Deep Reinforcement Learning-Based Effective Coverage Control With Connectivity Constraints

Shaofeng Meng and Zhen Kan, *Member, IEEE*

***Abstract*—This work studies dynamic coverage control of a multi-agent system using deep reinforcement learning. Dynamic coverage control is a type of cooperative control which requires a multi-agent system to dynamically monitor an area of interest over time. To develop motion control laws, most of previous works highly rely on the knowledge of system models, such as the environment model and agent kinematics/dynamics. However, acquiring an accurate model can be restrictive and even impossible in many practical applications. Another challenge is that agent often has a limited communication capability in practice. Two agents may only exchange information when they are within a certain distance. To address these challenges, a multi-agent deep reinforcement learning (MADRL) based control framework is developed to enable agents to learn control policies directly from interactions with the environment to achieve dynamic coverage control while preserving network connectivity. The developed MADRL is model free and employs decentralized execution and centralized training, in which agents coordinate using only local information and do not need to know other agents' strategies at execution phase. Numerical simulations demonstrate the effectiveness of the developed control strategy.**

***Index Terms*—Coverage control, deep reinforcement learning, multi-agent systems.**

I. Introduction

With the goal of monitoring areas of interest, coverage control of a multi-agent system has been widely used in a variety of applications, such as exploration, reconnaissance, surveillance, and target tracking [1]–[3]. In literature, there are two main classes of coverage control: the static and dynamic coverage control. Static coverage control generally solves the optimal placement of agents such that areas of interest can be fully covered by the union of agents' sensing zones [4], while dynamic coverage control focuses on taking advantage of the mobility of agents to dynamically monitor areas of interest sufficiently well over time. That is, every area of interest needs to be monitored by mobile agents for a sufficient amount of time [5]. However, static coverage control implicitly assumes either a sufficient number of agents or small-scale areas of interest [6], [7], otherwise there is no guarantee that all of the areas can be fully covered. Such assumptions can be relieved in dynamic coverage control, which allows the monitoring of large areas of interest by using a small number of agents [8], [9]. Nevertheless, in either static or dynamic coverage control, the models of the environment and agent kinematics/dynamics are often required to develop motion control laws for the agents. Acquiring accurate models can be restrictive and even impossible in many practical applications. Hence, this work is particularly motivated to develop a model free framework that enable agents to learn directly from interactions with the environment to achieve dynamic coverage control.

Reinforcement learning (RL) is a sequential decision-making process in which an agent interacts with and learns from the environment. In RL, optimal policies can be learned without knowing the model of the environment and the system's complex behaviors [10]. However, traditional RL is mainly restricted to discrete action space. When considering continuous action space or high dimensional data, deep reinforcement learning (DRL) that integrates neural networks can provide promising solutions [11]. For instance, robots can learn the control policies directly from real-world camera inputs rather than by hand-engineered controllers or learning from low-dimensional features of robot states [12]–[14]. Since multiple agents are more capable of complex tasks than single agent, multi-agent deep reinforcement learning (MADRL) is a recent research focus, which finds applications in multi-player online games [15], cooperative robots [16], [17], traffic control [18], [19], and power control systems [20], [21]. RL has also been applied for dynamic coverage control problems. Trajectory planning of multiple dronecells in vehicular networks was investigated in [22] via Q-learning; however, it can not deal with complex and continuous tasks. The works of [23], [24] adopted a centralized RL approach for a multi-agent system to achieve coverage goal and thus has limited scalability.

In this work, a small group of mobile agents with limited sensing capabilities is tasked with the objective of dynamically monitoring an area populated with a large number of points of interest. Due to limited sensing and limited number of agents, we are particularly interested in effective coverage control which aims to sense every point of interest for a sufficient amount of effort accumulatively. Each agent is further assumed to have a limited communication capability,

Manuscript received January 7, 2021; revised March 14, 2021; accepted March 31, 2021. Date of publication April 5, 2021; date of current version June 24, 2021. Recommended by Senior Editor M. Guay. *(Corresponding author: Zhen Kan.)*

The authors are with the Department of Automation, University of Science and Technology of China, Hefei 230026, China (e-mail: zkan@ustc.edu.cn).

Digital Object Identifier 10.1109/LCSYS.2021.3070850

i.e., two agents can only communicate when they are within a certain distance. Previous work has shown that maintaining the connectivity of agent networks is crucial for multi-agent distributed control, especially, in the dynamic coverage problem [7], [25], [26]. Therefore, the motion of agents needs to be constrained such that the underlying communication network is always connected to enable information exchange and cooperative coverage control. Deep reinforcement learning based framework is then developed to enable agents to learn control policies directly from interactions with the environment to achieve dynamic coverage control while preserving network connectivity. Numerical simulations demonstrate the effectiveness of the DRL-based control framework.

The main contributions of our work can be summarized as follows.

1) In contrast to traditional coverage control that relies on system models to develop motion control strategies, the DRL based control framework in this work is model free, where agents learn optimal control policies via interactions with the environment, without requiring any knowledge of system models.
2) We develop a MADRL-based algorithm to realize dynamic coverage control, where agents coordinate using only local information and do not need to know other agents' strategies at execution phase. As an extension of actor-critic policy gradient method, the multi-agent deep deterministic policy gradient (MADDPG) [27] is adapted for dynamic coverage control. To ensure network connectivity, we exploit the fact that algebraic connectivity can indicate network connectivity and integrate it into the MADRL-based algorithm as a learning objective. The DRL-based control framework employs decentralized execution and centralized training. We further improve MADDPG by setting virtual boundaries for the environment to improve the convergence of the training process.

The remainder of the paper is organized as follows: Section II formulates the dynamic effective coverage problem. Section III presents deep reinforcement learning based control strategy for effective coverage and maintenance of network connectivity. Simulation results and analysis are provided in Section IV. Section V discusses potential extensions.

II. Problem Formulation

Consider M static points of interest (PoIs) distributed in a two-dimensional workspace $\mathcal{W}$. The positions of PoIs are denoted by $p_j \in \mathbb{R}^2, j \in \{1, \dots M\}$. A group of N mobile agents is tasked to keep monitoring these PoIs and the agent positions are denoted by x_i, $i \in \{1, \dots N\}$.

Assumption 1: The agent is assumed to have a limited sensing zone, modeled by a disk area with radius $r \in \mathbb{R}^+$ centered at itself.

Since the sensing quality generally degrades with the distance to the PoI, the distance based sensing of agent i over PoI j is characterized by

$$\mathcal{P}_i(x_i(t), p_j) = \begin{cases} \frac{M_P}{r^4}(l_{ij}(t) - r^2)^2, & l_{ij} \le r^2, \\ 0, & l_{ij} > r^2, \end{cases} \quad (1)$$

TABLE I
NOTATIONS

Notation	Explanation
M, N	Number of PoIs and agents
T	Total number of steps in one episode
$\mathcal{P}, R$	Sensing quality and communication capability
$E_{\mathcal{V},j}$	Cumulative coverage of the PoI j by the entire group of agents
o_i^t, a_t^i, r_t	Observation, action and reward of agent i at t
λ_2	Second smallest eigenvalue of Laplacian matrix
M^*, T_c	Number of PoIs have been covered and the time duration that the group is connected in the episode
$M^*/M, T_c/T$	Coverage rate and connection rate in an episode

Fig. 1. An example problem scenario of dynamic coverage control, where a group of 4 UAVs is tasked to dynamically monitor 20 PoIs.

where $l_{ij}(t) \triangleq \|x_i(t) - p_j\|^2$, and $M_p \in \mathbb{R}^+$ is a peak sensing quality. The sensing model in (1) indicates that the sensing quality of PoI j reaches the peak when agent i coincides with it, and decreases monotonically when agent i moves away from it.

In this work, we consider the case that only a limited number of mobile agents are tasked to monitor a large number of PoIs within $\mathcal{W}$. That is, the union of sensing zones of N agents are far from enough to completely cover M PoIs, otherwise existing methods in literature can be trivially applied. Hence, we aim to take advantage of the agent mobility and develop a cooperative motion strategy to dynamically monitor the PoIs. Detailed definitions of the considered dynamic coverage are provided in Section III-A. To enable coordination, the interaction between agents (i.e., communication and information exchange) is modeled as an undirected graph $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V}$ and $\mathcal{E}(t)$ denote the set of agents and the set of communication links, respectively.

Assumption 2: The agent is assumed to possess a limited communication capability represented by a circular area with radius R, i.e., two agents can only exchange information within the distance of R.

Therefore, the edge $(i, k) \in \mathcal{E}$ is established if and only if $R_{ik} = \|x_i - x_k\|_2 \le R$. The neighbors of agent i are denoted by $\mathcal{N}_i = \{k \in \mathcal{V} | (i, k) \in \mathcal{E}\}$. The graph $\mathcal{G}$ is connected if there exists a path connecting any two nodes.

Example 1: Consider an example problem scenario in Fig. 1, which consists of 20 PoIs required to be dynamically monitored by a group of agents (i.e., UAVs). The UAV has a limited sensing capability and the mapped sensing zone on the ground is a disk area. The lines connecting

UAVs indicate the communication links between UAVs. UAVs are required to dynamically monitor all PoIs while ensuring network connectivity to enable information exchange and coordination.

In contrast to most works that rely on system models, we do not assume any knowledge of system models. Hence, the goal of this work is to develop a DRL-based framework such that the agents can learn the optimal control policies for dynamic coverage control and maintenance of network connectivity only by interactions with the environment and other agents.

III. Deep Reinforcement Learning Based Multi-Agent Effective Coverage

A. Effective Coverage

Inspired by our early result [28], the DRL-based effective coverage control in this work is to drive mobile agents to ensure that every PoI in the region of interest is cumulatively covered to a desired level over time. Specifically, the effective coverage imposed by agents i over PoI j is defined as

$$E_{i,j} \triangleq \int_0^t \mathcal{P}_i\big(x_i(\tau), p_j\big)d\tau, \tag{2}$$

which quantifies the cumulative coverage exerted by agents i on PoI j over $[0, t]$. Based on the individual coverage defined in (2), the cumulative coverage of the PoI j by the entire group of agents is defined as

$$E_{\mathcal{V},j} \triangleq \int_0^t \sum_{i\in\mathcal{V}} \mathcal{P}_i\big(x_i(\tau), p_j\big)d\tau. \tag{3}$$

It is worth noting that each PoI j is associated with a desired coverage $E_j{}^* \in \mathbb{R}^+$, which indicates how much coverage effort should be applied to it.

Assumption 3: For any PoI j, its desired coverage level $E_j{}^*$ is assumed to be known and predetermined.

The coverage task for PoI j is completed if $E_{\mathcal{V},j} \geq E_j{}^*$. Therefore, the dynamic effective coverage task is to ensure each PoI in the workspace is accumulatively sensed to the expected coverage level over time.

As mentioned before, a connected network $\mathcal{G}(t)$ ensures the information exchange between agents; however, due to limited communication capabilities, the movement of agents may lead to the disconnection of the network $\mathcal{G}(t)$. Let $\mathcal{A}(t) \in \mathbb{R}^{N\times N}$ be the adjacency matrix of $\mathcal{G}(t)$, where each entry $\mathcal{A}_{ik}(t)$ is the edge weight between the agents i and k. Due to the consideration of undirected graphs, we have $\mathcal{A}_{ik}(t) = \mathcal{A}_{ki}(t)$. The Laplacian matrix of $\mathcal{G}(t)$ can then be defined as $\mathcal{L}(t) = \mathcal{D}(t) - \mathcal{A}(t)$, where $\mathcal{D}(t) = \text{diag}\,[\sum_{k=1}^{N} \mathcal{A}_{ik}(t)]$ is a diagonal matrix. It is well known that $\lambda_2(t) > 0$ if and only if the graph $\mathcal{G}(t)$ is connected, where λ_2 is called the algebraic connectivity, i.e., the second smallest eigenvalue of $\mathcal{L}(t)$ [29]. Therefore, the goal of this work in Section II can be reformulated to achieve $E_{\mathcal{V},j} \geq E_j{}^*$ for any PoI j (i.e., effective coverage control) while ensuring $\lambda_2(t) > 0$ (i.e., network connectivity).

B. DRL-Based Algorithm

Q-learning is a classical method in reinforcement learning. When considering Multi-agent RL, since agents update their strategies independently during learning progresses, the environment is non-stationary from the perspective of any agent, which makes traditional Q-learning no longer guarantees convergence in multi-agent cases. In addition, both Q-learning and Deep Q-Networks (DQN) are difficult to apply to continuous state-action space since they need to find the optimal policies greedily at every step, which is impossible when the action space is continuous. Policy gradient methods are another class of popular RL algorithms which directly adjust the parameter of the policy to maximize the expected cumulative reward by applying gradient ascent. However, traditional policy gradient method updates its parameter in every episode, which undoubtedly reduces the learning speed. Deep deterministic policy gradient (DDPG) [30] is a variant of Actor-Critic (AC) method, which inherits the target network and replay buffer in DQN and uses deterministic policy updating method to yield continuous control actions. In this section, a DRL-based distributed multi-agent control strategy is developed and each agent learns via interactions with the environment and other agents to achieve dynamic effective coverage. Inspired by [27], the framework of centralized training and distributed implementing is adopted, in which the agents learn the policy only based on their local observation of the environment. Since each agent can only observe local environment information and does not have access to other agents' strategies at execution phase, the dynamic effective coverage is modeled by POMDP as follows.

1) Observation Space $\mathcal{O}$ and State Space S: Let the observation of agent i at time t be its current position in the workspace, i.e., $o_i^t = \{x_i^t, y_i^t\}$, and thus the observation space is $\mathcal{O} \triangleq \{o_i^t\}$ with $i = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$. The state space is defined as $\mathcal{S} \triangleq \mathcal{O} \cup \{E_{\mathcal{V},j}^t(p_j)\}$, $j = 1, 2, \ldots, M$, which includes the coverage effort up to time t and the observation space $\mathcal{O}$.
2) Action Space A: The agent's action at time t is jointly determined by the motion direction θ_t^i and the distance d_t^i traveled along this direction, i.e., $a_t^i = \{\theta_t^i, d_t^i\}$. The action space is defined as $A \triangleq \{a_i^t\}$, $i = 1, 2, \ldots, N$, $t = 1, 2, \ldots, T$.
3) Reward Function: The immediate reward function r_t^i consists of two components: the coverage reward and the connectivity penalty. Note that all agents have the same reward function due to the consideration of cooperative tasks. The coverage reward is designed as

$$c_t^j = \begin{cases} c^*, & E_{\mathcal{V},j}^t \geq E_j{}^*, \\ 0, & E_{\mathcal{V},j}^t < E_j{}^*, \end{cases} \tag{4}$$

where c^* is a predetermined positive constant. The group of agents receive positive rewards c_t^j if the PoI j has been covered to the desired level $E_j{}^*$, otherwise no reward is given to the group. Since a positive algebraic connectivity implies a connected network, the connectivity penalty is defined as

$$p_t = \begin{cases} p^*, & \lambda_2(\mathcal{L}(t)) \leq 0, \\ 0, & \lambda_2(\mathcal{L}(t)) > 0, \end{cases} \tag{5}$$

where p^* is a predetermined negative constant. A penalty will be given to the group if the communication network becomes disconnected. Based on (4) and (5), the reward function is designed as

$$r_t = \sum_{j\in\{1,\ldots M\}} c_t^j + p_t. \tag{6}$$

The DRL based algorithm is outlined in Algorithm 1. It is worth pointing out that the learning algorithm is an actor-critic approach and distributed in the sense that all agents only use their own local information to update the policies. By modeling the dynamic coverage as a POMDP, an extension of actor-critic policy gradient method, namely multi-agent deep deterministic policy gradient (MADDPG), is developed. Specifically, the agent's policies $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_N\}$ are parameterized by $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_N\}$. To handle continuous action space, the deterministic policy gradient is developed as

$$\nabla_{\theta_i} J(\mu_i) = \mathbb{E}_{o,a\sim\mathcal{D}}\big[\nabla_{\theta_i}\mu_i(a_i|o_i) \cdot \nabla_{a_i}\mathcal{Q}_i^{\mu}(S, a_1, \ldots, a_N)|_{a_i=\mu_i(o_i)}\big], \tag{7}$$

where the experience replay buffer $\mathcal{D}$ contains the tuples $(S, S^{'}, A, r)$, which stores experiences of all agents.

The critic network of agent i is updated by minimizing the loss function

$$L(\theta_i) = \mathbb{E}_{\mathcal{D}}\Bigg[\bigg(\mathcal{Q}_{\rangle}^{\mu}(S, a_1, \ldots, a_N) - \Big(r + \gamma\mathcal{Q}_{\rangle}^{\mu^{'}}\big(S^{'}, a_1^{'}, \ldots, a_N^{'}\big)|_{a_j^{'}=\mu_j^{'}(o_j)}\Big)\bigg)^2\Bigg], \tag{8}$$

where agent i only uses its own observation o_i to update its policy in (7), and require policies of other agents to update action-value function $\mathcal{Q}_i^{\mu}$ in (8). In addition, the weights of the target network are slowly updated with a discount $\tau = 0.01$.

To improve the training stability and efficiency, virtual boundaries of the training environment are considered. We notice that in the training process, especially in the first few time steps, agents may choose wrong directions and thus move away from the target region. To address this issue, we set virtual boundaries for the workspace, that is, reset the current episode and remove it from experience replay buffer $\mathcal{D}$ if agents reaches the boundaries. It is shown (in simulation section) that the use of boundaries can improve data efficiency in the experience replay buffer and thus lead to faster training.

Since agents take the observations of environment information as input and utilize the actor network $\mu_i(a_i|o_i)$ to generate actions, the input space of $\mathcal{Q}$ increases linearly with the number of agents N, and the total computational complexity of the fully connected layer is $\mathcal{O}(\sum_{p=1}^{P} n_p * n_{p-1})$, where n_p is the number of neural units in the fully connected layer.

IV. Simulation and Discussion

In this section, the performance of the proposed DRL based dynamic effective coverage is evaluated via numerical simulations. The algorithm is implemented using TensorFlow 2.3.1 and Python 3.5.4 on Ubuntu 16.04. The simulation code of the algorithm is available in our Github repository.[1] The workspace is set as 10×10 units, while its virtual boundary is set at 30×30 units. Suppose there are 20 static PoIs and 4 mobile agents randomly distributed within the area at the beginning of each episode. The mission of the agents is to learn the optimal actions (travel distances and directions) via interactions with the environment and other agents such that all PoIs can be effectively covered for a given period of time T while ensuring network connectivity during mission operation. The sensing radius of each agent is $r = 0.8$ units and the communication range is $R = 2$ units. The peak of sensing quality is $M_p = 1$. Without loss of generality, suppose that the PoIs are equally important, e.g., we set $E_j^* = 4$, $\forall j \in \{1, \ldots, N\}$ in the simulation. The agents receive rewards $c^* = 0.1$ if they complete the effective coverage task, and the agents are penalized by -1 if they lose network connectivity.

Algorithm 1 MADRL-Based Dynamic Coverage Algorithm

Initialize experience replay buffer $\mathcal{D}$.
for agent $i = 1, 2, \ldots, N$ **do**
  Randomly initialize critic network $\mathcal{Q}_i^{\mu}(S, a_1, \ldots, a_N)$ and actor network $\mu_i(a_i|o_i)$;
  Initialize target networks $\mathcal{Q}_{\rangle}^{\mu^{'}}$ and $\mu_i^{'}$;
end for
for episode = 1 to M **do**
  Initialize a random process $\mathcal{N}$ for action exploration;
  Receive initial state s_1;
  for $t = 1$ to T **do**
    for each agent i, select action $a_i = \mu_{\theta_i}(o_i) + \mathcal{N}_{\sqcup}$ w.r.t. the current policy and exploration;
    Execute actions $a = a_1, \ldots, a_N$ and observe reward r and new state $s^{'}$;
    for agent $i = 1, 2, \ldots, N$ **do**
      if agent i reaches boundary **then**
        Terminate episode and reset environment;
      end if
    end for
    Store $(s, s^{'}, a, r)$ in $\mathcal{D}$ and $s \leftarrow s^{'}$;
    for agent $i = 1, 2, \ldots, N$ **do**
      Sample a random minibatch of S samples $(s^j, s^{'j}, a^j, r^j)$ from $\mathcal{D}$;
      Update critic by minimizing the loss $L(\theta_i)$ in (8);
      Update actor using policy gradient $\nabla_{\theta_i} J(\mu_i)$ in (7);
    end for
    Update target network parameters for each agent i:

$$\theta_i^{'} \leftarrow \tau\theta_i + (1-\tau)\theta_i^{'}$$

  end for
end for

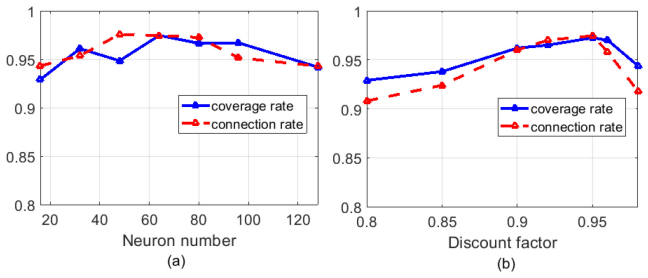

Fig. 2. The performance of coverage rate and connection rate under different hyper-parameters

[1] https://github.com/Sherry-97/Deep-Reinforcement-Learning-Based-Effective-Coverage-Control-with-Connectivity-Constraints

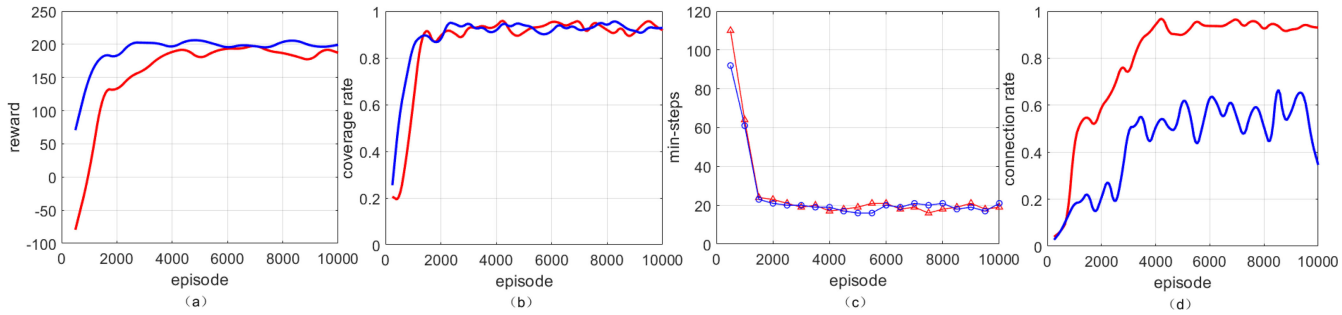

Fig. 3. The training curve of agent under CCA (red line) and NCCA (blue line), where the x-axis is episode, and y-axis is the reward, coverage rate, minimum steps, and connection rate, respectively.

The proposed model is trained for 10000 episodes, where each episode has 120 time steps.

To demonstrate the effectiveness of the DRL based coverage control, the evolution of accumulative rewards over time is shown in Fig. 3(a), where the discount factor $\gamma = 0.95$ is adopted. It is observed that, at the beginning of learning, the cumulative reward is negative and increases rapidly with time. This is because that, in the initial stage of training, most PoIs were not yet covered since agents did not learn the appropriate trajectories to provide enough coverage. In addition, random explorations of agents in reinforcement learning may cause the loss of network connectivity, resulting in punishments on the rewards. As the training continues, agents gradually learn how to effectively complete the coverage task while preserving network connectivity, which results in increasing gains of positive rewards. It is worth mentioning that due to the consideration of non-stationary environment, there are some fluctuations in the reward function curve after reaching a certain level.

Next, simulations are carried out to show how the hyperparameters (i.e., the discount factor and the number of neurons) affect the performance of the algorithm. The simulation results are shown in Fig. 2. We consider the following metrics. Let M^*/M denote the coverage rate at the end of an episode, where M^* is the number of PoIs that have reached the desired coverage level. Clearly, the coverage rate M^*/M directly indicates how well the dynamic effective coverage task has been completed in the episode. Let T_c/T denote the connection rate of the underlying communication network in an episode, where T_c is the number of time steps that the network of agent group is connected in this episode and T is the total number of time steps. We first show that, with the same discount factor, how the number of neurons affect the performance of the algorithm. As shown in Fig. 2(a), the coverage rate and connection rate first increase and then decrease as the neuron numbers increase and both shows the best performance with 64 neurons. It indicates that using more neurons can help learn the complex nonlinear relationship among state, action and reward; however, using too many neurons may lead to poor performance due to over fitting problem. Fig. 2(b) shows how the discount factor affects the performance of the algorithm when the number of neurons is fixed at 64. Both coverage rate and connection rate first increase and then decrease as γ increases and both reach the peak when γ is between 0.95 and 0.96. The reason that small γ does not perform well is that agents tend to be myopic with short-horizon rewards without considering future rewards. If a large γ is used, future rewards will be overwhelmed valued, resulting in the overlook of short-horizon rewards. Hence, γ needs to be fine tuned to yield desired performance.

To further demonstrate the performance of the proposed DRL-based algorithm for dynamic effective coverage task and maintenance network connectivity of agents, we investigate the training process of our algorithm and show the comparisons of the results with different settings. It is worth pointing out this work is one of the first attempts in literature to address network connectivity constraints using deep reinforcement learning algorithms. Therefore, we are interested in comparing the results with connectivity-constrained algorithm (CCA) and the results with non-connectivity-constrained algorithm (NCCA). For NCAA, agents get no penalty even if they lose network connectivity.

To evaluate its performance, in addition to coverage rate and connection rate, we also consider the min-steps as a performance metric, which is defined as the minimum steps required to complete the coverage task. In the training process, CCA and NCCA are compared in terms of reward function, coverage rate, min-steps, and the connection rate, respectively. It can be seen from the reward function curve in Fig. 3(a) that the training curve of CCA grows slower than the training curve of NCCA in the first 4000 episodes. This is because at this stage, as for CCA, agents aim to learn policies that make trade-off between dynamic coverage tasks and connectivity maintenance, which can also be seen in the curve of coverage rate and connection rate. The reward function will reach the same level as NCCA in the later stage after agents learned satisfactory policies which take two missions into consideration. From Fig. 3(b), the coverage rate of CCA is a little behind NCCA in the initial 1000 episodes. The coverage rate of CCA reaches the same level as NCCA in the later stage. That is, agents under CCA can learn dynamic coverage policies as effectively as under NCCA. In Fig. 3(c), the min-step on the other hand illustrates the efficiency of the agent to complete the dynamic coverage task. It can be seen that at the beginning of the training, agents need a long time or even cannot complete the task, while in the later stage of training, only a dozen steps are needed. In the whole training process, two curves maintain the same level, which also shows that agents under CCA can learn policies to complete coverage task well as NCCA. In addition, the network connection rate

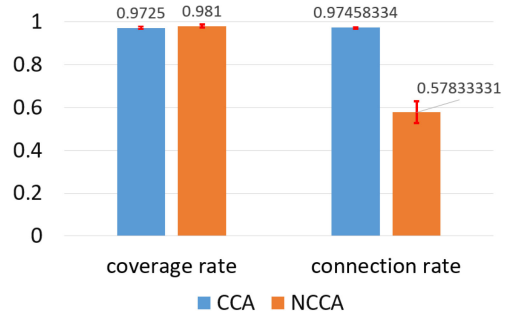

Fig. 4. Comparison of networks trained by CCA and NCCA about coverage rate and connection rate.

under NCCA as shown in Fig. 3(d) maintains at a relatively low level with great fluctuation, while the network connection rate under CCA achieves satisfactory convergence and achieves a high connection rate in the late training period.

Finally, we test the trained optimal networks produced from the DRL-based algorithm under CCA and NCCA, respectively, for 100 episodes. The comparisons of the coverage rate and the network connection rate are shown in Fig. 4. It can be seen that there is only negligible difference in coverage rate between CCA and NCCA, and both of them reach a high degree of completion. In addition, the test results of network trained using CCA show that the connection rate is about 97.5% with a small variance. However, the network trained using NCAA gains poor performance in maintaining connectivity of agent group, whose connection rate is only 57.8% with a large variance. This result indicates once again that our algorithm enables agents learning policies to ensure network connectivity while not affecting its coverage performance.

V. Conclusion

This work investigates network connectivity ensured dynamic coverage algorithm based on deep reinforcement learning. Simulation results indicate that our algorithm can complete the dynamic coverage task well while ensuring the connectivity of agent group. To better elaborate the practical utility of our method, additional work will consider more comparisons with model-based or approximate-model-based methods. Future research will also consider extending current results to more general environments, e.g., considering mobile PoIs, and focus on developing theoretical guarantees for the performance of deep RL algorithms.

References

[1] X. Kan, H. Teng, and K. Karydis, "Online exploration and coverage planning in unknown obstacle-cluttered environments," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5969–5976, Oct. 2020.

[2] M. Khaledyan, A. P. Vinod, M. Oishi, and J. A. Richards, "Optimal coverage control and stochastic multi-target tracking," in *Proc. IEEE Conf. Decis. Control*, 2019, pp. 2467–2472.

[3] S. Gao and Z. Kan, "Effective dynamic coverage control for heterogeneous driftless control affine systems," *IEEE Control Syst. Lett.*, vol. 5, no. 6, pp. 2018–2023, 2021.

[4] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Autom.*, vol. 20, no. 2, pp. 243–255, Apr. 2004.

[5] B. Liu, O. Dousse, P. Nain, and D. Towsley, "Dynamic coverage of mobile sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 2, pp. 301–311, Feb. 2013.

[6] M. Schwager, D. Rus, and J.-J. Slotine, "Unifying geometric, probabilistic, and potential field approaches to multi-robot deployment," *Int. J. Robot. Res.*, vol. 30, no. 3, pp. 371–383, 2011.

[7] M. Zhong and C. G. Cassandras, "Distributed coverage control and data collection with mobile sensor networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2445–2455, Oct. 2011.

[8] D. E. Soltero, M. Schwager, and D. Rus, "Decentralized path planning for coverage tasks using gradient descent adaptive control," *Int. J. Robot. Res.*, vol. 30, no. 3, pp. 1–25, 2014.

[9] Y. Wang and I. I. Hussein, "Awareness coverage control over large-scale domains with intermittent communications," *IEEE Trans. Autom. Control*, vol. 55, no. 8, pp. 1850–1859, Aug. 2010.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[11] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[12] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 421–436, 2018.

[13] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[14] M. Cai, M. Hasanbeig, S. Xiao, A. Abate, and Z. Kan, "Modular deep reinforcement learning for continuous motion planning with temporal logic," 2021. [Online]. Available: arXiv:2102.12855.

[15] O. Vinyals *et al.*, "Grandmaster level in starcraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[16] J. K. Gupta, M. Egorov, and M. J. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Int. Conf. Auton. Agents Multiagent Syst.*, 2017, pp. 66–83.

[17] C. Yu, Y. Dong, Y. Li, and Y. Chen, "Distributed multi-agent deep reinforcement learning for cooperative multi-robot pursuit," *J. Eng.*, vol. 2020, no. 13, pp. 499–504, 2020.

[18] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans.Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.

[19] T. Wu *et al.*, "Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8243–8256, Aug. 2020.

[20] M. K. Sharma, A. Zappone, M. Assaad, M. Debbah, and S. Vassilaras, "Distributed power control for large energy harvesting networks: A multi-agent deep reinforcement learning approach," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1140–1154, Dec. 2019.

[21] S. Wang *et al.*, "A data-driven multi-agent autonomous voltage control framework using deep reinforcement learning," *IEEE Trans. Power Syst.*, vol. 35, no. 6, pp. 4644–4654, Nov. 2020.

[22] M. Samir, D. Ebrahimi, C. Assi, S. Sharafeddine, and A. Ghrayeb, "Trajectory planning of multiple dronecells in vehicular networks: A reinforcement learning approach," *IEEE Netw. Lett.*, vol. 2, no. 1, pp. 14–18, Mar. 2020.

[23] C. H. Liu, Z. Chen, J. Tang, J. Xu, and C. Piao, "Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 2059–2070, Sep. 2018.

[24] M. S. Shokry, D. Ebrahimi, C. Assi, S. Sharafeddine, and A. Ghrayeb, "Leveraging uavs for coverage in cell-free vehicular networks: A deep reinforcement learning approach," *IEEE Trans. Mobile Comput.*, early access, Apr. 29, 2020, doi: 10.1109/TMC.2020.2991326.

[25] Y. Kantaros and M. M. Zavlanos, "Distributed communication-aware coverage control by mobile sensor networks," *Automatica*, vol. 63, pp. 209–220, Jan. 2016.

[26] Z. Kan, A. P. Dani, J. M. Shea, and W. E. Dixon, "Network connectivity preserving formation stabilization and obstacle avoidance via a decentralized controller," *IEEE Trans. Autom. Control*, vol. 57, no. 7, pp. 1827–1832, Jul. 2012.

[27] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Adv. Neural Inf. Process. Syst.*, 2017, pp. 6379–6390.

[28] Z. Kan, J. M. Shea, E. A. Doucette, J. W. Curtis, and W. E. Dixon, "Coverage control based effective jamming strategy for wireless networks," in *Proc. Amer. Control Conf.*, 2016, pp. 4655–4660.

[29] C. Godsil and G. Royle, *Algebraic Graph Theory*. Cham, Switzerland: Springer, 2001.

[30] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Int. Conf. Learn. Represent.*, San Juan, Puerto Rico, 2016, pp. 1–14.