

Distributed Reinforcement Learning for Robot Teams: A Review

Yutong Wang¹, Mehul Damani¹, Pamela Wang¹, Yuhong Cao¹ and Guillaume Sartoretti^{1*}

¹*Department of Mechanical Engineering, National University of Singapore,
9 Engineering Dr 1, 117575, Singapore.

*Corresponding author(s). E-mail(s): guillaume.sartoretti@nus.edu.sg;
Contributing authors: e0576114@u.nus.edu; damanimehul24@gmail.com;
wangyeelin@gmail.com; caoyuhong@u.nus.edu;

Abstract

Purpose of review: Recent advances in sensing, actuation, and computation have opened the door to multi-robot systems consisting of hundreds/thousands of robots, with promising applications to automated manufacturing, disaster relief, harvesting, last-mile delivery, port/airport operations, or search and rescue. The community has leveraged model-free multi-agent reinforcement learning (MARL) to devise efficient, scalable controllers for multi-robot systems (MRS). This review aims to provide an analysis of the state-of-the-art in distributed MARL for multi-robot cooperation.

Recent findings: Decentralized MRS face fundamental challenges, such as non-stationarity and partial observability. Building upon the “centralized training, decentralized execution” paradigm, recent MARL approaches include independent learning, centralized critic, value decomposition, and communication learning approaches. Cooperative behaviors are demonstrated through AI benchmarks and fundamental real-world robotic capabilities such as multi-robot motion/path planning.

Summary: This survey reports the challenges surrounding decentralized model-free MARL for multi-robot cooperation and existing classes of approaches. We present benchmarks and robotic applications along with a discussion on current open avenues for research.

Keywords: Multi-Robot Systems; Reinforcement Learning; Cooperation; Communication Learning; Mixed cooperative-competitive Settings; Motion Planning

1 Introduction

With the recent advances in sensing, actuation, and computation, we are quickly approaching a point at which real-life applications will involve the deployment of hundreds, if not thousands, of robots, for tasks such as automated manufacturing and deliveries [1], environmental monitoring and exploration [2], disaster relief/search and rescue [3]. However, as the number of agents (i.e.,

robots) in the system grows, so does the combinatorial complexity of coordinating them. Existing conventional methods such as handcrafted controllers struggle at keeping up with this increased need for performance, scalability, and flexibility, since they usually rely on the intuition/experience of their designer and on domain-specific knowledge. Building upon the success of (single-agent) reinforcement learning (RL), the community has started to look to machine learning methods –

and in particular multi-agent reinforcement learning (MARL) – to devise controllers for multi-robot systems, with a particular focus on cooperation among robots and scalability to larger teams.

In this survey, we first introduce RL and MARL, as well as the different reward and training setups. We discuss the challenges associated with MARL and MRS (Section 2), namely non-stationarity during training, partial observability, scalability, as well as challenges associated with communications (and, in particular, communication learning) among agents. We then detail the main research directions that have been proposed recently to tackle these challenges and improve agent cooperation (Section 3). In particular, recent advances have often adopted the “Centralized Training, Decentralized Execution” (CTDE) paradigm, where agents rely on global information during training to boost cooperation, while still learning policies that only rely on local information and interactions among agents at execution time, thus improving scalability. We report relevant approaches to decentralized multi-agent cooperation, namely independent learning, centralized critic, value decomposition, and communication learning methods. We then discuss how the fundamental agent capabilities developed by these works can be assessed and visualized, both through AI benchmarks and real-world robotic tasks (Section 4). Most experimental demonstrations of MARL cooperation methods on robot consider motion/path planning problems, both as a fundamental multi-robot problem and because many robot teams consider mobile robots, whose (motion) coordination is of primary concern. Finally, we take on a more critical view to describe open challenges in MARL, identifying and motivating future research in these areas (Section 5), and conclude this review in Section 6.

We note that our review is by no means exhaustive. For further information about the recent work in distributed MARL, we refer the reader to a set of excellent resources in this area [4–8]. Furthermore, and more generally for multi-agent systems (MAS), Jorge and Magnus discussed a number of decentralized control and coordination strategies [9]. Elio et al. [10] and Zhi et al. [11] reviewed recent advancements in multi-robot cooperative object transport and collaborative manipulation, respectively.

2 Background

2.1 Single-Agent Reinforcement Learning

Reinforcement Learning (RL) addresses the fundamental problem of sequencing intelligent decisions in an environment with discrete time-steps to maximize long-term cumulative return, and is often formalized as a Markov Decision Process (MDP). An MDP is defined by a tuple $(S, A, R, T, O, Z, \rho, \gamma)$, where S is the state space, A the action space, $R : S \times A \times S \rightarrow \mathbb{R}$ the reward function, defining the agent’s reward at each time-step. $T : S \times A \times S \rightarrow [0, 1]$ is the state transition function, defining the probability of reaching state s' by taking action a in state s . $O : S \times A \rightarrow Z$ is the observation function, from which the agent’s observation (sometimes referred to as its *state*) $z \in Z$ is sampled at each time-step. $\rho(s_0)$ is the distribution of initial states, and $\gamma \in [0, 1]$ is the *discount factor*, setting the trade-off between immediate and future rewards.

The goal of an agent is to learn an optimal decision strategy, more commonly referred to as a *policy*, $\pi^* : S \rightarrow A$, which produces a distribution over possible actions in each state, such that the long-term, cumulative discounted return is maximized:

$$\pi^* = P(a | s) = \arg \max_{\pi \in \Pi} \mathbb{E}_{s \sim \rho(s_0)} [V_\pi(s)], \quad (1)$$

where Π is the set of all policies and $V_\pi(s)$ is the *state value* function, i.e., the expected return when starting in state s and following policy π :

$$V_\pi(s) = \mathbb{E}_{\pi, T} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 = s \right]. \quad (2)$$

Similar to $V_\pi(s)$, $Q^\pi(s, a)$ is defined as the *state-action value* function, i.e., the expected return when starting in state s , taking action a and thereafter following policy π :

$$Q_\pi(s, a) = \mathbb{E}_{\pi, T} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid \begin{array}{l} s_0 = s \\ a_0 = a \end{array} \right]. \quad (3)$$

There are two broad classes of methods to learn an optimal policy: value-based and policy-based methods.

Value-based methods aim to estimate the optimal policy’s corresponding state value function

$V^*(\mathbf{s})$ or state-action value function $Q^*(\mathbf{s}, \mathbf{a})$. Once such a function is obtained, the optimal policy is obtained by acting greedily with respect to it. These functions can also be computed through tabular methods using algorithms such as value iteration and policy iteration, if the transition model is known. However, a transition model is generally not available in most realistic environments, and algorithms have to rely on sampling to approximate these functions. This can either be done by relying on full rollouts through the environment (Monte-Carlo methods), or via bootstrapping. DQN is a very popular sampling-based algorithm that relies on bootstrapping to learn the optimal action-value function, which is parameterized using a deep neural network [12].

In contrast to value-based methods, policy-based methods directly try to learn the optimal policy π^* . Typical policy-based setups involve a parameterized policy π_θ , whose parameters are optimized to maximize the objective in Eq. (1) using either gradient-based or gradient-free approaches. The recent advent of deep learning has made it possible to represent powerful non-linear policies through a deep neural network and optimize the network weights through gradient-based methods. REINFORCE is one of the most fundamental and popular policy gradient algorithms which uses actual returns from a trajectory to compute such weights gradients [13]. However, since it relies on noisy returns for optimization, REINFORCE generally suffers from high variance, drastically slowing down its convergence.

Actor-critic methods are a special class of policy-based methods which learn both an explicit policy representation and a state-value function for that policy. Instead of direct (often noisy) returns from the environment like in REINFORCE, the learning of the policy (actor) is guided by the value function (critic). In practice, this significantly reduces variance and stabilizes training, but at the cost of introducing some bias, which is generally an acceptable trade-off. Actor critic methods have grown massively in popularity since their introduction and include most state-of-the-art RL algorithms, such as A3C [14], PPO [15], SAC [16], and DDPG [17].

This section is meant to provide a basic background in RL and is in no way comprehensive. For

more detailed information, we refer the reader to a set of excellent resources in this area [18–21].

2.2 Multi Agent Reinforcement Learning

MARL generalizes the single-agent MDP to a Markov game, which can be described by a tuple $(N, S, A, R, T, O, Z, \rho, \gamma)$ where N is the number of agents, S the state space, A the joint action space, defined as $\mathbf{A} = A_1 \times A_2 \times \dots \times A_N$. The reward function $R : S \times \mathbf{A} \times S \rightarrow \mathbb{R}^n$ computes N rewards $[r_1, r_2, \dots, r_N]$ at each time-step, one for each agent. $T : S \times \mathbf{A} \times S \rightarrow [0, 1]$ is the transition function, which defines the probability of reaching state s' after taking joint action $\mathbf{a} = [a_1, a_2, \dots, a_n]$ in state s . $\mathbf{O} = [O_1, O_2, \dots, O_N]$ are the set of observation functions, from which each agent's observation $O_i(s) : S \times \mathbf{A} \rightarrow Z_i$ is sampled at each time-step, where $\mathbf{Z} = [Z_1, Z_2, \dots, Z_N]$ are the agents' observation spaces. Finally, $\rho(s_0)$ and γ are the initial state distribution and discount factor respectively.

The above formulation assumes that the multi-agent system is *heterogeneous*, i.e., each agent can have a distinct action space and observation function. On the other hand, *homogeneous* systems assume that agents have the same action space and observation function, i.e., $A_1 = A_2 = \dots = A_N$, $O_1 = O_2 = \dots = O_N$ and $Z_1 = Z_2 = \dots = Z_N$. However, agents in a homogeneous system can still have differing policies that allow them to perform specialized roles. When specialization is not a requirement, homogeneous systems can be extended by sharing the same policy between agents i.e., having homogeneous policies, which allows for efficient and more scalable learning by relying on parameter sharing [22].

MARL setups can be broadly classified by their reward functions, which can induce cooperative or competitive behaviors, as well as on the basis of their learning setups, which can drastically influence the type and quality of policies learned. These classifications are briefly described in the following sections.

2.3 Reward Setup

Fully Cooperative: In fully cooperative settings, all agents share the same reward. That is, $r_1 = r_2 = \dots = r_N$. These settings face the challenge of credit assignment, as individual agents are not

rewarded only on the basis of the action they took, but instead based on the joint action of the entire team. This leads to a high variance in rewards, which is particularly significant during early stages of training, where many exploratory actions are being taken. Alleviating this variance requires the use of methods capable of either implicit or explicit *credit assignment*. Common environments in this setting include the suite of StarCraft Micromanagement environments [23] and many Multi-Agent Particle environments [24].

Competitive: In competitive settings, agents receive and try to selfishly maximize their own local rewards, which are often conflicting in nature to other agents. That is, agents have conflicting goals and a high reward for some agents usually comes at the cost of low rewards for the other ones. Zero-sum games are a characteristic example of two player fully competitive environments, where the sum of all agent rewards is equal to 0. Examples of two-player zero-sum game environments include Chess/Shogi/Go [25] and Pong [26, 27]. Larger competitive benchmarks for MARL include Pommerman [28] and Neural-MMO [29]. It is also possible to have a hybrid setting, which consists of cooperative multi-agent teams in competition with other teams. That is, the reward setup is fully cooperative at the team level but competitive at the game level. An example of such an environment is the prey-predator game, where a team of pursuer agents is tasked with cooperatively capturing/caging evader agents [30, 31].

Mixed Cooperative-Competitive: In mixed environments, agents receive their own local (often shaped) rewards. These rewards, although given with the ultimate objective of maximizing a global metric or the team reward (sum of all local rewards), might induce selfish/undesirable behaviors in agents as they don't always directly align with the overall objective. Mixed environments lie at a unique intersection where they face a trade-off between efficient learning and optimality, i.e., shaped rewards accelerate learning but induce selfish behaviors which affect optimality. Examples of such settings are multi-robot path planning (Section 4.2) or multi-agent traffic signal control (MATSC) [32], where agents are locally rewarded for reaching a target location and minimizing the local traffic queue length respectively. However, the performance of these systems as a whole is judged on

global metrics - the throughput (agents reaching their goals per time-step) and average travel time of vehicles respectively.

This work focuses on robot teams in both fully cooperative and mixed cooperative-competitive settings.

2.4 Learning Setup

2.4.1 Centralized Learning

Centralized training models the multi-agent system as a single-agent and aims to learn a joint action from the joint observations of all agents. Centralized methods operate on the assumption that information can be centralized, both during training and execution, something which is often not achievable in practice. Additionally, centralized learners encounter a combinatorially growing action and observation space as the number of agents in the system grow. This makes optimization hard and introduces issues such as the lazy agent problem [33]. In conclusion, although they are convenient to model and enjoy access to all available environment information through centralization, centralized learners are both severely limited in scalability and difficult to optimize.

2.4.2 Independent Learning

In independent learning, each agent independently learns its own policy based on its own observation while considering other agents as a part of the environment. As a result, independent learning can be decentralized in both training and execution, allowing it to scale considerably more than centralized methods. However, by considering other agents as a part of the environment, independent learners forgo the Markov property and learn in an environment that is inherently non-stationary due to the changing policies of other agents. Additionally, in contrast to centralized learning, independent learners have limited observability and are unable to enjoy any gains which come from information centralization.

2.4.3 Centralized Training Decentralized Execution (CTDE)

The CTDE paradigm aims to get the best of both centralized and independent learning paradigms, by allowing agents to learn in a centralized setting

while executing policies in a decentralized setting. This allows agents to benefit from additional information available during training from centralization, as well as enjoy the scalability benefits of decentralized execution. Additional information can allow the learning of centralized critics [34, 35] and value factorization networks [33, 36], as well as allow techniques such as parameter sharing (sharing network parameters between different agents) to accelerate learning [22]. These works are discussed in more detail in Section 4. The CTDE regime has gained remarkable popularity in MARL, and is particularly desirable in settings where a simulator is available for learning, as is often the case for robotic tasks.

This work primarily focuses on the CTDE paradigm but also discusses independent learning in some detail.

2.5 Challenges of MARL

2.5.1 Non-Stationarity

From the perspective of a training agent, which considers other agents to be a part of the environment, a multi-agent environment is inherently *non-stationary* due to the changing policies of those other agents who are also learning at the same time. In other words, the transition function T is non-stationary from the perspective of a single learning agent that cannot observe the policies of other agents. This poses a complex learning problem, where the evolving policies of different agents interact with each other in an unpredictable manner and often lead to undesirable outcomes such as oscillating policies (Fig. 1 (a)). Convergence guarantees, which are well-defined in stationary environments (most single-agent environments), are not applicable in non-stationary environments as the Markov property does not hold there. Additionally, learning in a non-stationary environment implies that data collected in the past was governed by a different environment dynamics (i.e., “off-environment”) function based on the past policies of other agents. This affects the performance of all algorithms that maintain an experience buffer to learn from past data.

2.5.2 Scalability

Scalability is a multi-faceted challenge, which both encompasses training policies that can scale, i.e.,

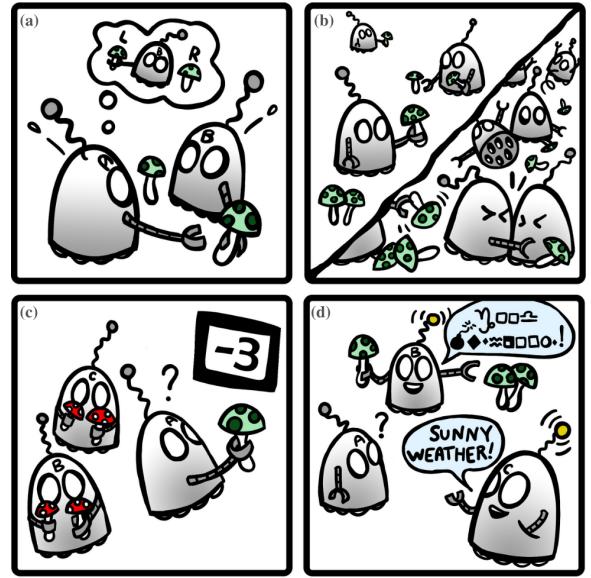


Fig. 1: Key challenges in MARL: (a) *Non-Stationarity* - Agents learn in an ever-changing environment, where other agents also constantly update their behavior. As a result, predictions based on past experiences may not be accurate anymore. Here, agent A (wrongly) predicts that agent B will go for the left mushroom. (b) *Scalability* - Agents trained in a smaller team may struggle at generalizing their strategy to a larger one. (c) *Partial Observability* - Agents may be confused by shared rewards, which often depend on the action of agents beyond their sensing range. (d) *Communication* - Agents need to identify and encode relevant information in a commonly agreed upon manner, and must often learn to select whom to speak/listen to, to avoid being overwhelmed. Here, agent A cannot understand the useful message from agent B, deciding instead to listen to the useless, yet understandable message from agent C. Image by and courtesy of Anna Szücs.

generalize to smaller/larger teams, and algorithms that can handle the training of large teams, without a considerable effect on performance and (computational) cost (Fig. 1 (b)). Both kinds of scalability are crucial in the real world, where it is common for robots to be deployed on tasks of varying scales and expensive for robots to be retrained due to the changes in the team size (such as agent addition or breakdowns). However, most algorithms face a trade-off between non-stationarity and scalability. Non-stationarity

can be alleviated by centralization, but centralization faces scalability challenges such as an exponentially growing state and action spaces (in particular, most algorithms that rely on communications see the number of messages transmitted scale exponentially) and the high cost of centralizing information. Similarly, independent learning can scale up, but faces growing non-stationarity as more learning agents are added to the system. The CTDE paradigm (Section 2.4.3) balances these trade-offs reasonably well. However, training functions such as a centralized critic or a value decomposition network is also limited as an extremely large network can be difficult to optimize. Out of the three learning setups, CTDE-based algorithms have the most potential to further mature and increase scalability by incorporating techniques such as parameter sharing [22].

2.5.3 Partial Observability

In many environments, agents are required to learn a policy based on limited (often noisy) information about the environment, i.e., partially observable settings. This is even more prominent in real-world applications where cost and characteristics of robot sensors can considerably restrict the information available to a robot. For example, a forward facing camera cannot provide information about the other directions around a robot. Partial observability is an even more profound challenge in multi-agent environments as an agent might not be able to observe other agents in the system. This exacerbates non-stationarity and makes learning extremely hard, especially in fully cooperative settings where an agent might not be able to infer why it received a low reward, because the agent responsible for the low reward is not currently observable (Fig. 1 (c)). Partial observability can be addressed through network architectures that incorporate memory [37].

2.5.4 Communication

Key issues such as non-stationarity and partial observability can be mitigated by allowing agents to communicate information within the team, to augment each others' knowledge/observations [38–40]. However, employing pre-defined/handcrafted communication protocols generally restricts the benefits and flexibility of communication, by relying greatly on the intuition

of its designer and/or on domain knowledge. On the other hand, allowing agents to learn what to share and how to encode this information, as well as whom to communicate/listen to, comes with fundamental challenges (Fig. 1 (d)), which are being addressed by the *communication learning* community (see Section 3.4). First, allowing agents to identify and encode relevant information into an efficient message is the first challenge, and there is currently no conclusive evidence on what information is required for agents to learn an optimal policy. Second, the choice/learning of a high-quality communication topology, which can accurately capture the intricate relationships between agents while meeting real-world communication constraints (e.g., bandwidth or range), is also nontrivial. Finally, scalability remains one of the hardest challenges in communication, where larger team sizes can see agents overwhelmed with (potentially contradictory) messages, essentially introducing noise in the agents' observations and leading to poor performances (i.e., the *chatter* problem).

3 Principles of Cooperation

This section describes the four classes of approaches to mitigate the challenges discussed in Section 2 and encourage agent cooperation. These approaches are illustrated in Fig. 2, while Table 1 summarizes the main representative cooperation algorithms reviewed in this section.

3.1 Independent Learners

Independent learners learn their own policy based on their own observation using either shared (fully cooperative) or local (mixed cooperative-competitive) rewards while considering other agents as a part of the environment. While vanilla independent learning often functions well in practice [27], more recent independent learning works have focused on developing methods that address non-stationarity in MARL environments, which is particularly significant in fully cooperative settings where agents receive a joint team reward and an agent's action being individually-optimal is not enough to ensure team-level optimality. Lauer et al. introduced an optimistic Q-Learning algorithm, which only updates Q-values when there is a guaranteed improvement and ignores

negative updates by attributing low returns to sub-optimal actions of other agents [41]. Softer versions of this were introduced with the aim of ultimately converging to accurate action values [42, 43]. Matignon et al. [42] used a constant smaller learning rate for negative updates, while Panait et al. [43] used a decaying temperature to initially show leniency in negative updates and progressively decrease leniency as a state-action pair's visit count increases. More generally, there is a spectrum of independent Q-Learning methods ranging from totally optimistic [41], moderately optimistic [42, 43] and vanilla Q-Learning. Although originally introduced for tabular settings, these methods have been successfully extended to deep RL settings [44, 45]. While the methods described above focus on fully cooperative settings, using intrinsic rewards that encourage social influence over other agents has also proven effective in mixed cooperative-competitive settings [46]. In addition to off-policy methods, which are more popular in MARL due to their sample-efficiency, recent methods utilizing state-of-the-art on-policy algorithms such as PPO have also shown great promise [47, 48]

While optimistic Q-Learning methods can address non-stationarity to a certain extent, many off-policy deep RL algorithms introduce another form of non-stationarity through their use of an experience buffer. In order to stabilize training and efficiently reuse data, these algorithms maintain a buffer which stores past environment transitions. However, in the case of a multi-agent environment, the experience buffer consists of off-environment, possibly obsolete data that can introduce training instabilities. Given the central role played by experience replay, Foerster et al. introduced two methods to stabilize experience replay in MARL [49]. The first method introduced an importance sampling correction to decay updates from transitions where the joint policy had changed significantly. The second method, which was partially inspired from hyper Q-Learning [50], introduced a fingerprint as an input feature to disambiguate the age of the data and serve as an indicator for the quality of other agents' policies. With a similar objective, Omidshafiei et al. introduced Concurrent Experience Replay Trajectories to sample concurrent experience from the replay buffers of different

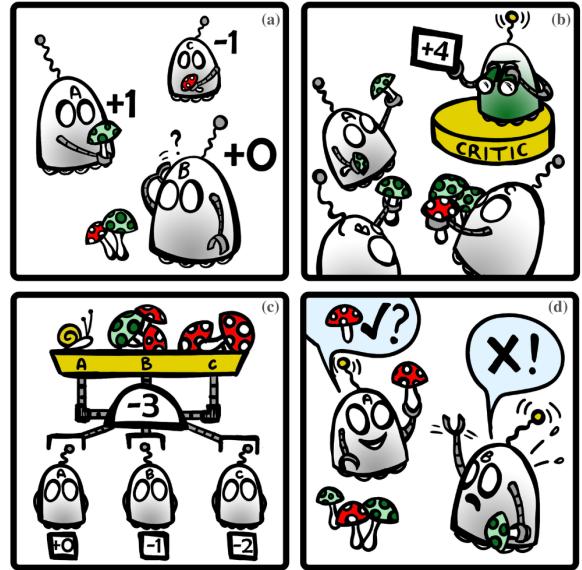


Fig. 2: Cooperation approaches in MARL. (a) *Independent Learning* - Agents learn policies (here, from individual rewards) by treating other agents as part of their common environment. (b) During training, a *Centralized Critic* can provide a more accurate cooperative baseline - the state value, i.e., the expected long-term return from the current state, by relying on augmented state and policy information from all agents. Learned policies remain decentralized. (c) *Factorized Value Functions* - Agents learn to explicitly address the *credit assignment* problem to transform a shared reward into individual contributions that can be used to update their individual policy, (d) *Communication Learning* - Agents learn to identify, encode, and share relevant information to augment each other's knowledge about the system. Image by and courtesy of Anna Szücs.

agents, thus introducing correlations in local policy updates [45].

3.2 Centralized Critic

The CTDE paradigm has made it possible to train agents in a centralized manner, allowing for the development of methods that reduce non-stationarity and achieve better credit assignment. Actor-critic algorithms are particularly well suited for the CTDE paradigm as the critic is required only during training and can thus be augmented with any desirable centralized information, such

as the policies of all agents or any extra available state information. This centralization of information during training can help alleviate the non-stationarity faced by independent learners. In fully cooperative settings with a single joint reward, it is often sufficient to have a single centralized critic for all the agents, while in competitive and mixed cooperative-competitive environments with local agent rewards, each agent might be required to train its own critic.

Lowe et al. proposed a generally applicable algorithm called MADDPG for both cooperative and competitive settings, which learns a unique centralized critic for each agent that is conditioned on extra state information and other agents' actions [34]. Many works have proposed extensions to improve the performance and scalability of MADDPG and of centralized critics in general. Iqbal et al. used attention to dynamically select relevant information for estimating the centralized critic's value function [51]. ATT-MADDPG uses attention on the centralized critic to explicitly model the dynamic joint policy of teammates in order to improve cooperation [52].

In addition to non-stationarity, centralized critics can also be used for credit assignment in fully cooperative settings. Foerster et al. proposed COMA, which uses a learned centralized critic to compute a counterfactual baseline for each agent that estimates the contribution of each agent to the shared reward [35]. Similarly, Zhou et al. proposed LICA, a framework for implicit credit assignment which directly ascends approximate joint action value gradients of the centralized critic, which is represented through a hyper-network to enable incorporation of latent state information directly into the policy gradients [53].

3.3 Factorized Value Functions

While centralized critics can guide policy learning efficiently in actor-critic methods, another way to utilize the CTDE paradigm effectively is to learn a team-value function and factorize it into agent-wise value functions that can be used during execution. That is, each agent has its own parameterized Q-function, the learning of which is guided by the joint team value function. The end-to-end framework allows for credit to be distributed implicitly from the centralized head to the individual learners, thus allowing for effective learning

in joint reward settings. In summary, centralized learning stabilizes the training, while value factorization ultimately ensures decentralized and scalable execution.

Value Decomposition Networks (VDN) factorize the joint team value function on the assumption that the joint function can be represented as a sum of individual value functions [33]. However, by enforcing the structural constraint of *additivity*, VDN limits the class of functions that can be learned, i.e., not all possible factorizations are additive in nature. QMIX enforces the constraint of *monotonicity* between the team and individual value functions, which allows it to represent the team value function as a non-linear combination of individual agent values using a mixing network [36]. This enables it to represent a richer class of functions than VDN. QTRAN aims to learn a more general factorization without any structural constraints by transforming the optimal value function into one which is easily factorizable and has the same optimal actions [54]. MAVEN extends QMIX and other value factorization methods by using a shared latent variable controlled by a hierarchical policy to guide committed and temporally extended exploration [55].

3.4 Communication Learning

Communication provides agents with the ability to share information and overcome many limitations in decentralized MARL. For example, sharing observations/knowledge can allow agents to augment their knowledge, effectively alleviating partial observability. Non-stationarity can also be mitigated by allowing agents to share their policies/actions (or intent) with each others, further boosting cooperation [30, 55]. Table 2 summarizes the communication learning algorithms reviewed in this paper.

3.4.1 Message Generation

The first challenge in communication learning is learned message generation. In the majority of existing works, agents learn to generate messages by encoding their current observation or the hidden state of a recurrent network fed with this observation. Within a time step, messages are generated in one of two ways - by feeding each agents' information into a message generation module [30, 31, 38–40, 56–68], or by allowing

each agent to sequentially process a "message" (or set of messages) passed among agents (following a given topology/topologies) [69–71].

The generation of these message can be trained in two ways - either via reinforced or differentiable communication learning [38]. Reinforced communication learning implicitly trains message generation using the agent's own reward signal, naturally extending vanilla reinforcement learning. Differentiable communication learning (DCL) explicitly trains message generation by backpropagating the learning gradients from the recipient agent(s) through the communication channel. Because of this richer feedback, DCL is preferentially used, as it is generally able to handle more complex tasks. However, DCL relies on the assumption of a differentiable communication channel, and thus cannot directly handle channels with unknown noise or binary/discrete messages.

Communications can be further enhanced by incorporating environmental or task information into their representation, and can even rely on the environment as the (implicit) communication channel. Agarwal et al. proposed to incorporate the environment's intrinsic high-level structure by exchanging messages along the edges of a shared agent-entity graph, where the environment is described as a collection of discrete entities, and the positions of those entities serve as transmitted information [72]. Cao et al. found that grounding the communication channel such that messages are binding and verifiable allows agents to learn to communicate in mixed cooperative-competitive settings [73]. Finally, ForMIC proposed to consider *implicit* (stigmergic) communications among agents for multi-robot foraging, where robots create trails of pheromones in the environment that decay in concentration over time, to indicate the location and utility of distant food resources [74].

3.4.2 Communication Topology

Communication topologies can be classified into three types: predefined fixed topology [30, 39, 56, 63, 69–71], predefined dynamic topology [57, 58, 60, 61, 65, 72, 75] and learned topology [31, 40, 59, 62, 64, 67]. Predefined fixed topologies are manually defined before training based on prior assumptions and remain fixed throughout training and execution, e.g., global communications (complete graph among agents). Similarly, predefined

dynamic topologies rely on a pre-defined, non-learnable condition to decide whether communication would occur between any given pair of agents at every time step. For example, many works have either assumed a fixed communication range around each agent [57, 60, 61, 65, 72], a threshold on the confidence level of each agent's individual decision [58], or the influence of the presence of other agents for decision adjustment [75].

In contrast, learned topologies update the communication rule among agents via training, using gradient backpropagation similar to standard MARL. The learned topology is usually dynamic, and the communication graph can change at every time step to match the needs of the task. Common techniques for determining the subset of agents that should communicate include: hard attention [62], soft-max layers (i.e., sampling from a learned probability distribution) [40], and weight generator networks that select a fixed-sized subset of agents to communicate to [31]. In practice, real-world constraints such as communication bandwidth need to be considered when applying communication topologies to robots, and some of these works specifically aim to achieve *sparse* topologies [31]. More recent work have further proposed to teach agents to minimize the amount of information shared at every time-step (i.e., select to generate shorter messages), allowing agents to even select when to avoid communication entirely [76].

4 Towards Multi-Robot Applications

In this section, we discuss AI benchmarks and real-world robotic tasks, which are used to assess the performance of cooperative multi-agent/-robot teams. Simplified benchmarks allow us to test specific aspects of cooperation in controlled environments, and provide a common basis on which to compare different approaches. In contrast, verifying MARL models on real robots in realistic scenarios helps to test the entire set of capabilities required for deployments, such as scalability and robustness to sensing/actuation noise.

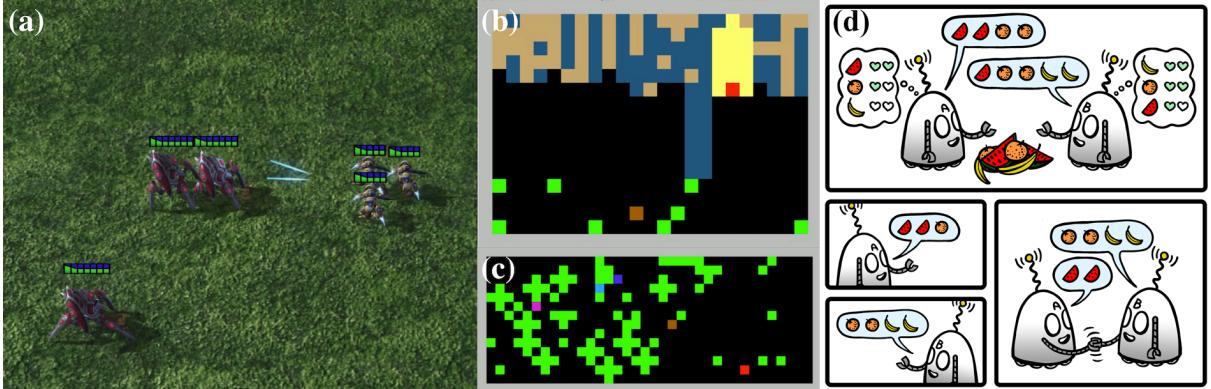


Fig. 3: (a) *Starcraft Multi-Agent Challenge* example task (3s_v_3z), where the goal of the learning agents (left agents) is to eliminate all enemy agents (right agents), by selecting the correct movement/attack actions cooperatively. (b) *Cleanup* is a public goods dilemma where agents need to maximize the team’s reward from individual, mixed cooperative-competitive rewards. Agents (red and brown squares) get rewards for consuming apples (green), which only (re-)grow when the river (blue) is clean from waste (brown), but do not directly get rewards for cleaning the river (yellow beam). (c) Similarly, *Harvest* is a common pool resource dilemma (tragedy of the commons), where apples (green squares) regrow at a rate proportional to the number of nearby apples, thus teaching agents (colored squares) to limit their (selfish) desires to consume apples, to reach a team-optimal behavior. (d) *Negotiation task*, where agents with individual needs have to divide up a set of items to maximize utility, based on successive communications. Figure (d) by and courtesy of Anna Szűcs.

4.1 AI Cooperation Benchmarks

Benchmarks serve as a common way to evaluate and compare the performance of different models. We discuss cooperative, mixed cooperative-competitive, as well as communication learning benchmarks.

4.1.1 Cooperative Setting Benchmarks

The most popular benchmarks for cooperative settings are the StarCraft II Multi-Agent Challenge (SMAC) [23] and the multi-agent particle environment [24] which both test the general ability of agents to work together in fully cooperative tasks with continuous observation and discrete action spaces. SMAC focuses on team based cooperation by simulating a skirmish between two platoons, one of which is controlled by MARL agents and the other by the built-in, non-learning game AI. An example of a symmetric, homogeneous battle scenario is shown in Fig 3 (a), where the goal of the agents is to eliminate all adversaries to win the skirmish. In order to make the tasks more challenging, SMAC includes asymmetric battle scenarios with heterogeneous team configurations,

which require advanced tactics as well as the ability to find and act one’s role in the team to ensure success. The multi-agent particle environment is a simpler benchmark consisting of a multi-agent particle world with basic simulated physics [24]. This environment can be easily modified for the design of new tasks, in addition to the basic set of benchmark tasks that are provided such as prey-predator.

4.1.2 Mixed Cooperative-Competitive Setting Benchmarks

Mixed cooperative-competitive benchmarks focus on scenarios that threaten the stability of purely cooperative tasks, allowing us to test if agents will be able to cooperate, even in situations where their individual rewards motivates them to act in a way that might be detrimental to the collective good. In particular, group dilemmas present agents with scenarios where they must work together to obtain higher team rewards, despite a built-in incentive to be selfish. That is, if an agent does not cooperate with the team, they can maximize their own short-term return, but all agents in the environment (including themselves) will suffer the cost

of their selfish actions in the long run. The main benchmarks look at sequential social dilemmas, such as the common pool resource dilemma, and public goods dilemma which have been implemented in *Cleanup* [77] and *Harvest* respectively. In *Cleanup* shown in Fig 3 (b), agents gain reward from consuming apples, which only regrow when the river is clean from waste. Although the team as a whole benefits from agents cleaning the river, these cleaning agents are not directly rewarded as they do it, while other agents are rewarded for consuming newly-regrown apples. In *Harvest* shown in Fig 3 (c), apples regrow based on how many apples are nearby. *Harvest* is a common-pool resource appropriation problem (*tragedy of the commons*), where agents directly gain reward from consuming apples, which regrow naturally proportionally to the amount of nearby apples. Therefore, agents need to limit their selfish desire to consume as many apples as they can, to help maximize the regrowth rate and maximize the overall long-term team return.

Team return – the overall sum of rewards received by all agents in the long run – is traditionally used to track training progress in fully cooperative/competitive tasks. However, team return is an insufficient metric in mixed cooperative-competitive settings, where improving agent cooperation does not necessarily immediately increase returns, and returns even often see a downwards trend before going back up once a nearer-team-optimal cooperation equilibrium is reached. That is, unlike single agent cases where it is easier for the agent to self-correct to a more sustainable strategy, agents in a MAS often fall into socially-deficient Nash equilibria, where changing their own behavior is insufficient to change the overall team behavior, and thus acting greedily remains the dominant strategy [78]. As a result, mixed cooperative-competitive benchmarks often turn to social outcome metrics [79], such as equality and sustainability, which help shed light on the true progress of the cooperation learning process.

For a more quantitative evaluation, simple social dilemma matrix games, extended from game theory, have been used to analyze how different learning aids affect the probability of convergence to better/worse equilibria. For example, general-sum coordination matrix games such as *Stag Hunt* can help demonstrate how gifting between agents

can improve the model’s tendency to converge to the true, pro-social equilibrium [80].

4.1.3 Communication Benchmarks

We examine communication learning benchmarks in both cooperative and mixed cooperative-competitive settings. These benchmarks generally evaluate the agents’ ability to share relevant information effectively, or to reach an agreement on how agents should compromise for the welfare of the team. In purely cooperative settings, the most popular communication benchmarks are *referential games*, where one agent is made aware of a target object, and has to generate a message that communicates to another agent which one to pick out of a set of candidate objects. Referential games can be used to examine the viability of different mediums of communication [81]. However, unlike non-communicative cooperative benchmarks, which mainly focus on the performance achieved within the game, referential games are also often used to measure the quality of the learned communication protocol. For example, they can shed light into the ease of teaching such protocols to new agents, or, through the linguistic analysis of the compositionality of the learned language, can help examine the generalizability of communication protocols [82].

Communication benchmarks in mixed cooperative-competitive setting aim to assess whether agents can learn to communicate successfully to reach a compromise for the good of the team, even when it does not necessarily benefit them individually. The main benchmarks used are *negotiation tasks*, where two agents with different hidden utility functions have to split a randomly initiated item pool by making successive proposals until both agents agree [83]. These tasks can be used to examine the conditions required for agents to learn to reach a *consensus* through sequential communications [73, 84].

4.2 Motion and Path Planning

Many robot teams are composed of mobile robots, of which controlling the (collision-free) motion is of primary concern. More involved multi-robot tasks, such as RoboSoccer [85], collaborative manipulation [86], multiple travelling salesman problem [87] and multi-UAV surveillance [88], also consider the coordination/distribution of robots,

and may add additional actions (e.g., kick the ball, or take measurements). As a result, multi-robot path/motion planning has become one of the most important and widely used tasks to evaluate the performance of deep RL methods for MRS. In doing so, the hope is that the learned capabilities needed for cooperation in multi-robot path/motion planning will further extend to other, more general multi-robot tasks [85, 86, 88]. Additionally, thanks to the recent developments in robot motion control, it has become easier to verify deep RL algorithms for multi-robot motion/path planning on real robots [89, 90].

Many approaches have proposed to rely on independent learning for multi-robot path planning problems. In particular, recently proposed approaches have built on both value-based and policy gradient-based RL algorithms for multi-robot path planning. Wang et al. modified the dueling DQN algorithm and trained policies from images generated from robot-centric relative perspectives [91]. Fan et al. proposed an approach that uses PPO, and allows agents to make decisions directly based on sensor-level inputs, which enables easier implementation on real robots [89]. The learned policy is also integrated into a hybrid control framework to further improve its robustness and effectiveness. Imitation learning has also been combined with deep RL to achieve better performance and sample efficiency for multi-robot path planning [92, 93], by directly training agents to imitate the type of cooperative behaviors exhibited by centralized path planners, while showing scalability to thousands of agents. Many of these works include experimental validation of the trained policies on hardware to verify their performance under realistic conditions [89, 90, 92].

More recent works in multi-robot path planning have relied on centralized critics and value factorization methods. Notably, Marchesini et al. proposed a centralized state-value network to inject global state information in the value-based updates of the agents, demonstrating better performance than independent learning and VDN [94]. Huang et al. relied on MADDPG with prioritized experience replay to efficiently utilize the collected experience [95]. He et al. applied MASAC based on cooperative waypoints search, and showed improvements in performance compared to other MARL algorithms such as MATD3

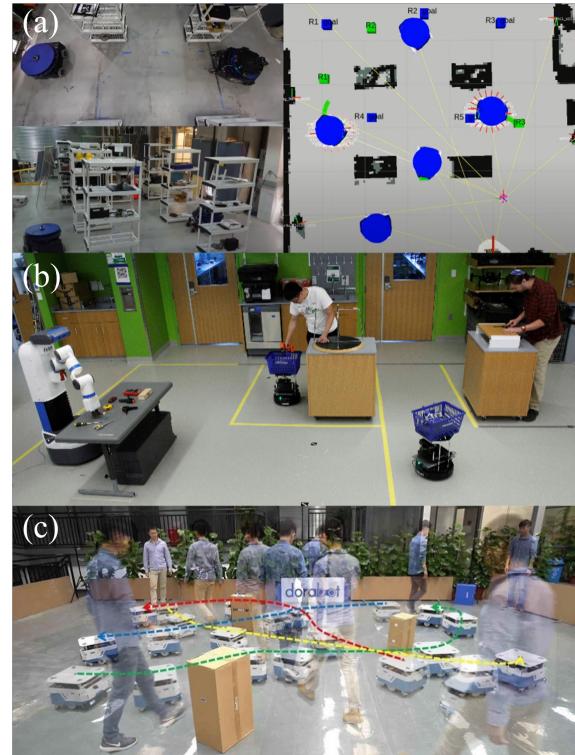


Fig. 4: Experimental validation of deep MARL path/motion planning policies on MRS, considered in [92], [90] and [89] respectively. Robots are assigned each a goal location and need to cooperatively plan collision-free paths to these goals. In (a) and (b), robots need to avoid static obstacles in warehouse-like scenarios, while (c) considers crowded scenarios involving static and dynamic obstacles. (b) and (c) have been reused with the permission of the authors and copyright owners.

and MAAC [96]. Value factorization approaches include de Witt et al., who introduced COMIX, which employs a decentralized joint action-value function with a per-agent factorization, and showed significant improvement over MADDPG and VDN [97]. Freed et al. proposed to rely on learned attention mechanisms to identify and decouple subsets of robots, which reduced the gradient estimator variances in a variety of motion-/path planning tasks, showing improvements over independent learning and COMA [98].

5 Open Avenues for Research

Safe MARL: When developing MARL algorithms intended to be directly implemented on robots in the real-world, safety is often of paramount importance. Safe RL is defined as training an agent to learn an efficient policy, while minimizing violations of safety constraints, such as avoiding collisions with other robots or obstacles/humans, or ensuring the robot’s safety/stability (e.g., enforcing actions that keep a flying robot in the air). These safety guarantees need to be guaranteed when deploying the final learned policies, and sometimes even during training. Numerous recent works have focused on safe single-agent RL. They can be broadly classified into two types: modifying the optimization criterion and regulating the exploration process [99].

However, the community is still in the early phases of transitioning from safe single-agent RL to safe MARL. Ensuring safety becomes much more difficult in the presence of multiple agents, as the environment’s instability increases, and solutions must now account for coupling among agents, particularly between those with competing interests. In the limited research on safe MARL so far, we note Shalev et al., who devised a safe MARL algorithm for autonomous driving [100], and Zhang et al., who proposed a multi-agent algorithm that provably guarantees safety for arbitrary learned policies [101].

Simulation to Real-World: Training in the real world is often costly, time-consuming, and sometimes dangerous or infeasible. Thus, using simulated environments has become a common choice for robotics tasks, with a myriad of environments developed to simulate real-world tasks, such as MINOS (indoor 3D Navigation) [102], Assistive Gym (physical human-robot interaction and robotic assistance) [103], SURREAL (robot manipulation) [104]. However, MARL research that successfully deploys policies trained in the virtual environment on physical robots has been rare so far [89, 105]. Due to complicated mechanical interactions, uncertainties in real system dynamics, time discretization, and numerous other challenges, building a simulator that perfectly reflects the real world is often impossible. On the other hand, most MARL algorithms are notoriously sensitive to their input, and simple changes can drastically affect performance on the

same task [106]. In particular, policies learned in low-precision simulations frequently cannot be transferred directly to real robots. As a result, techniques that allow policies to be trained in simulation, and then adapted for implementation on robots in the real world, is an open question of importance in the community.

Model-Based MARL: Most of the algorithms introduced above are model-free methods, i.e., agents learn policy purely by interacting with the environment. In contrast, model-based RL optimizes policies based on a known/learned model of the state-transition and/or rewards dynamics. With the development of deep learning, model-free RL has attracted the most interest in recent year. However, model-free RL requires large amounts of experiences to achieve acceptable performance, which, when applied on real robots, may render training times unreasonable and increase the risk of accidents and wear and tear on the hardware [107]. In contrast, single-agent model-based RL methods have been proposed recently, which exhibit improved sample-efficiency and safety [108, 109]. Given these promising results, we believe that model-based MARL may be one of the next frontiers for MARL. However, model-based MARL comes with a number of new challenges, as agents now need to learn the dynamics of an environment containing other agents, thus further increasing the impact of non-stationarity and partial-observability, as well as scalability.

Decentralized Joint Decision-Making: While existing communication learning methods can endow agents with the ability to augment each other’s knowledge about the environment, share intent, or reach a consensus through negotiation, we believe that the holy grail of communication learning remains methods that can allow agents to *consensually* generate joint plans from communications. That is, advanced communications between agents may allow the team to reach centralized performances, while remaining fully decentralized. In doing so, agents will most likely need to be able to diffuse relevant information in the team (including current knowledge and future intent/predictions), perform multiple back-and-forth communications to allow true dialogue to happen, and reach a consensus over joint actions to be selected for team-level optimality. Although some of these capabilities have already

been addressed by existing methods (as discussed in Section 3.4), their combination into a functioning, decentralized joint decision-making framework is not trivial. In particular, we note that, while decentralized consensus-based methods have been used to achieve decentralized training in MARL [110], only a few works so far have truly addressed consensual joint action selection [111].

6 Conclusions

This survey provides a broad overview of recent work in distributed model-free MARL for MAS/MRS, and reports recent applications of MARL to AI and fundamental robotics tasks. Cooperative MARL can be achieved by four general classes of approaches, namely independent learning, centralized critic, factorized value functions, and communication learning. These methods have been investigated and validated extensively on AI benchmarks, but we also highlight recent works that were demonstrated on MRS, especially in the area of multi-robot motion/path planning. Despite the exciting recent advances in the field, we also identify a few open avenues for future research in the application/improvement of MARL for multi-robot teams, such as safe MARL, sim2real transfer, model-based MARL, and decentralized joint decision-making. We remain highly optimistic about the progress of the MARL community, and hope that this survey will inspire further developments in this very active and captivating field, to bring us ever closer to the wide-spread, safe implementation of these methods in the real-world to support the robotics deployments of tomorrow.

7 Highlighted References to be added into the Reference List

- Reference [30]. This work allows globally-communicating agents to share intent by modeling the environment dynamics and other agents' actions.
- Reference [31]. This work allows agents to learn to estimate the importance of their observation/-knowledge, to selectively broadcasts continuous messages to the whole team.

•• Reference [46]. This work proposed to encourage cooperation among agents by relying on an intrinsic reward that aims at maximizing their influence over each other.

• Reference [48]. This work shows that independent learning using on-policy algorithms such as PPO can perform effectively in fully cooperative MARL environments.

•• Reference [51]. This work uses an attention mechanism in the centralized critic to dynamically select relevant information.

• Reference [53]. This work proposes a framework for implicit credit assignment which directly ascends approximate joint action value gradients of the centralized critic.

• Reference [54]. This work aims to learn a general value factorization without any structural constraints by transforming the optimal value function into one which is easily factorizable.

• Reference [55]. This work extends QMIX and other value factorization methods by using a hierarchical policy to guide committed and temporally extended exploration.

• Reference [61]. This work formalizes the multi-agent system as a graph and lets agents communicate with neighbors via graph convolution to solve the multi-agent pathfinding task.

• Reference [62]. This work uses a two-stage attention network to estimate whether two agents should communicate and the importance of that communication instance.

•• Reference [67]. This work incorporates relies on a conventional coupled planner to guide the learning of the communication topology in multi-agent pathfinding.

•• Reference [89]. This work presents a deep RL-based decentralized collision-avoidance framework for multi-robot path planning based on sensor inputs, with numerical and experimental validation results.

• Reference [92]. This work introduces a scalable framework for multi-agent pathfinding which utilizes RL and imitation learning to learn decentralized policies that can scale to more than a thousand agents.

Supplementary information. Not applicable.

Acknowledgments. Not applicable.

Declarations

Funding. This work was supported by the Singapore Ministry of Education Academic Research Fund Tier 1.

Conflict of interest/Competing interests. The authors declare that they have no conflict of interest nor competing interests.

Human and Animal Rights. This article does not contain any studies with human or animal subjects performed by any of the authors.

Ethics approval. Not applicable.

Consent to participate. Not applicable.

Consent for publication. Not applicable.

Availability of data and materials. Not applicable.

Code availability. Not applicable.

Authors' contributions. All authors contributed to the study conception, study design, original draft preparation, as well as review and editing. Yutong Wang performed figure design, and the literature search and data analysis for communication learning methods, challenges, and benchmarks, as well as for open avenues for research. Mehul Damani performed the literature search and data analysis for background, as well as communication-free cooperation methods and challenges. Pamela Wang co-performed the literature search and data analysis for communication learning and benchmarks and Figure 3. Yuhong Cao performed the literature search and data analysis for multi-robot applications and Figure 4. Guillaume Sartoretti performed project administration, supervision, and was involved in the literature search for all aspects of this survey.

Representative Works in Cooperative MARL				
Paper	Learning Setup	Coop. Technique	Summary	Challenges Addressed
Lauer et al. [41]	IND	IL	Update Q-Values only if there is guaranteed improvement	NST
Hysteretic-DQN [42]	IND	IL	Use smaller learning rate for negative Q-updates	NST
Lenient-DQN [43]	IND	IL	Show progressively decreasing lenience in negative Q-updates	NST
Stabilising ER [49]	IND	IL	Use importance sampling and fingerprints to stabilize experience replay in MARL	NST
MADDPG [34]	CTDE	CC	Train a centralized critic for each agent with augmented information	NST
COMA [35]	CTDE	CC	Use counterfactual baseline computed using centralized critic for credit assignment	CA,NST
Actor-Attention Critic [51]	CTDE	CC	Use an attention mechanism on the centralized critic to dynamically select relevant information	SCA, NST
VDN [33]	CTDE	VF	Decompose team value function into a sum of agent-wise value functions	CA,NST
QMIX [36]	CTDE	VF	Decompose team value function into a non-linear combination of agent-wise value functions using a mixing network	CA,NST
Omidshafiei et al. [45]	CTDE	IL	Use hysteretic learning, concurrent experience replay, recurrent networks and distillation to learn a multi-task MARL policy	PO,NST,SCA
Gupta et al. [22]	CTDE	IL	Use parameter sharing among homogeneous agents to learn more efficiently	SCA
CommNet [39]	IND	CL	Sum continuous messages between different layers of all agents' networks	NST,PO
BicNet [69]	IND	CL	Use bi-directional recurrent neural network in the actor-critic paradigm	NST,PO
IC3Net [40]	IND	CL	Selectively communicate through a gating mechanism	NST,PO,SCA
SchedNet [31]	CTDE	CL	Selectively broadcast continuous messages by learning importance of each agent's observation	NST,PO

Table 1: Summary of the major cooperative MARL algorithms reviewed in this paper. Learning setup (Sect. 2.4) is either independent (**IND**) or centralized training decentralized execution (**CTDE**). Cooperation techniques are classified into four based on Sect. 3, namely independent learners (**IL**), centralized critic (**CC**), value factorization (**VF**) and communication (**CL**). The major challenges in MARL (Sect. 2.5) addressed by these algorithms are non-stationarity (**NST**), credit assignment (**CA**), scalability (**SCA**), and partial observability (**PO**).

Reference	Scope of communication	Time-lagged	Setting	Shared Information
AC-CNet [56]	Global	No	Co	Observation
A-CCNet [56]	Global	No	Co	Observation & Action
IS [30]	Global	Yes	Co	Intention
CCOMA [57]	Partial	No	Co	Observation
VBC [58]	Targeted	No	Co	Observation
ATOC [59]	Targeted	No	Co	Observation & Intention
DGN [60]	Partial	No	Co	Observation
DHC [61]	Partial	No	Co	Observation
DCC [75]	Targeted & Partial	No	Co	Observation & Relative position
CommNet [39]	Global	No	Co	Observation
SchedNet [31]	Targeted	No	Co	Observation
Agarwal et.al. [72]	Global (unrestricted) & Partial (restricted)	No	Co	Observation & Environment Information
IC3Net [40]	Targeted	Yes	Co & Mix & Com	Observation
GA-Comm [62]	Targeted	No	Co	Observation
GA-AC [62]	Targeted	No	Co	Observation & Action
BicNet [69]	Global	No	Co	Observation & Action
MS-MARL [63]	Global	No	Co	Intention
TarMAC [64]	Targeted	Yes	Co & Mix & Com	Observation
Blumenkamp [65] and Prorok	Partial	No	Co & Mix	Observation
MD-MADDPG [70]	Global	No	Co	Observation
FlowComm [66]	Targeted	No	Co	Observation
DIAL [38]	Global	Yes	Co	Observation
RIAL [38]	Global	Yes	Co	Observation
PICO [67]	Partial & Targeted	No	Co	Observation
FCMNet [71]	Global	No	Co	Observation
Cao et al. [73]	Global	No	Mix	Intention & Proposal
Noukhovitch et al. [84]	Global	No	Mix	Proposal
Mihai et al. [81]	Global	No	Co	Observation
Li et al. [82]	Global	No	Co	Observation
Freed et al. [76]	Global	No	Co	Observation
ForMIC (implicit) [74]	Partial	Yes	Co	Memory

Table 2: Summary of the communication learning algorithms reviewed in this paper and detailed in Section 3.4. Communications can be classified as either explicit or implicit. Implicit communications are defined as indirect communication between agents in which agents transmit information via their shared environment. In contrast, explicit communications are separated from the environment, and let agents directly send messages to other agents (All algorithms in the table except ForMIC consider explicit communication). The scope of communication can either be global, partial, or targeted. Global communication refers to settings where agents broadcast messages to all other agents. In partial communications, agents only communicate to other agents within a certain range. Finally, targeted communication means that agents (learn to) select the specific agents to communicate/listen to at each time-step. Time-lagged communications consider cases where agents use messages received at one time-step, as input to their decision-making at the next time-step. Tasks setting include: cooperative (**Co**), mixed (**Mix**), and competitive (**Com**). Shared information refers to the high-level information encoded within messages shared.

References

- [1] Nägele L, Schierl A, Hoffmann A, Reif W. Multi-robot cooperation for assembly: Automated planning and optimization. In: International Conference on Informatics in Control, Automation and Robotics. Springer; 2019. p. 169–192.
- [2] Ma K, Ma Z, Liu L, Sukhatme GS. Multi-robot Informative and Adaptive Planning for Persistent Environmental Monitoring. In: Distributed Autonomous Robotic Systems, The 13th International Symposium, DARS 2016, Natural History Museum, London, UK, November 7-9, 2016. vol. 6; 2016. p. 285–298. Available from: https://doi.org/10.1007/978-3-319-73008-0_20.
- [3] Wang H, Zhang C, Song Y, Pang B. Master-Followed multiple robots cooperation SLAM adapted to search and rescue scenarios. In: IEEE International Conference on Information and Automation, ICIA 2017, Macau, SAR, China, July 18-20, 2017; 2017. p. 579–585. Available from: <https://doi.org/10.1109/ICInfA.2017.8078975>.
- [4] Oroojlooyjadid A, Hajinezhad D. A Review of Cooperative Multi-Agent Deep Reinforcement Learning. CoRR. 2019;abs/1908.03963. <https://arxiv.org/abs/1908.03963>.
- [5] Hernandez-Leal P, Kartal B, Taylor ME. A survey and critique of multiagent deep reinforcement learning. Auton Agents Multi Agent Syst. 2019;33(6):750–797. <https://doi.org/10.1007/s10458-019-09421-1>.
- [6] Nguyen TT, Nguyen ND, Nahavandi S. Deep Reinforcement Learning for Multi-agent Systems: A Review of Challenges, Solutions, and Applications. IEEE Trans Cybern. 2020;50(9):3826–3839. <https://doi.org/10.1109/TCYB.2020.2977374>.
- [7] Gronauer S, Diepold K. Multi-agent deep reinforcement learning: a survey. Artif Intell Rev. 2022;55(2):895–943. <https://doi.org/10.1007/s10462-021-09996-w>.
- [8] Papoudakis G, Christianos F, Rahman A, Albrecht SV. Dealing with Non-Stationarity in Multi-Agent Deep Reinforcement Learning. CoRR. 2019;abs/1906.04737. <https://arxiv.org/abs/1906.04737>.
- [9] Cortés J, Egerstedt M. Coordinated control of multi-robot systems: A survey. SICE Journal of Control, Measurement, and System Integration. 2017;10(6):495–503.
- [10] Tuci E, Alkilabi MHM, Akanyeti O. Cooperative Object Transport in Multi-Robot Systems: A Review of the State-of-the-Art. Frontiers Robotics AI. 2018;5:59. <https://doi.org/10.3389/frobt.2018.00059>.
- [11] Feng Z, Hu G, Sun Y, Soon J. An overview of collaborative robotic manipulation in multi-robot systems. Annu Rev Control. 2020;49:113–127. <https://doi.org/10.1016/j.arcontrol.2020.02.002>.
- [12] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, et al. Playing Atari with Deep Reinforcement Learning. CoRR. 2013;abs/1312.5602. <https://arxiv.org/abs/1312.5602>.
- [13] Sutton RS, McAllester DA, Singh SP, Mansour Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In: Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]; 1999. p. 1057–1063.
- [14] Mnih V, Badia AP, Mirza M, Graves A, Lillicrap TP, Harley T, et al. Asynchronous Methods for Deep Reinforcement Learning. In: Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016. vol. 48; 2016. p. 1928–1937. Available from: <http://proceedings.mlr.press/v48/mnih16.html>.
- [15] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal Policy Optimization Algorithms. CoRR. 2017;abs/1707.06347. <https://arxiv.org/abs/1707.06347>.

- [16] Haarnoja T, Zhou A, Abbeel P, Levine S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. vol. 80; 2018. p. 1856–1865. Available from: <http://proceedings.mlr.press/v80/haarnoja18b.html>.
- [17] Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, et al. Continuous control with deep reinforcement learning. In: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings; 2016. Available from: <http://arxiv.org/abs/1509.02971>.
- [18] Sutton RS, Barto AG. Reinforcement learning: An introduction. MIT press; 2018.
- [19] Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA. Deep Reinforcement Learning: A Brief Survey. IEEE Signal Process Mag. 2017;34(6):26–38. <https://doi.org/10.1109/MSP.2017.2743240>.
- [20] Kaelbling LP, Littman ML, Moore AW. Reinforcement Learning: A Survey. J Artif Intell Res. 1996;4:237–285. <https://doi.org/10.1613/jair.301>.
- [21] François-Lavet V, Henderson P, Islam R, Bellemare MG, Pineau J. An Introduction to Deep Reinforcement Learning. Found Trends Mach Learn. 2018;11(3-4):219–354. <https://doi.org/10.1561/2200000071>.
- [22] Gupta JK, Egorov M, Kochenderfer MJ. Cooperative Multi-agent Control Using Deep Reinforcement Learning. In: Autonomous Agents and Multiagent Systems - AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers. vol. 10642; 2017. p. 66–83. Available from: https://doi.org/10.1007/978-3-319-71682-4_5.
- [23] Samvelyan M, Rashid T, de Witt CS, Farquhar G, Nardelli N, Rudner TGJ, et al. The StarCraft Multi-Agent Challenge. In: Proceedings of the 18th International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019; 2019. p. 2186–2188. Available from: <http://dl.acm.org/citation.cfm?id=3332052>.
- [24] Mordatch I, Abbeel P. Emergence of Grounded Compositional Language in Multi-Agent Populations. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018; 2018. p. 1495–1502. Available from: <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17007>.
- [25] Silver D, Hubert T, Schrittwieser J, Antonoglou I, Lai M, Guez A, et al. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. CoRR. 2017;abs/1712.01815. <https://arxiv.org/abs/1712.01815>.
- [26] Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. Nat. 2015;518(7540):529–533. <https://doi.org/10.1038/nature14236>.
- [27] Tampuu A, Matiisen T, Kodelja D, Kuzovkin I, Korjus K, Aru J, et al. Multiagent Cooperation and Competition with Deep Reinforcement Learning. CoRR. 2015;abs/1511.08779. <https://arxiv.org/abs/1511.08779>.
- [28] Resnick C, Eldridge W, Ha D, Britz D, Foerster JN, Togelius J, et al. Pommerman: A Multi-Agent Playground. In: Joint Proceedings of the AIIDE 2018 Workshops co-located with 14th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2018), Edmonton, Canada, November 13-14, 2018. vol. 2282; 2018. Available from: <http://ceur-ws.org/>

Distributed Reinforcement Learning for Robot Teams: A Review

[Vol-2282/MARLO_104.pdf.](#)

- [29] Suarez J, Du Y, Isola P, Mordatch I. Neural MMO: A Massively Multiagent Game Environment for Training and Evaluating Intelligent Agents. CoRR. 2019;abs/1903.00784. <https://arxiv.org/abs/1903.00784>.
- [30] Kim W, Park J, Sung Y. Communication in Multi-Agent Reinforcement Learning: Intention Sharing. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021; 2021. Available from: <https://openreview.net/forum?id=qpsl2dR9twy>.
- [31] Kim D, Moon S, Hostallero D, Kang WJ, Lee T, Son K, et al. Learning to Schedule Communication in Multi-agent Reinforcement Learning. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019; 2019. Available from: <https://openreview.net/forum?id=SJxu5iR9KQ>.
- [32] Chu T, Wang J, Codecà L, Li Z. Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control. IEEE Trans Intell Transp Syst. 2020;21(3):1086–1095. <https://doi.org/10.1109/TITS.2019.2901791>.
- [33] Sunehag P, Lever G, Gruslys A, Czarnecki WM, Zambaldi VF, Jaderberg M, et al. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018; 2018. p. 2085–2087. Available from: <http://dl.acm.org/citation.cfm?id=3238080>.
- [34] Lowe R, Wu Y, Tamar A, Harb J, Abbeel P, Mordatch I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA; 2017. p. 6379–6390. Available from: <https://proceedings.neurips.cc/paper/2017/hash/68a9750337a418a86fe06c1991a1d64c-Abstract.html>.
- [35] Foerster JN, Farquhar G, Afouras T, Nardelli N, Whiteson S. Counterfactual Multi-Agent Policy Gradients. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018; 2018. p. 2974–2982. Available from: <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17193>.
- [36] Rashid T, Samvelyan M, de Witt CS, Farquhar G, Foerster JN, Whiteson S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. vol. 80; 2018. p. 4292–4301. Available from: <http://proceedings.mlr.press/v80/rashid18a.html>.
- [37] Hausknecht MJ, Stone P. Deep Recurrent Q-Learning for Partially Observable MDPs. In: 2015 AAAI Fall Symposia, Arlington, Virginia, USA, November 12-14, 2015; 2015. p. 29–37. Available from: <http://www.aaai.org/ocs/index.php/FSS/FSS15/paper/view/11673>.
- [38] Foerster JN, Assael YM, de Freitas N, Whiteson S. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. In: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain; 2016. p. 2137–2145. Available from: <https://proceedings.neurips.cc/paper/2016/hash/c7635bfd99248a2cdef8249ef7bfbef4-Abstract.html>.

- [39] Sukhbaatar S, Szlam A, Fergus R. Learning Multiagent Communication with Backpropagation. In: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain; 2016. p. 2244–2252. Available from: <https://proceedings.neurips.cc/paper/2016/hash/55b1927fdafef39c48e5b73b5d61ea60-Abstract.html>.
- [40] Singh A, Jain T, Sukhbaatar S. Learning when to Communicate at Scale in Multiagent Cooperative and Competitive Tasks. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019; 2019. Available from: <https://openreview.net/forum?id=rye7knCqK7>.
- [41] Lauer M, Riedmiller MA. An Algorithm for Distributed Reinforcement Learning in Cooperative Multi-Agent Systems. In: Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000; 2000. p. 535–542.
- [42] Matignon L, Laurent GJ, Fort-Piat NL. Hysteretic q-learning : an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 29 - November 2, 2007, Sheraton Hotel and Marina, San Diego, California, USA; 2007. p. 64–69. Available from: <https://doi.org/10.1109/IROS.2007.4399095>.
- [43] Panait L, Sullivan K, Luke S. Lenient learners in cooperative multiagent systems. In: 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006; 2006. p. 801–803. Available from: <https://doi.org/10.1145/1160633.1160776>.
- [44] Palmer G, Tuyls K, Bloembergen D, Savani R. Lenient Multi-Agent Deep Reinforcement Learning. In: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018; 2018. p. 443–451. Available from: <http://dl.acm.org/citation.cfm?id=3237451>.
- [45] Omidshafiei S, Pazis J, Amato C, How JP, Vian J. Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability. In: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. vol. 70; 2017. p. 2681–2690. Available from: <http://proceedings.mlr.press/v70/omidshafiei17a.html>.
- [46] Jaques N, Lazaridou A, Hughes E, Gülcehrə Ç, Ortega PA, Strouse D, et al. Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA. vol. 97; 2019. p. 3040–3049. Available from: <http://proceedings.mlr.press/v97/jaques19a.html>.
- [47] Sun M, Devlin S, Hofmann K, Whiteson S. Monotonic Improvement Guarantees under Non-stationarity for Decentralized PPO. CoRR. 2022;abs/2202.00082. <https://arxiv.org/abs/2202.00082>.
- [48] Yu C, Velu A, Vinitsky E, Wang Y, Bayen AM, Wu Y. The Surprising Effectiveness of MAPPO in Cooperative, Multi-Agent Games. CoRR. 2021;abs/2103.01955. <https://arxiv.org/abs/2103.01955>.
- [49] Foerster JN, Nardelli N, Farquhar G, Afouras T, Torr PHS, Kohli P, et al. Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning. In: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. vol. 70; 2017. p. 1146–1155. Available from: <http://proceedings.mlr.press>

Distributed Reinforcement Learning for Robot Teams: A Review

v70/foerster17b.html.

- [50] Tesauro G. Extending Q-Learning to General Adaptive Multi-Agent Systems. In: Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]; 2003. p. 871–878. Available from: <https://proceedings.neurips.cc/paper/2003/hash/e71e5cd119bbc5797164fb0cd7fd94a4-Abstract.html>.
- [51] Iqbal S, Sha F. Actor-Attention-Critic for Multi-Agent Reinforcement Learning. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA. vol. 97; 2019. p. 2961–2970. Available from: [http://proceedings.mlr.press/v97/iqbal19a.html](https://proceedings.mlr.press/v97/iqbal19a.html).
- [52] Mao H, Zhang Z, Xiao Z, Gong Z. Modelling the Dynamic Joint Policy of Teammates with Attention Multi-agent DDPG. In: Proceedings of the 18th International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019; 2019. p. 1108–1116. Available from: <http://dl.acm.org/citation.cfm?id=3331810>.
- [53] Zhou M, Liu Z, Sui P, Li Y, Chung YY. Learning Implicit Credit Assignment for Cooperative Multi-Agent Reinforcement Learning. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual; 2020. Available from: <https://proceedings.neurips.cc/paper/2020/hash/8977ecbb8cb82d77fb091c7a7f186163-Abstract.html>.
- [54] Son K, Kim D, Kang WJ, Hostallero D, Yi Y. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA. vol. 97; 2019. p. 5887–5896. Available from: <http://proceedings.mlr.press/v97/son19a.html>.
- [55] Mahajan A, Rashid T, Samvelyan M, Whiteson S. MAVEN: Multi-Agent Variational Exploration. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada; 2019. p. 7611–7622. Available from: <https://proceedings.neurips.cc/paper/2019/hash/f816dc0acf7e498e10496222e9db10-Abstract.html>.
- [56] Mao H, Gong Z, Ni Y, Liu X, Wang Q, Ke W, et al. ACCNet: Actor-Coordinator-Critic Net for "Learning-to-Communicate" with Deep Multi-agent Reinforcement Learning. CoRR. 2017;abs/1706.03235. <https://arxiv.org/abs/1706.03235>.
- [57] Su J, Adams SC, Beling PA. Counterfactual Multi-Agent Reinforcement Learning with Graph Convolution Communication. CoRR. 2020;abs/2004.00470. <https://arxiv.org/abs/2004.00470>.
- [58] Zhang SQ, Zhang Q, Lin J. Efficient Communication in Multi-Agent Reinforcement Learning via Variance Based Control. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada; 2019. p. 3230–3239. Available from: <https://proceedings.neurips.cc/paper/2019/hash/14cfdb59b5bda1fc245aadae15b1984a-Abstract.html>.
- [59] Jiang J, Lu Z. Learning Attentional Communication for Multi-Agent Cooperation. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada; 2018. p. 7265–7275. Available from: <https://>

- proceedings.neurips.cc/paper/2018/hash/6a8018b3a00b69c008601b8becae392b-Abstract.html.
- [60] Jiang J, Dun C, Huang T, Lu Z. Graph Convolutional Reinforcement Learning. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020; 2020. Available from: <https://openreview.net/forum?id=HkxdQkSYDB>.
- [61] Ma Z, Luo Y, Ma H. Distributed Heuristic Multi-Agent Path Finding with Communication. In: IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021; 2021. p. 8699–8705. Available from: <https://doi.org/10.1109/ICRA48506.2021.9560748>.
- [62] Liu Y, Wang W, Hu Y, Hao J, Chen X, Gao Y. Multi-Agent Game Abstraction via Graph Attention Neural Network. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020; 2020. p. 7211–7218. Available from: <https://ojs.aaai.org/index.php/AAAI/article/view/6211>.
- [63] Kong X, Xin B, Liu F, Wang Y. Revisiting the Master-Slave Architecture in Multi-Agent Deep Reinforcement Learning. CoRR. 2017;abs/1712.07305. <https://arxiv.org/abs/1712.07305>.
- [64] Das A, Gervet T, Romoff J, Batra D, Parikh D, Rabbat M, et al. TarMAC: Targeted Multi-Agent Communication. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA. vol. 97; 2019. p. 1538–1546. Available from: [http://proceedings.mlr.press/v97/das19a.html](https://proceedings.mlr.press/v97/das19a.html).
- [65] Blumenkamp J, Prorok A. The Emergence of Adversarial Communication in Multi-Agent Reinforcement Learning. In: 4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA. vol. 155; 2020. p. 1394–1414. Available from: <https://proceedings.mlr.press/v155/blumenkamp21a.html>.
- [66] Du Y, Liu B, Moens V, Liu Z, Ren Z, Wang J, et al. Learning Correlated Communication Topology in Multi-Agent Reinforcement learning. In: AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021; 2021. p. 456–464. Available from: <https://dl.acm.org/doi/10.5555/3463952.3464010>.
- [67] Li W, Chen H, Jin B, Tan W, Zha H, Wang X. Multi-Agent Path Finding with Prioritized Communication Learning. CoRR. 2022;abs/2202.03634. <https://arxiv.org/abs/2202.03634>.
- [68] Pesce E, Montana G. Connectivity-driven Communication in Multi-agent Reinforcement Learning through Diffusion Processes on Graphs. CoRR. 2020;abs/2002.05233. <https://arxiv.org/abs/2002.05233>.
- [69] Peng P, Yuan Q, Wen Y, Yang Y, Tang Z, Long H, et al. Multiagent Bidirectionally-Coordinated Nets for Learning to Play StarCraft Combat Games. CoRR. 2017;abs/1703.10069. <https://arxiv.org/abs/1703.10069>.
- [70] Pesce E, Montana G. Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication. Machine Learning. 2020;109(9-10):1727–1747. <https://doi.org/10.1007/s10994-019-05864-5>.
- [71] Wang Y, Sartoretti G. FCMNet: Full Communication Memory Net for Team-Level Cooperation in Multi-Agent Systems. CoRR. 2022;abs/2201.11994. <https://arxiv.org/abs/2201.11994>.
- [72] Agarwal A, Kumar S, Sycara KP, Lewis M. Learning Transferable Cooperative Behavior in Multi-Agent Teams. In: Proceedings of the 19th International Conference on

Distributed Reinforcement Learning for Robot Teams: A Review

- Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020; 2020. p. 1741–1743. Available from: <https://dl.acm.org/doi/abs/10.5555/3398761.3398967>.
- [73] Cao K, Lazaridou A, Lanctot M, Leibo JZ, Tuyls K, Clark S. Emergent Communication through Negotiation. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings; 2018. Available from: <https://openreview.net/forum?id=Hk6WhagRW>.
- [74] Shaw S, Wenzel E, Walker A, Sartoretti G. ForMIC: Foraging via Multiagent RL With Implicit Communication. IEEE Robotics Autom Lett. 2022;7(2):4877–4884. <https://doi.org/10.1109/LRA.2022.3152688>.
- [75] Ma Z, Luo Y, Pan J. Learning Selective Communication for Multi-Agent Path Finding. CoRR. 2021;abs/2109.05413. <https://arxiv.org/abs/2109.05413>.
- [76] Freed B, James R, Sartoretti G, Choset H. Sparse Discrete Communication Learning for Multi-Agent Cooperation Through Backpropagation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021; 2020. p. 7993–7998. Available from: <https://doi.org/10.1109/IROS45743.2020.9341079>.
- [77] Leibo JZ, Zambaldi VF, Lanctot M, Marecki J, Graepel T. Multi-agent Reinforcement Learning in Sequential Social Dilemmas. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017; 2017. p. 464–473. Available from: <http://dl.acm.org/citation.cfm?id=3091194>.
- [78] Claus C, Boutilier C. The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA; 1998. p. 746–752. Available from: <http://www.aaai.org/Library/AAAI/1998/aaai98-106.php>.
- [79] Pérolut J, Leibo JZ, Zambaldi VF, Beattie C, Tuyls K, Graepel T. A multi-agent reinforcement learning model of common-pool resource appropriation. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA; 2017. p. 3643–3652. Available from: <https://proceedings.neurips.cc/paper/2017/hash/2b0f658cbffd284984fb11d90254081f-Abstract.html>.
- [80] Wang WZ, Beliaev M, Biyik E, Lazar DA, Pedarsani R, Sadigh D. Emergent Prosociality in Multi-Agent Games Through Gifting. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021; 2021. p. 434–442. Available from: <https://doi.org/10.24963/ijcai.2021/61>.
- [81] Mihai D, Hare JS. Learning to Draw: Emergent Communication through Sketching. CoRR. 2021;abs/2106.02067. <https://arxiv.org/abs/2106.02067>.
- [82] Li F, Bowling M. Ease-of-Teaching and Language Structure from Emergent Communication. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada; 2019. p. 15825–15835. Available from: <https://proceedings.neurips.cc/paper/2019/hash/b0cf188d74589db9b23d5d277238a929-Abstract.html>.
- [83] Lewis M, Yarats D, Dauphin YN, Parikh D, Batra D. Deal or No Deal? End-to-End Learning for Negotiation Dialogues. CoRR. 2017;abs/1706.05125. <https://arxiv.org/abs/1706.05125>.

- [org/abs/1706.05125](https://arxiv.org/abs/1706.05125).
- [84] Noukhovitch M, LaCroix T, Lazaridou A, Courville AC. Emergent Communication under Competition. In: AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021; 2021. p. 974–982. Available from: <https://dl.acm.org/doi/10.5555/3463952.3464066>.
- [85] Liu S, Lever G, Wang Z, Merel J, Eslami SMA, Hennes D, et al. From Motor Control to Team Play in Simulated Humanoid Football. CoRR. 2021;abs/2105.12196. <https://arxiv.org/abs/2105.12196>.
- [86] Ding G, Koh JJ, Merckaert K, Vanderborght B, Nicotra MM, Heckman C, et al. Distributed Reinforcement Learning for Cooperative Multi-Robot Object Manipulation. In: Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020; 2020. p. 1831–1833. Available from: <https://dl.acm.org/doi/abs/10.5555/3398761.3398997>.
- [87] Cao Y, Sun Z, Sartoretti G. DAN: Decentralized Attention-based Neural Network to Solve the MinMax Multiple Traveling Salesman Problem. CoRR. 2021;abs/2109.04205. <https://arxiv.org/abs/2109.04205>.
- [88] Hu J, Zhang H, Song L, Schober R, Poor HV. Cooperative Internet of UAVs: Distributed Trajectory Design by Multi-Agent Deep Reinforcement Learning. IEEE Trans Commun. 2020;68(11):6807–6821. <https://doi.org/10.1109/TCOMM.2020.3013599>.
- [89] Fan T, Long P, Liu W, Pan J. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. Int J Robotics Res. 2020;39(7). <https://doi.org/10.1177/0278364920916531>.
- [90] Xiao Y, Hoffman J, Xia T, Amato C. Learning Multi-Robot Decentralized Macro-Action-Based Policies via a Centralized Q-Net. In: 2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020; 2020. p. 10695–10701. Available from: <https://doi.org/10.1109/ICRA40945.2020.9196684>.
- [91] Wang D, Deng H, Pan Z. MRCDRL: Multi-robot coordination with deep reinforcement learning. Neurocomputing. 2020;406:68–76. <https://doi.org/10.1016/j.neucom.2020.04.028>.
- [92] Sartoretti G, Kerr J, Shi Y, Wagner G, Kumar TKS, Koenig S, et al. PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning. IEEE Robotics Autom Lett. 2019;4(3):2378–2385. <https://doi.org/10.1109/LRA.2019.2903261>.
- [93] Damani M, Luo Z, Wenzel E, Sartoretti G. PRIMAL\$.2\$: Pathfinding Via Reinforcement and Imitation Multi-Agent Learning - Lifelong. IEEE Robotics Autom Lett. 2021;6(2):2666–2673. <https://doi.org/10.1109/LRA.2021.3062803>.
- [94] Marchesini E, Farinelli A. Centralizing State-Values in Dueling Networks for Multi-Robot Reinforcement Learning Mapless Navigation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2021, Prague, Czech Republic, September 27 - Oct. 1, 2021; 2021. p. 4583–4588. Available from: <https://doi.org/10.1109/IROS51168.2021.9636349>.
- [95] Huang Y, Wu S, Mu Z, Long X, Chu S, Zhao G. A multi-agent reinforcement learning method for swarm robots in space collaborative exploration. In: 2020 6th International Conference on Control, Automation and Robotics (ICCAR); 2020. p. 139–144.
- [96] He Z, Dong L, Song C, Sun C. Multi-agent Soft Actor-Critic Based Hybrid Motion Planner for Mobile Robots. CoRR. 2021;abs/2112.06594. <https://arxiv.org/abs/2112.06594>.

Distributed Reinforcement Learning for Robot Teams: A Review

- [97] de Witt CS, Peng B, Kamienny P, Torr PHS, Böhmer W, Whiteson S. Deep Multi-Agent Reinforcement Learning for Decentralized Continuous Cooperative Control. CoRR. 2020;abs/2003.06709. <https://arxiv.org/abs/2003.06709>.
- [98] Freed B, Kapoor A, Abraham I, Schneider JG, Choset H. Learning Cooperative Multi-Agent Policies with Partial Reward Decoupling. CoRR. 2021;abs/2112.12740. <https://arxiv.org/abs/2112.12740>.
- [99] García J, Fernández F. A comprehensive survey on safe reinforcement learning. *J Mach Learn Res.* 2015;16:1437–1480.
- [100] Shalev-Shwartz S, Shammah S, Shashua A. Safe, Multi-Agent, Reinforcement Learning for Autonomous Driving. CoRR. 2016;abs/1610.03295. <https://arxiv.org/abs/1610.03295>.
- [101] Zhang W, Bastani O. MAMPS: Safe Multi-Agent Reinforcement Learning via Model Predictive Shielding. CoRR. 2019;abs/1910.12639. <https://arxiv.org/abs/1910.12639>.
- [102] Savva M, Chang AX, Dosovitskiy A, Funkhouser TA, Koltun V. MINOS: Multimodal Indoor Simulator for Navigation in Complex Environments. CoRR. 2017;abs/1712.03931. <https://arxiv.org/abs/1712.03931>.
- [103] Erickson ZM, Gangaram V, Kapusta A, Liu CK, Kemp CC. Assistive Gym: A Physics Simulation Framework for Assistive Robotics. In: 2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020; 2020. p. 10169–10176. Available from: <https://doi.org/10.1109/ICRA40945.2020.9197411>.
- [104] Fan L, Zhu Y, Zhu J, Liu Z, Zeng O, Gupta A, et al. SURREAL: Open-Source Reinforcement Learning Framework and Robot Manipulation Benchmark. In: 2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings. vol. 87; 2018. p. 767–782. Available from: <http://proceedings.mlr.press/v87/fan18a.html>.
- [105] Freed B, Sartoretti G, Choset H. Simultaneous Policy and Discrete Communication Learning for Multi-Agent Cooperation. *IEEE Robotics Autom Lett.* 2020;5(2):2498–2505. <https://doi.org/10.1109/LRA.2020.2972862>.
- [106] Henderson P, Islam R, Bachman P, Pineau J, Precup D, Meger D. Deep Reinforcement Learning That Matters. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018; 2018. p. 3207–3214. Available from: <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16669>.
- [107] Polydoros AS, Nalpantidis L. Survey of Model-Based Reinforcement Learning: Applications on Robotics. *J Intell Robotic Syst.* 2017;86(2):153–173. <https://doi.org/10.1007/s10846-017-0468-y>.
- [108] Thuruthel TG, Falotico E, Renda F, Laschi C. Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators. *IEEE Trans Robotics.* 2019;35(1):124–134. <https://doi.org/10.1109/TRO.2018.2878318>.
- [109] Thananjeyan B, Balakrishna A, Rosolia U, Li F, McAllister R, Gonzalez JE, et al. Safety Augmented Value Estimation From Demonstrations (SAVED): Safe Deep Model-Based RL for Sparse Cost Robotic Tasks. *IEEE Robotics Autom Lett.* 2020;5(2):3612–3619. <https://doi.org/10.1109/LRA.2020.2976272>.
- [110] Zhang K, Yang Z, Basar T. Decentralized multi-agent reinforcement learning with networked agents:

- recent advances. Frontiers Inf Technol Electron Eng. 2021;22(6):802–814. <https://doi.org/10.1631/FITEE.1900661>.
- [111] Zhang K, Yang Z, Liu H, Zhang T, Basar T. Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. vol. 80; 2018. p. 5867–5876. Available from: <http://proceedings.mlr.press/v80/zhang18n.html>.