

## **A Survey of Multi-Agent Reinforcement Learning from the Perspective of Task Characteristics**

Yongli Li, Shuochen Ji, Mingyao Gong\*, Yiping Zhu, Xu Zhao, and Lirong Zhou

School of Aeronautical Engineering, JILIN UNIVERSITY OF CHEMICAL TECHNOLOGY, Jilin  
132000, China

---

### **Abstract**

Multi-agent reinforcement learning (MARL), the relationship between agents' rewards determines part of the learning problem. Shared rewards give rise to credit assignment and coordinated action selection; opposed rewards require responses to an adapting opponent. Both relationships occur in mixed tasks. This survey takes these task differences as the starting point for reviewing MARL, complementing classifications based on algorithms, training frameworks, or application domains. The analysis follows task characteristics, interaction types, and algorithmic mechanisms across cooperative, competitive, and mixed cooperative-competitive settings. It covers the basic formulations and training and execution paradigms, then compares representative methods in terms of their mechanisms, applicability, and limitations. Path planning, task allocation, pursuit-evasion and adversarial engagement, autonomous driving, and resource allocation illustrate how information access and operational constraints affect method selection. The concluding discussion concerns non-stationarity, scaling, safety, and the evidence needed for deployment. The comparisons support method selection under stated task conditions, without treating applicability as independent of the task being addressed.

### **Keywords**

**Agents; Reinforcement Learning; Multi-agent Reinforcement Learning; Task Characteristics.**

---

### **1. Introduction**

Unmanned systems and robot swarms bring a particular difficulty to autonomous control: several agents influence the outcome, but their interests need not coincide. Coordination under a common objective differs from decision-making against an adversary. Even within one application, the reward relationship can change the kind of interaction a policy must represent. MARL provides a learning formulation for these coupled decisions by treating agents as participants in repeated interaction with both the environment and one another. The policies must still meet the application's demands for adaptation and real-time decision-making.

During training, the behavior being learned by one agent forms part of the conditions under which the others learn. Their value estimates may consequently become outdated as other policies are updated. This source of non-stationarity accompanies a separate scaling problem: additional agents enlarge the joint state-action space. In cooperative tasks, the team return must also be related to individual decisions. These difficulties make the task description important to method selection. An account of rewards and interaction patterns is needed to establish what the agents must learn and which information their policies can use.

Existing surveys cover MARL challenges and theory, cooperative learning, Internet applications, and multi-robot systems [1-8]. Their algorithmic and application-based classifications serve different purposes. A further comparison is needed to connect a task objective to an interaction model and then to a learning mechanism. Classifying a method as value-based, for example, does not specify whether its purpose is to allocate a team reward or respond to an opponent. Those requirements follow from the task, even when the underlying method family is the same. The relationship between these requirements and algorithm design is the focus of this review.

Accordingly, the literature is grouped into cooperative, competitive, and mixed cooperative-competitive tasks according to reward relationships and interaction patterns. Within each group, the review examines the interactions represented during learning, the mechanisms used to learn them, and the conditions that limit their use. Sections 2 and 3 establish the background and examine the corresponding algorithms. Section 4 considers their use in applications, with attention to task objectives and constraints, while Section 5 discusses deployment challenges and conclusions. This organization connects task characteristics to interaction types and algorithmic mechanisms throughout the discussion.

## **2. Background and Fundamental Problems**

The following background distinguishes how a policy is learned from how information is used during training and execution. Basic reinforcement learning and multi-agent formulations are introduced first, followed by the main method families and information-sharing paradigms. These distinctions underpin the task-based analysis in the subsequent sections.

### **2.1 Fundamentals of Reinforcement Learning and MARL**

Reinforcement learning concerns an agent that learns to pursue a goal through interaction with its environment. At each step, it selects an action from the current state; the environment then changes state and supplies a reward. The learning objective is to maximize expected cumulative return over repeated interactions. A common formulation is the Markov decision process, written as the five-tuple  $\langle S, A, P, R, \gamma \rangle$ . Here,  $S$  and  $A$  denote the state and action sets, respectively,  $P$  describes state transitions,  $R$  specifies rewards, and  $\gamma$  is the discount factor.

A Markov game extends this formulation to a system of  $N$  agents. Agent  $i$  selects an action from its local observation, and the resulting joint action determines how the environment state evolves. The difficulty is that the other agents also update their policies: an individual learner therefore encounters changing decision behavior rather than a fixed environment. Value estimates must accommodate this non-stationarity, making policy convergence and the development of coordinated behavior more difficult than in the single-agent case.

### **2.2 Classification of Methods**

#### **2.2.1 Value-Based Methods**

Q-learning, SARSA, and DQN are representative value-based methods. They learn a state-action value function and use its estimates to select actions, rather than optimizing a policy directly. Their relatively simple architectures are suited to discrete action spaces and small multi-agent systems. Scaling them to more complex cooperative tasks is harder. The joint action space expands rapidly as agents are added, while concurrent policy changes can destabilize value estimates. Ease of implementation thus does not remove the difficulties of learning a joint value function in a large system.

#### **2.2.2 Policy-Based Methods**

In policy-based learning, the policy is an explicit parameterized function whose parameters are adjusted to increase expected cumulative return. This formulation accommodates continuous actions and provides flexibility for high-dimensional control. The policy itself is optimized, instead of being obtained indirectly from value estimates. In a multi-agent system, the gradient is estimated while the policies governing other agents' actions are also being updated. The resulting variance and changing

interaction conditions can make training unstable. Centralized information and restrictions on policy updates are used to make the learning process more stable.

### 2.2.3 Actor-Critic Methods

The actor and critic have separate roles: the actor specifies a policy, and the critic estimates values used to improve it. Access to information need not be identical for the two. In a MARL implementation with centralized training and decentralized execution, a critic may receive joint actions, global state information, or information about other agents, while the deployed actor uses local observations. MADDPG [11] and MAPPO [10] follow this arrangement. Centralization thus concerns the information used for learning, without imposing a central decision-maker on the execution stage.

## 2.3 Training and Execution in MARL

Two information choices distinguish the training and execution paradigms: what the learner can access when updating its policy, and what the deployed policy needs to select an action. The usual combinations are centralized training with centralized execution (CTCE), decentralized training with decentralized execution (DTDE), and centralized training with decentralized execution (CTDE) [12]. For CTCE, joint observations and actions remain available to a central decision-maker in both stages. System-wide optimization is possible, but deployment depends on retaining that central controller and its access to global information as the joint space grows.

In DTDE, learning is carried out separately by agents using local observations, as is execution. The distributed architecture avoids the persistent centralized information requirement of CTCE. Its difficulty appears during learning: an agent generally treats the others as part of its environment without directly accounting for their policy updates. The observations generated by that environment then reflect changing behaviors, with adverse effects on sample efficiency and coordination.

CTDE reserves global information for training. Once training is complete, the actors choose actions from their local observations, so centralized information is not a prerequisite for every deployed decision. Compared with DTDE, the extra training information can help value estimation account for other agents' behavior and partly alleviate non-stationarity. It does not remove the cost of scale: centralized estimators still accept larger inputs as agents are added. More demanding training and limited generalization of the local policies remain relevant when selecting CTDE for a particular application.

## 3. MARL Algorithms Organized by Task Characteristics

**Table 1.** Basic characteristics of the three MARL task categories

| Type                          | Rewards and interactions                                        | Key challenges                                                            | Representative algorithms        |
|-------------------------------|-----------------------------------------------------------------|---------------------------------------------------------------------------|----------------------------------|
| Fully cooperative             | Shared team reward or alignment of objectives                   | Credit assignment; coordinated exploration; joint action-space growth     | VDN, QMIX, QPLEX, MAPPO          |
| Fully competitive             | Opposed rewards, including approximately zero-sum payoffs       | Responding to opponents; finding equilibria; generalizing across policies | Minimax-Q, self-play, NFSP, PSRO |
| Mixed cooperative-competitive | Team cooperation alongside competition and individual interests | Changing policies, relevant interaction modeling, value estimation        | MADDPG, MATD3, MAAC, MASAC       |

The distinction used in this review concerns how an outcome rewards the participating agents [9]. In a fully cooperative task, improvement is judged against a common team objective. For a two-agent zero-sum task, the payoff gained by one agent is lost by the other; this is the representative fully

competitive case. Mixed tasks allow team cooperation to coexist with competition and individual returns. Their reward relationships are therefore not described by uniform alignment or opposition. Table 1 summarizes the learning difficulties and representative methods associated with these reward structures.

### 3.1 Fully Cooperative MARL Algorithms

Fully cooperative MARL concerns agents whose objectives are aligned, usually through a shared team reward. Their actions influence both the environment and one another, and performance is measured by the collective return. A shared reward does not, by itself, specify which agent contributed to an outcome or which action each agent should take. The learning problem is therefore to represent contributions within the joint action space while obtaining policies that individual agents can execute locally. Credit assignment and decentralized decision-making are closely connected in this setting.

Value decomposition addresses this connection by relating a team value function to local agent values. VDN, introduced by Sunehag et al., expresses the joint action value as their sum, simplifying learning at the cost of excluding nonlinear coordination relationships [13]. QMIX replaces the sum with a nonlinear mixing network [14]. Its monotonicity constraint makes greedy actions under the local values consistent with greedy maximization of the represented joint value function, but restricts what that function can express. QTRAN uses value transformation to relax factorization conditions [15]. QPLEX takes a different route, using advantage decomposition to strengthen consistency between individual and global maximization [16].

Policy optimization offers another route to cooperation. MAPPO trains decentralized policies with a centralized value function and uses mechanisms such as PPO's clipped objective to stabilize updates [10]. For heterogeneous agents, HAPPO combines advantage decomposition with sequential policy updates [17]. These methods and the value decomposition approaches address the same task requirements through different learning mechanisms: attributing team returns, managing the joint action space, and obtaining stable decentralized policies. Their main contributions and limitations are compared in Table 2.

**Table 2.** Representative fully cooperative MARL algorithms

| Algorithm | Main contributions                                                                     | Main limitations                                                                    | Ref. |
|-----------|----------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|------|
| VDN       | Represents the team value as a sum of local values, simplifying joint value learning   | Additive representation excludes nonlinear coordination                             | [13] |
| QMIX      | Mixes local values monotonically to support consistent decentralized action selection  | Monotonicity excludes non-monotonic coordination relationships                      | [14] |
| QTRAN     | Uses value transformation to relax additive and monotonic factorization restrictions   | Constrained objectives complicate optimization and training stability               | [15] |
| QPLEX     | Uses advantage decomposition to reinforce individual-global maximization consistency   | Greater architectural and training complexity                                       | [16] |
| MAPPO     | Trains decentralized policies with centralized value estimates for more stable updates | Ongoing on-policy sampling; sensitivity to advantage estimation and hyperparameters | [10] |
| HAPPO     | Updates heterogeneous agents sequentially                                              | More involved updates and substantial training overhead                             | [17] |

### 3.2 Fully Competitive MARL Algorithms

In fully competitive tasks, improving one agent's outcome conflicts with the interests of another. The relationship is explicit in a two-agent zero-sum game, where an increase in one payoff is matched by a decrease in the other. A useful policy must therefore account for the opponent's possible responses,

not merely maximize returns against its present behavior. This requirement places opponent modeling, policy robustness, and equilibrium computation at the center of competitive MARL. Table 3 compares representative approaches and their limitations.

Littman's Minimax-Q evaluates actions in a Markov game against an optimal opponent response, incorporating minimax optimization into its value updates [18]. Explicit evaluation becomes difficult once actions are high-dimensional or continuous. Training against other policies offers a different formulation of the opponent: in self-play, current or earlier versions of a policy generate the interactions from which a learner improves [19]. NFSP maintains both a learned best response and an average policy, with the aim of approximating equilibrium play in imperfect-information games [20].

PSRO places the opponent distribution at the center of training [21]. A maintained population supplies the policies in a meta-game; approximate best responses are learned against its current meta-strategy distribution. The learner is thus exposed to a set of policies instead of being trained only against a single opponent. Wider coverage may reduce opponent-specific overfitting, but it requires population maintenance and meta-game computation. For larger competitive tasks, these costs accompany the move from explicitly computing equilibria toward learning responses that generalize across opponent policies.

**Table 3.** Representative fully competitive MARL algorithms

| Algorithm | Main contributions                                                                         | Main limitations                                                                       | Ref. |
|-----------|--------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|------|
| Minimax-Q | Applies minimax updates to value learning in zero-sum Markov games                         | Difficult to extend to high-dimensional, continuous, or imperfect-information settings | [18] |
| Self-play | Trains against current or historical policies to develop adaptive competitive behavior     | Risk of policy cycling and overfitting to past opponents                               | [19] |
| NFSP      | Learns best-response and average policies to approximate equilibria                        | Two policies to maintain; demanding sampling and computation                           | [20] |
| PSRO      | Trains responses to a meta-strategy over a policy population, broadening opponent coverage | Costly population maintenance and meta-game computation                                | [21] |

### 3.3 Mixed Cooperative-Competitive MARL Algorithms

Mixed cooperative-competitive tasks combine reward relationships that are neither uniformly aligned nor uniformly opposed. Agents may cooperate within a team, compete over resources or against another team, and pursue individual returns. Nash-Q, Correlated-Q, and Friend-or-Foe Q-learning represent early attempts to handle such structures through Nash equilibria, correlated equilibria, and friend-or-foe relationships, respectively. Their requirements on the state-action space and game model make direct use in high-dimensional continuous control difficult. Table 4 summarizes representative methods that address mixed interactions and the limitations of these methods.

In MADDPG, each agent acts through its own actor. Its critic is trained with observations and actions from other agents as well as its own, giving value estimation access to information about concurrent learning [11]. This is a continuous-action framework and is not restricted to mixed tasks; cooperative and competitive problems can also use it. The cost lies in the critic's input size, which increases with the number of agents. MATD3 addresses a different weakness of centralized value learning, overestimation bias, by using twin critics together with target policy smoothing and delayed policy updates [22].

**Table 4.** Representative mixed cooperative-competitive MARL algorithms

| Algorithm | Main contributions                                                                                 | Main limitations                                           | Ref. |
|-----------|----------------------------------------------------------------------------------------------------|------------------------------------------------------------|------|
| MADDPG    | Combines centralized critics with local actors for continuous-action tasks with mixed interactions | Centralized critic inputs expand with the agent population | [11] |
| MATD3     | Uses twin critics and associated update mechanisms to reduce value overestimation                  | Additional networks increase computational requirements    | [22] |
| MAAC      | Selects decision-relevant agent information through attention                                      | Depends on reliably identifying relevant interactions      | [23] |

MAAC addresses the selection of information supplied to an agent's critic [23]. Its attention mechanism selects information from interaction partners relevant to the current decision, instead of treating every other agent's information alike. This distinction matters when only some interactions bear on the learner's action. For mixed tasks, selective interaction modeling and centralized value estimation address related but different requirements: identifying which agents matter and estimating returns while their policies change. Stable exploration must also be maintained under these changing conditions.

## 4. Advances in MARL Applications to Real-World Tasks

Pathfinding robots, autonomous vehicles, and network resource managers do not face the same constraints on learning. Their rewards may be shared or conflicting, and the observations and actions available to an individual agent differ. The application studies below are examined in terms of these task conditions. The discussion extends beyond benchmark games to multi-robot and unmanned systems, transportation, and resource management, while keeping method selection tied to the information and operational constraints of each setting.

### 4.1 MARL Applications to Cooperative Tasks

A cooperative policy is judged by what the team accomplishes, although each agent may observe only part of the task and have restricted communication. In the applications reviewed here, this separation between a common objective and limited local information occurs in path planning and navigation, coverage search and perception, task allocation and multi-task coordination, and formation or swarm control. The decision changes across these tasks, from a route or assignment to a continuous control action; the need for coordinated behavior remains.

For centralized planners that rely on global information and static environmental assumptions, online replanning becomes difficult as the team grows. Path conflicts and repeated searches must then be resolved while plans are updated. PRIMAL learns decentralized pathfinding from reinforcement and imitation signals under partial observability [24]. For unmanned aerial vehicle (UAV) search in an unknown environment, Advantage QPLEX combines advantage decomposition with target probability maps [25]. A distributed reinforcement learning formulation is also used to reduce overlap between drone fields of view [26]. When the target itself moves, action masking and MAPPO address action selection and training instability [27]. These methods concern different aspects of cooperative search, from local path decisions to coordination over likely target locations and the selection of search actions.

Task allocation shifts the decision from selecting a route to deciding which task to accept and how to use the available resources. New task arrivals and unequal vehicle or robot capabilities make the assignment change over time. Under the shared-reward formulation, distributed learners acquire task acceptance and resource-use policies [28]. Yin et al.'s multi-UAV scheduling method combines QMIX with transfer reinforcement learning: value decomposition addresses the growing joint action

space, and knowledge transfer is used to improve training efficiency [29]. For maritime rescue, PG-MAPPO applies policy optimization to dynamically allocating tasks among UAVs [30].

Formation control introduces continuous dynamics into the cooperative decision problem. Maintaining formation is insufficient on its own: agents must also avoid obstacles and operate with the communication available to them. Local adjacency relationships provide the interaction representation in graph reinforcement learning for formation control under restricted communication [31]. Safe MARL represents safety requirements as constraint costs in a Markov game, alongside the team-return objective [32]. The first mechanism deals with the information available for coordination; the second makes individual robot safety part of the learning formulation. They illustrate why continuous cooperative control must be considered together with communication and safety constraints.

#### **4.2 MARL Applications to Competitive Tasks**

Competitive applications involve explicit conflicts over outcomes. Pursuit-evasion, interception and avoidance, unmanned-system offensive and defensive operations, game playing, and cybersecurity attack and defense all fall within this scope. Performance against the current opponent is only part of the problem. A policy must also contend with opponents that adapt during training and with behaviors it has not previously encountered, making opponent uncertainty a central application requirement.

Pursuit-evasion connects adversarial learning to physical decision-making. Differential games, optimal control, and rule-based strategies provide traditional formulations, but adaptation becomes difficult when continuous actions are high-dimensional, observations are partial, and several agents coordinate against an opponent [33]. Wan et al. use deep MARL to learn encirclement, interception, and evasion through dynamic interaction [34]. Peng et al. study coordinated multi-UAV pursuit using local observations under limited sensing [35]. Self-play, curriculum learning, and MAPPO have also been combined so that pursuers and evaders improve their policies through continued competition [36].

Games offer a complementary setting in which rules and experimental conditions can be reproduced. AlphaGo Zero combines self-play from scratch with search in a perfect-information game [37]. OpenAI Five uses large-scale self-play and distributed training to learn team strategies in Dota 2 [38], whereas AlphaStar uses policy populations and league training in one-versus-one StarCraft II [39]. Board games, poker, and real-time strategy games thus support the study of competitive learning under different information structures. The results demonstrate the strength of self-play and population-based training, but do not remove questions about training costs or generalization across opponents [40].

#### **4.3 MARL Applications to Mixed Cooperative-Competitive Tasks**

Mixed applications have to represent cooperation and conflicting payoffs within the same task. A UAV team can coordinate internally while confronting another team. Vehicles share a requirement for safe traffic interaction while competing for road access, and network participants seek resources that other participants also need. Adversarial swarms, autonomous driving, and communication or computational resource allocation are considered here on that basis. For these applications, both the individual return and its relationship to team or system performance matter.

For an adversarial UAV team, information sharing and task division concern cooperation within the team; penetration, interception, and countermeasures concern conflict with the opposing side. The reviewed methods make changes to how experience is used in learning these interactions. DP-MADDPG employs decomposition and prioritized experience replay for partially observable air combat, aiming to improve experience utilization and decision-making [41]. CEL-MADDPG introduces curriculum-based experience when several UAVs learn to encircle a maneuvering target, with training stability as a principal concern [42]. The common issue is learning coordinated behavior while the task also depends on an adversary's response.

In traffic, the cost of an action cannot be assessed only through the return to the vehicle taking it. Following, yielding, and merging involve interactions needed for safety, yet access to road space and right of way can conflict with an individual vehicle's travel-time objective [43]. Safe MARL treats traffic efficiency and collision risk within constrained optimization and game-theoretic formulations [44]. A policy evaluation should accordingly report collision rates, rule and constraint compliance, and decision time in addition to average return. These measures describe operational requirements that an improvement in return alone does not establish.

The shared quantities in network and edge-computing tasks are spectrum, transmit power, and computational capacity. Their allocation links the demand of an individual user, vehicle, base station, or server to the efficiency of the wider system [45]. The cited studies adapt MARL in different ways. Federated multi-agent deep reinforcement learning coordinates resources in vehicular communication with less centralized data collection [46]. In UAV edge computing, MATD3-based offloading addresses energy efficiency and overestimation in value learning [47]. Distributed MARL is also applied to computation offloading and resource allocation in large heterogeneous edge systems [48]. The choice among these learning arrangements depends on the resource constraints and payoff relationships represented in the task.

#### 4.4 Comparative Analysis of Task Types

**Table 5.** Typical applications and key challenges across task types

| Task type                     | Typical applications                                | Key challenges                                                                | Common methods                                          |
|-------------------------------|-----------------------------------------------------|-------------------------------------------------------------------------------|---------------------------------------------------------|
| Fully cooperative             | Path planning and coverage search                   | Path conflicts; redundant search; sparse reward signals                       | Value decomposition, reward shaping, MAPPO              |
| Fully cooperative             | Task allocation                                     | Dynamic task demand; resource limits; heterogeneous capabilities              | QMIX, transfer learning, PG-MAPPO                       |
| Fully cooperative             | Formation and swarm control                         | Continuous dynamics; restricted communication; collision risk                 | Actor-critic, graph networks, safe MARL                 |
| Fully competitive             | Pursuit-evasion; unmanned-system attack and defense | Partial observations; changing opponents; generalization across policies      | Self-play, curriculum learning, PSRO                    |
| Fully competitive             | Game playing and cybersecurity                      | Long-horizon decisions; hidden information; changing attack sequences         | Policy populations, Markov games, hierarchical policies |
| Mixed cooperative-competitive | Swarm confrontation and autonomous driving          | Coordinating teammates, responding to rival teams, meeting safety constraints | MADDPG, MATD3, safe MARL                                |
| Mixed cooperative-competitive | Communication and computational resource allocation | Meeting latency, energy, fairness, and throughput requirements                | Federated MARL, distributed actor-critic                |

Task performance is expressed differently across the three categories. For cooperative applications, completion rate, coverage, and aggregate team return describe the shared outcome. Local observations and limited communication constrain how that outcome can be achieved, even where training has access to global information. In competitive applications, the partly observed quantities include an opponent's intentions and policy. Win rates and offensive or defensive payoffs then need to be read together with performance against different opponents.

Mixed tasks require useful information from teammates and indications of opposing behavior. They also require a way of disregarding interactions that do not affect the current decision. Individual returns are evaluated alongside system efficiency and overall stability. Table 5 collects the principal application settings, challenges, and commonly used methods. It relates evaluation to the task objective without assuming that one set of performance measures is sufficient for all three task categories.

An algorithm's placement in one of these categories should not be read as an exclusive assignment. MADDPG, for example, was discussed in Section 3.3 because it models mixed interactions, although it also supports cooperative and competitive tasks. The classification instead supplies a sequence for examining a method: specify the objective and rewards, identify the interactions they entail, and assess whether the value representation and training framework address those interactions. Comparisons also require sufficiently similar task conditions and experimental settings. Applicability demonstrated in one setting is evidence for that setting, not for all tasks sharing the same broad label.

## **5. Outlook and Conclusion**

### **5.1 Future Challenges and Directions**

The difficulty of deployment is evident at two levels. During learning, each agent's updates alter the behavior encountered by the others, so value estimation and convergence must be considered under changing policies. At the system level, adding agents enlarges the joint state-action space and increases demands on samples, computation, and communication. Tighter task constraints add to these demands. A method that can be trained in a controlled setting still has to be assessed against the requirements of the intended application.

Assessment itself remains a limitation. Differences in environments, implementations, hyperparameters, and evaluation protocols make results from separate studies difficult to compare. More consistent experimental conditions and evaluation standards are needed to distinguish a change in algorithmic performance from a change in the task or training setup.

For heterogeneous agents and more constrained tasks, the dynamics model and prior task knowledge deserve a direct role in the learning formulation. Control theory, game theory, constrained optimization, and model predictive methods offer possible means of introducing that information and expressing safety requirements. Their use should be assessed in terms of stability, safety, and interpretability as well as learning performance. Improved simulation and experiments on physical systems are needed for that assessment. The resulting evidence would concern a complete combination of task model, learning method, and deployment conditions, rather than an algorithm considered in isolation.

### **5.2 Conclusion**

The review has used reward relationships to distinguish cooperative, competitive, and mixed cooperative-competitive tasks. This makes it possible to read algorithmic developments against the interactions they are intended to learn. Under shared rewards, the central issues are credit assignment and decentralized coordination; with opposing payoffs, opponent responses and policy generalization become central. Mixed settings require attention to both payoff relationships and the relevance of different interaction partners. Value decomposition, policy optimization, opponent-based training, and centralized critics address different parts of these problems.

In the applications considered, those distinctions are qualified by information access and operational constraints. Path planning and task allocation emphasize coordinated use of limited information and resources; pursuit-evasion and adversarial engagement require policies that account for opponents; autonomous driving and resource allocation also bring individual objectives into contact with system-level requirements. The framework linking task characteristics, interaction types, and algorithmic mechanisms is used to delimit a method's applicability and the conditions for comparison. Its purpose

is to relate a learning approach to a specified problem, without implying that it will remain appropriate when the task conditions change.

## References

- [1] Wong, A., Bäck, T., Kononova, A. V., et al. (2023). Deep multiagent reinforcement learning: challenges and directions. *Artificial Intelligence Review*, 56(6).
- [2] Hernandez-Leal, P., Kartal, B., & Taylor, M. E. (2019). A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6), 750–797.
- [3] Oroojlooy, A., & Hajinezhad, D. (2023). A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53(11), 13677–13722.
- [4] Zhang, K., Yang, Z., & Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. In *Handbook of reinforcement learning and control* (pp. 321–384).
- [5] Li, T., Zhu, K., Luong, N. C., et al. (2022). Applications of multi-agent reinforcement learning in future internet: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 24(2), 1240–1279.
- [6] Orr, J., & Dutta, A. (2023). Multi-agent deep reinforcement learning for multi-robot applications: A survey. *Sensors*, 23(7), 3625.
- [7] Gronauer, S., & Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2), 895–943.
- [8] Ning, Z., & Xie, L. (2024). A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence*, 3(2), 73–91.
- [9] Buşoniu, L., Babuška, R., & De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1* (pp. 183–221).
- [10] Yu, C., Velu, A., Vinitisky, E., et al. (2022). The surprising effectiveness of PPO in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35, 24611–24624.
- [11] Lowe, R., Wu, Y., Tamar, A., et al. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 30.
- [12] Amato, C. (2024). An introduction to centralized training for decentralized execution in cooperative multi-agent reinforcement learning. *arXiv*. [<https://arxiv.org/abs/2409.03052>](<https://arxiv.org/abs/2409.03052>)
- [13] Sunehag, P., Lever, G., Gruslys, A., et al. (2017). Value-decomposition networks for cooperative multi-agent learning. *arXiv*. [<https://arxiv.org/abs/1706.05296>](<https://arxiv.org/abs/1706.05296>)
- [14] Rashid, T., Samvelyan, M., De Witt, C. S., et al. (2020). Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178), 1–51.
- [15] Son, K., Kim, D., Kang, W. J., et al. (2019). QTRAN: Learning to factorize with transformation. In *International Conference on Machine Learning* (pp. 5887–5896). PMLR.
- [16] Wang, J., Ren, Z., Liu, T., et al. (2020). QPLEX: Duplex dueling multi-agent Q-learning. *arXiv*. [<https://arxiv.org/abs/2008.01062>](<https://arxiv.org/abs/2008.01062>)
- [17] Kuba, J. G., Chen, R., Wen, M., et al. (2021). Trust region policy optimisation in multi-agent reinforcement learning. *arXiv*. [<https://arxiv.org/abs/2109.11251>](<https://arxiv.org/abs/2109.11251>)
- [18] Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994* (pp. 157–163). Morgan Kaufmann.
- [19] Heinrich, J., Lanctot, M., & Silver, D. (2015). Fictitious self-play in extensive-form games. In *International Conference on Machine Learning* (pp. 805–813). PMLR.
- [20] Heinrich, J., & Silver, D. (2016). Deep reinforcement learning from self-play in imperfect-information games. *arXiv*. [<https://arxiv.org/abs/1603.01121>](<https://arxiv.org/abs/1603.01121>)
- [21] Lanctot, M., Zambaldi, V., Gruslys, A., et al. (2017). A unified game-theoretic approach to multiagent reinforcement learning. *Advances in Neural Information Processing Systems*, 30.
- [22] Ackermann, J., Gabler, V., Osa, T., et al. (2019). Reducing overestimation bias in multi-agent domains using double centralized critics. *arXiv*. [<https://arxiv.org/abs/1910.01465>](<https://arxiv.org/abs/1910.01465>)

- [23] Iqbal, S., & Sha, F. (2019). Actor-attention-critic for multi-agent reinforcement learning. In International Conference on Machine Learning (pp. 2961–2970). PMLR.
- [24] Sartoretti, G., Kerr, J., Shi, Y., et al. (2019). PRIMAL: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters*, 4(3), 2378–2385.
- [25] Kong, X., Yang, J., Chai, X., et al. (2025). An advantage duPLEX dueling multi-agent Q-learning algorithm for multi-UAV cooperative target search in unknown environments. *Simulation Modelling Practice and Theory*, 142, 103118.
- [26] Pham, H. X., La, H. M., Feil-Seifer, D., et al. (2018). Cooperative and distributed reinforcement learning of drones for field coverage. *arXiv*. [<https://arxiv.org/abs/1803.07250>](<https://arxiv.org/abs/1803.07250>)
- [27] Liu, Y., Li, X., Wang, J., et al. (2024). Reinforcement-learning-based multi-UAV cooperative search for moving targets in 3D scenarios. *Drones*, 8(8), 378.
- [28] Qin, W., Sun, Y. N., Zhuang, Z. L., et al. (2021). Multi-agent reinforcement learning-based dynamic task assignment for vehicles in urban transportation system. *International Journal of Production Economics*, 240, 108251.
- [29] Yin, Y., Guo, Y., Su, Q., et al. (2022). Task allocation of multiple unmanned aerial vehicles based on deep transfer reinforcement learning. *Drones*, 6(8), 215.
- [30] Wu, X., Yan, Q., Wang, J., et al. (2025). Dynamic task allocation for UAV swarms in maritime rescue scenarios based on PG-MAPPO. *IEEE Internet of Things Journal*.
- [31] Wang, H., Qiu, T., Liu, Z., et al. (2020). Multi-agent formation control with obstacle avoidance under restricted communication through graph reinforcement learning. *IFAC-PapersOnLine*, 53(2), 8150–8156.
- [32] Gu, S., Kuba, J. G., Chen, Y., et al. (2023). Safe multi-agent reinforcement learning for multi-robot control. *Artificial Intelligence*, 319, 103905.
- [33] Kun, Y., Ao, S., Xu, N., et al. (2025). A review of reinforcement learning approaches for pursuit-evasion games. *Chinese Journal of Aeronautics*, 103940.
- [34] Wan, K., Wu, D., Zhai, Y., et al. (2021). An improved approach towards multi-agent pursuit-evasion game decision-making using deep reinforcement learning. *Entropy*, 23(11), 1433.
- [35] Peng, Z., Wu, G., Luo, B., et al. (2025). Multi-UAV cooperative pursuit strategy with limited visual field in urban airspace: A multi-agent reinforcement learning approach. *IEEE/CAA Journal of Automatica Sinica*.
- [36] Lin, Y., Gao, H., & Xia, Y. (2025). Distributed pursuit-evasion game decision-making based on multi-agent deep reinforcement learning. *Electronics*, 14(11), 2141.
- [37] Silver, D., Schrittwieser, J., Simonyan, K., et al. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676), 354–359.
- [38] Berner, C., Brockman, G., Chan, B., et al. (2019). Dota 2 with large scale deep reinforcement learning. *arXiv*. [<https://arxiv.org/abs/1912.06680>](<https://arxiv.org/abs/1912.06680>)
- [39] Vinyals, O., Babuschkin, I., Czarnecki, W. M., et al. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782), 350–354.
- [40] Bighashdel, A., Wang, Y., McAleer, S., et al. (2024). Policy space response oracles: A survey. *arXiv*. [<https://arxiv.org/abs/2403.02227>](<https://arxiv.org/abs/2403.02227>)
- [41] Yang, J., Yang, X., & Yu, T. (2024). Multi-unmanned aerial vehicle confrontation in intelligent air combat: A multi-agent deep reinforcement learning approach. *Drones*, 8(8), 382.
- [42] Li, B., Wang, J., Song, C., et al. (2024). Multi-UAV roundup strategy method based on deep reinforcement learning CEL-MADDPG algorithm. *Expert Systems with Applications*, 245, 123018.
- [43] Dinneweth, J., Boubezoul, A., Mandiau, R., et al. (2022). Multi-agent reinforcement learning for autonomous vehicles: A survey. *Autonomous Intelligent Systems*, 2(1), 27.
- [44] Zheng, Z., & Gu, S. (2025). Safe multiagent reinforcement learning with bilevel optimization in autonomous driving. *IEEE Transactions on Artificial Intelligence*, 6(4), 829–842. [<https://doi.org/10.1109/TAI.2024.3497919>](<https://doi.org/10.1109/TAI.2024.3497919>)
- [45] Hady, M. A., Hu, S., Pratama, M., et al. (2025). Multi-agent reinforcement learning for resource allocation optimization: A survey. *Artificial Intelligence Review*, 58(11), 354.

- [46]Liu, Q., & Ma, Y. (2025). Communication resource allocation method in vehicular networks based on federated multi-agent deep reinforcement learning. *Scientific Reports*, 15(1), 30866.
- [47]Xu, S., Liu, Q., Gong, C., et al. (2025). Energy-efficient multi-agent deep reinforcement learning task offloading and resource allocation for UAV edge computing. *Sensors*, 25(11), 3403.
- [48]Gao, Z., Yang, L., & Dai, Y. (2023). Large-scale computation offloading using a multi-agent reinforcement learning in heterogeneous multi-access edge computing. *IEEE Transactions on Mobile Computing*, 22(6), 3425–3443.  
[<https://doi.org/10.1109/TMC.2022.3141080>](<https://doi.org/10.1109/TMC.2022.3141080>)