of taking an action a in state s under a policy π :

$$Q_\pi(s, a) = \mathbb{E}[G | s_t = s, a_t = a], \quad (1)$$

with the recursive Bellman update:

$$Q_\pi(s, a) = \mathbb{E}_{s'}[R(s, a) + \gamma \mathbb{E}_{a' \sim \pi}[Q_\pi(s', a')]], \quad (2)$$

where $R(s, a)$ and γ represent the reward function and the discount factor, respectively.

Deep Q-Networks (DQN) [10] extend Q-learning to high-dimensional inputs using neural networks to approximate $\hat{Q}(s, a)$. DQN introduces stability through a target network, whose parameters are periodically updated, and an experience replay buffer, and minimizes the loss:

$$L(w) = \mathbb{E}_{s, a, r, s'}[(r + \gamma \max_{a'}(\hat{Q}(s', a'; w^-)) - \hat{Q}(s, a; w))^2], \quad (3)$$

where w and w^- are the parameters of the prediction and target networks, respectively.

Extending value-based methods to multi-agent settings introduces key challenges. In Independent Q-Learning (IQL) [11], each agent learns its own Q-function individually. However, as agents update their policies concurrently, the environment becomes non-stationary from each agent's perspective, undermining convergence guarantees. Fully centralized methods address non-stationarity by learning a joint Q-function, but suffer from poor scalability due to exponential growth in the joint action and observation spaces [12].

To balance coordination and scalability, value function factorization methods were proposed under the centralized training with decentralized execution (CTDE) paradigm [13]. Approaches like Value Decomposition Networks (VDN) [14] and QMIX [15] learn individual Q-functions $\hat{Q}^i(s^i, a^i)$ and combine them into a global estimate Q_{total} . VDN uses a simple summation:

$$Q_{\text{total}} = \sum_{i=1}^N \hat{Q}^i(s^i, a^i), \quad (4)$$

while QMIX learns a monotonic mixing function:

$$Q_{\text{total}} = f_s(\hat{Q}^1(s^1, a^1), \dots, \hat{Q}^n(s^n, a^n)), \quad \text{with } \frac{\partial f_s}{\partial \hat{Q}^i} \geq 0. \quad (5)$$

These methods have achieved strong results in discrete settings but remain difficult to apply in continuous action spaces, where the action space is infinite and value estimates over all actions become computationally infeasible. As a result, MARL in continuous domains has mostly relied on policy-gradient methods, despite their sample inefficiency.

B. Policy-Based Learning: Single to Multi-Agent

Policy-based methods are widely used in continuous control tasks, where value-based approaches struggle with infinite action spaces. Instead of estimating action values, they directly optimize a parameterized policy π_θ . The policy gradient (PG) theorem [2] defines the update rule as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim p_\pi, a \sim \pi_\theta}[\nabla_\theta \log \pi_\theta(a|s) Q_\pi(s, a)], \quad (6)$$

where p_π represents the state distribution under policy π . Variants of PG methods differ in how they estimate Q_π . REINFORCE [16] uses Monte Carlo returns, while actor-critic methods [17] reduce variance by learning a critic.

The PG framework also supports deterministic policies (DPG) [18], where actions are selected via $\mu_\theta(s)$, a deterministic policy defined as $\mu_\theta : \mathcal{S} \mapsto \mathcal{A}$, with a policy vector of m parameters $\theta \in \mathbb{R}^m$ and the gradient becomes:

$$\nabla_\theta J(\mu_\theta) = \mathbb{E}_{s \sim p_\mu}[\nabla_\theta \mu_\theta(s) \nabla_a Q_\mu(s, a)|_{a=\mu_\theta(s)}]. \quad (7)$$

Deep DPG (DDPG) [5] extends DPG using deep neural networks for the actor and critic, with stability ensured through target networks and experience replay.

In the multi-agent setting, Multi-Agent DDPG (MADDPG) [3] builds on DDPG by assigning each agent its own actor and centralized critic. Each critic conditions on the joint observations and actions of all agents, while policies are executed independently (for details, see [3]). Though effective in many domains, MADDPG often suffers from sample inefficiency and convergence to local optima, especially in complex environments with high-dimensional action spaces.

These limitations have motivated recent efforts to improve the sample efficiency of actor-critic methods in multi-agent settings. FACMAC [19] introduces a factored centralized critic and a centralized gradient estimator to support coordinated learning in continuous domains. VFS [20], on the other hand, enhances value prediction by perturbing multiple critics during training. While both methods improve performance, they remain grounded in policy-gradient techniques, which inherently suffer from sampling inefficiencies. In contrast, our work adopts a value-based approach to continuous multi-agent reinforcement learning, aiming to overcome these limitations through more efficient action evaluation and coordination.

III. PROPOSED METHOD

To address the inefficiencies of policy-based approaches, we introduce Coordinated Q-Functionals (CoQF)—a novel value-based algorithm that enables agents to learn in

continuous-action multi-agent domains. CoQF brings the sample efficiency of value-based learning to continuous domains by extending the concept of Q-functionals [8] to multi-agent settings under the centralized training and decentralized execution (CTDE) paradigm to ensure scalability and coordination. To evaluate the effectiveness of this approach, we also present two baseline variants: Independent Q-Functionals (Ind-QF) and Centralized Q-Functionals (Ctr-QF). Before introducing our proposed MARL algorithms, we begin with a brief review Q-functionals in the single-agent setting, which serve as the foundation for the methods presented later—Ind-QF, Ctr-QF, and CoQF.

A. Q-functionals

Q-functionals [8] reformulate the Q-function to enable parallel evaluation of continuous actions. Unlike traditional Q-networks that map a state-action pair directly to a scalar estimate (i.e., $\hat{Q}(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$), Q-functionals decouple the state and action representation. Specifically, each state is mapped to a function over the action space:

$$\hat{Q}_f(s, a) : \mathcal{S} \mapsto (\mathcal{A} \mapsto \mathcal{R}). \quad (8)$$

This is achieved by learning a set of basis function coefficients conditioned on the state, which can then be multiplied with action representations to compute Q-values efficiently, through matrix operations. In other words, state-action values can be represented as:

$$Q_f(s, a) = C(s)^\top \Phi(a), \quad (9)$$

where $C(s)$ are the learned coefficients and $\Phi(a)$ is the action feature vector.

This structure enables fast, batched evaluation of many candidate actions in continuous spaces using matrix operations, addressing a key limitation of traditional value-based methods in single-agent settings.

B. Ctr-QF: Centralized Q-Functionals

Ctr-QF defines a single, centralized Q-functional over the joint state-action space. The mapping becomes:

$$\hat{Q}_f^{\text{ctr}}(s, a) : \mathcal{S} \mapsto (\mathcal{A} \mapsto \mathcal{R}), \quad (10)$$

where s and a represent the joint states and actions of all agents. While this formulation captures inter-agent dependencies, it scales poorly: both the action and observation spaces grow exponentially with the number of agents, undermining the sample efficiency that Q-functionals are designed to provide.

C. Ind-QF: Independent Q-Functionals

Ind-QF applies Q-functionals to each agent independently, allowing agent i to learn its own functional \hat{Q}_f^i as in standard Independent Q-Learning [11]. Although straightforward in design, this approach fails to account for the non-stationarity introduced by simultaneous learning among agents, and thus lacks convergence guarantees—even under unlimited exploration. In our experiments, we use Ind-QF as a baseline to highlight the advantage of incorporating coordination in multi-agent settings.

D. CoQF: Coordinated Q-Functionals

CoQF addresses the limitations of both Ind-QF (non-stationarity) and Ctr-QF (poor scalability) by combining their strengths. Each agent learns its own Q-functional, maintaining decentralized policy execution, while training is coordinated through a centralized mixing function over the agents' individual Q-values. This allows the model to capture inter-agent dependencies while keeping the action and observation spaces for each agent tractable.

The mixing mechanism in CoQF is flexible and can be instantiated as: an additive sum (as in (4), VDN [14]), a monotonic mixing network (as in (5), QMIX [15]), or a more expressive structure (e.g., QTRAN [21] or Weighted QMIX [22]). The design choice depends on the coordination complexity required by the environment.

What sets CoQF apart from prior value-based MARL methods is its seamless integration of Q-functionals with continuous action spaces, where traditional value-based methods struggle. By representing states via learned basis functions, agents can evaluate many candidate actions efficiently, selecting the most promising ones without relying on high-variance gradients. Moreover, CoQF follows CTDE paradigm, ensuring that agents remain independently deployable at test time while benefiting from global learning signals during training.

In the experiments, we benchmark CoQF against four DDPG-based algorithms—Independent DDPG (Ind-DDPG), Centralized DDPG (Ctr-DDPG), MADDPG, and FACMAC—and show that it offers improved learning stability and competitive performance across a range of continuous multi-agent tasks. As illustrated in Fig. 1, our framework consists of three primary components for each learner: (i) episode generation, (ii) training (of prediction blocks), and (iii) target (block) updates.

(i) Episode Generation

At each timestep, agents interact with the environment using actions selected from their prediction blocks, as illustrated in Fig. 1(a). Each agent uses a neural network to output a set of coefficients $C(s)$ for its current state. These coefficients define a function over the action space, enabling the agent to compute Q-values as:

$$Q_f(s, a) = C(s)^\top \Phi(a), \quad (11)$$

where $\Phi(a)$ is a vector representation of action a , based on a predefined basis function (e.g., polynomial). The number of coefficients k depends on the order \mathcal{O} of the basis and the action dimension D , computed as $\binom{D+\mathcal{O}}{\mathcal{O}}$.

For example, with a second-order *polynomial* basis and a 2D action space $a = [a_1, a_2]$, the coefficient and action feature vectors are:

$$C(s) = [c_0 \quad c_1 \quad c_2 \quad c_3 \quad c_4 \quad c_5]^\top, \\ \Phi(a) = [1 \quad a_1 \quad a_1^2 \quad a_2 \quad a_2^2 \quad a_1 a_2]^\top,$$

yielding the Q-value:

$$Q_f(s, a) = c_0 + c_1 a_1 + c_2 a_1^2 + c_3 a_2 + c_4 a_2^2 + c_5 a_1 a_2. \quad (12)$$

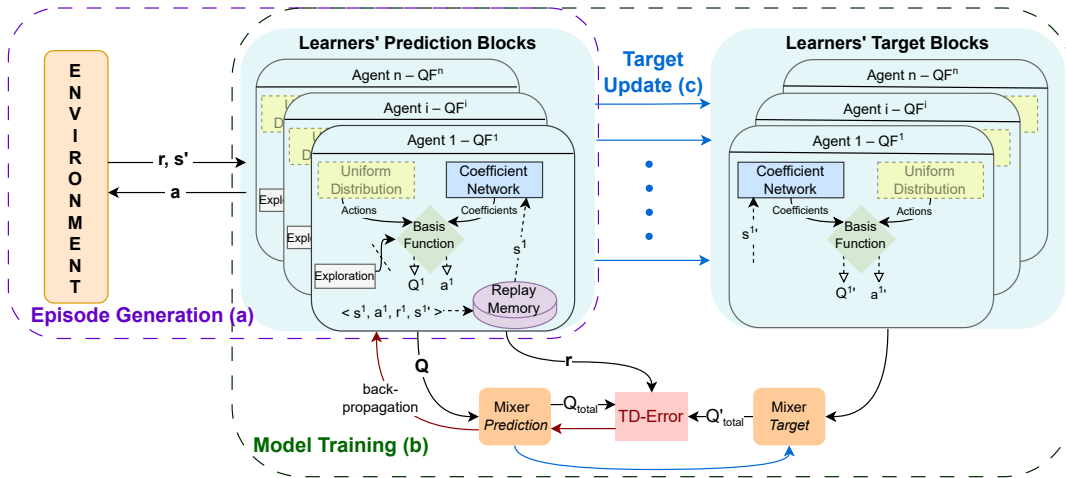


Fig. 1: CoQF framework illustrated across three components: (a) agents generate episodes by interacting with the environment using Q-functionals, (b) model training involves computing temporal-difference errors and updating the prediction networks via backpropagation (red arrows), and (c) target networks for both the learners and the mixer are incrementally updated (blue arrows) to stabilize learning.

Thanks to this functional representation, agents can efficiently evaluate many actions in parallel via matrix multiplication and select the one with the highest Q-value. During training, agents explore using either *Gaussian* noise added to the chosen action or ϵ -greedy sampling, where a random action is taken with probability ϵ . These strategies are disabled during evaluation.

After taking actions, agents receive individual rewards and next states from the environment. Each agent stores its transition tuple $\langle s^i, a^i, r^i, s^{i'} \rangle$ in its own replay buffer B^i .

(ii) Training

Training begins by sampling mini-batches of transitions from each agent’s replay buffer, B^i . Each agent computes its current Q-values using its Q-functional, and a central mixer network aggregates these into a global value, Q_f^{total} while target version of the mixer, operating on next-state Q-values, produces the target value, $Q_f^{\text{total}-}$. These are used to compute the temporal difference (TD) loss:

$$e_{\text{TD}} = \sum_{s, a, r, s'}^{b \sim B} [(\hat{Q}_f^{\text{total}}(s, a; w) - y(r, s'))^2], \quad (13)$$

where the bold symbols are vectors denoting the set of corresponding values for all agents, and the TD target, $y(r, s')$, is defined as:

$$y(r, s') = \sum_{i=1}^N r^i + \gamma \max_{a'} (\hat{Q}_f^{\text{total}}(s', a'; w^-) |_{a' \sim \mathcal{U}}), \quad (14)$$

with the next joint actions a' sampled from a uniform distribution, \mathcal{U} . The mixing function, \hat{Q}_f^{total} , can take the form of VDN, QMIX, or other value factorization structures. Training minimizes the TD error and encourages agents to coordinate actions that maximize total reward, while preserving decentralized execution using their own Q-functionals.

(iii) Target Updates

To improve stability in continuous domains, CoQF uses *soft target updates* method rather than hard copies, as it

has been shown to be more effective in continuous action settings [5]. After each step, the target network parameters w^- are updated incrementally:

$$w^- = \tau w + (1 - \tau)w^- \quad (15)$$

where $\tau \ll 1$ (e.g., 0.005). This gradual update reduces training variance and leads to smoother convergence compared to periodically copying the prediction network, as done in DQN-style methods.

IV. EXPERIMENTS

A. Environments

We evaluate the performance of CoQF across a diverse set of multi-agent continuous tasks in two well-established environments: the Multi-Agent Particle Environment (MPE) [23] and the Multi-Walker Environment (MWE) [24].

B. Multi-Agent Particle Environment (MPE)

We build on OpenAI’s MPE framework [3] and design two cooperative scenarios—Landmark Capturing and Predator-Prey—each instantiated in two configurations. Agents operate in a bounded 2D plane and select actions corresponding to horizontal and vertical movement within $[-1, 1]$.

Landmark Capturing: In this task, N agents must coordinate to cover N landmarks, where the goal is to maximize overall landmark coverage through cooperation. The reward for an agent is determined by its proximity to the nearest landmark, defined as $e^{-d^2/c}$ if within a threshold d_{th} , and by the total distance to all landmarks otherwise. Agents learn to divide coverage responsibilities, effectively deciding which agent should capture which landmark. Each agent observes its own velocity and the relative positions of all landmarks. We evaluate two settings: 2 agents and 2 landmarks (2A2L, Fig 2(a)), and 5 agents with 5 landmarks (5A5L, Fig 2(b)).

Predator-Prey: We adapt the classic *simple tag* scenario into a predator-prey setup where three slower agents (predators)

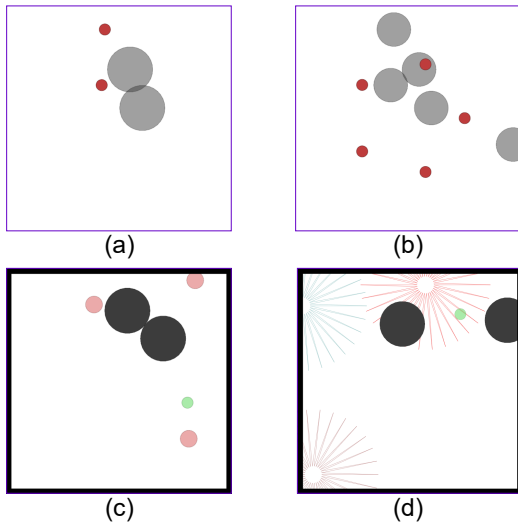


Fig. 2: MPE scenarios used for evaluation: (a) 2A2L, (b) 5A5L, (c) S-PP and (d) IC-PP.

attempt to capture a faster, heuristic-driven prey. Two large landmarks serve as obstacles. Agents receive observations that include their own position and velocity, the relative position of landmarks and other agents, as well as the prey’s position and velocity. We implement two versions to study varying levels of cooperative pressure: (i) *Standard Predator-Prey (S-PP)* rewards predators with 10 points if at least two of them make contact with the prey. Otherwise, a penalty based on the Euclidean distance to the prey is applied. While encouraging proximity, this version can lead to *inadvertent* cooperation (see Fig 2(c)). (ii) *Increased Cooperation Predator-Prey (IC-PP)* adds a field-of-view constraint, requiring predators to coordinate their approach. A successful capture requires one agent to touch the prey, but only those with the prey in view at that moment receive the reward, encouraging more *intentional* cooperation (Fig 2(d)).

C. Multi-Walker Environment (MWE)

Originally introduced in [25], the Multi-Walker Environment is a continuous control benchmark where agents jointly transport an object by maintaining balance and locomotion. In the default version, each walker is an agent controlling both legs. We extend this environment by splitting each walker into two agents, each controlling a single leg, thereby amplifying the coordination challenge. This modification transforms the task into a hybrid of locomotion and high-level collaboration: agents must synchronize not only within

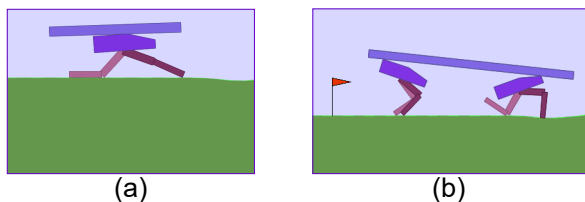


Fig. 3: MWE tasks used for evaluation: (a) 2A1W, and (b) 4A2W.

a walker but also between multiple walkers carrying the same object. While structurally similar to MaMuJoCo’s Walker2D setup [26], our version demands inter-agent cooperation to keep the package stable and progressing forward.

Each leg-agent controls two torque values in $[-1, 1]$. Observations are composed of both local features (ground contact, hip/knee angles, joint velocities) and global features (walker hull angle, angular velocity, package orientation, and velocity in x and y). Agents receive +5 for forward movement and incur a -100 penalty upon falling. Episodes terminate if any walker or the package collapses, if the package exits the terrain, or if a step limit is reached. We evaluate two settings: a single walker split into two agents (2A1W) and two walkers split into four agents (4A2W).

D. Models and Hyperparameters

Each agent in CoQF learns a Q-functional using a Multi-Layer Perceptron (MLP), which maps state inputs to a set of coefficients. The number of coefficients is determined by the combinatorial expression $\binom{O+D}{D}$, where D is the action space dimensionality and O is the order of the selected basis function. After conducting a systematic hyperparameter search—including various basis types (e.g., Polynomial, Fourier, Legendre) and functional orders (1 through 4)—we identified a second-order polynomial basis as the most effective and used it in all experiments.

For baselines, we used the official implementations of MADDPG¹ and FACMAC², selecting optimal hyperparameters based on performance across two random seeds, following the same tuning protocol as CoQF.

Our results are visualized in Figures 4, 5, and 6, where solid lines denote the mean test rewards of agents, and shaded regions indicate the 95% confidence intervals computed over multiple runs. Agents are evaluated using greedy action selection, with assessments conducted at task-specific intervals over a fixed number of test episodes. Evaluation schedules are as follows: (i) 2A2L and 5A5L, every 0.2k training episodes over 10 test episodes; (ii) S-PP, every 2.4k training episodes over 10 test episodes; (iii) IC-PP, every 0.8k training episodes over 10 test episodes; (iv) 2A1W, every 40k training steps over 100 test episodes; (v) 4A2W, every 60k training steps over 100 test episodes. In addition to these learning curves, comprehensive numerical comparisons are presented in Tables I, II, and III, which report average performance metrics over 10000 post-training test episodes.

E. Experimental Results

We evaluate CoQF on six cooperative continuous control tasks, comparing its performance to two of its ablations, Ind-QF and Ctr-QF, as well as four baselines: Ind-DDPG, Ctr-DDPG, MADDPG, and FACMAC, a state-of-the-art policy-based method designed for improved sample efficiency. To account for reward structure variations, we evaluate both a team-reward (MADDPG-Team) and an individual-reward (MADDPG-Ind) version of MADDPG.

¹<https://github.com/openai/maddpg>

²<https://github.com/oxwhirl1/facmac>

TABLE I: Average performance metrics with 95% confidence intervals on MPE scenarios after training is completed.

Methods	2-Agent 2-Landmarks		5-Agents 5-Landmarks		Increased Cooperation Predator-Prey		
	Team Reward	Success Rate	Team Reward	Success Rate	Team Reward	Total #captures	3-Agent #captures
Ctr-QF	32.64 ± 4.26	0.09 ± 0.01	6.22 ± 0.54	0.00 ± 0.00	65.27 ± 1.60	4.80 ± 0.11	0.37 ± 0.03
Ctr-DDPG	25.86 ± 4.98	0.00 ± 0.00	5.89 ± 0.32	0.00 ± 0.00	72.77 ± 5.35	4.30 ± 0.11	1.04 ± 0.21
Ind-QF	74.98 ± 2.13	0.69 ± 0.07	207.67 ± 4.09	0.94 ± 0.02	97.61 ± 1.13	4.91 ± 0.06	2.03 ± 0.12
Ind-DDPG	77.38 ± 0.29	0.83 ± 0.01	169.55 ± 5.93	0.13 ± 0.11	86.43 ± 0.30	4.19 ± 0.02	1.91 ± 0.04
CoQF	80.46 ± 0.57	0.88 ± 0.03	210.69 ± 3.02	0.98 ± 0.01	104.02 ± 0.96	4.65 ± 0.05	2.71 ± 0.04
MADDPG-Ind	75.05 ± 3.08	0.82 ± 0.08	176.28 ± 5.09	0.16 ± 0.09	81.92 ± 0.86	3.78 ± 0.05	2.11 ± 0.03
MADDPG-Team	73.90 ± 3.17	0.82 ± 0.08	166.25 ± 6.64	0.02 ± 0.05	78.87 ± 0.79	3.75 ± 0.70	1.93 ± 0.01
FACMAC	71.30 ± 1.45	0.70 ± 0.07	191.26 ± 2.14	0.32 ± 0.09	106.85 ± 0.29	4.94 ± 0.02	2.62 ± 0.05

1) *Landmark Capturing (MPE)*: We begin with the Landmark Capturing task in the MPE, as this environment was originally used to evaluate MADDPG, offering a natural point of comparison. Results for the 2A2L and 5A5L scenarios are shown in Fig. 4(a) and (b). In both settings, centralized models (Ctr-QF and Ctr-DDPG) struggle to learn effective policies. This performance drop can be attributed to (i) the high-dimensional joint observation space and (ii) the limited sensory input provided to each agent—agents observe their velocity and relative landmark positions but lack global context, making coordination difficult.

Although Ctr-QF and Ctr-DDPG perform slightly better in the simpler 2A2L setup, they are significantly outperformed by other methods, particularly in the more complex 5A5L scenario. Meanwhile, FACMAC, both MADDPG variants, and independent learners (Ind-DDPG and Ind-QF) show slower learning and lower final performance than CoQF across both tasks. While most methods reach optimal performance in 2A2L (where each landmark is captured), their performance diverges in 5A5L: policy-based methods converge to suboptimal behaviors due to sample inefficiency, whereas value-based methods, especially CoQF, continue improving.

While Ind-QF outperforms policy-based methods (in 5A5L) it suffers from instability due to non-stationarity, as seen in the performance dip around episode 2000. In contrast, CoQF leverages centralized training and mixing to achieve faster and more stable convergence, yielding the highest average return and success rate across both tasks (Table I).

2) *Predator-Prey (MPE)*: Next, we evaluate performance in the IC-PP task, where agents must not only capture a prey but also observe it during capture, promoting intentional cooperation. Table I shows that although Ind-QF records more successful captures, CoQF achieves higher average

team rewards, suggesting it better aligns individual actions to maximize joint returns. As seen in Fig. 5(a), both MADDPG variants converge faster but settle into local optima that fail to encourage cooperative trapping. In contrast, value-based methods, especially CoQF, are better able to coordinate multi-agent behavior for higher team utility. FACMAC and CoQF perform similarly in this task, as confirmed by behavior trajectories showing nearly identical group strategies with minor execution differences.

3) *Multi-Walker (MWE)*: We also test the models with locomotion-based scenarios: 2A1W and 4A2W in the Multi-Walker Environment. Agents here must learn to stabilize and transport a package across terrain—requiring complex coordination and continuous control. Animations of trained agent reveals a range of emergent strategies, including: (i) flinging the package forward without regard for balance, (ii) standing in place while holding the package, (iii) learning to take small, balanced steps, (iv) and, successfully transporting the package to the end of the terrain.

Only behavior (iv) represents a fully optimal solution. Although methods like FACMAC and Ind-QF initially demonstrate performance comparable to CoQF, as shown in Fig. 6, deeper analysis reveals their limitations in maintaining stable behavior over time. As shown in Table II, both algorithms exhibit significant drops in average reward and consistency during extended evaluation, indicating challenges in generalizing beyond their training trajectories. Ctr-DDPG, interestingly, achieves high balance rates in 4A2W but rarely completes the task within the episode limit, converging to behavior (ii) or (iii). By contrast, CoQF consistently produces stable, forward-moving policies, leading to better outcomes in both average reward and success rate. These results further highlight the benefit of efficient action evaluation and coordination in continuous domains. While extended training or increased episode lengths might allow other algorithms to catch up, our results confirm CoQF’s superior sample efficiency and robustness across tasks.

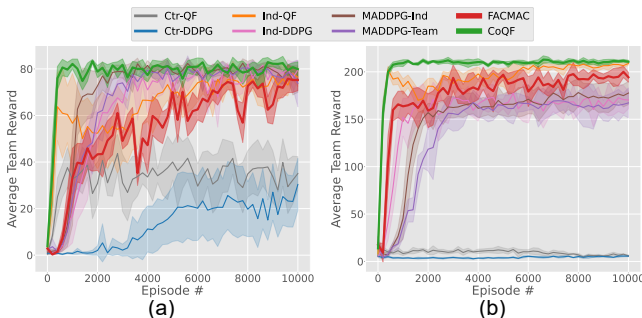


Fig. 4: Results of the landmark capturing scenarios: (a) 2A2L, and (b) 5A5L (details in Section IV-E.1).

TABLE II: Average performance metrics with 95% confidence intervals on MWE tasks after training is completed.

Methods	2-Agents (1-Walker)		4-Agents (2-Walkers)	
	Team Reward	Balance Rate	Team Reward	Balance Rate
Ctr-QF	50.76 ± 13.09	0.72 ± 0.07	-59.51 ± 8.15	0.46 ± 0.03
Ctr-DDPG	-0.26 ± 20.10	0.60 ± 0.16	43.23 ± 14.22	1.00 ± 0.00
Ind-QF	78.18 ± 2.43	0.85 ± 0.02	30.66 ± 22.62	0.73 ± 0.09
Ind-DDPG	-7.27 ± 12.41	0.39 ± 0.08	30.26 ± 6.83	0.79 ± 0.03
CoQF	94.61 ± 1.89	0.95 ± 0.02	91.31 ± 5.80	0.96 ± 0.02
MADDPG-Ind	-54.57 ± 18.05	0.20 ± 0.09	-69.84 ± 13.37	0.13 ± 0.09
MADDPG-Team	-20.78 ± 18.71	0.34 ± 0.12	10.03 ± 13.35	0.80 ± 0.08
FACMAC	67.10 ± 17.53	0.90 ± 0.10	62.70 ± 33.99	0.87 ± 0.11

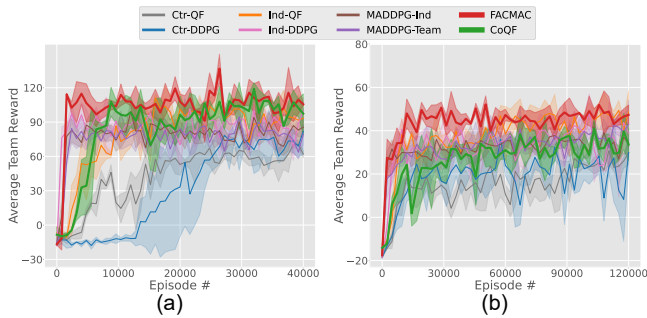


Fig. 5: Results of the predator-prey scenarios: (a) IC-PP and (b) S-PP. In IC-PP, CoQF and FACMAC show comparable performance, as also reflected in Table I and discussed in Section IV-E.2. In S-PP, where cooperation is largely inadvertent, CoQF is outperformed by Ind-QF and FACMAC, as explained in the limitation Section IV-F.

F. Limitation

To assess CoQF’s performance in settings where coordination may arise incidentally rather than by design, we evaluate it in the S-PP scenario. In such *inadvertent cooperation* settings, agents pursue independent objectives but their behaviors can appear coordinated—often due to shared environment constraints or overlapping goals. This effect is observed when the same policy, trained independently for a single predator, is deployed across multiple agents. Despite no explicit coordination, these agents may still converge on strategies resulting in prey capture. As shown in Fig. 5(b), Ind-QF performs strongly in S-PP, outperforming most baselines except FACMAC. However, the observed success stems from incentive alignment rather than genuine coordination.

This is further highlighted in Table III, which reports the number of three-agent captures—instances where all predators collaborate to surround the prey. Both Ind-QF and FACMAC exhibit low rates of such events, indicating that their strategies do not reflect intentional planning or joint action. To better evaluate models in scenarios that demand explicit cooperation, we turn to the IC-PP variant, where rewards are structured to require agents to synchronize their behaviors for team-level success.

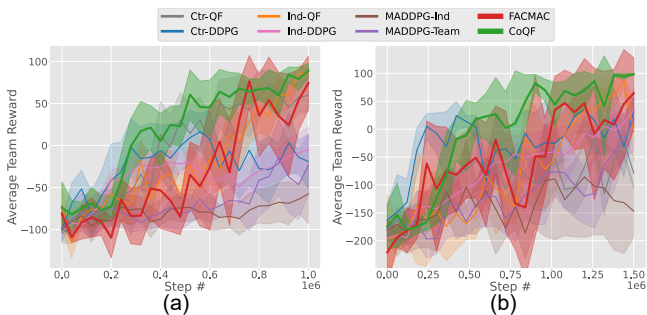


Fig. 6: Learning curves of the MWE scenarios: (a) 2A1W, and (b) 4A2W (see Section IV-E.3 for details), showing how each algorithm progresses over time. The overlapping trends, caused by the small number of test episodes (due to long episode durations), make early differences difficult to observe. However, as illustrated in Table II—based on extensive evaluation after training—CoQF demonstrates clear performance advantages.

V. DISCUSSION

This work introduces a value-based framework for multi-agent reinforcement learning in continuous action spaces. While existing methods like MADDPG and MAPPO extend single-agent policy gradient techniques to the multi-agent setting, our approach—built on Q-functionals—demonstrates the viability and advantages of value-based learning in these more complex environments. Across benchmark tasks, including landmark capture (2A2L, 5A5L), cooperative predator-prey (IC-PP), and multi-agent locomotion (2A1W, 4A2W), CoQF consistently outperforms both value-based baselines and policy-based methods, including FACMAC.

In the landmark capturing tasks, optimal behavior requires agents to distribute themselves across distinct targets. While CoQF and Ind-QF converge to such optimal strategies, particularly in the more complex 5A5L setting, other methods (e.g., Ind-DDPG, MADDPG, FACMAC) tend to collapse into suboptimal behaviors where multiple agents pursue the same landmark, reducing success rates (Table I). These patterns were confirmed via success rate metrics and visualizations of trained agents. In the IC-PP scenario, CoQF and FACMAC were the only methods to learn intentional coordination strategies, delaying prey capture until teammates could observe the event and share the reward. In contrast, Ind-QF maximized capture count without regard for cooperation, and MADDPG exhibited delayed responses that reduced overall efficiency.

In the multi-walker environment, CoQF outperformed all baselines in both task success and stability. Agents learned to maintain balance and coordinate locomotion over long horizons, reliably delivering the package to the endpoint (Table II). Animated evaluations revealed that CoQF agents executed smoother and more successful transport strategies compared to others, which often dropped the package or failed to make meaningful progress. Although learning curves as in Fig. 6 may suggest similar trends across methods, deeper evaluation reveals clear differences. As shown in Table II, performance gaps become apparent under long test horizons, where CoQF consistently achieves higher rewards and balance rates. The limited number of test episodes shown in figures (due to long episode lengths and environment stochasticity) emphasizes the importance of post-training summary statistics. In this light, the 10000-episode averages reported in the table offer a more reliable basis for comparison.

A natural question arises about the role of explicit explo-

TABLE III: Average performance metrics with 95% confidence intervals on the S-PP task after training completion.

Methods	Standard Predator-Prey		
	Team Reward	Total #captures	3-Agent #captures
Ctr-QF	27.44 ± 0.66	1.82 ± 0.03	0.00 ± 0.00
Ctr-DDPG	20.13 ± 1.06	1.51 ± 0.05	0.00 ± 0.00
Ind-QF	47.85 ± 0.47	2.73 ± 0.02	0.10 ± 0.00
Ind-DDPG	36.69 ± 0.48	2.22 ± 0.03	0.07 ± 0.00
CoQF	33.97 ± 0.58	2.02 ± 0.03	0.07 ± 0.00
MADDPG-Ind	34.11 ± 0.11	2.05 ± 0.00	0.06 ± 0.00
MADDPG-Team	38.13 ± 0.28	2.25 ± 0.01	0.07 ± 0.00
FACMAC	46.72 ± 1.62	2.67 ± 0.07	0.15 ± 0.02

ration in CoQF. Since the method already samples multiple actions per state and supports decentralized execution, one might expect sufficient exploration from structure alone. However, our experiments show that additional exploration strategies, such as Gaussian or ϵ -greedy noise, can accelerate learning, particularly in more complex settings. We evaluated several exploration methods and selected the one yielding the fastest and most stable convergence for each task.

Finally, we emphasize that the mixing function in CoQF can take various forms. While we adopt a simple additive structure in this paper, the framework supports more expressive options, including monotonic or constrained mixers such as those in QMIX, QTRAN, or weighted QMIX. Although more complex mixing may be necessary for high-dimensional or competitive domains, we found additive mixing was sufficient for the cooperative tasks explored here.

VI. CONCLUSION AND FUTURE WORK

We introduce CoQF, a value-based algorithm developed to address the limitations of policy-gradient methods in cooperative multi-agent reinforcement learning with continuous actions. By representing agent states through basis functions and combining their action-values via a mixing architecture, CoQF enables efficient and coordinated policy learning across diverse multi-agent tasks. Empirical results highlight two key advantages: (i) CoQF consistently achieves optimal performance where policy-based methods—including FAC-MAC, a state-of-the-art variant—often converge to suboptimal behaviors, and (ii) in scenarios where multiple methods reach optimality, CoQF converges significantly faster, demonstrating superior sample efficiency and scalability.

While CoQF already demonstrates more stable convergence than existing methods, occasional fluctuations suggest room for improvement. Incorporating techniques such as target policy smoothing [27] and exploring more expressive mixing functions may enhance performance, particularly in high-dimensional or non-stationary environments. Additionally, although CoQF is designed for cooperative multi-agent learning, it may be extended to partially competitive settings. For instance, in scenarios like predator-prey, separate CoQF policies could be trained for opposing teams to enable adversarial interactions. While not explored in this study, it is hypothesized that CoQF can support such dynamics when competition is framed within a broader cooperative context, offering a compelling direction for future research.

VII. ACKNOWLEDGMENTS

This work is supported in part by NSF (IIS-2112633) and the Army Research Lab (W911NF20-2-0089).

REFERENCES

- [1] C. J. Watkins and P. Dayan, “Q-learning,” *Machine learning*, vol. 8, pp. 279–292, 1992.
- [2] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” *Advances in neural information processing systems*, vol. 12, 1999.
- [3] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, “Multi-agent actor-critic for mixed cooperative-competitive environments,” *Advances in neural information processing systems*, vol. 30, 2017.

- [4] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, “The surprising effectiveness of ppo in cooperative, multi-agent games,” *arXiv preprint arXiv:2103.01955*, 2021.
- [5] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.
- [6] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [7] G. Papoudakis, F. Christianos, L. Schäfer, and S. V. Albrecht, “Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks,” *arXiv preprint arXiv:2006.07869*, 2020.
- [8] S. Lobel, S. Rammohan, B. He, S. Yu, and G. Konidaris, “Q-functionals for value-based continuous control,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 7, 2023, pp. 8932–8939.
- [9] F. A. Oliehoek, C. Amato, et al., *A concise introduction to decentralized POMDPs*. Springer, 2016, vol. 1.
- [10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., “Human-level control through deep reinforcement learning,” *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [11] M. Tan, “Multi-agent reinforcement learning: Independent vs. cooperative agents,” in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.
- [12] C. Claus and C. Boutilier, “The dynamics of reinforcement learning in cooperative multiagent systems,” *AAAI/IAAI*, vol. 1998, no. 746–752, p. 2, 1998.
- [13] F. A. Oliehoek, M. T. Spaan, and N. Vlassis, “Optimal and approximate q-value functions for decentralized pomdps,” *Journal of Artificial Intelligence Research*, vol. 32, pp. 289–353, 2008.
- [14] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, W. M. Czarnecki, W. M. Czarnecki, et al., “Value-decomposition networks for cooperative multi-agent learning,” *arXiv preprint arXiv:1706.05296*, 2017.
- [15] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, “Monotonic value function factorisation for deep multi-agent reinforcement learning,” *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.
- [16] R. J. Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning,” *Machine learning*, vol. 8, pp. 229–256, 1992.
- [17] V. Konda and J. Tsitsiklis, “Actor-critic algorithms,” *Advances in neural information processing systems*, vol. 12, 1999.
- [18] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, “Deterministic policy gradient algorithms,” in *International conference on machine learning*. Pmlr, 2014, pp. 387–395.
- [19] B. Peng, T. Rashid, C. Schroeder de Witt, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, “Facmac: Factored multi-agent centralised policy gradients,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 208–12 221, 2021.
- [20] E. Marchesini and C. Amato, “Improving deep policy gradients with value function search,” *arXiv preprint arXiv:2302.10145*, 2023.
- [21] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, “Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning,” in *International conference on machine learning*. PMLR, 2019, pp. 5887–5896.
- [22] T. Rashid, G. Farquhar, B. Peng, and S. Whiteson, “Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning,” *Advances in neural information processing systems*, vol. 33, pp. 10 199–10 210, 2020.
- [23] I. Mordatch and P. Abbeel, “Emergence of grounded compositional language in multi-agent populations,” *arXiv preprint arXiv:1703.04908*, 2017.
- [24] J. Terry, B. Black, N. Grammel, et al., “Pettingzoo: Gym for multi-agent reinforcement learning,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 032–15 043, 2021.
- [25] J. K. Gupta, M. Egorov, and M. Kochenderfer, “Cooperative multi-agent control using deep reinforcement learning,” in *Autonomous Agents and Multiagent Systems: AAMAS Workshops*, 2017, pp. 66–83.
- [26] R. de Lazzano, K. Andreas, J. J. Tai, S. R. Lee, and J. Terry, “Gymnasium robotics,” 2023. [Online]. Available: <http://github.com/Farama-Foundation/Gymnasium-Robotics>
- [27] S. Fujimoto, H. Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods,” in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.