Computer Science and Information Systems 20(3):1061–1084 https://doi.org/10.2298/CSIS221222037A

Point of Interest Coverage with Distributed Multi-Unmanned Aerial Vehicles on Dynamic Environment

Fatih Aydemir[1,2] and Aydin Cetin[2]

[1] STM Defence Technologies Engineering and Trade. Inc.
06530 Ankara Turkiye
fatih.aydemir@stm.com.tr
[2] Gazi University, Faculty of Technology, Department of Computer Engineering
06560 Ankara Turkiye
acetin@gazi.edu.tr

Abstract. Mobile agents, which learn to optimize a task in real time, can adapt to dynamic environments and find the optimum locations with the navigation mechanism that includes a motion model. In this study, it is aimed to effectively cover points of interest (PoI) in a dynamic environment by modeling a group of unmanned aerial vehicles (UAVs) on the basis of a learning multi-agent system. Agents create an abstract rectangular plane containing the area to be covered, and then decompose the area into grids. An agent learns to locate on a center of grid that are closest to it, which has the largest number of PoIs to plan its path. This planning helps to achieve a high fairness index by reducing the number of common PoIs covered. The proposed method has been tested in a simulation environment and the results are presented by comparing with similar studies. The results show that the proposed method outperforms existing similar studies and is suitable for area coverage applications.

Keywords: unmanned aerial vehicle, multi-agent system, reinforcement learning, dynamic-area coverage, grid decomposition.

1. Introduction

1.1. Background, Definition, and Motivation

Point of interest coverage (PoI) is a problem that has a huge scope, where a group of agents is tasked with exploring and mapping [16] an unknown environment while also monitoring [9] as many PoI as possible. The aim of the PoI coverage is to ensure that a group of agents visit and monitor predefined locations, known as PoIs. These agents, which could be robots, drones, or even people, should move across a dynamic environment in order to cover a set of PoI. Topics such as how much of the PoI is monitored, how fast the process is managed, and how high the fault tolerance is, determine the quality of coverage. In applications where non-mobile agents are used, the agent remains stable after initial setup. This makes it difficult to adapt to changes in the environment and reduces the fault tolerance of the coverage process. It is needed the mobile agent to be designed with complex control modules if a single mobile agent is used. However, the same capability can be achieved with the cooperation of a group of simple mobile agents.

A group of mobile agents can be designed as a multi-agent system (MAS) that handle coverage problem in a coordinated, interactive, dependent, or independent manner [5]. These systems have defined a collection of agents that produce a common goal-oriented behavior called "collective intelligence" [30]. Methods developed with collective intelligence which are based on centralized algorithms can be computationally expensive and inflexible in dynamic environments. There has been growing curiosity about using distributed multi-agent reinforcement learning (MARL) to deal with the PoI coverage problem [32]. In MARL, each agent learns to make decisions independently, based on its local observations and interactions with the environment. This helps to create more scalable and adaptable solutions since the agents learn to coordinate and collaborate without the need for a centralized controller.

One key challenge in using MARL for PoI coverage is the dynamic nature of the environment [3]. MARL algorithms can be developed to be responsive, meaning that they can adjust their behavior based on the current state of the environment. The requirement for effective coordination among the agents presents another challenge when employing MARL for PoI coverage. For the agents to be able to coordinate their actions in a way that is both effective and efficient, it is necessary to design sophisticated communication and coordination algorithms. Although there are some challenges, MARL can enable to design of effective approaches for solving the PoI coverage problem. By using methods based on MARL, a group of agents can learn to work together and coordinate in order to cover the maximum number of PoIs in an area, even when the environment is changing and unpredictable.

1.2. Literature Review

Area coverage-based studies have been carried out by deployment strategies in order to solve problems such as maximum PoI coverage and maximum area surveillance. Mozaffari et al. [21] have investigated how effectively multiple unmanned aerial vehicles (UAVs) could be deployed to serve as wireless base stations and cover ground users. In this context, the authors proposed a 3-dimensional (3-D) deployment strategy based on circle packing theory to maximize area coverage while increasing the lifetime of the network. In [33], an algorithm based on the Artificial Bee Colony (ABC) algorithm has been applied for increasing area coverage and quality of connectivity between sensor nodes. Experimental results were compared with random distribution and Genetic Algorithm. According to the results, the proposed algorithm has a longer lifetime of communication and a higher coverage rate than the others. Gupta et al. [10] studied sensor node placement using a genetic algorithm-based approach that all target points k-coverage and sensor nodes m-connected. Kalantari et al. [14] tried to find the number of UAV base stations needed to cover an area by considering the system requirements and constraints of the environment. The researchers proposed a heuristic algorithm based on Particle Swarm Optimization to locate base stations in a 3-D plane [25]. Njoya et al. [22] proposed a hybrid method that aims to spread sensor nodes effectively considering connectivity between sensor nodes to cover the target area. Jagtap and Gomathi [12] have studied the Euclidean Spanning Tree Model (ECST) and ECST-adaptive velocity added (VABC) (ECST-AVABC) methods for solving the mobile sensor deployment problem, which includes target coverage and network connectivity. Additionally, the researchers designed an AVABC optimization method by acquiring the least amount of movement of mobile sensors through the network. In [26], a novel method for increasing the WSN's coverage and longevity while preserving network connectivity was presented. The proposed method GCVD, which is based on the Voronoi diagram (VD) and the geometric center (GC), allows mobile sensors to only move within a predefined distance. This movement strategy helps maximize area coverage with minimal energy consumption while guaranteeing the connectivity of nodes. In [15], a deployment strategy using IoT devices association policy has been investigated for fixed-wing UAVs while gathering data from IoT devices. It was assumed that each UAV follows a circular path over devices associated with it. The problem of device association was designed as a Multi-Knapsack Problem that takes into account the service capacities of UAVs as well as the load requests of devices. Ganganath et al. [7] have studied two different methods based on anti-flocking for dynamic coverage with mobile sensor networks in environments with and without obstacles.

The studies that designed with collaborative behavior strategies have been studied, where it was recommended an online coverage algorithm. In [31], the area coverage problem was handled using a target assignment with path planning approach, which has online execution process. Liu et al. [18] proposed a learning method called DRL-EC3 in order to perform the UAV positioning and orienting process with less energy consumption. This method was designed based on the deep deterministic policy gradient (DDPG) [17] algorithm that single actor-critic network was used in the training phase. An improved version of the DRL-EC3 model was presented to increase fault tolerance for dynamic changes in the environment [19]. In this improved model, each UAV had its own actor-critic network that helps find out convenient actions to maximize area coverage with minimum energy consumption. In the target area, it was also aimed to establish a network between the UAVs performing distributed manner. Nemer et. al. [23] designed an actor-critic method on the bases of a state-based potential game (SBG-AC) that guides UAVs in moving from the initial position to the final position to obtain maximum PoI coverage with minimum energy consumption. The method has a similar motion model with the improved DRL-EC3 both methods use the distributed version of DDPG. Cabreira et al. [2] proposed a coverage path planning algorithm based on the grid-based approach presented in [29]. The original cost function was replaced with an energy-cost function to minimize energy consumption while mapping an area with UAVs. Aydemir and Cetin [1] proposed a method to maximize the covered area in a dynamic environment using multi-UAVs. Agents modeled with deep reinforcement learning techniques produced a model-free policy using a central module in the learning phase, but the central module was not used during the execution phase. The proposed method built as distributed system aims to place the active agents at the most appropriate points in the target area by creating a graph.

In agent-based methods, mobile agents focus on improving the coverage process by spreading over the area. The main difference between them is how the motion model is designed. In systems managed by a central controller, the process is managed from a single point, and errors that may occur in the centralized controller directly affect the entire system. Heuristic methods can optimize the dynamic range coverage process, but agents do not include the actions of other agents in the optimization process. With this approach, it may take a long time to converge to the optimal result. In methods using anti-flocking, agents evaluate the instantaneous situation rather than the entire coverage process. For this reason, the most appropriate result of the instant situation is obtained.

1.3. Contribution of the Paper

In the dynamic range coverage process, UAVs equipped with sensors can be used to achieve low cost with high flexibility [31]. In this paper, a multi-agent deep RL-based method in which each UAV is represented as an agent is proposed to cover the maximum number of PoI in the target area. RL is a reward-driven machine learning approach that an agent learns in an interactive environment through its experiences. An RL agent interacts with the environment and receives positive or negative reward points for each action. Classical RL methods are individual learning processes in that an agent learns to change its behavior to maximize the reward PoInts it receives. In multi-agent RL (MARL), the actions of all agents are evaluated for the common goal of the group. Collective behavior refers to a system that provides maximum coverage when evaluated for the agent group. When this behavior is evaluated for an agent, it represents a subsystem that finds an appropriate location for itself with limited communication capabilities.

In the proposed method, the learning-execution process of the agent group is based on the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [20] algorithm, which is one of the central learning-decentralized execution methods. In the MADDPG algorithm, each agent is trained by the DDPG algorithm, where the actor has access to only local observations. The proposed method, which is a modified version of MADDPG, takes into account the state and actions in a collective intelligence manner using an actor-critic network. This approach helps the MAS cover area by avoiding collisions and orienting the agents to the most suitable locations as soon as possible. In addition, it enables the achievement of maximum PoI coverage in regularly-irregularly shaped areas with connected agents. It is assumed that the agents are homogeneous and have a circular coverage. Agents create an abstract rectangular plane using the $x_{\min,}y_{\min,}x_{\max}$ and $y_{\max}$ values of the target area to be covered and then divide this area into grids. Then, they form path planning for positioning into the nearby grid which has the largest number of PoI. Path planning helps to achieve high fairness index by reducing the number of commonly covered PoIs. While minimizing the energy consumption is handled with path planning, the grid decomposition assists irregularly shaped areas to be covered.

1.4. Organization of the Paper

The remainder of the paper is organized as follows. In Section 2, a MAS model for dynamic area coverage problems is presented by giving detailed information about the proposed method. In Section 3, experimental studies and simulation results are reported. In Section 4, the results are analyzed, and a road-map is drawn for further work.

2. Material-Method

Each agent in the group is modeled as a UAV agent that has the same motion model as a real UAV. It is aimed to meet the following objectives by collaborating with agents:

– Building a distributed system equipped with learning capabilities.
– Covering the maximum number of PoIs.
– Minimizing energy consumption caused by agent movements.
– Maintaining the connection between agents.

- Moving without exceeding the borders of the target area.
- Optimizing agent movements and avoiding collisions between agents.
- Ensuring task continuity in dynamic environments.

With learning capabilities, agents can adapt to changes and achieve collective success. With the design in accordance with the distributed system architecture, it can be ensured that the system can make decisions independently from the central control. Thus, the proposed method designed with the MARL approach helps the agents produce collective success independent of central control.

2.1. Reinforcement Learning

In RL, an agent interacts with the environment and optimizes its behavior using the reward received. As seen in Fig. 1, the agent takes an action to change its state, and then receives a reward from the environment that indicates the quality of the action. One of the key advantages of reinforcement learning is that it enables the agent to learn from experience, without being explicitly programmed with a set of rules or procedures.

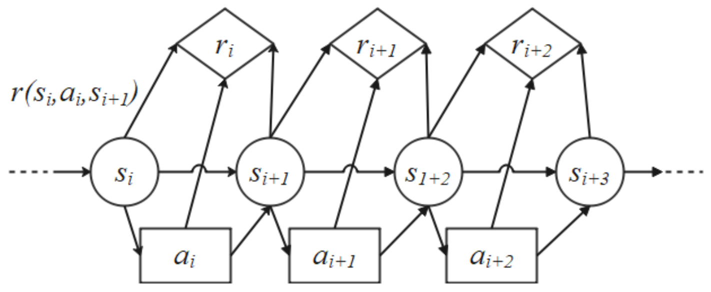

Fig. 1. RL agents

RL is modeled as Markov Decision Process (MDP), which are defined as fully observable environments. The assumption of full observability of MDPs allows the agent to access the current state of the system at each step [8]. Partially-observable Markov Decision Process (POMDP) is a type of MDP that connects unobservable system states to observations probabilistically and is used to model dynamic systems. Depending on the system state and the agent's future actions, the agent may take actions that affect the system in order to maximize expected future rewards. The goal is to find the most appropriate policy that guides the agent's actions. Unlike MDPs, an agent in POMDPs cannot directly observe the entire system state but makes circumstantial observations.

Multi-agent RL involves multiple agents learning to interact with each other and the environment. It focuses on examining the behavior of multiple learning agents coexisting in a shared environment. MARLs exist in which collaborative agents are modeled as nodes that leverage information shared through a communication network. A node means that agents can instantly communicate, exchange information with each other. Therefore, agents know their local and neighbor information. Information sharing helps to obtain more and more stable information about the environment in which the agents are located.

This multi-agent behavior approach is called connected agents. The proposed method in this paper is used connected agents approach to find more appropriate locations using reachable agents' observations.

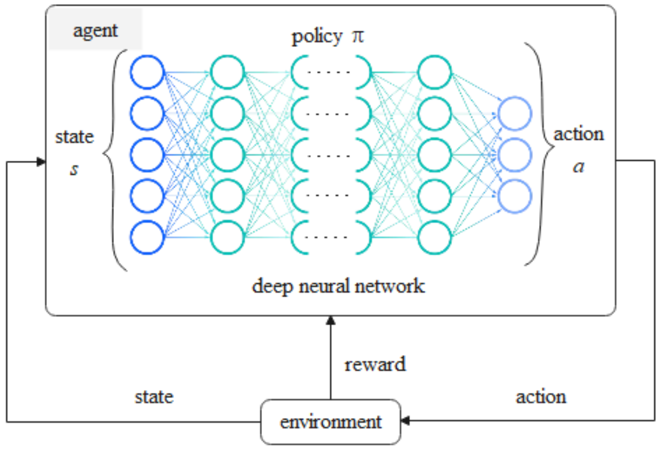

Fig. 2. Deep RL agent

According to [4], deep learning is a machine learning approach that learns multiple representation levels corresponding to different levels of abstraction. In a simple case, it contains two groups of neurons, one receiving the input signal and the other sending the output signal. When the input layer receives an input, it transmits the processed version of the input to the next layer. In a deep network, there are many layers between input and output, consisting of multiple linear and multiple processing layers [28].

As seen in Fig. 2 Deep RL (DRL) improves RL by using deep neural networks as the underlying model to represent the policy, value function or model of the environment. This approach allows the agent to handle more complex, high-dimensional observation spaces and make decisions based on more sophisticated representations of the state of the environment. In other words, DRL combines the power of deep learning with the problem-solving framework of reinforcement learning to create more advanced agents.

Multi-agent deep RL (MADRL) is a multi-agent system that consists of agents which compete or cooperate to solve complex tasks. Each agent has direct access to local observations only. These observations can be many things, a view of the environment, locations relative to landmarks, and even relative locations of other agents. Agents in MADRL are designed distributed manner, and during the learning phase, are guided by a central module or critic. While each agent has only local information and local policies, agents have a critic which reviews their information and advises them on how to update their policies. This approach is called centralized learning. During execution, the removal of the central module limits the state and action space effects. The central module is intended to provide sufficient information for the most appropriate local policy at execution time.

2.2. Multi-agent Deep Deterministic Policy Gradient

DDPG [27] is an actor-critic method that adapts the core achievement of deep Q-learning [6] to continuous action. The MADDPG algorithm has been proposed as an extension of the actor-critic policy gradient methods, in which agents can only access local information and share their policies with other agents. The main idea of the policy gradient (PG) approach is to set a policy parameter to maximize a given target by taking steps in the direction of the given gradient. Using a critic within the system is a common solution to handle the dynamic state of the environment. Therefore, this central critic can be used as a reliable guide to increase the flexibility of agents with local observations. In MADDPG, each agent has two networks: an actor-network and a critic network. The actor-network calculates the action to take based on the situation in which the agent is located, while the critic network evaluates the consequences of the action to improve the performance of the network of actors. Experience replay buffer used for critic network update helps to break correlations in training data and make training more stable. In the training phase, each agent is trained by a DDPG algorithm, where the actor has access to local observations. The centralized critic, on the other hand, combines all states and actions as input and uses the local reward function to obtain the corresponding Q-value. In the execution phase, the critic network is removed, and agents only use the actor-network. This means that the execution is decentralized. In fact, MADDPG can be thought of as a multi-agent version of DDPG. The main goal is to decentralize the execution. Q-learning and DDPG perform poorly in multi-agent environments as they do not use the knowledge of other agents in the group. The MADDPG approach overcomes this challenge by using the observations and actions of all agents.

2.3. Proposed Method

The main purpose of this paper is to construct a method in which a group of UAVs tasked with dynamic area coverage is modeled with a multi-agent deep reinforcement learning (MADRL) approach. In the MADRL approach, UAVs are modeled as mobile agents operating in a dynamic environment and interacting with each other for a common goal. In this way, an intelligent system can be achieved by generating strategies that provide high fairness index with low energy consumption while maximizing coverage.

To simplify agent-based coverage and energy consumption, the target area is divided into grids and the center of each grid is called the grid center (GC). Each agent is intended to be deployed to a GC in a reasonable amount of time.

Assumption 1. It is assumed that all agents in the team have the same properties and move in a 2-D plane. Each agent is represented by A_i where $A_i \in A | i = 1, ..., N$.

Assumption 2. It is assumed that each agent knows its location. The proposed method uses a geospatial approach to reach the target area, discover other agents in the target area and interact with the agents in the communication range, $(x_i^A(t), y_i^A(t))$. Agents have information on how many agents are on the team. However, it does not have the location information of the agents that are not within communication distance. The positions of all agents in the environment at time t are indicated by:

$$A^N = \{(x_1^A(t), y_1^A(t)), x_2^A(t), y_2^A(t)), ..., x_m^A(t), y_m^A(t))\} \tag{1}$$

Assumption 3. Each agent is assumed to have a circular shape (diameter: $\emptyset_A$) and a circular detection zone. The distance between A_i and A_j is expressed as $d_{ij} = |A_i A_j|$. For agents A_i and A_j, the following equation must be satisfied:

$$\left|(x_i^A(t), y_i^A(t)), (x_j^A(t), y_j^A(t))\right| \leq d_{ij} \tag{2}$$

Each agent has limitations such as communication/detection range. At each learning step, agents divide the target area into grids. When the communication distance is shorter than the grid separation distance, the connection between agents will be broken. However, any agent action to restore communication may cause other agents' positions to change. As a result, each action taken to find a suitable solution increases energy consumption.

In this section, the details of the proposed method are presented in order to achieve maximum coverage with minimum energy consumption under the frame of a high fair index of connected agents (which can exchange information) in the target area.

Grid Decomposition. Within the system, each UAV is represented by an agent. In a 2-D environment, agents learn to locate grid centers in the target area. The target area may not be a regular shape. An abstract area is created for the regular or irregular target area. The abstract area is the smallest regular rectangular area containing the target area. Agents are directed to abstract grid centers that are divided into grids. Agents try to find the fastest and most convenient solution by going to the grid closest to them and containing the maximum number of PoIs. It is also aimed to provide communication between agents in the target area. The smallest regular quadrilateral area containing the target area, with T being the target area and k^T the vertices of the target area are expressed as $\left\{(x_i^T, y_i^T), ..., (x_n^T, y_n^T) \forall x^T, y^T \in k^T\right\}$ for $(x_{\min}^T, y_{\max}^T), (x_{\max}^T, y_{\max}^T), (x_{\max}^T, y_{\min}^T)$, and $(x_{\min}^T, y_{\min}^T)$, respectively. GCs are donated by $j \in \{1, ..., N\}$, while the set of GCs is given by $GC_j \subseteq T$. The set of all PoIs within the target area is defined as:

$$PoI_T = \{PoI_1, ..., PoI_n\} \tag{3}$$

The grid decomposition algorithm is given in Algorithm 1.

Algorithm 1 Grid decomposition

```
1:  Initialize: $L_I(i,k) \leftarrow$ (GC index,number of PoI)– empty list for grids
2:  Set: distance for grid decomposition $m, m \leq A_m | A_m$ communication distance of an agent
3:  Start: $x_{temp} = x_{\min} + (m/2), y_{temp} = y_{\min} + (m/2)$, $PoI_T \leftarrow$ PoIs in the target area
4:  for from $y_{temp}$ to $y_{\max}$ do
5:      $j = 0$
6:      for from $x_{temp}$ to $x_{\max}$ do
7:          $GC_j = (x_{temp}, y_{temp})$
8:          $x_{temp} = x_{temp} + m$
9:          $PoI_{temp} = GC_j.buffer(m/2) \cap PoI_T$
10:         $L_I.insert(j, PoI_{temp}.size())$
11:         $j++$
12:     $x_{temp} = x_{temp} + (m/2)$
13:     $y_{temp} = y_{temp} + m$
    return $L_1$
```

First, as seen in Alg. 1, the coordinates of the smallest regular rectangular area containing the target area are found. Then, using the specified decomposition distance (usually the agent's communication/sensing distance), GCs are found where the agent will be located. As seen in Fig. 3, the number of PoIs in GCs is calculated.

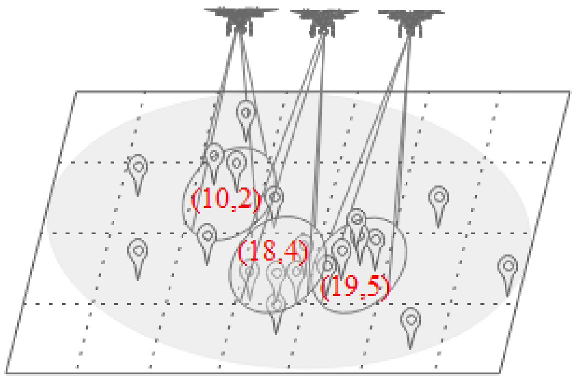

Fig. 3. Grid center and PoI pairs

The goal here is not to create a grid and put agents at its center, but to place agents at the calculated PoInts and to create a coverage area equal to the communication distance of the agents. The circular coverage areas are intersected with PoIs. Finally, it is thrown into a list containing the GC index and the number of PoIs it contains.

Reward Strategy. Minimizing travel distance has many benefits, such as saving time, energy, and equipment usage time. Inspired by [24], a reward structure has been designed to perform target assignments and minimize travel distance. As seen in Alg. 2, the reward is determined by the distance of each agent from each GC. At each step, the distances of the agents to the GCs are calculated, but maximizing the reward received would be an unsuccessful strategy here. Maximizing the reward achieved will distract agents from the target area and prevent areas from being covered. For this reason, the smallest of the distance values obtained is selected. Then, this distance value is multiplied by -1 to get a negative value. In this way, the reward of the agents for distributing the areas separated into the grids converges from negative to 0. In this way, reward-driven learning, which is the basis of RL, is realized.

Algorithm 2 Reward function 1

```
1: Start: $r_1 = 0$
2: for from $GC^0$ to $GC^N$ do
3:     $d \leftarrow$ calculate the distance between agents and GCs
4:     $r_1 = r_1 + (-\min(d))$
   return $r_1$
```

Agents moving toward GCs are penalized if they collide with each other. Therefore, agents must learn to cover target areas while avoiding collisions. The distance between the agents is calculated at each step so that a collision can be determined. Considering that the agents are modeled as a circle if the distance is smaller than the sum of the radius of the agents, it means that the collision has occurred.

Algorithm 3 Reward function 2

```
1: Start: $r_2 = 1, \emptyset_A$
2: for from $agent = 0$ to $E^N$ do
3:     if $d_{A_i A_j} < \emptyset_A$ then
4:         $r_2 = \gamma_r * r_2$
   return $r_2$
```

In the collision calculation method, whose pseudo-code is given in Alg. 3, γ_r represents the discount factor for collision avoidance. Collisions for A_i are represented by p_i where $w(ij)$ is used to stand for the calculation whether there is a collision between A_i and A_j.

$$p_i = \prod_j w(ij), j \in \{1, ..., N\} all A_j \in E_i^C \tag{4}$$

$$w(ij) = \begin{cases} 1, (\emptyset_i + \emptyset_j)/2 \leq d_{ij} \\ \gamma_r, (\emptyset_i + \emptyset_j)/2 > d_{ij} \end{cases} \tag{5}$$

It is calculated to find the percentage of PoIs that needs to be covered.

$$r_c = \sum_i^n c_i \forall i, n\{A_i, ..., A_n\} \in A^R \tag{6}$$

$$r_3 = \left(1 - \frac{r_c}{PoI^N}\right) * 100 \forall i, n\{PoI_i, ..., PoI_n\} \in PoI_T \tag{7}$$

The reward function for orienting agents towards GCs with many PoIs is given in Alg. 4:

Algorithm 4 Reward function 3

```
1: Start: $r_3 = 0$
2: for from $agent = 0$ to $E^N$ do
3:     Get: reachable agents of $A_i$
4:     Get: PoIs covered by the reachable agents ($A^R$)
5:     Sum: $r_c \leftarrow$ number of the PoIs covered ($c$), $L_I(i,k)$ using Eq. 6:
6:     Calculate: $r_3 \leftarrow$ the percentage of PoIs not covered by all PoIs using Eq. 7
   return $r_3$
```

The proposed method has 3 reward functions to maximize covered area with minimum energy consumption. However, these reward functions designed for different purposes affect the coverage quality at different weights. Therefore, coefficient matrix represented as $[k_1, k_2, k_3]$ is used to calculate the collaborative reward of the system. K is a matrix that contains the coefficients of all the reward functions in the system. The collaborative reward function obtained using reward functions designed for the shortest route, collision avoidance, and covered PoIs is as follows:

$$K = [k_1, k_2, k_3] \tag{8}$$

$$r_i = k_1 r_1 * k_2 r_2 * k_3 r_3 \tag{9}$$

In the collaborative reward function, the reward points for collision avoidance route planning are multiplied by the percentage of PoIs not covered. The path planning process is a negative value; the higher the number of PoIs not covered, the lower the cumulative score will be. This approach speeds up the learning process by growing the reward range. Thus, the RL agents in the proposed method try to converge to 0 with the reward-driven approach using Eq. 8.

Multi-agent System Model. A MAS model is designed to solve the complexity of collaboration between agents. The position of agents A_i moving on a 2-D plane at time t is expressed as $(x_i^A(t), y_i^A(t))$. The target area T, and the GCs are represented by $j \in \{1, ..., N\}$ for all $GC_j \subseteq T$. In the MAS model, it is aimed to meet the following objectives:

– Agents should be distributed across GCs to cover the maximum number of PoIs.
– Each GC should only be covered by one agent.
  i.e., $\forall i, j | GC_i^{'} \neq GC_j^{'} | i, j \in \{1, ..., GC^T\}$
– Agents positioned on GCs within the target area should be at a distance to communicate with each other.
– Agents should avoid the collision, so two agents cannot be in the same position at that same time t.
  i.e., $\forall i, j, (x_i^A(t), y_i^A(t)) \neq (x_j^A(t), y_j^A(t))$

The agent policy is represented by θ and the policy parameter by μ. In the N-agent MAS model, the policies for state transitions are expressed as $\mu = \{\mu_1, \mu_2, ..., \mu_N, \}$, and the parameters as $\theta = \{\theta_1, \theta_2, ...\theta_N\}$. Additionally, the model uses an experience replay buffer for model-free training. The agent stores information in the form of $(x, x', a_1, ..., a_N, r_1, ..., r_N)$, showing the joint status at each step, the next joint status, the joint action, and the rewards received by each of the agents. Then, it again does a sampling of its buffer to train the agent. The agent's critic is updated using the sampled information. Thus, the loss function using the sampled temporal difference error for the critic is defined as:

$$y = r_i + \gamma Q_i^{\mu'}(x', a_1', ..., a_N')|_{a_j' = \mu_j'(o_j)} \tag{10}$$

$$L(\theta_i) = \frac{1}{S} \sum_j (y^j - Q_i^{\mu}(x^j, a_1^j, ..., a_N^j))^2 \tag{11}$$

In the above equation, a', γ, S, o and Q_i^{μ} represent the next joint action, the discount factor, the size of the randomly selected sample from the replay buffer, the partial observations of the environment, and the central action-value function, respectively. The sampled policy gradient for updating the actor is defined as follows:

$$\nabla_{\theta_i} J \approx \frac{1}{S} \sum_j \nabla_{\theta_i} \mu_i(o_j^i) \nabla_{a_i} Q_i^{\mu}(x^j, a_j^i, ..., a_N^j)|_{a_i=\mu_i(o_i^j)} \tag{12}$$

Construction of the proposed method In this paper, a method is proposed that aims to cover the maximum number of PoIs in regular-irregular shaped areas with MAS. In the method designed in the context of the actor-critic, it is aimed to achieve high coverage of the connected agents with low energy consumption. According to the experimental results, the proposed method was able to successfully complete the coverage task in the dynamic environment. In the testing phase, the agents created distributed but collective actions for the common purpose of the group using their own critic and actor networks. In addition, the results show that policies with high fairness index were produced under communication restrictions by adapting to the changing number of agents in the dynamic environment. On the other hand, in the proposed method, the state space depends on local observations. Therefore, the growing action-state space is eliminated. This approach has helped produce an effective policy in less time while the designed reward structure enabled the generation of collective behavior without the need for reward sharing, as seen in Fig. 4. Moreover, the proposed model-free method, which is based on local observations independent of central control, is capable of guiding real applications with its reward strategy.

The behavior strategy of the proposed method is based on the commonly used combination of ”attractiveness” and ”avoidance”. Inspired by [34], “attraction” is considered a positive reward when agents are attracted to GCs in the target area as expressed in Algorithm 2. If the agents go outside the communication distance, it is considered a negative reward. In addition, as presented in Algorithm 4, agents receive positive rewards for PoIs they cover with reachable agents. This positive reward prevents agents from exceeding the communication distance. Therefore, a positive reward is obtained in terms of ”attractiveness”. Inspired by [11], the collision distance between agents is defined, and based on this distance, a negative reward (avoidance) is given whenever any two agents are too close to each other. Thus, according to Algorithm 3, each agent receives a positive reward for being farther from the others than the specified distance and a negative reward for being too close to others.

The proposed method has a model-free learning structure and operates the learning process using the replay buffer. The centralized module guides agents on how to update their policies during the training period. The training process is divided into sections, and the location of the agents is randomly determined before each section is run. Agents store a tuple of action, status, and reward information in the replay buffer to rerun the scenario during the training process. At each time step, samples are taken from the replay buffer. Critics are updated with Eq. 11, while the policy gradient of agents is updated with Eq. 12. Each agent determines the most appropriate joint action to increase area coverage. In the execution process, the central module is removed, and the actor-network is used for local observations. The pseudo-code of the proposed method is given in Alg. 5.

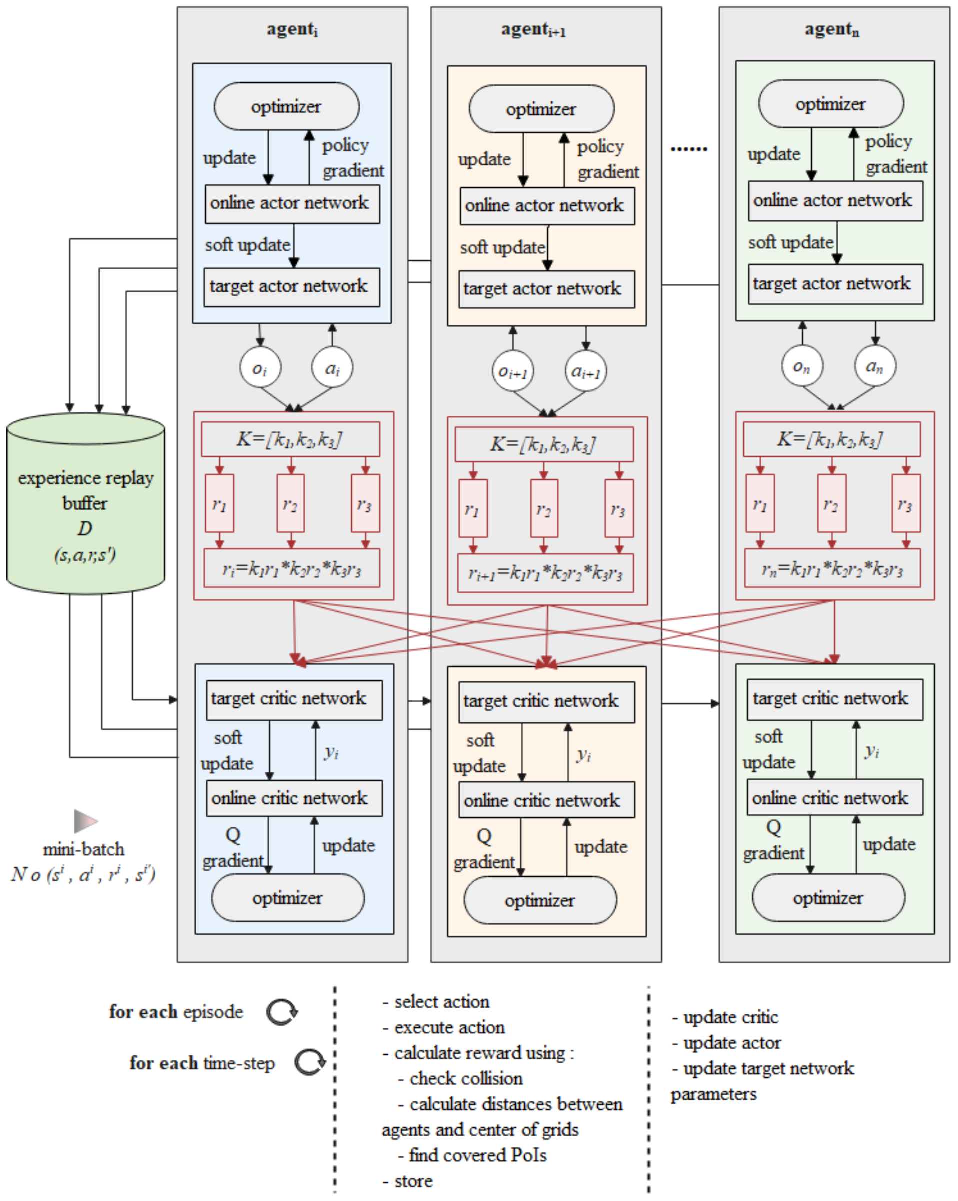

Fig. 4. The framework of the proposed multi-agent deep RL method

Algorithm 5 Proposed method

```
1:  Initialize: learning and reward discount factor
2:  for from $episode = 0$ to $M$ do
3:      Start: $N$ and target area $T$
4:      Find: $GC^T$ using Algorithm 1
5:      Get: the initial state of $x$
6:      for from $stept = 1$ to $max_s tep$ do
7:          Set: for each agent $i$, selected action with noise: $N_t, a_i = \mu_{\theta_i}(o_i) + N_t$, policy and
    observation: $w.r.t$
8:          Execute: actions $a = (a_1, ..., a_N)$
9:          Calculate: $r$ using Algorithm 4
10:         Store: $(x.a.r.x^{'})$ to replay buffer $D$
11:         $x \leftarrow x^{'}$
12:         for from $agent_i = 0$ to $N$ do
13:             Sample: mini-batch of $S$ samples from replay buffer $D$
14:             Update: critic network using Eq. 11
15:             Update: actor-network using the sampled policy gradient using Eq. 12
16:     Update: target network parameters for each agent
```

Training process Each agent has its own actor and critic network. As described in Sect. 2.3, the method learns from experiences (i.e. action, state, and reward) stored in the replay buffer. In other words, actors and critics of all agents are updated by using the mini-batch, which is randomly sampled at each time step during the learning process. In Algorithm 5, the pseudo-code of the learning approach in the training process is given. In the proposed method, the process from the starting position of the agents to positioning in the target area is expressed as an episode. Each episode for training consists of time steps represented t. In the training loop, the system gets the initial state s_1, and then the initial conditions of the environment are created. Each agent i selects an action according to the actor μ_{θ_i} by observation Q_i. Noise is added to the selected action to prevent the agent from choosing the most appropriate local policy and performing further exploration. Agents performing the selected action obtain a reward value r_t and a new state s_{t+1}. If the selected action forces the agent to leave the target area or collide with other agents, it will be penalized by Eq. 8. Therefore, the agent learns to avoid this action and not to choose the state transition while the last values of (s_t, a_t, r_t, s_{t+1}) are stored in the replay buffer. At the end of the training period, each agent at time step t randomly selects the mini-batch S from the replay buffer D and then updates the critic using Eq. 11. After updating the critic, the actor is updated with Eq. 12. Finally, the target network is gradually updated with the loss function and learning rate.

3. Experimental Studies

A simulation environment is designed using the multi-agent actor-critic platform developed by the OpenAI team [20] to evaluate the proposed method.

3.1. Experimental Settings

A coordinate plane, which is formed of both a horizontal line and a vertical line as 1 unit, is used as a simulation environment where the geometric center is the origin. The agents and target area are located on the plane. A circular shape is used to express an agent. Also, the area coverage of an agent is represented by a circle around the agent. At the beginning of the training episode, the locations of the agents, the shape and the location of the target area, the number and location(s) of PoI(s), and the communication distance of agents are randomly generated. The number of episodes of training is determined as 5000 while the number of step size of the episode is set to 250. The rest of the parameters are specified as follows; the number of units in the multi-layer perceptron is 64, the learning rate is 0.001, the batch size is 1024. The learning rate γ_r to be used in the r_2 is chosen as 1. Due to the random generation of the initial positions of the agents, the simulation scenarios are repeated 50 times. The average of the metrics obtained are taken is to evaluate the proposed method's performance.

Two analyses were carried out to determine discount factor and find out the effects of the reward functions to coverage quality. Firstly, it was fixed the number of agents to 5, and then discount factor was set the number of between 0.8 and 0.99 to try to find appropriate one. According to the results, in particular, as g was increased, benchmark metrics went up until the peak value, then fluctuated it as seen in Fig. 5. According to these result, the best results were obtained when the discount factor was set to 0.88. Secondly, it was tried to find appropriate coefficient values to maximize collaborative reward. In this scenario, three experimental studies applied while only one coefficient function was changed at each experimental study. The coefficient matrices used for collaborative reward function are expressed as $[k_1, 1, 1], [1, k_2, 1], [1, 1, k_3]$ where $k_1, k_2, k_3 \in k = \{0.1, 0.5, 1, 1.5, 2\}$. As seen in Fig. 6, when the weights of the reward functions used for target assignment and area coverage increased, the number of covered PoIs increased while the average step size decreased. However, as the collision avoidance function weight was increased too much, agents focused on collision avoidance and the quality of coverage decreased. According to these results, when the coefficient matrix was set to $[k_1 = 2, k_2 = 0.5, k_3 = 2]$, coverage quality could increase satisfactorily.

The proposed model is compared with DRL-EC3, improved-DRL-EC3, and SBG-AC, using the same simulation settings. Three topics are used for the comparison and validation of the simulation results:

- The effect of increasing the number of agents on coverage: It is the average PoI score covered by the system. It is calculated using algorithm 4.
- The effect of increasing the number of agents on energy consumption: energy efficiency is expressed by the ratio of the number of agents to the PoIs covered. It is the normalized version of the PoIs covered, that is, the ratio of the result obtained by Algorithm 4 to the number of agents in the system.
- Fairness index for covered PoIs: The Jain [13] fairness index for PoI coverage scores. Where N represents the number of PoIs in the environment, $c_t(i)$ represents the number of agents containing the PoI at time t. The case of $J = 1$ indicates perfect fairness between agents.

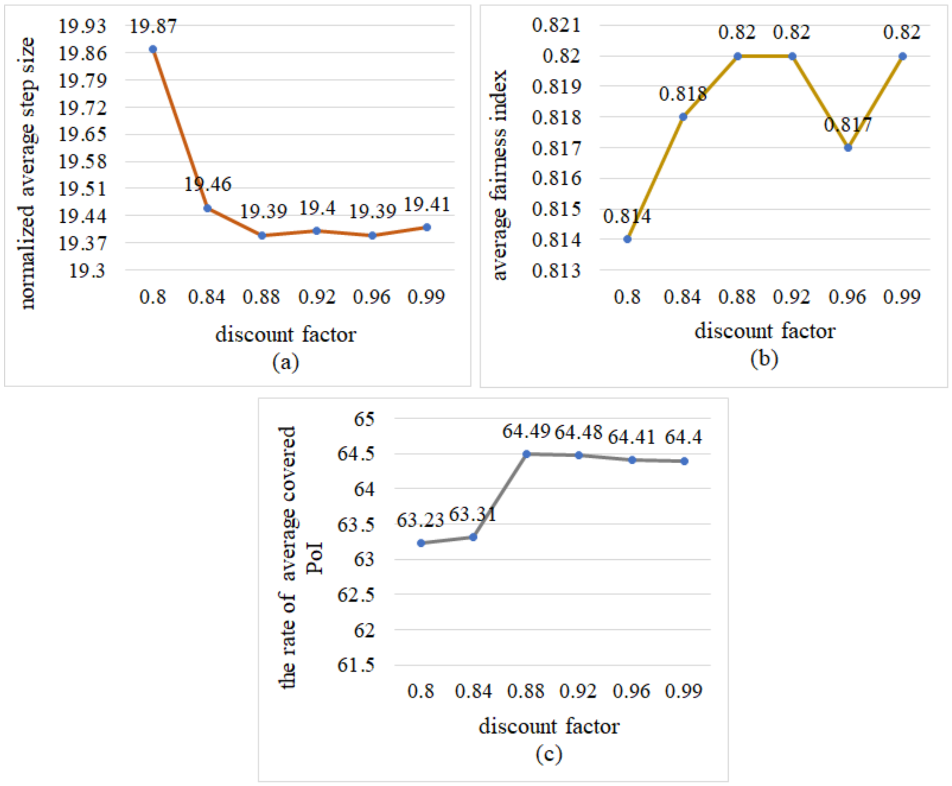

Fig. 5. The effects of the discount factor on coverage quality

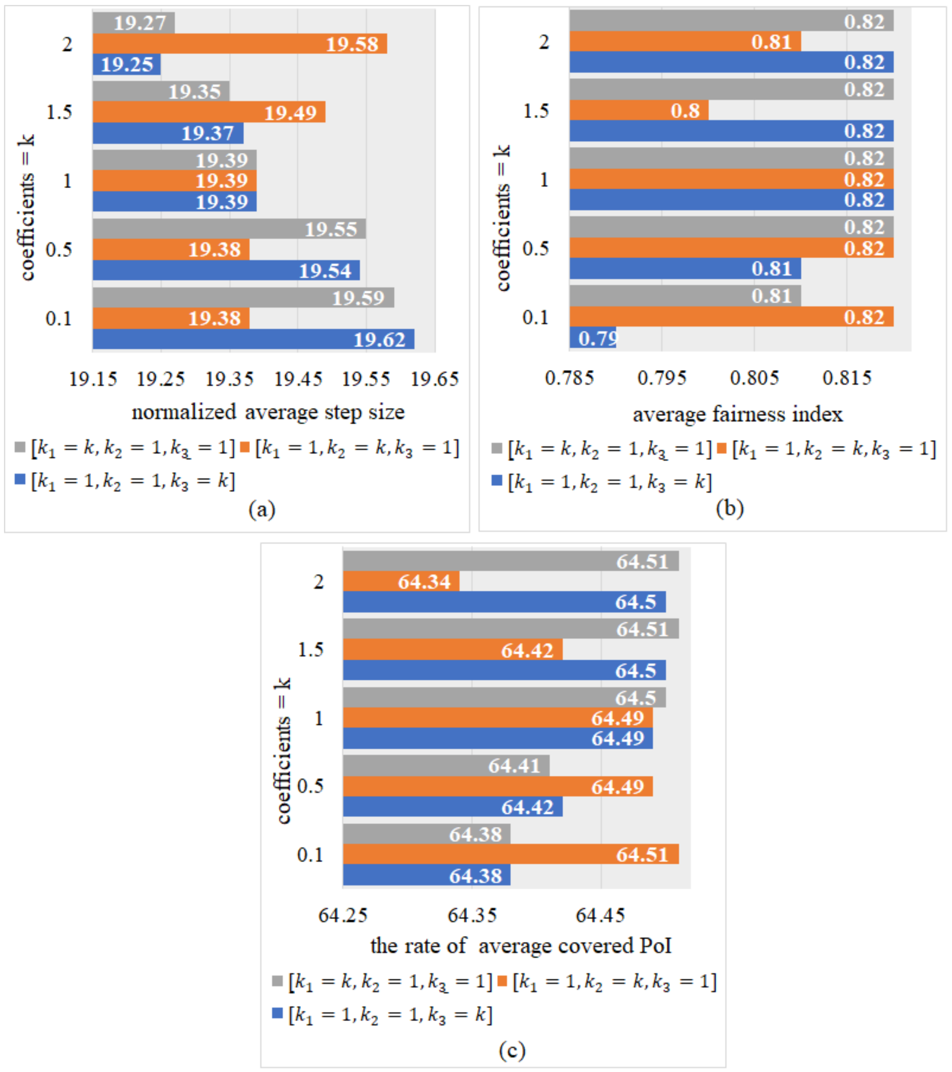

Fig. 6. The effects of the collaborative reward function coefficients on coverage quality

$$J = \frac{\left(\sum_{i=1}^{N} c_t(i)\right)^2}{N \sum_{i=1}^{N} c_t(i)^2} \tag{13}$$

3.2. Experimental Results

In the first experiment, the effect of increasing the number of agents is examined and the performance of the 4 models was compared. The coverage ratios to be used for comparison were calculated according to the method given in Algorithm 4. As seen in Fig. 7, the proposed method achieved approximately 10.1% more coverage than DRL-EC3, 5.06% more coverage than improved-DRL-EC3, and 3.18% more coverage than SBG-AC. For example, when the number of agents was 4, the proposed method covered approximately 59.8%, while DRL-EC3 covered approximately 53.9%, improved-DRL-EC3 approximately 55.2%, and SBG-AC approximately 57.4%. In the case of 8 agents, the coverage obtained by the proposed method was approximately 90.9%, 80.2% obtained by DRL-EC3, 87.3% obtained by improved-DRL-EC3, while the coverage obtained by SBG-AC was 88.7%. There was a similar trend for other scenarios. Therefore, the average covered PoI rate achieved by the proposed method increased regularly. The increase in the number of agents has allowed agents to connect with different patterns in the PoI coverage process.

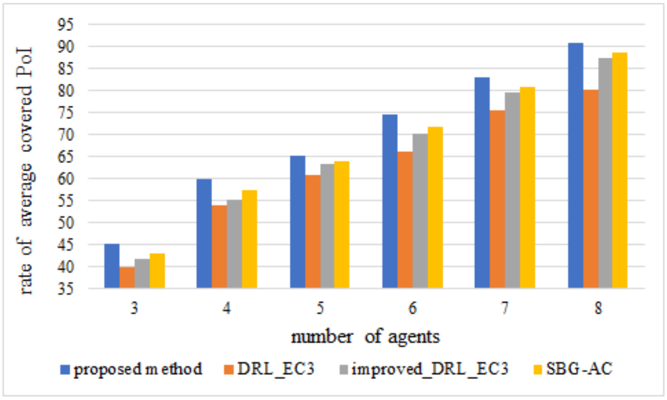

Fig. 7. Agent relationship with the covered PoIs

In the second experimental study, an energy consumption comparison of 4 models was made. In the calculation of energy consumption, the number of actions performed by the agents was used. In each episode, the coverage rate achieved by the system is recorded. At the end of the training, the coverage scores are summed and divided by the number of actions. The result obtained is divided by the number of agents in the system so that the number of PoIs covered per action is found. Finally, it is calculated how many steps an agent takes from the initial position to the final position to cover one PoI. This result represents the normalized average energy consumption. As seen in Fig. 8, DRL-EC3, improved-DRL-EC3, and SBG-AC models consumed approximately 9.35%, 4.44%, and 2.8% more energy, respectively, than the proposed method. For example, when the number of agents was 4 and 8, the normalized average step size is given as follows; the proposed model was 16.85 and 22.1, DRL-EC3 was 18.55 and 24.94, improved-DRL-EC3 was 18.12 and 22.94, and SBG-AC was 17.77 and 22.53. According to these results, it was observed that there was no significant change in energy consumption values as the number of agents increased. Due to the competitive and cooperative nature of multi-agent systems, the number of agents does not have a large impact on policies. The proposed method has reached the maximum coverage rate by consuming less energy. It is thought that there are 2 reasons for this; (i) a specific reward strategy is designed for path planning; (ii) the reward is calculated by the status of each agent it can access, not just the agent's own status.

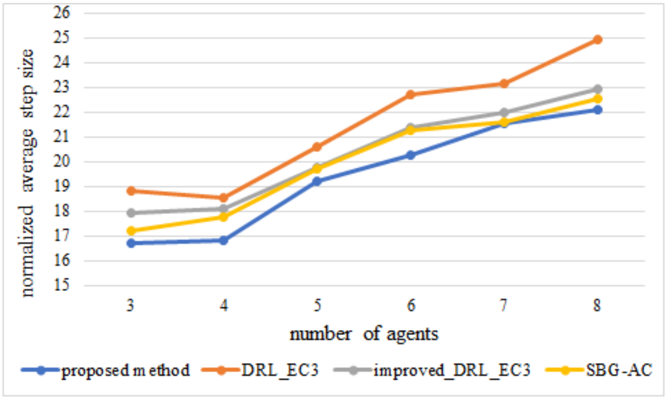

Fig. 8. Number of actions-agent relationships

Finally, four models were compared in terms of fairness index according to the number of agents. In Fig. 9, the fairness index results obtained by different numbers of agents for the 4 models are presented. The proposed model appeared to have a better fairness index than DRL-EC3, improved-DRL-EC3, and SBG-AC with an average increase of about 11.14%, 2.34%, and 1.76%, respectively. For example, when the number of agents is 3, the proposed model achieved 0.71 fairness index, DRL-EC3 obtained 0.588 fairness index, improved-DRL-EC3 obtained 0.698 fairness index and SBG-AC obtained 0.698 fairness index. When the number of agents was 6, the fairness indices of the proposed model, DRL-EC3, improved-DRL-EC3, and SBG-AC were 0.937, 0.801, 0.911, and 0.919, respectively. A similar trend is observed for other scenarios as well. The larger the number of agents, the greater the number of grids covered. This relationship leads to a direct improvement in the fairness index. In addition, with fewer steps in the proposed method, a similar fairness index rate was achieved with other methods due to purpose-based reward strategies.

In this section, the behavior of the proposed model against issues such as energy, coverage, and fairness index are examined. Then, using these three metrics, the proposed method is compared with DRL-EC3, improved-DRL-EC3, and SBG-AC models. The simulation results are summarized in Table 1.

Table 1. Simulation results

	proposed method	DRL-EC3	improved-DRL-EC3	SBG-AC
Approach	MARL	MARL	MARL	MARL
Algorithm	MADDPG	DDPG	MADDPG	MADDPG
Reward strategy	Agent&group-specific	Agent specific	Agent specific	Group specific
Evaluation Metric Average rate of PoIs covered	3 agents: 45.2 4 agents: 59.8 5 agents: 65.1 6 agents: 74.6 7 agents: 83.1 8 agents: 90.9	3 agents: 39.8 4 agents: 53.9 5 agents: 60.7 6 agents: 66.2 7 agents: 75.6 8 agents: 80.2	3 agents: 41.8 4 agents: 55.2 5 agents: 63.3 6 agents: 70.2 7 agents: 79.7 8 agents: 87.3	3 agents: 42.9 4 agents: 57.4 5 agents: 63.8 6 agents: 71.7 7 agents: 80.9 8 agents: 88.7
Evaluation Metric Normalized average step size	3 agents: 16.72 4 agents: 16.85 5 agents: 19.21 6 agents: 20.28 7 agents: 21.54 8 agents: 22.1	3 agents: 18.83 4 agents: 18.55 5 agents: 20.59 6 agents: 22.68 7 agents: 23.15 8 agents: 24.94	3 agents: 17.95 4 agents: 18.12 5 agents: 19.76 6 agents: 21.37 7 agents: 21.98 8 agents: 22.94	3 agents: 17.2 4 agents:17.77 5 agents: 19.69 6 agents: 21.26 7 agents: 21.61 8 agents: 22.53
Evaluation Metric Average fairness index	3 agents: 0.71 4 agents: 0.75 5 agents: 0.82 6 agents: 0.937 7 agents: 0.946 8 agents: 0.957	3 agents: 0.588 4 agents: 0.694 5 agents: 0.762 6 agents: 0.801 7 agents: 0.844 8 agents: 0.86	3 agents: 0.698 4 agents: 0.724 5 agents: 0.812 6 agents: 0.911 7 agents: 0.915 8 agents: 0.94	3 agents: 0.698 4 agents: 0.731 5 agents: 0.814 6 agents: 0.919 7 agents: 0.923 8 agents: 0.945

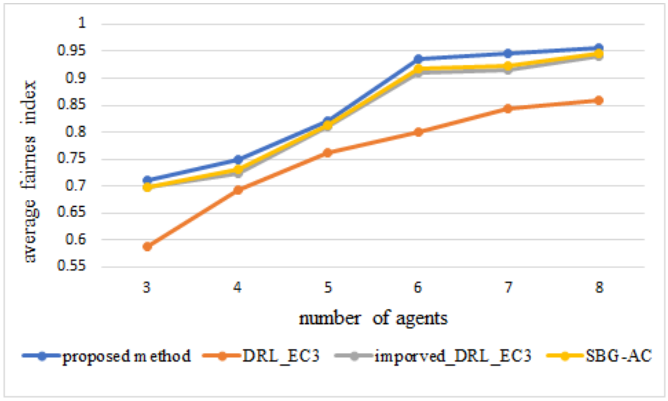

Fig. 9. Average fairness index with respect to number of agents

In this paper, 3 different reward strategies are designed as purpose-based while there is only one reward strategy in the DRL-EC3, improved-DRL-EC3 and SBG-AC methods. With the collaborative design, low energy consumption-high coverage ratio has been tried to be achieved. The use of 3 different reward strategies increases the space of the reward that can be obtained and clarifies the relationship between situation-action pairs. Therefore, while learning time decreases, learning quality increases. There are two main reasons for this result; (i) connected agents are positioned on grids without going out of communication range, (ii) agents receive high rewards as they distribute into grids and reduce intersections. DRL-EC3 and improved-DRL-EC3 use only agent-specific reward strategies that the reward received is only affected by agent-environment interaction. However, SBG-AC and the proposed method are interested in the current state of the environment, in which all agents share a common reward. Contrary to the proposed method, there is no reward strategy related to target assignment in DRL-EC3, improved-DRL-EC3, and SBG-AC methods. In the proposed method, agents try to maximize their reward points by using the information of connected agents. This approach forces agents to connect with each other and make collaborative decisions. In addition, successful results were obtained in terms of energy consumption, since the agents tried to locate the GC with the maximum number of PoIs in the closest distance to them. This approach represents the path planning algorithm with target assignment in MAS. With the help of this algorithm, the intersection of the agents in the coverage areas is minimized and a high fairness index result is obtained.

4. Conclusions

In this paper, a method is proposed that aims to cover the maximum number of PoIs in regular-irregular shaped areas with MAS. In the method designed in the context of the actor-critic, it is aimed to achieve high coverage of the connected agents with low energy consumption. According to the experimental results, the proposed method was able to successfully complete the coverage task in the dynamic environment. In the testing phase, the agents created distributed but collective actions for the common purpose of the group using their own critic and actor networks. In addition, the results show that policies with high fairness index were produced under communication restrictions by adapting to the changing number of agents in the dynamic environment. On the other hand, in the proposed method, the state space depends on local observations. Therefore, the growing action-state space is eliminated. This approach has helped produce an effective policy in less time while the designed reward structure enabled the generation of collective behavior without the need for reward sharing. Moreover, the proposed model-free method, which is based on local observations independent of central control, is capable of guiding real applications with its reward strategy.

References

1. Aydemir, F., Cetin, A.: Multi-agent dynamic area coverage based on reinforcement learning with connected agents. Computer Systems Science and Engineering 45(1), 215–230 (2023)
2. Cabreira, T.M., Ferreira, P.R., Franco, C.D., Buttazzo, G.C.: Grid-based coverage path planning with minimum energy over irregular-shaped areas with uavs. In: 2019 International Conference on Unmanned Aircraft Systems (ICUAS). pp. 758–767 (2019)
3. Canese, L., Cardarilli, G.C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., Spanò, S.: Multi-agent reinforcement learning: A review of challenges and applications. Applied Sciences 11(11) (2021)
4. Deng, L., Yu, D.: Deep learning: Methods and applications. Found. Trends Signal Process. 7(3-–4), 197—387 (2014)
5. Dorri, A., Kanhere, S.S., Jurdak, R.: Multi-agent systems: A survey. IEEE Access 6, 28573–28593 (2018)
6. Fan, J., Wang, Z., Xie, Y., Yang, Z.: A theoretical analysis of deep q-learning. arXiv (2019), [Online]. Available: https://arxiv.org/abs/1901.00137 (current December 2022)
7. Ganganath, N., Cheng, C.T., Tse, C.K.: Distributed antiflocking algorithms for dynamic coverage of mobile sensor networks. IEEE Transactions on Industrial Informatics 12(5), 1795–1805 (2016)
8. Ge, Y., Zhu, F., Huang, W., Zhao, P., Liu, Q.: Multi-agent cooperation q-learning algorithm based on constrained markov game. Computer Science and Information Systems 17(2), 647–664 (2020)
9. Gupta, H., Verma, O.P.: Monitoring and surveillance of urban road traffic using low altitude drone images: A deep learning approach. Multimedia Tools Appl. 81(14), 19683—-19703 (2022)
10. Gupta, S.K., Kuila, P., Jana, P.K.: Genetic algorithm approach for k-coverage and m-connected node placement in target based wireless sensor networks. Computers & Electrical Engineering 56, 544–556 (2016)
11. Hüttenrauch, M., Sosic, A., Neumann, G.: Local communication protocols for learning complex swarm behaviors with deep reinforcement learning. In: Dorigo, M., Birattari, M., Blum, C., Christensen, A., Reina, A., Trianni, V. (eds.) Swarm Intelligence, ANTS 2018. Lecture Notes in Computer Science(), vol. 11172, pp. 71–83. Springer, Cham, Rome, Italy
12. Jagtap, A.M., Gomathi, N.: Minimizing movement for network connectivity in mobile sensor networks: An adaptive approach. Cluster Computing 22(1), 1373—-1383 (2019)
13. Jain, R., Chiu, D.M., WR, H.: A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. CoRR (1998)
14. Kalantari, E., Yanikomeroglu, H., Yongacoglu, A.: On the number and 3d placement of drone base stations in wireless cellular networks. In: 2016 IEEE 84th Vehicular Technology Conference (VTC-Fall). pp. 1–6 (2016)

15. Kuo, Y.C., Chiu, J.H., Sheu, J.P., Hong, Y.W.P.: Uav deployment and iot device association for energy-efficient data-gathering in fixed-wing multi-uav networks. IEEE Transactions on Green Communications and Networking 5(4), 1934–1946 (2021)
16. Lee, H.R., Lee, T.: Multi-agent reinforcement learning algorithm to solve a partially-observable multi-agent problem in disaster response. European Journal of Operational Research 291(1), 296–308 (2021)
17. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. arXiv (2015), [Online]. Available: https://arxiv.org/abs/1509.02971 (current December 2022)
18. Liu, C.H., Chen, Z., Tang, J., Xu, J., Piao, C.: Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach. IEEE Journal on Selected Areas in Communications 36(9), 2059–2070 (2018)
19. Liu, C.H., Ma, X., Gao, X., Tang, J.: Distributed energy-efficient multi-uav navigation for long-term communication coverage by deep reinforcement learning. IEEE Transactions on Mobile Computing 19(6), 1274–1285 (2020)
20. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 6382—-6393. Curran Associates Inc., Red Hook, NY, USA (2017)
21. Mozaffari, M., Saad, W., Bennis, M., Debbah, M.: Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage. IEEE Communications Letters 20(8), 1647–1650 (2016)
22. Ndam Njoya, A., Ari, A., Awa, M., Titouna, C., Labraoui, N., Effa, Y., Abdou, W., Gueroui, A.: Hybrid wireless sensors deployment scheme with connectivity and coverage maintaining in wireless sensor networks. Wireless Personal Communications 112, 1893—-1917 (2020)
23. Nemer, I.A., Sheltami, T.R., Belhaiza, S., Mahmoud, A.: Energy-efficient uav movement control for fair communication coverage: A deep reinforcement learning approach. Sensors 22(5), 1–27 (2022)
24. Qie, H., Shi, D., Shen, T., Xu, X., Li, Y., Wang, L.: Joint optimization of multi-uav target assignment and path planning based on multi-agent reinforcement learning. IEEE Access 7, 146264–146272 (2019)
25. Shi, W., Li, J., Xu, W., Zhou, H., Zhang, N., Zhang, S., Shen, X.: Multiple drone-cell deployment analyses and optimization in drone assisted radio access networks. IEEE Access 6, 12518–12529 (2018)
26. Shu, T., Dsouza, K.B., Bhargava, V., Silva, C.: Using geometric centroid of voronoi diagram for coverage and lifetime optimization in mobile wireless sensor networks. In: 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE). pp. 1–5 (2019)
27. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic policy gradient algorithms. In: 31st International Conference on Machine Learning, ICML 2014, Proceedings of Machine Learning Research. vol. 32, pp. 387–395 (2014)
28. Song, H., Lee, S.Y.: Hierarchical representation using nmf. In: Lee, M., Hirose, A.and Hou, Z., Kil, R. (eds.) Neural Information Processing, ICONIP 2013. Lecture Notes in Computer Science, vol. 8226, pp. 466–473. Springer
29. Valente, J., Sanz, D., Cerro, J., Barrientos, A., de Frutos, M.: Near-optimal coverage trajectories for image mosaicing using a mini quad-rotor over irregular-shaped fields. Precision Agriculture 14, 115—-132 (2013)
30. Woolley, A.W., Aggarwal, I., Malone, W.T.: Collective intelligence and group performance. Current Directions in Psychological Science 24(6), 420–424 (2015)
31. Xiao, J., Wang, G., Zhang, Y., Cheng, L.: A distributed multi-agent dynamic area coverage algorithm based on reinforcement learning. IEEE Access 8, 33511–33521 (2020)

32. Ye, Z., Wang, K., Chen, Y., Jiang, X., Song, G.: Multi-uav navigation for partially observable communication coverage by graph reinforcement learning. IEEE Transactions on Mobile Computing pp. 1–1 (2022)
33. Yue, Y., Cao, L., Luo, Z.: Hybrid artificial bee colony algorithm for improving the coverage and connectivity of wireless sensor networks 108, 1719—-1732 (2019)
34. Zoss, B.M., Mateo, D., Kuan, Y.K., Tokić, G., Chamanbaz, M., Goh, L., Vallegra, F., Bouffanais, R., Yue, D.K.: Distributed system of autonomous buoys for scalable deployment and monitoring of large waterbodies. Auton. Robots 42(8), 1669—1689 (2018)

Fatih Aydemir received the B.S. degree in computer engineering from 9 Eylul University, Turkey, and the M.S. degree in computer engineering from Gazi University, Turkey. He is currently pursuing the Ph.D. degree in computer engineering with the Gazi University. He is also Leader Software Engineer and working for STM Defence Technologies Engineering and Trade. Inc., Ankara, Turkey.

Aydin Cetin is currently with Gazi University Department of Computer Engineering. He received his MS degree from LSU in Electrical and Computer Engineering and PhD. Degree in Electrical Ed. from Gazi University. His current research areas of interest include AI, Smart and Autonomous things, Agent systems and optimization techniques. He is member of IEEE, IEEE Computational Intelligence and PE Societies, and IEEE Big Data, Cyber Security and IoT communities.

Received: December 22, 2022; Accepted: March 25, 2023.