Ingénierie des Systèmes d'Information
Vol. 30, No. 12, December, 2025, pp. 3117-3130
Journal homepage: http://iieta.org/journals/isi

Multi-Head DDPG for Pursuit-Evasion with Interpretable Behavioral Decomposition

Saida Lehis[1,2*], Abderrahim Siam[2], Hamouma Moumen[1], Wahid Chergui[2], Mohammed El Habib Souidi[2], Abdelaali Bekhouche[2]

[1] Department of Computer Science, Mostefa Ben Boulaid University, Batna 2, Batna 05001, Algeria
[2] Department of Computer Science, ICOSI Laboratory, Abbes Laghrour University, Khenchela 40004, Algeria

Corresponding Author Email: lehis.saida@univ-khenchela.dz

https://doi.org/10.18280/isi.301204

Received: 21 September 2025
Revised: 30 November 2025
Accepted: 14 December 2025
Available online: 31 December 2025

Keywords:
Deep Reinforcement Learning, multi-agent systems, decentralized coordination, self-organization, behavioral decomposition, pursuit-evasion, Multi-Head architecture, force-based control

ABSTRACT

Designing scalable and interpretable control strategies for decentralized multi-agent systems remains a challenge in reinforcement learning (RL). This challenge is particularly evident in pursuit–evasion tasks, which require coordination under partial observability, without explicit communication or centralized guidance. Although deep RL methods achieve strong performance, they typically operate as black boxes, limiting trust and deployment in safety-critical domains. We propose a Multi-Head DDPG architecture that decomposes control into three interpretable force components - pursuit, cohesion, and separation - weighted adaptively to generate context-aware actions. This design enables emergent role differentiation and interpretable self-organization in the model. In grid-based pursuit–evasion benchmarks, our method outperforms DQN, PPO, and standard DDPG in terms of success rate, convergence speed, and generalization, while also yielding transparent collective behaviors. Overall, the results show that weighted force-based behavioral decomposition provides a principled pathway toward achieving both high-performance and explainable multi-agent control.

1. INTRODUCTION

Reinforcement Learning (RL) is a computational framework in which agents learn sequential decision-making policies by interacting with their environment and optimizing long-term cumulative rewards through trial and error [1]. Its extension to multi-agent systems, known as Multi-Agent Reinforcement Learning (MARL), addresses scenarios where multiple agents must adapt their policies concurrently under partial observability, non-stationarity, and limited communication [2]. In recent years, the integration of Deep Reinforcement Learning (DRL) into MARL has significantly advanced the field, allowing agents to approximate policies and value functions directly from high-dimensional inputs [3-7]. This synergy has positioned DRL-based MARL as a cornerstone paradigm for distributed coordination in domains such as swarm robotics, autonomous driving, and collaborative multi-robot systems.

In this context, the pursuit-evasion problem has emerged as a canonical benchmark for evaluating coordination under decentralization. In this task, a team of pursuers must cooperate to capture one or more evaders in discrete or continuous environments, typically without centralized supervision [8-24]. The challenge lies in the simultaneous requirements for adaptation under uncertainty, cooperative strategy formation and emergent role differentiation. Although traditional solutions rooted in control theory, game-theoretic models, or heuristic rules [13], Shoham and Leyton-Brown [5] offered interpretable baselines, they lack the flexibility and scalability required in dynamic and high-dimensional settings.

Despite the successes of DRL-based MARL and the development of strategies such as Centralized Training with Decentralized Execution (CTDE) [8], selective parameter sharing [9], and emergent communication [10], a critical limitation persists: most DRL policies behave as opaque black boxes. They often achieve state-of-the-art performance in UAV swarms or multi-robot systems but provide little insight into why specific coordination patterns or roles emerge, thereby hindering trust, debugging, and safe deployment [14]. Among existing DRL algorithms, the Deep Deterministic Policy Gradient (DDPG) [12] has been widely applied to continuous control owing to its sample efficiency and actor-critic structure. However, its monolithic design maps observations directly to single actions, resulting in slow convergence, high sensitivity to hyperparameters, and a lack of mechanisms for role differentiation and self-organization [8, 21].

To overcome these limitations, we introduce a Multi-Head DDPG architecture that explicitly decomposes the actor's policy into three interpretable behavioral primitives—pursuit, cohesion, and separation—each represented as a force-based component. Unlike the monolithic DDPG, the final action is computed as an adaptive weighted combination of these primitives, thereby producing a modular and transparent control policy. This formulation not only enhances performance but also enables emergent self-organization, as agents autonomously specialize in complementary roles without central supervision. This work makes three main

contributions. First, we propose a structural innovation in the form of a Multi-Head Actor architecture that decomposes continuous control into interpretable force-based behavioral primitives, improving policy transparency and modularity in decentralized MARL. Second, we introduce a mechanistic innovation based on adaptive and context-aware weighting of these primitives, enabling dynamic role specialization and self-organized coordination during learning. Third, we provide a comprehensive empirical validation of decentralized pursuit–evasion benchmarks. The results show that the proposed structural and mechanistic designs jointly yield superior performance, faster convergence, and improved generalization compared to DQN, PPO, and standard DDPG.

An interpretability analysis of the learned behavioral weights further illustrates how coordinated strategies and role differentiation emerge from this framework.

The remainder of this paper is organized as follows: Section 2 reviews the related work on MARL, pursuit-evasion, and interpretable control. Section 3 formalizes the problem. Section 4 introduces the proposed Multi-Head DDPG framework. Section 5 presents the experimental evaluations, including comparative benchmarks and interpretability analysis. Finally, Section 6 concludes with the key findings and future research directions.

2. RELATED WORK

The pursuit-evasion problem has long served as a benchmark for studying coordination in MARL because it naturally requires decentralized control, adaptability, and emergent teamwork [3]. Existing approaches can be broadly categorized along a spectrum between interpretable but limited methods and scalable but opaque methods.

Early approaches. Heuristics, fuzzy control, and game-theoretic models, such as Nash equilibria or rule-based behavioral controllers inspired by collective motion models [25-36], provide transparent and explainable policies. Their interpretability makes them useful for understanding coordination mechanisms; however, their handcrafted nature limits their scalability to high-dimensional or stochastic environments. These methods highlight the trade-off between interpretability and adaptability, which is a limitation that motivated the adoption of learning-based approaches.

DRL-based approaches. With the advent of DRL, agents can learn policies directly from raw or high-dimensional inputs, enabling scalable coordination strategies [6, 12]. State-of-the-art performance has been demonstrated in complex multi-agent tasks [1, 2]. However, the resulting policies typically behave as black-box models, offering limited interpretability. Thus, while DRL methods provide scalability and robustness, they exacerbate the lack of transparency regarding internal decision-making processes.

Communication and CTDE. Another research line has focused on improving coordination through communication learning and centralized training methods. Foerster et al. [10] introduced Reinforced Inter-Agent Learning (RIAL) and Differentiable Inter-Agent Learning (DIAL), showing that agents can autonomously develop communication protocols. Building on this, the now-dominant CTDE paradigm allows leveraging global information during training while retaining distributed autonomy at execution [20, 21]. These approaches achieve strong performance, particularly in UAV coordination, but the internal reasoning processes remain opaque. Thus, they maximize scalability but further reduce the interpretability.

Interpretable MARL and behavior-based decomposition. Recent work in MARL has increasingly emphasized interpretability, motivated by the need for transparency, debugging, and trustworthy deployment in safety-critical multi-robot systems [11, 14]. Explainable AI studies have highlighted that high-performing deep policies often lack explicit causal structure, hindering decision justification and failure diagnosis in multi-agent settings [1, 2]. To address this, behavioral decomposition and hierarchical reinforcement learning represent policies as compositions of reusable primitives or options rather than monolithic mappings, improving both learning efficiency and interpretability [22, 29]. However, such abstractions are often detached from physically meaningful interaction mechanisms. In parallel, swarm robotics and multi-agent navigation have long relied on attractive–repulsive interaction rules, including artificial potential fields and social-force models, to encode goal attraction, collision avoidance, and group dynamics [4, 26]. Boids models and flocking theory further formalize cohesion, separation, and alignment as decentralized coordination primitives [27, 36]. Several DRL approaches implicitly incorporate these interaction priors, for example through socially aware reward shaping or interaction features, demonstrating improved coordination and navigation performance [17, 18, 32]. Nevertheless, in most existing works, force-based components remain handcrafted or auxiliary, and learned policies do not provide an explicit, interpretable decomposition of behavioral roles. Importantly, these interaction models are rarely integrated as first-class components of the policy representation itself.

Emergent coordination without communication. A complementary direction emphasizes self-organization without explicit communication between the agents. Sun et al. [16] proposed fuzzy self-organization for subgroup formation, Christianos et al. [9] leveraged selective parameter sharing to enhance scalability, and Hüttenrauch et al. [18] introduced mean feature embeddings to enhance local spatial awareness. In pursuit-evasion tasks, De Souza et al. [17] combined DRL with curriculum learning to balance group and individual incentives, whereas Yu et al. [19] extended Double-DQN and Duelling-DQN to stabilize cooperative pursuits under partial observability. These methods demonstrate that scalable emergent teamwork is possible; however, because policies remain opaque, role differentiation and self-organization emerge implicitly rather than being explicitly modelled.

Our contribution. The proposed Multi-Head DDPG addresses this gap by explicitly decomposing agent control into three interpretable behavioral primitives: pursuit, cohesion, and separation. Unlike existing DRL methods, our approach maintains scalability while directly exposing the modulation of behavior using adaptive weights. This not only improves quantitative performance but also enables emergent self-organization and transparent role differentiation, reconciling the tension between performance and interpretability of the model.

3. FORMULATION OF THE PURSUIT-EVASION DILEMMA

The multi-agent pursuit-evasion (PE) problem serves as a long-standing benchmark for cooperative and adversarial

decision-making, particularly in the field of MARL. It involves two classes of agents, pursuers and evaders, interacting in a shared environment characterized by decentralization and partial observability. Pursuers aim to coordinate their efforts to capture evaders, whereas evaders seek to maximize their survival by avoiding detection and confinement. This dilemma encapsulates fundamental challenges of distributed perception, decentralized control, and emergent self-organization [1, 2].

In this study, we adopt a grid-based discrete environment of size M × N, which is a widely used abstraction in pursuit-evasion research that balances tractability with the ability to model realistic multi-agent dynamics [19, 22, 23]. Each agent occupies a single grid cell and evolves in discrete time steps t = 0, 1, 2, … At each step, an agent may move to one of the four cardinal neighboring cells (up, down, left, right) or remain stationary, subject to the grid boundaries and occupancy constraints. Formally, let P = {p_1, p_2, …, p_{np}} denote the set of pursuers and E = {e_1, e_2, …, e_{ne}} the set of evaders, with $A = P \cup E$ representing the complete population of agents. The position of agent $a \in A$ at time t is denoted $pos_a^t \in \mathbb{Z}^2$, and its discrete action space is given by: $\mathcal{A}_a$ = {up, down, left, right, stay} with valid transitions restricted to:

$$\begin{aligned} pos_a^{t+1} \in \mathcal{N}(pos_a^t) \\ = \{pos_a^t, pos_a^t + (0,1), pos_a^t \\ - (0,1), pos_a^t + (1,0),\ pos_a^t \\ - (1,0)\} \end{aligned} \tag{1}$$

where, $\mathcal{N}(pos_a^t)$ denotes the Von Neumann neighborhood of the agent's position. This neighborhood includes the four orthogonally adjacent cells along the cardinal axes, while excluding diagonal cells. The Von Neumann neighborhood is extensively employed in grid-based multi-agent environments because it provides a simple yet realistic model of local connectivity, facilitating efficient interaction dynamics and collision handling [28].

The pursuit-evasion task is formally modeled as a Partially Observable Markov Game (POMG) [24], defined by the tuple

$$G = \langle \mathcal{A}, S, \{\mathcal{O}_a\}_{a\in\mathcal{A}}, \{\mathcal{A}_a\}_{a\in\mathcal{A}}, T, \{\mathcal{R}_a\}_{a\in\mathcal{A}}, \rho \rangle \tag{2}$$

where, S is the global state space of agent configurations, $\mathcal{O}_a$ is the local observation space for agent a, T is the stochastic transition function, $\mathcal{R}_a$ is the agent-specific reward, and $\rho \in [0, 1]$ is the discount factor.

Observability is egocentric, meaning that each agent perceives the relative positions of other agents within a fixed sensing radius r_s.

The local observation set of agent a at time t is therefore defined as:

$$\begin{aligned} O_a^t = \{pos_j^t - pos_a^t \mid j \\ \in \boldsymbol{\mathcal{A}} \wedge \|pos_j^t - pos_a^t\|_2 \le r_s\} \end{aligned} \tag{3}$$

This egocentric design ensures translational invariance, scalability to varying numbers of agents, and adaptability to different grid sizes while reducing the input dimensionality by retaining only spatially local motion cues. By excluding environmental obstacles, we further isolate the intrinsic coordination phenomena that emerge from agent-agent interactions, making this formulation a robust testbed for investigating emergent self-organization and cooperative strategies in decentralized multi-agent systems [8, 21].

3.1 Definition of capture

A capture event occurs when an evader is fully surrounded by pursuers such that all adjacent cells are simultaneously occupied. Let the position of evader $e_j \in E$ at time t be $pos_{e_j}^t \in Z^2$. Its neighborhood is defined as:

$$\mathcal{N}\left(pos_{e_j}^t\right) = \left\{ \begin{matrix} (x,y) \in \mathbb{Z}^2 \,\middle|\, (x,y) \in \left\{pos_{e_j}^t + \delta\right\}, \\ \delta \in \{(0,1),(0,-1),(1,0),(-1,0)\} \end{matrix} \right\} \tag{4}$$

An evader is captured at time t if at least k of the cells in its Von Neumann neighborhood $\mathcal{N}\left(pos_{e_j}^t\right)$ are simultaneously occupied by distinct pursuer agents $p_i \in P$:

$$\left| (x,y) \in \mathcal{N}\left(pos_{e_j}^t\right) \middle| \exists p_i \in P : pos_{p_i}^t = (x,y) \right| \ge \mathrm{k} \tag{5}$$

Here, the neighborhood $\mathcal{N}\left(pos_{e_j}^t\right)$ contains at most four orthogonally adjacent cells. Therefore, parameter k ≤ 4 denotes a capture threshold, corresponding to the minimum number of neighboring cells that must be occupied for a capture event to occur. Setting k = 4 enforces strict encirclement, requiring full occupation of the neighborhood, whereas smaller values (e.g., k = 2) relax this constraint and reduce the level of coordination required among pursuers.

This parameterization allows the task difficulty to be systematically adjusted, ranging from partial containment to complete encirclement, and enables controlled evaluation of MARL algorithms under varying cooperation demands. Such capture criteria naturally promote coordinated behaviors, such as flanking, blocking, and convergence, fostering emergent role differentiation and self-organization [18].

3.2 Reward and learning objectives

The objective of pursuers is to maximize the total number of successful captures through decentralized coordination. Following standard MARL formulations [9, 17], we employ a sparse, event-driven reward structure. For each pursuer $p_i \in P$, the immediate reward at time t is defined as:

$$r_i^t = \begin{cases} +R_c, \text{if agent i contributes to a capture at time t} \\ -r_m, \text{if agent i makes an invalid or redundant move} \\ 0, \ \text{otherwise} \end{cases} \tag{6}$$

where, $R_c > 0$ is the cooperative capture reward and $r_m > 0$ is a penalty for inefficiency. Each agent aims to optimize its discounted return within a finite time horizon T, which is defined as:

$$R_i = \sum_{t=0}^{T} \rho^t r_i^t \tag{7}$$

with $\rho \in [0, 1]$ as the discount factor. This design ensures that captures can only be achieved through collective effort, while discouraging selfish or redundant actions. Meanwhile, evaders follow reactive strategies aimed at maximizing their distance

from nearby pursuers, thus exerting adversarial pressure and promoting the development of robust pursuit policies.

3.3 Significance for self-organization

The pursuit-evasion paradigm extends beyond theoretical interest: it models practical scenarios in robotic surveillance, UAV swarms, and wildlife monitoring where agents operate under strict communication and sensing constraints [25-27]. Its decentralized formulation, absence of global state information, and reliance on local interactions make it a natural benchmark for studying emergent coordination and interpretable self-organization. By analyzing how global strategies, such as encirclement or flanking, arise from local force-based rules, this framework provides valuable insights into the design of scalable, robust, and explainable MARL systems.

4. BACKGROUND AND PROPOSED MULTI-HEAD ARCHITECTURE

4.1 Deep Deterministic Policy Gradient: Limitations and Multi-Head enhancement

The DDPG algorithm, introduced by Lillicrap et al. [12], is an off-policy actor-critic method designed for continuous action spaces. It combines the Deterministic Policy Gradient (DPG) [7] with deep neural networks to approximate both the policy and value function, while employing replay buffers and target networks to improve training stability. DDPG has been successfully applied in robotics [31], navigation [32], and multi-agent learning [8].

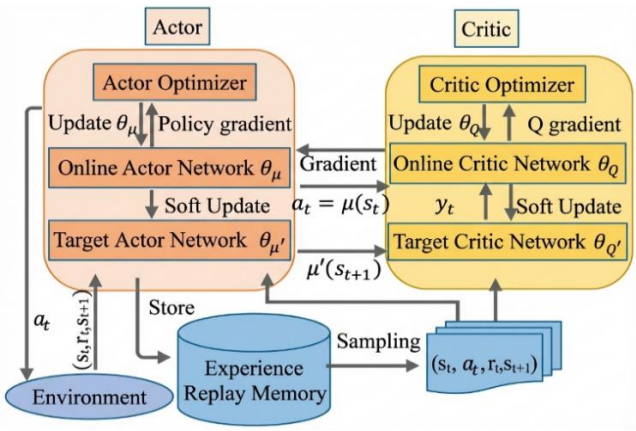

Figure 1. Structure of the Deep Deterministic Policy Gradient (DDPG) algorithm

As illustrated in Figure 1, the standard DDPG consists of four neural networks: an online actor, an online critic, and their respective target networks. The actor maps the states to continuous actions, whereas the critic evaluates these actions using the Q-function. Target networks stabilize training through soft updates, and the replay buffer enables efficient off-policy learning.

Despite these strengths, the standard DDPG exhibits key limitations in multi-agent settings. Its monolithic architecture maps observations directly to single actions, which (i) slows convergence and increases sensitivity to hyperparameters, (ii) produces opaque policies that hinder interpretability, and (iii) prevents role differentiation, thereby limiting the emergence of complex cooperative behaviors such as encirclement.

To overcome these shortcomings, we introduce a Multi-Head DDPG actor that decomposes the control signal into three interpretable primitives—pursuit, cohesion, and separation—weighted adaptively at each time step. This modular design preserves the stability and sample efficiency of the DDPG while enhancing policy transparency and enabling emergent self-organization in decentralized pursuit-evasion tasks.

4.2 Multi-Head Actor with force modulation

4.2.1 Architectural overview

The proposed Multi-Head Actor architecture constitutes a substantial extension of the conventional DDPG framework [12], introducing explicit behavioral modularity that promotes emergent coordination in decentralized pursuit-evasion tasks. Unlike the monolithic policy outputs of standard DDPG, our actor decomposes decision-making into three interpretable control primitives inspired by swarm intelligence [35], bio-inspired collective behavior [26], and modular reinforcement learning [29]: pursuit ($\overrightarrow{F_{\text{goal}}}$), cohesion ($\overrightarrow{F_{coh}}$), and separation ($\overrightarrow{F_{sep}}$). This architecture is illustrated in Figure 2.

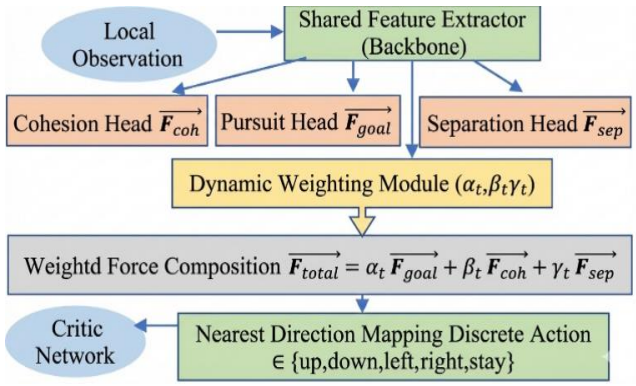

Figure 2. Multi-Head Actor architecture

The network design begins with a shared feature extractor, realized as a multilayer perceptron, which encodes the agent's local observation O_a^t into a latent representation. This common backbone ensures that all behavioral components operate with a unified semantic understanding of the environment. From this latent representation, three parallel output heads compute their respective continuous force vectors.

In parallel, a dynamic role-weighting module, implemented as a separate branch with a softmax activation, generates a triplet of normalized behavioral weights $(\alpha_t,\beta_t,\gamma_t) \in [0, 1]^3$, subject to the convexity constraint: $\alpha_t + \beta_t + \gamma_t = 1$. These weights adaptively modulate the relative influence of each control primitive based on the agent's local context, thereby supporting dynamic role allocation. The final continuous control vector is then obtained as follows:

$$\overrightarrow{F_{total}} = \alpha_t \overrightarrow{F_{goal}} + \beta_t \overrightarrow{F_{coh}} + \gamma_t \overrightarrow{F_{sep}} \tag{8}$$

Since the pursuit-evasion environment operates on a discrete grid, $\overrightarrow{F_{total}}$ is discretized into one of the available movement actions A = {up, down, left, right, stay} using a nearest-direction mapping.

This architectural choice offers three core advantages: (i)

interpretability, as the learned role weights can be directly inspected over time, revealing the agent's decision-making process; (ii) context-sensitive coordination, allowing agents to autonomously adapt their behavioral emphasis to situational demands; and (iii) robustness and generalization, since modular primitives can be dynamically recombined to handle unseen or evolving scenarios.

By embedding these behavioral priors into the policy, the Multi-Head Actor bridges the gap between high-performance DRL [12] and explainable self-organization [11]. This results in scalable, interpretable, and emergent group behaviors in fully decentralized multi-agent systems.

4.2.2 Behavioral decomposition: Force-based modulation

The proposed Multi-Head Actor architecture aims to achieve interpretable and modular control by decomposing the policy into three fundamental behavioral primitives: pursuit, cohesion, and separation. Consider an agent a ∈ P_a, which represents a pursuer located at position $\overrightarrow{pos_a}$ at time step t. Based on its local perception O_a^t, the agent calculates three normalized force vectors, each corresponding to a distinct behavioral primitive. These primitives function as modular components within the control policy, facilitating interpretable and adaptive multi-agent coordination.

Pursuit Force($\overrightarrow{F_{goal}}$): Generates a force vector oriented toward the nearest target, enabling direct interception behavior.

$$\overrightarrow{F_{\text{goal}}} = \frac{\overrightarrow{pos_e} - \overrightarrow{pos_a}}{\|\overrightarrow{pos_e} - \overrightarrow{pos_a}\|_2} \tag{9}$$

$$\overrightarrow{pos_e} = \arg\min_{\vec{p}_j \in (E \cap Q_a^t)} \left\|\overrightarrow{pos_j} - \overrightarrow{pos_a}\right\|_2 \tag{10}$$

Here, $\overrightarrow{pos_e}$ denotes the position of the closest evader within the sensing range of the agent. The resulting vector is unit-normalized to preserve directionality while removing magnitude scaling. From a behavioral perspective, a high modulation coefficient α_t indicates an aggressive pursuit strategy whereby the agent prioritizes rapid engagement with its target.

Cohesion Force ($\overrightarrow{F_{coh}}$): Produces a force vector that drives the agent toward the centroid of its teammates within a fixed radius r_s [27] and outside a repulsion zone r_{rep} (to avoid conflict with separation), thereby enhancing spatial coordination.

$$N_a = \left\{ j \in P : \left| r_{rep} < \left\|\overrightarrow{pos_j} - \overrightarrow{pos_a}\right\|_2 \leq r_s \right\} \right. \tag{11}$$

$$\overrightarrow{F_{coh}} = \frac{1}{|N_a|} \sum_{j \in N_a} \frac{\overrightarrow{pos_j} - \overrightarrow{pos_a}}{\left\|\overrightarrow{pos_j} - \overrightarrow{pos_a}\right\|_2} \tag{12}$$

The neighbour set N_a consists of all pursuers in the annular region $r_{rep} < d \leq r_s$. Each direction vector toward a teammate was normalized, and the resulting mean vector points toward the centroid of the group. A high value of β_t reflects formation-preserving behavior, promoting coordinated movement and group cohesion, which can improve encirclement strategies (Figure 3).

Separation Force ($\overrightarrow{F_{sep}}$): Outputs a force vector that repels the agent from nearby pursuers to prevent overcrowding and collisions [27].

$$\overrightarrow{F_{sep}} = -\sum_{j \in N_a'} \frac{\overrightarrow{pos_j} - \overrightarrow{pos_a}}{\left\|\overrightarrow{pos_j} - \overrightarrow{pos_a}\right\|_2^2} \tag{13}$$

$$N_a' = \left\{ j \in (P \cup E) \left| \left\|\overrightarrow{pos_j} - \overrightarrow{pos_a}\right\|_2 \leq r_{rep} \right. \right\} \tag{14}$$

The set N_a' includes all agents within a predefined repulsion radius r_{rep}. Inverse-square distance weighting ensures that the repulsion strength increases significantly in close proximity. Behaviorally, a high coefficient γ_t corresponds to collision-avoidance prioritization, which is a critical factor in dense formations or confined environments.

Each force vector is calculated locally based on the geometric relationships within the grid. All forces are normalized to ensure stability and proper orientation. These normalized forces are then combined to form the final motion vector $\overrightarrow{F_{total}}$.

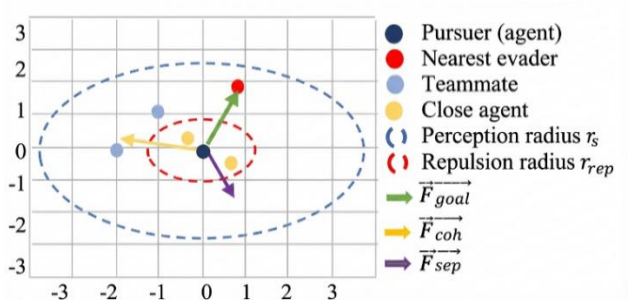

Figure 3. Illustration of force-based modulation around an agent

4.2.3 Theoretical foundation and behavioral primitives

Within the proposed Multi-Head DDPG framework, the actor's decision-making process is realized through a context-dependent weighted integration of three behavioral primitives: pursuit, cohesion, and separation. Unlike standard DDPG architectures that output a single continuous action [12], this formulation introduces behavioral modularity and implicit role differentiation using force-based heads. The design of these primitives is directly inspired by the principles of swarm intelligence [3] and bio-inspired collective behavior models [35], where complex group dynamics emerge from simple local interaction rules.

The choice of pursuit, cohesion, and separation as the core set of primitives is based on both theoretical and empirical evidence. In the classical boids model of Reynolds [36], similar rules (attraction, alignment, and separation) were shown to be sufficient to generate flocking, swarming, and encirclement canonical patterns of distributed self-organization. Pursuit dynamics extend this foundation by explicitly modeling predator-prey interactions [36], making the triplet a minimal yet expressive basis for coordinating agents in pursuit evasion. From a control-theoretic perspective, pursuit drives goal-directedness, cohesion maintains group integrity, and separation ensures collision avoidance and spatial safety of the flock. Together, these forces span the exploration-exploitation trade-off at the collective level: aggressive engagement versus cooperative stability.

4.2.4 Weighted combination and policy output

Formally, each agent receives a local observation $O_a^t \in R^d$, which is processed by a multilayer perceptron (MLP). The

MLP applies successive affine transformations interleaved with nonlinear activations, mapping the raw input into a compact latent embedding $z_t \in R^k$. This latent representation encodes the salient spatial and relational features of the environment while attenuating irrelevant noise. Crucially, it serves as a shared backbone for all behavioral primitives, ensuring that pursuit, cohesion, and separation are grounded in a unified perceptual representation while still allowing each to specialize modularly [8, 10].

From this latent representation, three parallel output heads compute unnormalized scalar activations ($\widetilde{\alpha_t}, \widetilde{\beta_t}, \widetilde{\gamma_t}$), each corresponding to one behavioral primitive:

$$[\widetilde{\alpha_t}, \widetilde{\beta_t}, \widetilde{\gamma_t}] = ActorHeads(z_t) \tag{15}$$

To guarantee interpretability and stability, these activations are normalized using a softmax transformation, producing a triplet of adaptive behavioral weights:

$$[\alpha_t, \beta_t, \gamma_t] = softmax([\widetilde{\alpha_t}, \widetilde{\beta_t}, \widetilde{\gamma_t}]) \tag{16}$$

with the explicit formulation:

$$\begin{aligned} \alpha_t &= \frac{exp^{\widetilde{\alpha_t}}}{(exp^{\widetilde{\alpha_t}} + exp^{\widetilde{\beta_t}} + exp^{\widetilde{\gamma_t}})} \\ \beta_t &= \frac{exp^{\widetilde{\beta_t}}}{(exp^{\widetilde{\alpha_t}} + exp^{\widetilde{\beta_t}} + exp^{\widetilde{\gamma_t}})} \\ \gamma_t &= \frac{exp^{\widetilde{\gamma_t}}}{(exp^{\widetilde{\alpha_t}} + exp^{\widetilde{\beta_t}} + exp^{\widetilde{\gamma_t}})} \end{aligned} \tag{17}$$

This formulation guarantees that: $(\alpha,\beta,\gamma) \in [0, 1]^3$ and $\alpha + \beta + \gamma = 1$, allowing a probabilistic interpretation of role allocation. At each time step, the model dynamically adjusts the emphasis on pursuit, cohesion, or separation according to the agent's local context.

The final continuous control vector is obtained through a convex combination of the pre-computed force vectors, as defined in Eq. (8):

$$\overrightarrow{F_{total}} = \alpha_t \overrightarrow{F_{goal}} + \beta_t \overrightarrow{F_{coh}} + \gamma_t \overrightarrow{F_{sep}} \tag{18}$$

The resultant vector $\overrightarrow{F_{total}}$ constitutes an optimal synthesis of behavioral primitives, as it preserves the directional information from pursuit, cohesion, and separation while adaptively weighting them according to the agent's local context. Unlike the selection of a single primitive, this convex combination ensures continuity of behavior, robustness to dynamic environments, and faithful preservation of the underlying strategic intent when mapped onto a discrete action space.

4.2.5 Direction discretization via angular mapping

Although the pursuit-evasion environment is defined over a discrete action space, we deliberately adopted a continuous control algorithm (DDPG) as the foundation of our proposed framework. The motivation stems from the fact that the underlying decision process is inherently continuous: agents interact through force-based primitives (pursuit, cohesion, and separation), which are naturally expressed as continuous motion vectors in R^2. Similar continuous-to-discrete control strategies have been explored in prior reinforcement learning and robotics literature, where continuous policy learning is combined with discretized action execution to preserve expressiveness during optimization [7, 12, 30, 32, 34, 37, 38]. Directly modeling these interactions in discrete space obscures directional nuances and restricts the expressivity of emergent coordination. In contrast, continuous policies allow the actor network to modulate the relative weights $(\alpha_t, \beta_t, \gamma_t)$ smoothly, ensuring richer behavioral diversity and more precise adaptations to local contexts before discretization.

To execute actions in the grid-based environment, the continuous control vector $\overrightarrow{F_{total}}$ (Eq. (8)) is projected into the discrete action space via angular mapping. Specifically, the orientation angle is computed as:

$$\theta = arctan2\left(F_y, F_x\right) \tag{19}$$

where, arctan2(.) ensures quadrant disambiguation and robust handling of directional signs [33]. The angle θ is then compared with the canonical direction angles corresponding to the five discrete moves, and the action minimizing the angular distance is selected (Table 1).

Table 1. Definition of cardinal target direction

Discrete Action at	Direction Vector (dx,dy)	Angular Range (Degrees)
Up	(0, +1)	[45°,135°]
Down	(0, -1)	[225°,315°]
Left	(-1, 0)	[135°,225°]
Right	(+1, 0)	[-45°,45°]
Stay (No movement)	(0, 0)	N/A

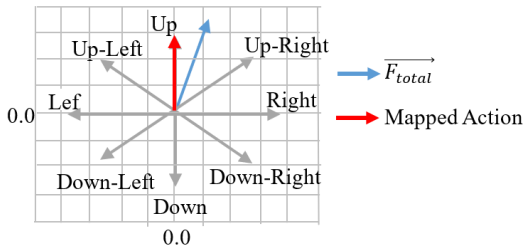

Figure 4. Mapping resultant vector to discrete action

This nearest-direction projection guarantees that the chosen discrete action remains the closest approximation of the intended continuous force, thus preserving the strategic intent of the policy (Figure 4). Importantly, no major loss of information occurs because the grid constraints inherently restrict the expressivity to five moves; hence, any control architecture, whether continuous or discrete, must eventually map to this finite set. The advantage of the continuous formulation lies in its intermediate flexibility: two continuous vectors pointing in slightly different directions but mapped to the same discrete move still influence the learning process differently, as they produce distinct gradients during the actor-critic optimization. This ensures that the training signal preserves fine-grained information even if the final executed action is discrete.

Such discretization strategies have long been adopted in robotics and multi-agent navigation, where continuous velocity commands are transformed into grid-aligned primitives for path planning and cooperative behaviors [25, 30]. The use of arctan2(·) further provides a deterministic

resolution in ambiguous cases (e.g., diagonal forces), ensuring consistency across agents and stability in swarm dynamics [33].

The algorithm of the Multi-Head Actor Network has been presented in Algorithm 1.

Algorithm 1. Multi-Head Actor Network with Force-Based Modulation

Initialize the actor network $\mu(O_a^t; \theta_\mu)$ with a shared backbone and three behavioral heads (pursuit, cohesion, separation) producing raw outputs $(\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma})$.
Initialize the critic network $Q(s, a; \theta_Q)$.
Create target networks by deep-copying parameters: $\theta_{\mu'} \leftarrow \theta_\mu$ and $\theta_{Q'} \leftarrow \theta_Q$.
Initialize an empty replay buffer B of capacity D.
Initialize the exploration noise process $\mathcal{N}_t$.
Begin
For episode = 1 to M:
a. Reset environment E and receive initial local observations O_a^t for all agents
b. For t = 0 to T:
i. Forward pass through the actor to compute $(\widetilde{\alpha_t}, \widetilde{\beta_t}, \widetilde{\gamma_t})$ using Eq. (15)
ii. Apply softmax to obtain normalized weights $(\alpha_t, \beta_t, \gamma_t)$ using Eq. (16)
iii. Compute forces vectors $\overrightarrow{F_{\text{goal}}}$, $\overrightarrow{F_{coh}}$, $\overrightarrow{F_{sep}}$ using Eqs. (9)-(14)
iv. Form weighted action a_t^{cont} using Eq. (20) and add exploration noise: a_t^{noisy} using Eq. (21)
v. Discretize a_t^{disc} using Eq. (19)
vi. Execute action a_t^{disc} in E, and observe $(O_a^{t+1}, r_t, s_{t+1}, done_t)$
Store $(s_t, O_a^t, a_t^{noisy}, r_t, s_{t+1}, done_t)$ in B
vii. Sample a mini-batch $(s_j, O_a^j, a_j^{noisy}, r_j, s_{j+1}, done_j)$ from B
viii. Critic update:
compute target action: $a_j' = \mu'(O_a^{j+1}; \theta_{\mu'})$ compute target value:

$$y_j = r_j + (1 - done_j)\gamma Q'(s_{j+1}, a_j' | \theta_{Q'})$$

Minimize critic loss:

$$L_Q = \frac{1}{N}\sum_{j=1}^{N}[y_j - Q(s_j, a_j^{noisy} | \theta_Q)]^2$$

Update critic $\theta_Q \leftarrow \theta_Q -$ ηcritic $\nabla\theta_Q L_Q$
ix. Actor update: compute policy gradient $\nabla_{\theta_\mu} J$ using Eq. (24) and update actor $\theta_\mu \leftarrow \theta_\mu +$ ηactor $\nabla_{\theta_\mu} J$
x. Soft-update target using Eq. (25)
End for (timestep loop)
End for (episode loop)

The proposed angular mapping bridges the gap between continuous force-based reasoning and discrete-action execution. This enables our Multi-Head DDPG to exploit the expressive power of continuous control during learning while maintaining compatibility with the grid-based pursuit-evasion task. This design choice ensures that the emergent behaviors are strategically optimal, interpretable, and robust.

4.3 Training procedure of the Multi-Head Actor network

The proposed Multi-Head Actor Network is trained under the CTDE paradigm, ensuring that each agent operates solely on local observations during execution while exploiting global information during centralized training via a shared replay buffer. This paradigm has been shown to be effective in MARL in cooperative and competitive environments [8]. The optimization builds upon the DDPG framework [12], which is adapted here to incorporate the force-based action representation and dynamic role-weighting mechanism described in Section 4.2.

Let μ $(O_a^t; \theta\mu)$ denote the deterministic actor network parameterized by $\theta\mu$, mapping the local observation O_a^t to a continuous control vector $a_t^{count} \in R^2$. The network architecture consists of a shared backbone (MLP) that encodes O_a^t into a latent vector z_t, which is then processed by three behavioral heads (pursuit, cohesion, separation). These heads produce raw activations $(\widetilde{\alpha_t}, \widetilde{\beta_t}, \widetilde{\gamma_t})$, which are normalized via the softmax function into adaptive weights $(\alpha_t, \beta_t, \gamma_t)$ as in Eq. (16). Based on the encoded observation, three behavioral forces $\overrightarrow{F_{\text{goal}}}$, $\overrightarrow{F_{coh}}$, and $\overrightarrow{F_{sep}}$ are computed (Eqs. (9)-(14)). The actor's continuous control action is then obtained as the weighted combination.

$$a_t^{cont} = \alpha\overrightarrow{F_{goal}} + \beta\overrightarrow{F_{coh}} + \gamma\overrightarrow{F_{sep}} \tag{20}$$

Exploration noise $\mathcal{N}_t$ (e.g., Gaussian noise [37]) is added in the continuous space before discretization:

$$a_t^{noisy} = a_t^{cont} + \mathcal{N}_t \tag{21}$$

and mapped to the discrete action space A = {up, down, left, right, stay} via nearest-direction mapping to produce a_t^{disc} (using Eq. (19)) for environment execution. Transitions $(s_t, O_a^t, a_t^{noisy}, r_t, s_{t+1}, done_t)$ are stored in the replay buffer B [6], where $done_t \in \{0,1\}$denotes the done flag indicating whether the episode terminates at step t ($done_t = 1$) or continues ($done_t = 0$). This signal is crucial for learning because it prevents the critic from propagating future value estimates beyond terminal states. The critic $Q(s, a; \theta_Q)$ is then updated by minimizing the temporal-difference loss as follows:

$$L_Q(\theta_Q) = \frac{1}{N}\sum_{j=1}^{N}[y_j - Q(s_j, a_j^{noisy} | \theta_Q)]^2 \tag{22}$$

with target values:

$$y_j = r_j + (1 - done_j)\,\gamma Q'\left(s_{j+1}, \mu'\left(O_a^{j+1} \middle| \theta_{\mu'}\right) \middle| \theta_{Q'}\right) \tag{23}$$

where, μ' are target networks with parameters $\theta_{\mu'}$ and $\theta_{Q'}$. The actor parameters are updated using the deterministic policy gradient [7]:

$$\nabla_{\theta_\mu} J(\theta_\mu) \approx \frac{1}{N}\sum_{j=1}^{N}\left(\nabla_a Q\left(s_j, a | \theta_Q\right)\Big|_{a=\mu(O_a^t;\theta_\mu)} \nabla\theta_\mu\, \mu\left(O_a^t | \theta_\mu\right)\right) \tag{24}$$

followed by Polyak averaging:

$$\theta_{Q'} \leftarrow \tau\theta_Q + (1 - \tau)\theta_{Q'}, \quad \theta_{\mu'} \leftarrow \tau\theta_\mu + (1 - \tau)\theta_{\mu'} \tag{25}$$

4.4 Interpretable role allocation and emergent self-organization

The Multi-Head DDPG architecture provides interpretability by explicitly representing each agent's decision as a weighted combination of three forces: pursuit, cohesion, and separation. The agent's real-time priorities are revealed by a triplet of normalized weights (α, β, γ), which can be directly inspected during execution.

This formulation facilitates emergent role differentiation, as agents are not constrained to preassigned static strategies. For example, in open environments, an agent's policy tends to prioritize pursuit (α), whereas in crowded spaces, it increasingly emphasizes cohesion (β) and separation (γ). This dynamic specialization (e.g. as a chaser or blocker) arises naturally from local observations without requiring central control or explicit communication.

This mechanism constitutes a form of distributed self-organization that echoes the collective intelligence observed in biological swarms [27, 36]. It illustrates how complex global strategies, such as encirclement, can emerge from simple, interpretable local rules. The ability to monitor the force weights transforms the model from a mere control system into a diagnostic and explanatory tool, bridging the gap between reinforcement learning and explainable artificial intelligence (XAI) [11]. As a result, the system becomes more scalable, robust, and resilient for real-world deployment.

5. EXPERIMENTAL SETUP AND RESULTS

This section presents the empirical evaluation of the proposed Multi-Head DDPG architecture in a decentralized multi-agent pursuit–evasion scenario. The analysis is organized around four core dimensions: (i) quantitative performance, (ii) learning stability and behavioral dynamics, (iii) generalization and robustness, and (iv) interpretability through emergent self-organization. Additionally, we benchmarked the proposed method against established baselines and provided a dedicated analysis of weighted force coupling and control vector construction. Figures 5-10 jointly illustrate how learning dynamics, interpretability, and emergent self-organization arise from the proposed modular actor design.

5.1 Experimental setup

To assess the performance and robustness of our approach, we implemented a 2D grid-based pursuit-evasion simulation environment with fixed dimensions of N × N. The environment consists of ten cooperative pursuer agents and two evading targets, where agents must coordinate to intercept all evaders as quickly and efficiently as possible (see Figure 5 for an illustration of the environment).

Algorithm-specific hyperparameters (e.g., network architecture, learning rates, replay buffer size, and target update coefficients) are summarized in Table 2.

Each agent operates under a partially observable decentralized framework and receives a local observation vector at each time step t. The agents can choose a discrete movement action from the set A = {up, down, left, right, stay}. A capture event is registered when the Euclidean distance between the pursuer and evader satisfies $\|pos_p - pos_e\|_2 \leq d_{cap}$ with $d_{cap} = 1$, corresponding to the pursuer being in the same or an immediately adjacent cell in the Von Neumann neighborhood (no diagonals).

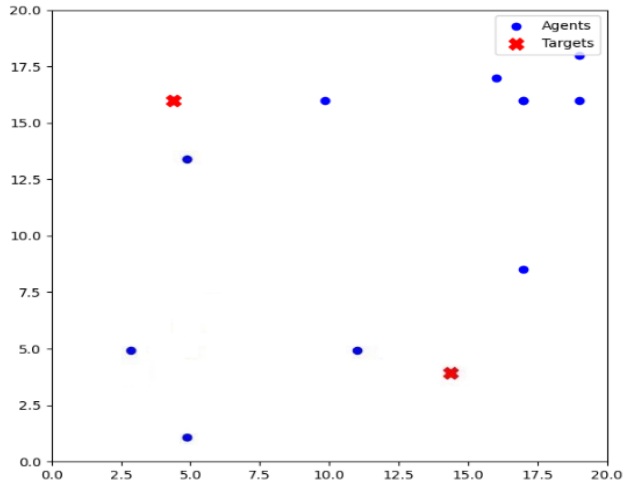

Figure 5. Pursuit-evasion environment (20 × 20 Grid)

Table 2. Hyperparameter configuration for baseline algorithms

Parameter	DQN	PPO	DDPG	Multi-Head DDPG
Network architecture	2 × 128 MLP	2 × 128 MLP	2 × 128 MLP	Shared 2 × 128 MLP + 3 heads
Actor learning rate	—	3e−4	1e−4	1e−4
Critic learning rate	1e−4	3e−4	1e−3	1e−3
Replay buffer size	1e5	—	1e5	1e5
Target update τ	—	—	0.005	0.005

To ensure a fair and reproducible comparison across the learning algorithms, the following experimental conditions were uniformly applied:

- Agent configuration: 10 pursuers and 2 evaders.
- Grid size: 20 × 20 cells.
- Training duration: 1,000 episodes, with each episode capped at 100 times steps.
- Discount factor: All models used γ = 0.95.
- Batch size: 128.
- Optimizer: Adam for all neural networks [1].

Exploration strategies followed standard algorithm-specific practices. DQN employed an epsilon-greedy strategy with linear annealing, PPO relied on stochastic policy sampling, and actor–critic methods (standard DDPG and Multi-Head DDPG) used additive Gaussian noise $N(0, \sigma^2)$ [34, 37]. Additional experiments using Ornstein–Uhlenbeck noise [12] confirmed the robustness of the results with respect to the choice of exploration process.

To mitigate stochasticity and initialization variance, each algorithm was trained and evaluated over ten independent runs with different random seeds. All reported metrics are presented as mean ± standard deviation after convergence.

All experiments were conducted on a workstation equipped with an Intel Core i9-12900K CPU (3.2 GHz), 64 GB RAM, and an NVIDIA RTX 3090 GPU (24 GB VRAM), running Windows 11 with PyTorch 2.0.

Algorithm-specific hyperparameters (e.g., network architecture, learning rates, replay buffer size, and target update coefficients) are summarized in Table 2.

5.2 Comparative methods

To rigorously validate the contribution of our approach, we compared the proposed architecture with three representative and widely recognized RL baselines:

Deep Q-Network (DQN): A value-based RL algorithm effective for discrete action spaces [6]. This serves as a fundamental baseline for measuring the benefits of continuous-action extensions.

Proximal Policy Optimization (PPO): A robust and sample-efficient policy-gradient method considered a state-of-the-art baseline for policy optimization in multi-agent systems [15].

DDPG (Standard): A continuous-action actor-critic method [12], that serves as a direct baseline to isolate the effects of our behavioral decomposition and modular actor.

Multi-Head DDPG (Proposed): Our extension of the DDPG introduces a modular actor with three behavioral heads (pursuit, cohesion, and separation). These heads are dynamically weighted to form a force-based control vector that enables interpretable and adaptive decision-making.

This spectrum of baselines allows us to evaluate the performance across discrete versus continuous action spaces, monolithic versus modular architectures, and value-based versus policy-gradient paradigms.

5.3 Quantitative results

To quantitatively assess our method, we conducted ten independent training runs with different random seeds to mitigate stochastic variance. Upon convergence, each trained policy was evaluated over 1,000 noise-free episodes to obtain stable and unbiased performance estimates. Evaluation was based on three key metrics: Success Rate (percentage of episodes in which all evaders were captured), Average Reward (mean cumulative return per episode, reflecting both efficiency and success), and Steps to Capture (mean number of timesteps required to capture the final evader). The findings are presented in Table 3. Importantly, the reward values reported here correspond to normalized evaluation rewards, ensuring comparability across methods.

To assess the statistical significance of the observed performance differences, we conducted pairwise Welch's t-tests between the proposed Multi-Head DDPG and each baseline across the 10 independent runs. For all three evaluation metrics (success rate, average reward, and steps to capture), the improvements achieved by the proposed method were found to be statistically significant ($p < 0.05$).

Relative to the standard DDPG baseline, the proposed Multi-Head DDPG achieves a +12.6% absolute improvement in success rate, a +2.7 point increase in average reward, and a −10.8 timestep reduction in capture time. Compared to PPO, it yields a +17.8% absolute success rate gain, a +4.4 point reward increase, and a −15.3 timestep reduction. Against DQN, improvements are even more pronounced, with a +24.0% absolute success rate gain, a +5.2 point reward increase, and a −19.6 timestep reduction.

These consistent improvements across baselines validate our hypothesis that explicit behavioral decomposition into pursuit, cohesion, and separation forces, modulated by adaptive role weights, enables superior coordination under partial observability. Shorter capture times indicate more decisive and coordinated pursuit maneuvers, whereas higher rewards reflect greater robustness and generalization.

Table 3. Final performance comparison on the pursuit-evasion task (10 Runs, Mean ± Standard Deviation)

Algorithm	Success Rate	Avg Reward	Steps to Capture
Multi-Head DDPG (Ours)	85.2 ± 2.1	14.5 ± 0.9	28.9 ± 2.3
DDPG (Standard)	72.6 ± 3.4	11.8 ± 1.3	39.7 ± 2.6
PPO	67.4 ± 2.2	10.1 ± 1.1	44.2 ± 2.9
DQN	61.2 ± 2.7	9.3 ± 1.5	48.5 ± 3.1

5.4 Learning curves and convergence behavior

To investigate stability and sample efficiency, we monitored the evolution of raw average episodic reward during training for all four algorithms (DQN, PPO, DDPG, Multi-Head DDPG) over 1,000 episodes (10 runs averaged). This comparison is visually summarized in Figure. 6. Unlike the normalized metrics reported in Table 3, these learning curves illustrate the unnormalized reward dynamics throughout training.

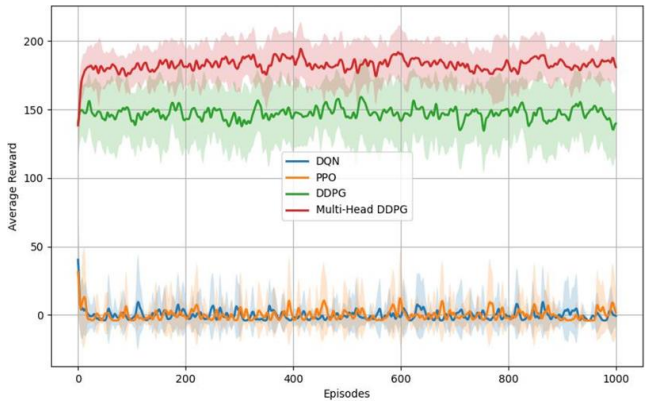

Figure 6. Training curves of the raw (non-normalized) average episodic reward for DQN, PPO, standard DDPG, and the proposed Multi-Head DDPG, averaged over 10 independent runs

The curves illustrate learning stability, convergence speed, and asymptotic performance; normalized rewards are reported separately in Table 3.

Because DDPG was originally designed for continuous spaces [12], its outputs were projected onto discrete directions via force-based angular mapping. This ensures a fair comparison with the DQN and PPO, which natively operate in discrete domains.

The proposed Multi-Head DDPG exhibited the most favorable learning dynamics, achieving rapid convergence within the first 100 episodes and maintaining the highest asymptotic reward (~180-190). In contrast, the standard DDPG plateaus at approximately 150 with a higher variance, indicating sensitivity to exploration noise and hyperparameters. PPO demonstrated slower and unstable convergence, ultimately yielding a near-zero average reward, whereas DQN failed to progress beyond negative to near-zero values, confirming the inadequacy of value-based methods in this setting.

These results highlight three main findings: (i) accelerated

convergence and superior asymptotic performance of the Multi-Head DDPG relative to all baselines; (ii) reduced variance across runs, evidencing robustness against non-stationarity; and (iii) clear limitations of monolithic or value-based approaches, which lack the representational flexibility required for fine-grained coordination. Overall, this analysis confirms that modular behavioral decomposition with adaptive weighting substantially enhances both learning efficiency and reliability in decentralized multi-agent pursuit-evasion tasks.

5.5 Behavioral role differentiation and interpretability

A distinctive advantage of the proposed Multi-Head DDPG lies in its interpretable actor design, where each action is parameterized by a triplet of role weights (α, β, γ) corresponding to the pursuit, cohesion, and separation forces, respectively. This explicit decomposition transforms latent policy modulations into observable quantities, thereby providing a direct window into the decision-making process of agents. Figure 7 illustrates the temporal evolution of the role weights across 1,000 steps for all ten pursuers, whereas Figure 8 summarizes their mean profiles over the entire horizon. After an initial transient phase (~150-200 steps), the majority of agents converge to stable yet heterogeneous configurations, reflecting emergent specialization. For instance, Agent 2 predominantly acts as a pursuit specialist (mean $\alpha \approx 0.94$), Agent 3 emphasizes cohesion ($\beta \approx 0.97$), and Agent 1 adopts a separation-dominant strategy ($\gamma \approx 0.96$) with a reduced pursuit. Conversely, Agent 7 exhibited high pursuit and cohesion but persistently low separation, corresponding to a risk-taking chaser. In contrast, several agents (e.g. Agents 0, 4, 5, 8, and 9) maintained balanced weight distributions (α, β, $\gamma \approx 0.95$-0.98), consistent with more generalist and adaptable behaviors.

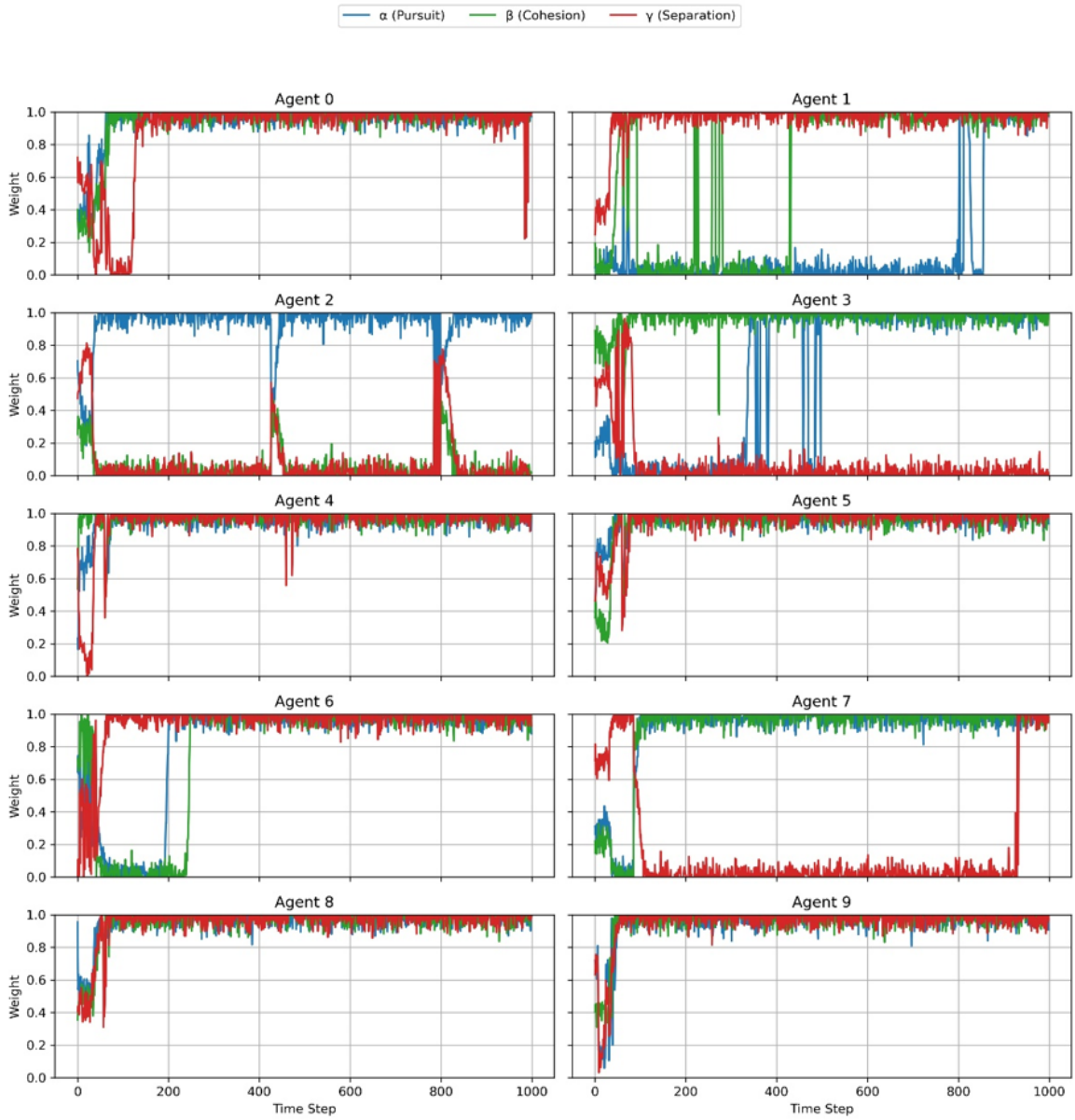

Figure 7. Temporal evolution of role weights (α, β, γ) for each pursuer

The heatmap in Figure 8 quantitatively confirms these trends by distinguishing between specialized and balanced roles without any pre-assignment or external coordination. Such differentiation arises end-to-end from decentralized learning dynamics under partial observability, thereby validating the self-organizing capacity of the architecture.

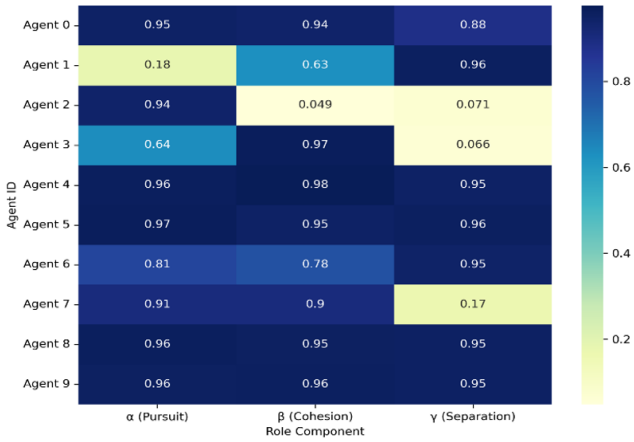

Figure 8. Heatmap of mean role weights (α, β, γ) per agent

Importantly, the ability to monitor (α, β, γ) over time not only enhances interpretability but also facilitates human-in-the-loop supervision, systematic diagnosis of emergent behaviors, and more reliable deployment in safety-critical multi-agent scenarios.

5.6 Weighted force coupling and control vector construction

To elucidate the internal decision-making process of the proposed architecture, the control vector at each time step is expressed as a weighted linear combination of three interpretable primitive forces—pursuit, cohesion, and separation—as formalized in Eq. (8). Figures 8 and 9 provide complementary perspectives: the trajectories of the adaptive role weights (α, β, γ) for all agents and the corresponding force magnitudes ||F_goal||, ||F_coh||, ||F_sep||, together with the resultant ||F_total||.

Three salient observations emerge from this analysis. First, separation consistently accounted for the largest share of the force budget. As illustrated by the recurrent peaks of ||F_sep|| (often in the range of 3-4) for Agents 5-9, collision avoidance dominated when the agents operated in congested configurations. In contrast, ||F_goal|| and ||F_coh|| remained comparatively smaller (typically below 1.5), except during sparse regimes or post-dispersion phases. Second, despite the high variability in the primitive forces, the resultant ||F_total|| exhibits smooth temporal evolution, demonstrating that the weighted coupling acts as a stabilizing mechanism. This effectively ensures coherent motion policies and mitigates the risk of erratic behavior. Third, the modulation of role weights directly reflects context-sensitive adaptation: pursuit forces weaken when agents close in on targets, whereas cohesion and separation forces increase under high-density interactions, consistent with transient rises in β and γ.

Overall, these findings highlight the operational transparency of the proposed architecture. The explicit mapping between role weights and force magnitudes provides an interpretable bridge between internal policy modulation and observable control signals, thereby reinforcing both the robustness and explainability of the emergent behaviors of the agent.

5.7 Generalization and robustness

To evaluate the generalization ability and robustness of the learned policies beyond their training distribution, we designed a set of out-of-distribution (OOD) scenarios that reflect the practical challenges of real-world multi-agent coordination. Specifically, we considered three settings:

Increased Agent Density — 15 agents instead of 10, leading to greater interaction complexity and collision potential.

Faster Evaders — evaders move more rapidly, making pursuit and interception more difficult to achieve.

Presence of Dynamic Obstacles — introduces environmental unpredictability, requiring enhanced situational awareness.

The success rates across these settings are reported in Table 4, which compares the performance of four algorithms: DQN, PPO, standard DDPG, and our proposed Multi-Head DDPG.

Several key observations emerge from this evaluation.

Superior generalization of Multi-Head DDPG: In all three scenarios, our method achieved the highest success rate, with margins of over 10% compared to the next best baseline. This consistent superiority indicates that the learned policy does not overfit the training conditions and adapts effectively to novel dynamics and spatial configurations.

Table 4. Success rates in out-of-distribution scenarios

Scenario	DQN	PPO	DDPG Std	Multi-Head DDPG
Dense Swarm (15 agents)	54.3%	61.8%	67.1%	81.7%
Faster Evaders	45.6%	52.0%	60.4%	75.3%
With Obstacles	40.1%	49.2%	57.3%	69.8%

Impact of modular force-based decomposition. The strength of the proposed architecture lies in its decomposition of control into interpretable behavioral forces (pursuit, cohesion, and separation) with adaptive weighting. This modular representation allows the policy to generalize its behavioral response, even when facing previously unseen variations in the agent's behavior or environmental structure.

Limitations of value-based and monolithic approaches. DQN, which lack continuous modulation capabilities, suffer the most across all settings. PPO and standard DDPG performed moderately better but still lacked the flexibility of our Multi-Head modulation strategy when exposed to high-density or nonstationary environments.

In summary, these results confirm that the Multi-Head DDPG offers both robust performance and strong generalization capacity, reinforcing the hypothesis that modular policy structures are more suitable for scalable real-world multi-agent coordination.

5.8 Emergent self-organization

We assessed whether the proposed Multi-Head DDPG facilitates emergent self-organization through both individual and collective analyses. At the agent level, the temporal trajectories of role weights (α, β, γ) indicated heterogeneous specialization, with some agents consistently prioritizing pursuit, while others emphasized cohesion or separation. Crucially, these roles are not predefined but emerge autonomously from decentralized interactions under partial observability.

This demonstrates the ability of the architecture to induce role differentiation without external supervision.

At the collective level, coordination metrics demonstrated structured group behaviors, including reduced collision rates, compact clustering around evaders, and spontaneous

emergence of encirclement and blocking formations. These dynamics are exemplified in Figure 10, Multi-agent simulation demonstrating emergent cooperative coordination and encirclement of evaders, which visually depicts how agents self-organize into cohesive formations to contain the targets. Such patterns were absent in the DQN, PPO, and standard DDPG, underscoring the distinct advantages of the proposed architecture.

Overall, these results show that Multi-Head DDPG does more than improve task performance; it induces interpretable emergent coordination grounded in local decision-making. This capacity to monitor and quantify emergent dynamics strengthens the link between reinforcement learning and explainable multi-agent control, enhancing the scalability and robustness of decentralized systems.

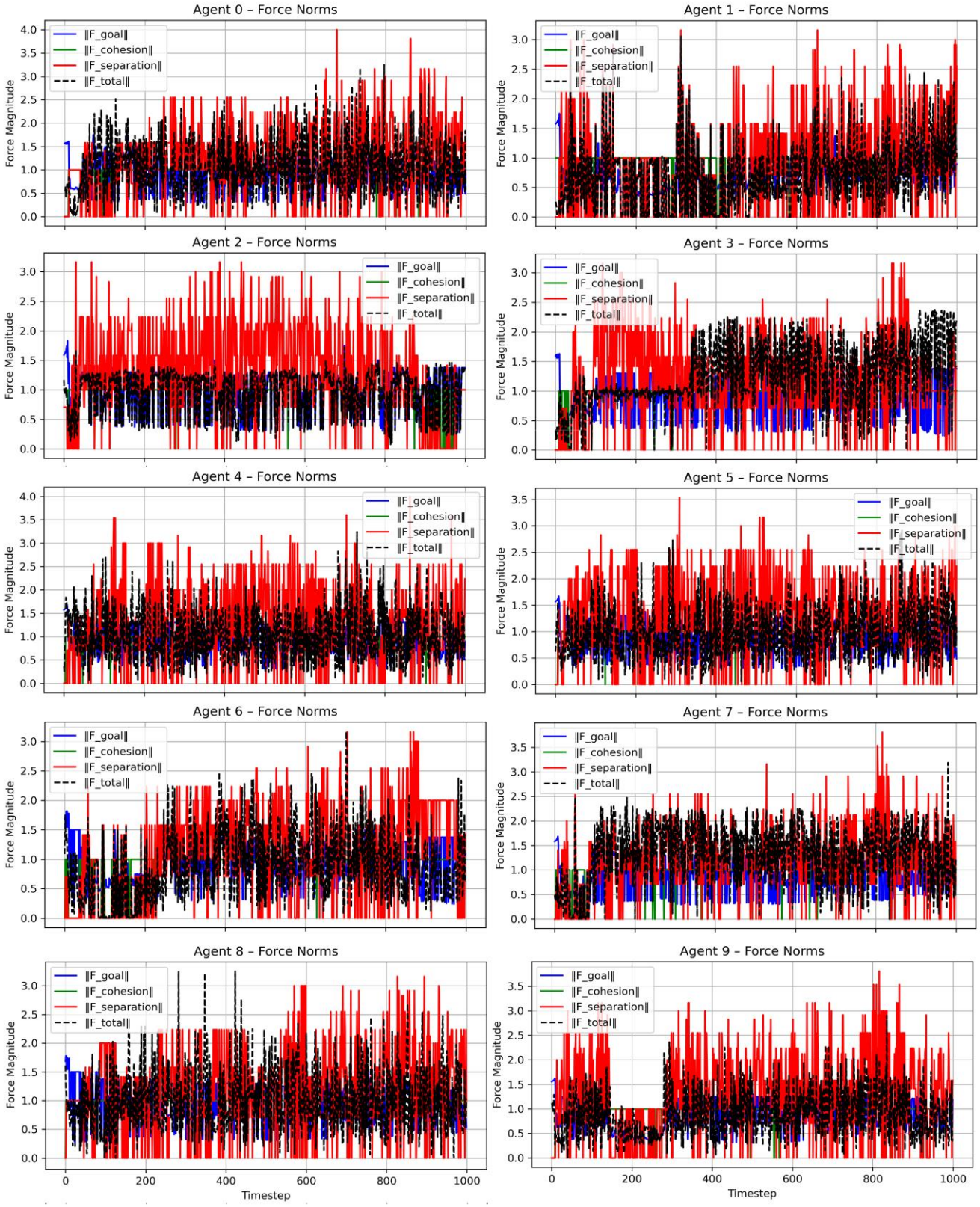

Figure 9. Coupling between adaptive role weights and corresponding force magnitudes

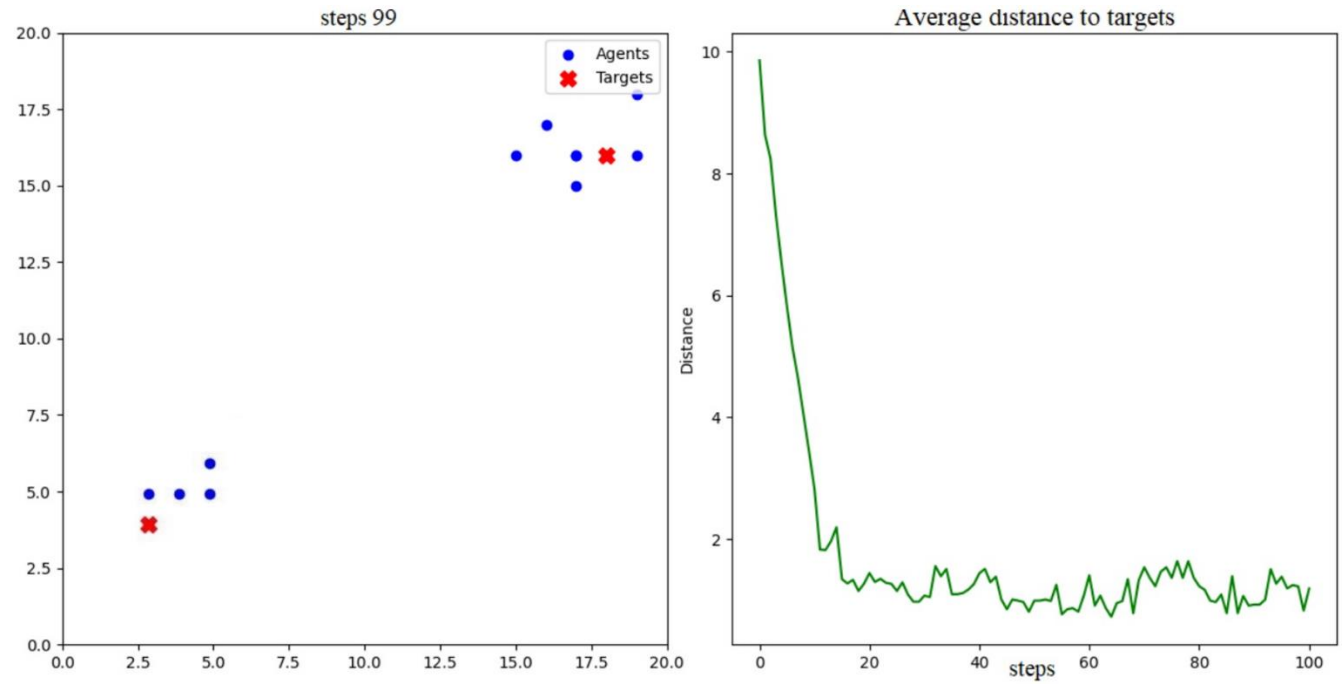

Figure 10. Pursuit-evasion episode illustrating emergent self-organization

6. CONCLUSION

This study introduces a novel extension of the DDPG algorithm for decentralized multi-agent coordination, leveraging a Multi-Head Actor architecture that adaptively balances three interpretable behavioral primitives—pursuit, cohesion, and separation— dynamically modulated weights (α, β, γ).

Extensive empirical evaluation against strong baselines (DQN, PPO, standard DDPG) shows that the proposed approach achieves consistent improvements across four dimensions: **(i)** superior task performance and efficiency, **(ii)** accelerated convergence with enhanced stability, **(iii)** interpretability through role-specific weight trajectories, and **(iv)** robust generalization to out-of-distribution scenarios involving denser swarms, faster evaders, and cluttered environments. Further coupling analysis confirm that emergent macroscopic behaviors arise coherently from modular force decomposition, offering both transparency and reliability in collective decision-making.

Beyond performance, this study underscores the potential of modular policy structures to bridge reinforcement learning with explainable multi-agent control. Future research directions include hierarchical role coordination, human-in-the-loop guidance, hardware deployment in robotic swarms, and theoretical analyses of convergence and stability guarantees.

In conclusion, the Multi-Head DDPG framework provides a scalable, interpretable, and high-performance foundation for cooperative autonomy, with direct applicability to swarm robotics, surveillance, and multi-agent search-and-rescue missions.

ACKNOWLEDGMENT

We are grateful to our colleagues and laboratories for supporting this research. Special thanks are extended to Drs. Wahid Chergui, Mohammed El Habib Souidi and Abdelaali Bekhouche for their valuable help and advice.

REFERENCES

[1] Wong, A., Bäck, T., Kononova, A.V., Plaat, A. (2023). Deep multiagent reinforcement learning: Challenges and directions. Artificial Intelligence Review, 56(6): 5023-5056. https://doi.org/10.1007/s10462-022-10299-x

[2] Gronauer, S., Diepold, K. (2022). Multi-agent deep reinforcement learning: A survey. Artificial Intelligence Review, 55(2): 895-943. https://doi.org/10.1007/s10462-021-09996-w

[3] Ye, D., Zhang, M., Vasilakos, A.V. (2016). A survey of self-organization mechanisms in multiagent systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47(3): 441-461. https://doi.org/10.1109/TSMC.2015.2504350

[4] Liu, H., Long, X., Li, Y., Yan, J., et al. (2025). Adaptive multi-UAV cooperative path planning based on novel rotation artificial potential fields. Knowledge-Based Systems, 317: 113429. https://doi.org/10.1016/j.knosys.2025.113429

[5] Shoham, Y., Leyton-Brown, K. (2008). Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press. https://doi.org/10.1017/CBO9780511811654

[6] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540): 529-533. https://doi.org/10.1038/nature14236

[7] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M. (2014). Deterministic policy gradient algorithms. In International Conference on Machine Learning, pp. 387-395.

[8] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I. (2017). Multi-agent actor-critic for mixed

cooperative-competitive environments. arXiv preprint arXiv:1706.02275. https://doi.org/10.48550/arxiv.1706.02275

[9] Christianos, F., Papoudakis, G., Rahman, M.A., Albrecht, S.V. (2021). Scaling multi-agent reinforcement learning with selective parameter sharing. In Proceedings of the 38th International Conference on Machine Learning, pp. 1989-1998.

[10] Foerster, J., Assael, I.A., De Freitas, N., Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. Advances in Neural Information Processing Systems, 29.

[11] Gunning, D., Aha, D. (2019). DARPA'S explainable artificial intelligence (XAI) program. AI Magazine, 40(2): 44-58. https://doi.org/10.1609/aimag.v40i2.2850

[12] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., et al. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971. https://doi.org/10.48550/arXiv.1509.02971

[13] Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., Shochet, O. (1995). Novel type of phase transition in a system of self-driven particles. Physical Review Letters, 75(6): 1226. https://doi.org/10.1103/PhysRevLett.75.1226

[14] Doshi-Velez, F., Kim, B. (2017). Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608. https://doi.org/10.48550/arXiv.1702.08608

[15] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347. https://doi.org/10.48550/arxiv.1707.06347

[16] Sun, L., Chang, Y.C., Lyu, C., Shi, Y., Shi, Y., Lin, C.T. (2023). Toward multi-target self-organizing pursuit in a partially observable Markov game. Information Sciences, 648: 119475. https://doi.org/10.1016/j.ins.2023.119475

[17] De Souza, C., Newbury, R., Cosgun, A., Castillo, P., Vidolov, B., Kulić, D. (2021). Decentralized multi-agent pursuit using deep reinforcement learning. IEEE Robotics and Automation Letters, 6(3): 4552-4559. https://doi.org/10.1109/LRA.2021.3068952

[18] Hüttenrauch, M., Šošić, A., Neumann, G. (2019). Deep reinforcement learning for swarm systems. Journal of Machine Learning Research, 20(54): 1-31.

[19] Yu, C., Dong, Y., Li, Y., Chen, Y. (2020). Distributed multi-agent deep reinforcement learning for cooperative multi-robot pursuit. The Journal of Engineering, 2020(13): 499-504. https://doi.org/10.1049/joe.2019.1200

[20] Park, C., Lee, H., Yun, W. J., Jung, S., Kim, J. (2022). Coordinated multi-agent reinforcement learning for unmanned aerial vehicle swarms in autonomous mobile access applications. arXiv preprint arXiv:2304.08493. https://doi.org/10.48550/arxiv.2304.08493

[21] Azzam, R., Boiko, I., Zweiri, Y. (2023). Swarm cooperative navigation using centralized training and decentralized execution. Drones, 7(3): 193. https://doi.org/10.3390/drones7030193

[22] Stone, P., Veloso, M. (1999). Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. Artificial Intelligence, 110(2): 241-273. https://doi.org/10.1016/S0004-3702(99)00025-9

[23] Chen, Y., Shi, Y., Dai, X., Meng, Q., Yu, T. (2025). Pursuit-evasion game with online planning using deep reinforcement learning. Applied Intelligence, 55(6): 1-17. https://doi.org/10.1007/s10489-025-06396-3

[24] Hansen, E.A., Bernstein, D.S., Zilberstein, S. (2004). Dynamic programming for partially observable stochastic games. In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 709-715.

[25] Şahin, E. (2004). Swarm robotics: From sources of inspiration to domains of application. In International Workshop on Swarm Robotics, pp. 10-20. https://doi.org/10.1007/978-3-540-30552-1_2

[26] Couzin, I.D., Krause, J., Franks, N.R., Levin, S.A. (2005). Effective leadership and decision-making in animal groups on the move. Nature, 433(7025): 513-516. https://doi.org/10.1038/nature03236

[27] Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. IEEE Transactions on Automatic Control, 51(3): 401-420. https://doi.org/10.1109/TAC.2005.864190

[28] Von Neumann, J., Burks, A.W. (1966). Theory of self-reproducing automata.

[29] Doya, K., Samejima, K., Katagiri, K.I., Kawato, M. (2002). Multiple model-based reinforcement learning. Neural Computation, 14(6): 1347-1369. https://doi.org/10.1162/089976602753712972

[30] LaValle, S.M. (2006). Planning Algorithms. Cambridge University Press. https://doi.org/10.1017/CBO9780511546877

[31] Gu, S., Holly, E., Lillicrap, T., Levine, S. (2017). Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, pp. 3389-3396. https://doi.org/10.1109/ICRA.2017.7989385

[32] Chen, Y.F., Everett, M., Liu, M., How, J.P. (2017). Socially aware motion planning with deep reinforcement learning. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1343-1350. https://doi.org/10.1109/IROS.2017.8202312

[33] Siegwart, R., Nourbakhsh, I.R., Scaramuzza, D. (2011). Introduction to Autonomous Mobile Robots. MIT Press.

[34] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv preprint arXiv:1801.01290. https://doi.org/10.48550/arXiv.1801.0129

[35] Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. In Proceedings of ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, pp. 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[36] Reynolds, C.W. (1987). Flocks, herds and schools: A distributed behavioral model. In Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, pp. 25-34. https://doi.org/10.1145/37402.37406

[37] Fujimoto, S., van Hoof, H., Meger, D. (2018). Addressing function approximation error in actor-critic methods. arXiv preprint arXiv:1802.09477. https://doi.org/10.48550/arXiv.1802.09477

[38] Tang, Y., Agrawal, S. (2020). Discretizing continuous action space for on-policy optimization. In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 5981-5988. https://doi.org/10.1609/aaai.v34i04.6059