Multiagent Collaboration for Maximizing Channel Awareness in Mobile Directional CRNs

Thi Thu Hien Pham, Arooj Masood, Thi My Tuyen Nguyen, Chunghyun Lee, and Sungrae Cho

***Abstract*—In spectrum sensing, secondary users (SUs) often face limitations in their sensing capabilities, making cooperation among SUs crucial. By leveraging collaboration, we aim to maximize comprehensive awareness of primary channel states while minimizing sensing overhead. To achieve this, we develop an adaptive algorithm that utilizes SU mobility and directional antennas, enabling operation in dynamic environments where prior network knowledge is unavailable. This study proposes a two-stage learning approach. In the first stage, a PU location estimation (PLE) algorithm assists SUs in identifying relevant sensing targets. In the second stage, a multiagent reinforcement learning-based algorithm is introduced to mitigate sensing collisions, thereby enhancing channel state awareness. Simulation results demonstrate that the proposed scheme outperforms baseline algorithms in convergence speed, sensing collision rate, and channel state awareness, offering an efficient solution for cognitive radio networks. Specifically, the proposed approach achieves up to an 81% and 24% increase in comprehensive channel awareness, and a 53% and 13% reduction in sensing overhead compared to the random scheme without and with PLE, respectively.**

***Index Terms*—Centralized training decentralized execution (CTDE), cognitive radio network (CRN), cooperative spectrum sensing (CSS), decentralized partially observable Markov decision process (Dec-POMDP), multiagent reinforcement learning (MARL)**

I. Introduction

By 2030, the global number of Internet of Things (IoT) devices is projected to nearly double, surpassing 29 billion from the 15.1 billion recorded in 2020 [1]. This rapid expansion underscores the urgent need for innovative solutions to mitigate spectrum scarcity. In this regard, cognitive radio networks (CRNs) [2] have emerged as a promising paradigm, enabling secondary users (SUs) to opportunistically access underutilized portions of the spectrum allocated to licensed primary users (PUs). To enhance spectrum sensing efficiency, cooperation among SUs is crucial. Hence, cooperative spectrum sensing (CSS) [3] has been widely recognized in the CRN domain for its ability to leverage the spatial diversity of multi-SU environments, thereby improving cooperative gain. Numerous CSS approaches have been proposed in the literature [4]–[23], each employing distinct coordination mechanisms. Centralized approaches, as highlighted in studies such as [4]–[13], rely on a central node for coordination, which may introduce potential bottlenecks. Additionally, the selection between omnidirectional antennas [5]–[8], [14]–[16] and directional antennas (DA) [4], [11], [17] presents further challenges. Most existing approaches [4]–[6], [10], [14], [15], [17]–[20], [22], [23] focus on stationary environments where both SUs and PUs remain spatially fixed, highlights the need for exploring CSS in mobile CRN scenarios. Decentralized or distributed CSS schemes have been proposed to reduce coordination costs by distributing tasks across multiple SUs to address coordination issues in centralized methods. Moreover, given the substantial advantages of DA in terms of accuracy, efficiency, and connectivity [4], [24], [25], equipping SUs with such antennas is an auspicious direction. In scenarios where mobile directional SUs collaborate without central node coordination, selecting channels and sensing directions for cooperative spectrum sensing becomes increasingly challenging.

This study addresses the CSS challenge in a mobile, decentralized, and directional CRN, maximizing collective channel state awareness while minimizing unnecessary sensing. Rather than evaluating sensing accuracy metrics such as false alarm and miss detection probabilities, we focus on selecting the appropriate sensing target–specifically, the direction and channel. This step is as crucial as managing sensing accuracy since an SU that repeatedly selects the wrong direction or channel gains no benefit, even with perfect sensing accuracy. For instance, if the SU consistently senses in a direction that does not cover the PU or in channels that the PU is not using, achieving a false alarm probability of 0 or a miss detection probability of 0 becomes ineffective. Moreover, due to the inherent limitations of individual sensing capabilities, SUs must cooperate to assess the status of as many channels as possible before excluding the busy ones from use. The relevance of accuracy metrics such as false alarm and miss detection probabilities arises only when SUs collectively decide the proper sectors and channels to sense. Thus, sensing decisions must be made intelligently based on the SU's location while minimizing sensing collisions. To tackle this challenge, we propose a multiagent collaboration framework using multiagent reinforcement learning (MARL) to maximize comprehensive channel state awareness in mobile directional CRNs. We formulate a stochastic sequential optimization problem for sensing target selection and introduce a two-stage learning approach. The first stage estimates PU locations to establish foundational environmental knowledge, while the second stage mitigates sensing collisions. By leveraging a MARL strategy

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2023-00209125), and in part by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-RS-2022-00156353) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) *(Corresponding authors: Sungrae Cho).*

Thi Thu Hien Pham, Arooj Masood, Thi My Tuyen Nguyen, Chunghyun Lee, and Sungrae Cho are with the School of Computer Science and Engineering, Chung-Ang University, Seoul 156-756, Republic of Korea (email: ptthien@uclab.re.kr, arooj@uclab.re.kr, tuyen@uclab.re.kr, chlee@uclab.re.kr, srcho@cau.ac.kr).

with centralized training and decentralized execution (CTDE), our method dynamically optimizes channel-PU pair selection, enhancing spectrum awareness and reducing sensing overhead, while enabling SUs to make efficient and autonomous sensing decisions without the need for centralized coordination. The key contributions of this study are as follows:

- We highlight the challenge of maximizing the comprehensive awareness rate of primary channel states owned by the PUs while accounting for SUs' sensing limitations. This essential aspect has not been explored in prior research.
- We investigate this challenge in a dynamic environment where SUs mobility and DA offer substantial benefits to the SU network [24]–[28], yet remain largely unexplored.
- We formulate a stochastic sequential optimization problem to dynamically select channel-PU pairs for sensing, aiming to improve channel state awareness while minimizing unnecessary sensing overhead.
- To solve this problem, we propose a two-stage learning approach: (1) a PU location estimation (PLE) algorithm to help SUs identify relevant sensing targets and (2) a sensing collision mitigation (SCM) algorithm to improve channel state awareness.
- Once the estimated PU location is obtained from the first learning stage, an SCM algorithm is proposed to determine the optimal sensing channel-PU pair, allowing SUs to maximize their comprehensive awareness rate while minimizing sensing overhead. The algorithm employs the CTDE-based MARL algorithm called QMIX [29].
- We validate the proposed scheme through simulations, demonstrating its superiority over baseline algorithms regarding convergence performance, sensing collision rate, and channel state awareness. Notably, it achieves up to an 81% and 24% increase in comprehensive channel awareness rate, and a 53% and 13% reduction in sensing overhead compared to the random scheme without and with PLE, respectively.

The paper is organized as follows. Section II reviews related research, and Section III introduces the system model. The problem formulation is presented in Section IV. Section V explains the proposed solution, including the PLE method and SCM algorithm. Experimental results and data analysis are discussed in Section VI. Finally, Section VII concludes the paper.

II. Related work

A. Cooperative Spectrum Sensing

CSS occurs when multiple SUs collaborate to enhance a common objective, namely cooperative gain [3]. Cooperative gain can manifest in various forms, including sensing accuracy [5]–[7], [12], [13], [16], [17], [19], [21], [23], sensing overhead [4], [10], [17], sum throughput [8], [10], [22], and channel availability [14], [15]. CSS methods are typically categorized into three main approaches: centralized, decentralized, and distributed. In centralized CSS [4]–[13], a fusion center (FC) coordinates sensing operations by collecting and aggregating sensing reports from all SUs. In contrast, decentralized CSS [14]–[21] eliminates the need for an FC while requiring SUs to exchange information for cooperative decision-making. On the other hand, distributed CSS [22], [23] enables SUs to operate independently without communication, relying on pre-trained policies to make sensing decisions. Earlier literature often groups decentralized and distributed schemes under the broad term "distributed."

In centralized CSS approaches, the coordination node is pivotal in determining the final sensing result and optimizing sensing parameters for subsequent steps across all SUs. Consequently, research has predominantly focused on optimizing collaborative sensing parameters for SUs [4] or developing fusion algorithms [5], [6], [10]. Concretely, parameter optimization strategies focus on refining collaborative sensing parameters. For instance, [4] introduced a cooperative scheme in which the FC employs a numerical gradient optimization algorithm to determine the optimal sensing parameters, including sensing duration, threshold, channel, and beam selection. The objective is to maximize PU detection probability while minimizing sensing overhead. In contrast, the primary objective of fusion schemes is to enhance sensing performance by aggregating data from multiple SUs, which sense channels and relay their findings to the FC. The FC then applies predefined or adaptive decision rules to determine the outcome. Notably, [5] utilized a convolutional neural network (CNN) for fusion, while [10] introduced a weighted fusion rule based on the results from evaluating the reliability of historical sensing information of all SUs. [6] implemented a deep Q-learning framework with a CNN-based Q-network to learn an optimal fusion function.

Although centralized approaches leverage spatial diversity among SUs to improve cooperation gains, they suffer from high operational costs and potential bottlenecks at the central node. Decentralized and distributed CSS schemes have been developed to address these limitations. Decentralized CSS strategies eliminate the need for an FC while maintaining effective cooperation among SUs. Researchers in [14], [15] proposed a multiagent deep deterministic policy gradient-based framework, enabling SUs to dynamically select cooperation partners and optimize channel selection to increase the likelihood of detecting available channels. In contrast, distributed CSS approaches ensure cooperation by allowing SUs to undergo centralized training before deployment, facilitating mutual learning without real-time information exchange during operation. For example, [22] applied the CTDE-based QMIX algorithm [29], where SUs strategically select sensing and access channels to maximize sum throughput while avoiding collisions with PUs and other SUs. Moreover, [23] proposed a collaborative model fusion technique that enables SUs to train band-specific sub-networks locally and, after fusion into a unified model, operate in a fully distributed manner for efficient and accurate wideband spectrum sensing. Despite their advantages, these approaches are primarily designed for conventional CRNs with stationary SUs using omnidirectional antennas. However, as mobility becomes a defining feature of next-generation wireless networks and millimeter-wave technologies gain prominence, these models may require substantial adaptations to remain effective.

TABLE I: Summary of cooperative spectrum sensing (CSS) approaches.

Reference	CSS scheme	Cooperation gain	Mobile SUs	DA	DRL-based
[4]	Centralized	Sensing overhead, PU detection probability		✓	
[5], [6]	Centralized	Sensing accuracy			✓
[7]	Centralized	Sensing accuracy	✓		
[8]	Centralized	Sum throughput	✓		
[9]	Centralized	Sum throughput	✓		✓
[10]	Centralized	Sensing overhead, Sum throughput			
[11]	Centralized	PU detection probability	✓	✓	
[12], [13]	Centralized	Sensing accuracy	✓		
[14], [15]	Decentralized	Channel availability			✓
[16]	Decentralized	Sensing accuracy	✓		✓
[17]	Decentralized	Sensing overhead, Sensing accuracy		✓	
[18]	Decentralized	Sum throughput			✓
[19]	Decentralized	Sensing accuracy			✓
[20]	Decentralized	Sum throughput, Fairness			✓
[21]	Decentralized	Sensing accuracy	✓		✓
[22]	Distributed	Sum throughput			✓
[23]	Distributed	Sensing accuracy			
Ours	Decentralized	Channel state awareness, Sensing overhead	✓	✓	✓

B. Cooperative Spectrum Sensing with Directional Antennas and Mobile Cognitive Radio Networks

DAs have recently gained popularity for their ability to steer and precisely concentrate signals. Studies indicate that the directional CRN model enhances SU connectivity and improves spectrum availability detection compared to the omnidirectional model [24], [25]. Despite these advantages, research on CSS in directional CRNs remains limited. One notable study [4] explored this scenario by leveraging spatial characteristics to reduce sensing overhead, allowing multiple SUs to sense distinct directions simultaneously and rapidly determine the PU's location. Similarly, [17] utilized spatial correlation among nearby SUs to further minimize sensing overhead and enhance accuracy by maximizing the overlap of their sensing beams. On the other hand, [11] proposed a cooperative spectrum sensing system that optimizes SU positions and their directional antenna orientations to maximize energy detection performance.

Mobility is another crucial factor in wireless networks. Within CRNs, mobility substantially impacts key performance metrics such as transmission time, misdetection rate, and error probability [26], [27]. A study by [28] demonstrated that mobility in CRNs reduces misdetection probability and improves sensing scheduling efficiency. However, research on CSS for mobile CRNs remains scarce due to the complexities introduced by network dynamics. A recent study [7] proposed a dynamic double-sensing threshold scheme, enabling all SUs to achieve a more accurate final sensing result through a fusion process at the FC. Other studies [8] have examined mobile CRNs as an energy-harvesting network, focusing on maximizing SU throughput under energy constraints. Additionally, [16] investigated mobile CRNs where SU detection capabilities vary due to mobility, introducing a deep reinforcement learning-based cooperation scheme to optimize sensing accuracy. [12] developed an online learning-based CSS algorithm that dynamically responds to mobility-induced variations in signal quality and node reliability due to their movement. On the other hand, [13] introduced a path planning algorithm that directs PUs toward areas with higher spectrum prediction uncertainty, which ensures more informative data collection with minimal redundancy, resulting in improved sensing accuracy.

Research into CSS schemes for mobile, directional CRNs presents a promising avenue. Furthermore, the combination of mobility and DAs enables SUs to identify and exploit spectrum opportunities better, achieving more effective utilization than static or omnidirectional models. Despite this potential, CSS in mobile, directional CRNs remains an unexplored area of research. Motivated by these advantages, this study introduces a novel decentralized CSS scheme tailored to this unique model. The proposed scheme uses the CTDE architecture and the QMIX algorithm to enhance channel state awareness while minimizing sensing overhead by eliminating redundant sensing operations. Table I summarizes the key distinctions between our proposed method and existing approaches.

III. System Model

A. Network and Antenna Model

This study considers a primary network of U stationary PUs and M mobile SUs. The SUs operate in a decentralized manner, forming a dynamic network that communicates directly without a central base station. Each SU is equipped with directional antennas, which can be implemented using beamforming-enabled antenna arrays, for communication and spectrum sensing. These SUs move following the random waypoint mobility model [30], with all SUs aware of each others' locations. SUs can detect and utilize available spectrum bands left unused by PUs. We assume that PUs operate on K licensed orthogonal channels and use omnidirectional antennas, whereas SUs leverage directional antennas for improved performance. The benefits of using directional antennas in decentralized mobile networks have been demonstrated in previous studies, such as [31]–[35], highlighting their advantages in similar scenarios. We define L as the number of sectors per SU. In practice, directional antennas consist of a main beam and several relatively insignificant side lobes. Thus, without loss of generality, we ignore the side lobes and adopt a sector-based model, where each sector has a width of $\phi = \frac{2\pi}{L}, \phi < \pi$.

SU sectors are indexed relative to the environment (i.e., North, South, East, and West) rather than the SU itself.

The area containing SUs and PUs is defined as $z \times z$. Additionally, s represents the sensing range of SUs, while p denotes the protection range of PUs. A moving SU may cause a PU to enter or exit its sensing range. Thus, for each SU, it might fall into one of the following two key events:
I : A PU enters the sensing range of a moving SU.
O : No PU is within the sensing range of the moving SU.

B. PU Traffic Model and SU Spectrum Sensing Method

We assume that all PUs in the network are all hopping nodes [36] and utilize a channel-hopping mechanism for their transmissions. At each time instant, a PU switches to a different channel based on a predefined hopping pattern [37]. Notably, the channel-hopping behavior of each PU is modeled as a Markov chain [38]. A Markov chain consists of a fixed number of states K, each representing a specific channel. The transition from one channel (state) to another is governed by a transition probability matrix, $\mathbf{P} = [P_i]_{0 \leq i < K}, P_i = [P_j]_{0 \leq j < K}$, where $\forall i, \sum_j P_{ij} = 1$. Notably, the SU has no prior knowledge of this matrix.

For spectrum sensing, we employed an energy detection method, which is widely recognized as a computationally efficient technique in cognitive radio systems for detecting whether a channel is occupied (busy) or idle [3]. There have been previous hypothesis testing models, such as those proposed in [39], [40], where the authors introduced a dual-threshold approach: one threshold distinguishes between white and gray opportunities, while the other differentiates between gray and black opportunities. In contrast, our approach adopts a binary hypothesis testing model with a single threshold to classify channel states as white (available) or black (occupied). This approach was chosen for its simplicity, ensuring efficient processing while meeting our objectives. It is important to note that the primary aim of our algorithm is to enhance the comprehensive awareness rate of SUs—specifically, to maximize the SU's ability to gather as much information as possible about channel states. Notably, this goal of maximizing awareness remains independent of the hypothesis testing method used for sensing. Whether a binary hypothesis testing model with a single threshold or a multi-threshold method [39], [40] is employed, the number of channels an SU can sense remains unchanged. The choice of hypothesis testing method primarily affects how we classify each channel state (i.e., as white or black) but does not influence the SU's sensing capability. Moreover, in single-threshold spectrum sensing, the decision between H_1 (occupied) and H_0 (idle) is based solely on whether the received signal energy λ exceeds the detection threshold. Therefore by adjusting this detection threshold, we can control how weak signals (gray opportunities) are classified. Let us denote the sensed channel bandwidth by w, the sensing duration by τ, and the sampling rate by $f_{sampling}$. The number of samples that can be analyzed within one sensing slot is given by $N = f_{sampling}\tau = 2w\tau$. In event I, let $y_I(\sigma)$ represent the received signal energy at any SU. The occupancy state of PUs at any SU is then formulated as a binary hypothesis testing problem as follows:

$$y_I(\sigma) = \begin{cases} \sum_{\sigma=1}^{N} |u_\sigma|^2, & H_0, \\ \sum_{\sigma=1}^{N} |hs_\sigma + u_\sigma|^2, & H_1, \end{cases} \tag{1}$$

where H_0 and H_1 represent the channel states when it is idle and busy, respectively. Moreover, $u_\sigma = u_\sigma^r + ju_\sigma^i$ denotes the complex Gaussian noise sample at time index σ, with u_σ^r and u_σ^i being its real and imaginary parts, respectively. $s_\sigma = s_\sigma^r + js_\sigma^i$ denotes the transmitted complex signal sample at time index σ, with s_σ^r and s_σ^i being its real and imaginary parts, respectively, and $h = h^r + jh^i$ denotes the complex channel coefficient. Therefore, we attain $(u_\sigma)^2 = (u_\sigma^r)^2 + (u_\sigma^i)^2$, and $(hs_\sigma + u_\sigma)^2 = (h^r s_\sigma^r - h^i s_\sigma^i + u_\sigma^r)^2 + (h^r s_\sigma^i + h^i s_\sigma^r + u_\sigma^i)^2$.

With respect to event O, the received signal energy at any SU is given by $y_O(\sigma)$ as follows:

$$y_O(\sigma) = \sum_{\sigma=1}^{N} |u_\sigma|^2, \text{under both } H_1, H_0. \tag{2}$$

The test statistic is represented as $\lambda = \frac{1}{N}|y(\sigma)|$, which is then compared to a predefined detection threshold to determine whether any PUs occupies the channel.

C. SU Transmission Protocol and Time Frame Structure

We assume that SUs employ a directional routing protocol, as described in [41], where each SU maintains a local table of its directional neighbor routes. When they transmit, they determine the destination and then use the table to search for the shortest path. In the proposed solution, before transmitting, SUs must lock the sectors covering any PUs, along with the channels currently occupied by those PUs. This process modifies the initial routing table by eliminating routes that pass through these locked sectors and channels. Our proposed algorithm enhances the SU's awareness of the state of all channels associated with nearby PUs. As a result, the SU gains knowledge of which specific PUs currently occupy channels. By leveraging this information and PU location data, the SU can accurately determine which channels and sectors should be locked. In summary, the primary objective is for SUs to identify channel-PU pairs that are currently occupied. This ensures that SUs avoid transmitting on sectors covering those PUs and the channels they use, thereby minimizing interference with the primary network.

All SUs are synchronized in a time-slot manner, where each time slot is referred to as a period. Each period consists of three phases: sensing, sharing, and transmission. In every time slot, one SU begins by sensing a set of k specific channel-PU pairs during the sensing phase. The SU then shares its sensing results with other SUs in its proximity, particularly those with sectors covering the same PUs. Subsequently, each SUs determines which channel-sector pairs should be locked based on the gathered information. Finally, the SUs select one of the remaining channel-sector pairs according to their transmission protocol for sending packets during the transmission phase. In event O, SUs do not need to sense any channels because no PUs are within their sensing range. As a result, locking channels and sectors is unnecessary in this case. Therefore, SUs can utilize the entire time-slot duration

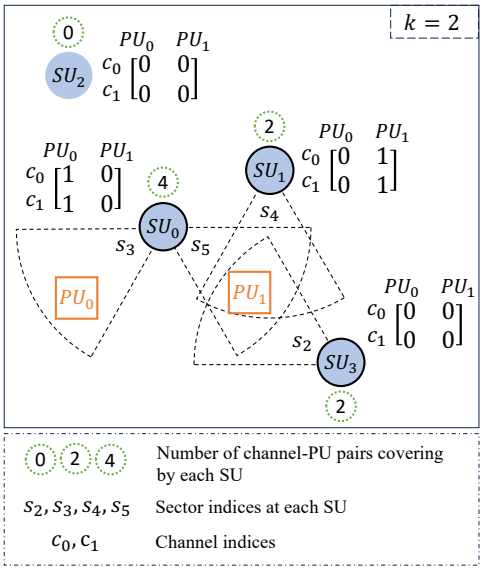

Fig. 1: An illustration of the optimal sensing decisions by secondary users.

for transmission, reducing sensing overhead and potentially increasing transmission throughput.

IV. Problem Formulation

We consider a cognitive radio access network environment consisting of U stationary PUs, each owning K primary channels, and coexisting with M mobile SUs that follow a random waypoint mobility model. At each time step, the SUs relocate to new positions, where they either experience event I (when one or more PUs fall within the SU's sensing range) or event O (when no PU is within the SU's sensing range). Since each SU can sense a maximum of k channels, and the number of channel-PU pairs within its sensing range often exceeds k, our objective is to develop a cooperation solution that maximizes the SUs' awareness of the states of the channel-PU pairs they cover while minimizing sensing overhead. Specifically, when SUs are in an event O, they are expected to forgo sensing and use the entire time-slot duration for transmission. Conversely, in event I, SUs must select k channel-PU pairs to sense, ensuring that the chosen PUs fall within their sectors. Additionally, the selected channel-PU pairs should differ from those sensed by nearby SUs covering the same PUs. In other words, SUs with overlapping coverage should avoid sensing collisions in terms of both channel and PU indices to maximize knowledge acquisition regarding the surrounding channel-PU states.

As illustrated in Fig. 1, no PUs are within the sensing range of SU_2 (i.e., in event O); thus, SU_2 should allocate the entire period on transmission. In contrast, two PUs are within the sensing range of SU_0 (i.e., PU_0 and PU_1), and one PU is within the sensing ranges of SU_1 and SU_3 (i.e., PU_1). Therefore, SU_0, SU_1, and SU_3 form a temporal group because they cover the same PU (i.e., PU_1). Consequently, these three SUs should ensure no sensing collision on PU_1. Let c_0 and c_1 represent the indices of two distinct channels, where c_0 corresponds to the first channel index and c_1 corresponds to the second channel index. An example of an optimal sensing assignment for all SUs is as follows:

- SU_0 senses channel c_0 and c_1 w.r.t. PU_0.
- SU_1 senses channels c_0 and c_1 w.r.t. PU_1.
- SU_2 skips sensing for this time slot.
- SU_3 skips sensing for this time slot.

Through this cooperation, SU_0, SU_1, and SU_3 can collectively obtain complete information about the surrounding channel-PU states by sharing their sensing results after the sensing phase. Moreover, SU_2 can conserve its sensing energy since there is no PU in its sensing range, SU_3 can also reduce its sensing effort while still benefiting from the shared information. Ultimately, mitigating sensing collisions among SUs within the same group is an effective approach to enhance comprehensive awareness of channel-PU activity, while also helping to reduce sensing overhead by avoiding unnecessary individual sensing efforts.

At each time step, the network consists of serveral temporal SU groups which are formed dynamically by SUs. Let $\mathcal{G}_t^m$ denote the temporal group formed by SU_m, which consists of SU_m itself and other SUs covering the same PUs at time step t. At each time step, every SU in $\mathcal{G}_t^m$ must decide whether to skip sensing or, if sensing is chosen, which specific channel–PU pair to sense. We define $c_t^{m,i,j}$ to represent a single decision made by SU_m on a single channel-PU pair, where i and j correspond to the channel and PU indices, respectively. Let $\mathbf{c}_t^m = [c_t^{m,i,j}]_{0\le i<K, 0\le j<U}$ denote the decision matrix made by SU_m on all channel-PUs pairs at the time step t. Eventually, the number of channel–PU pair states that SU_m can observe after performing sensing and sharing the results with other members of the group $\mathcal{G}_t^m$ is given by:

$$NoP_t^m = \sum_{i=0}^{K-1} \sum_{j\in\mathcal{U}_t^m} \mathbf{1}(c_t^{m,i,j} > 0) \tag{3}$$

where $c_t^{m,i,j} = 1$ indicates that SU_m selects channel i and PU_j for sensing, and $c_t^{m,i,j} = 0$ otherwise. With $\mathcal{U}_t^m (0 \le |\mathcal{U}_t^m| \le U)$ denotes the set of PUs covered by SU_m at time step t, we have $j \in \mathcal{U}_t^m$, which implies that the chosen sensing PU_j falls within the sensing range of SU_m. The total number of channel-PU pairs covered by SU_m is then given by $K|\mathcal{U}_t^m|$, where K is the total number of primary channels and $|\mathcal{U}_t^m|$ is the cardinality of the set $\mathcal{U}_t^m$. Accordingly, the comprehensive awareness rate of SU_m at time step t is expressed as:

$$\omega_t^m = \frac{NoP_t^m}{K|\mathcal{U}_t^m|}. \tag{4}$$

Thus, the problem is formulated as follows:

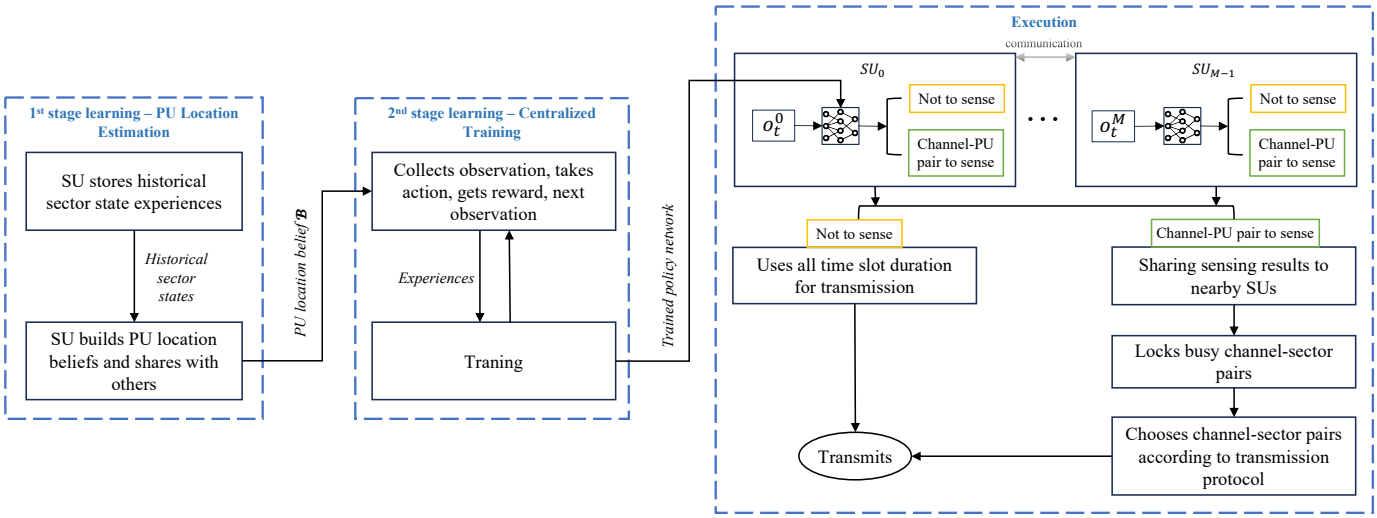

Fig. 2: Workflow of the proposed two-stage learning solution.

$$\max_{\mathbf{C}_t=\{\mathbf{c}_t^m\}} \quad \sum_{t\geq 0}\sum_{m=0}^{M-1}\sum_{i=0}^{K-1}\sum_{j=0}^{U-1}\omega_t^m \mathbf{1}_{\{I,j\}}(d_t^{m,j})c_t^{m,i,j} + (1-\mathbf{1}_{\{I,j\}}(d_t^{m,j}))(1-c_t^{m,i,j}) \tag{5}$$

$$\text{s.t.} \quad c_t^{m,i,j} \in \{0,1\}, \forall m,i,j, \tag{5a}$$

$$\sum_{i=0}^{K-1}\sum_{j=0}^{U-1} c_t^{m,i,j} \in \{0,k\}, \tag{5b}$$

where the first term of the equation represents the awareness rate of SU_m in event I. The indicator variable $\mathbf{1}_{\{I,j\}}(d_t^{m,j})$ determines whether event I applies to PU_j at SU_m, and $d_t^{m,j} = \sqrt{(x_{SU_m} - x_{PU_j})^2 + y_{SU_m} - y_{PU_j})^2}$ denotes the current distance between SU_m and PU_j. The first term, $\omega_t^m \mathbf{1}_{\{I,j\}}(d_t^{m,j})c_t^{m,i,j}$, is activated when SU_m chooses to sense the ith channel of PU_j (i.e., $c_t^{m,i,j} = 1$), and PU_j is within the SU's sensing range (i.e., event I, $\mathbf{1}_{\{I,j\}}(d_t^{m,j}) = 1$). This term is weighted by ω_t^m, the awareness rate of SU_m, which promotes sensing actions that contribute to greater awareness at the group level (i.e., actions that can prevent overlapping sensing). Meanwhile, the second term aims to maximize the likelihood that sensing is skipped when the SU is not in event I. The second term, $(1-\mathbf{1}_{\{I,j\}}(d_t^{m,j}))(1-c_t^{m,i,j})$, is activated when PU_j is not within the sensing range of SU_m (i.e., event O, $\mathbf{1}_{\{I,j\}}(d_t^{m,j}) = 0$), and the SU correctly chooses to skip sensing (i.e., $c_t^{m,i,j} = 0$). This term rewards energy-efficient behavior and avoids unnecessary sensing attempts that cannot yield useful information. Moreover, $\mathbf{C}_t$ represents the joint decision of all SUs at time t. The decision matrix $\mathbf{c}_t^m = [c_t^{m,i,j}]_{0\leq i<K, 0\leq j<U}$ made by SU_m specifies whether it performs sensing. If sensing is selected, it further determines the channel-PU pairs to be sensed. Specifically, the constraint in (5b) ensures that an SU can either skip sensing or select exactly k channel-PU pairs from the total $K \times U$ pairs within each time slot t.

The optimization problem presented in (5) is challenging due to several reasons. First, SUs have no prior knowledge about the environment–specifically, the number and locations of PUs are unknown. Second, there is no central coordinator to assign sensing tasks among the SUs. Third, the movement of SUs constantly changes the environment, making the sensing environment highly dynamic. Lastly, SUs can only access partial observation of the environment. These challenges collectively render traditional optimization methods impractical for solving the problem in a decentralized manner at every time step. Eq. (5), however, defines a cumulative objective function over time. More specifically, at each time step, the optimization aims to achieve two closely related objectives: (i) maximizing the comprehensive awareness rate ω_t^m, and (ii) increasing the chance of skipping sensing when the PU is not in event I. As defined in Equation (4), the awareness rate is calculated as the ratio between the number of channel-PU pairs that SU_m is actually aware of (i.e., NoP_t^m) and the total number of pairs it could potentially be aware of based on its coverage (i.e., $K|\mathcal{U}_t^m|$). Since the denominator (that is, $K|\mathcal{U}_t^m|$) is influenced by the environment and cannot be controlled directly, to maximize the comprehensive awareness rate, the actual number of channel–PU pairs that SU_m is aware of (i.e., NoP_t^m) should be maximized. This can be achieved when each SU collects non-redundant channel-PU observations. When multiple SUs sense the same channel-PU

pair, sensing collision occurs. Hence, to improve the comprehensive awareness rate, optimization must encourage different and non-redundant sensing in the same group $\mathcal{G}_t^m$. At the same time, SUs must also determine whether they are in event I (i.e., whether any PU is nearby) or not. This requirement implies that SUs must estimate the location of PUs before making sensing decisions. Based on these requirements, the original problem is decomposed into two sub-problems: (i) PU location estimation (PLE), enabling each SU to determine if it should sense at all, and (ii) sensing collision mitigation (SCM), ensuring that SUs in a group select a different subset of channel–PU pairs to sense. To address the first sub-problem, we propose a PLE method, detailed in Sub-section V-A. Subsequently, we employ a deep MARL approach to solve the second sub-problem. Specifically, we reformulate the objective as a decentralized partially observable Markov decision process (Dec-POMDP) [42], in Sub-sections V-B and V-C, and present the proposed SCM algorithm to solve it.

V. Proposed Solution

This section presents a two-stage learning solution to address the above problem. The overall process is depicted in Fig. 2, where the SUs estimate PU locations in the first stage and train their policy network according to the proposed solution, SCM, in the second stage. Once deployed (i.e., execution phase), all SUs acquire enough knowledge and a well-trained policy that helps them operate independently with limited information sharing from neighboring SUs that cover the same PUs. Concretely, after running the PLE algorithm, SUs estimate the locations of all PUs. This shared location information enables SUs to identify PUs within their sensing range. Additionally, since PU indices remain consistent across all SUs, each SU can determine which PUs fall within its sensing range upon relocation. Our cooperative sensing scheme then leverages this location data to optimize channel selection intelligently. For instance, if two SUs cover the same PU with four channels but one SU can sense only two, the scheme ensures they select different channels to avoid overlap. Once an SU selects a channel-PU pair, it must sense the PU's signal in the corresponding sector covering that PU. The SUs then collectively acquire knowledge about all four channels by sharing their sensing results. First, Section V-A explains the PU estimation method (i.e., the first learning stage). After that, section V-C introduces the SCM algorithm (i.e., the second learning stage).

A. Primary User Location Estimation

The objective is to enable SUs to estimate PU locations by leveraging their mobility and spatial sensing capabilities, thereby allowing them to skip sensing when no PU is nearby (i.e., when the system is not in event I). We assume that any arbitrary SU within the network can estimate the PU locations. The estimated position information is then shared with all other SUs in the network. Since PUs are assumed to be stationary, this location information remains valid over a long duration and does not require frequent re-estimation. To estimate PU locations, the algorithm leverages the spatial mobility of SUs to identify potential regions where PUs may exist, which are referred to as PU location beliefs. During each episode, these beliefs are refined by eliminating areas that are believed to contain no PUs, continuing until the belief region falls below a predefined threshold. At the start of each episode, the SU collects historical sensing data over a defined time window. Using this data, it constructs two sets: the PU-covering sector set, containing regions likely to include PUs, and the non-PU-covering sector set, containing regions believed to be PU-free. The PU location beliefs are then updated by intersecting the current belief regions with the PU-covering sector set and removing the non-PU-covering sector set. This process gradually narrows down the potential locations of PUs. Let $S^{m,l}$ denote a particular sector of SU_m, where $l \in [0, L)$ represents the sector index. Then, we obtain the following:

$$\begin{aligned} S^{m,l} =& \{p \mid (x_p - x_t^m)^2 + (y_p - y_t^m)^2 \leq s^2, \\ & \beta^s \leq \arctan 2(y_p - y_t^m, x_p - x_t^m) \leq \beta^s + \phi\}, \end{aligned} \tag{6}$$

where s is SU's sensing range, p denotes a particular point with coordinates (x_p, y_p), β^s represents the initial angle, measured counterclockwise from the positive x-axis to the first radius of the lth sector, and (x_t^m, y_t^m) are the current coordinates of SU_m. The sector width is given by $\phi = \frac{2\pi}{L}$. We define F as the set of points bounded by an arbitrary simple closed curve within the considered zone and express the intersection with sector $S^{m,l}$ as: $F \cap S^{m,l} = \{p | p \in F, p \in S^{m,l}\}$. Similarly, the difference between F and $S^{m,l}$ is given by: $F \setminus S^{m,l} = \{p | p \in F, p \notin S^{m,l}\}$. The area of F can be computed using the shoelace theorem as follows:

$$A(F) = \frac{1}{2}\left| \sum_{i=1}^{f} x_i y_{i+1} + x_f y_1 - \sum_{i=1}^{f} x_{i+1} y_i - x_1 y_f \right|, \tag{7}$$

where (x_i, y_i) are the coordinates of the ith point in the convex hull set C_F of F, which is determined using the Graham Scan algorithm [43], and $f = |C_F|$.

In the proposed solution, the SU maintains an estimation belief, denoted as $\mathcal{B}$, representing an approximation of the PU locations. After each movement, the SU selects a sector for comprehensive channel sensing and records historical sector data, including the chosen sector index and associated PU coverage status. The primary user location estimations (PLEs) are updated based on this collected information. The detailed PU estimation process is outlined in Algorithm 1.

This algorithm consists of E_f episodes with T_f time steps. At each time step:

1) The SU relocates to a new position.
2) It selects a sensing sector $S_t^{m,l}$ based on its current local estimation beliefs, prioritizing sectors that have the greatest intersection with those beliefs (*Lines 4–15*).
3) The experiences accumulated over T_f time steps are divided into two distinct sets:
   - PU-covering sector set $\mathcal{S}_t^c$.
   - Non-PU-covering sector set $\mathcal{S}_t^n$ (*Lines 21–22*).
4) The $\mathcal{S}_t^c$ set is subsequently used to update $\mathcal{B}$ by calculating the intersection between these sectors (*Lines 23–29*).

Algorithm 1 Primary user location estimation (PLE) algorithm

1: Initialize $e = 0$
2: **repeat**
3: **while** $t < T_f$ **do**
4: SU_m moves to a new way point
5: **if** $\mathcal{B} = \varnothing$ **then**
6: Randomly chose a sensing sector $S_t^{m,l}, l < L$
7: **else**
8: **for each** $S^{m,l}, l < L$ **do**
9: $I_l \leftarrow S^{m,l} \cap (\bigcup \mathcal{B})$
10: **end for**
11: **if** $\exists \operatorname{argmax}_l A(I_l)$ **then**
12: $l \leftarrow \operatorname{argmax}_l A(I_l)$
13: **else**
14: $S_t^{m,l} \leftarrow \varnothing$
15: **end if**
16: **end if**
17: **if** $S_t^{m,l} \neq \varnothing$ **then**
18: Perform sensing with sector $S_t^{m,l}$
19: Store $S_t^{m,l}$ with its PU-covering status
20: **end if**
21: **end while**
22: $\mathcal{S}_t^c \leftarrow \{S_t^{m,l} | S_t^{m,l} \ni$ PU$\}$
23: $\mathcal{S}_t^n \leftarrow \{S_t^{m,l} | S_t^{m,l} \not\ni$ PU$\}$
24: **for each** $S_t^{m,l} \in \mathcal{S}_t^c$ **do**
25: **if** $\forall B_k \in \mathcal{B}, S_t^{m,l} \cap B_k = \varnothing$ **then**
26: $\mathcal{B} \leftarrow \mathcal{B} + S_t^{m,l}$
27: **else if** $\exists B_k \in \mathcal{B}, |S_t^{m,l} \cap B_k| = 1$ **then**
28: $B_k \leftarrow S_t^{m,l} \cap B_k$
29: **end if**
30: **end for**
31: **for each** $S_t^{m,l} \in \mathcal{S}_t^n$ **do**
32: **if** $\forall B_k \in \mathcal{B}, B_k \setminus (S_t^{m,l} \cap B_k) = \varnothing$ **then**
33: Delete B_k
34: **else**
35: $B_k \leftarrow B_k \setminus (S_t^{m,l} \cap B_k)$
36: **end if**
37: **end for**
38: $e \leftarrow e + 1$
39: **until** $e = E_f$ or $\forall B_k \in \mathcal{B}, A(B_k) < \mu$

5) The estimation belief $\mathcal{B}$ is further updated by substracting non-PU-covering sectors in $\mathcal{S}_t^n$ (*Lines 30–36*).
6) The algorithm stops updating when all element areas in $\mathcal{B}$ reach the threshold μ (*Line 37*). At this stage, $\mathcal{B}$ consists of relatively small sets. The final PLEs are determined by computing the center points of all these sets, with the total number of sets in $\mathcal{B}$ corresponding to the estimated total number of PUs in the area.

B. Decentralized Partially Markov Decision Process

The optimization objective in (5) aims to maximize the cumulative comprehensive awareness rate of all SUs, while also increasing the likelihood of skipping sensing when an SU is not in event I over time. In this context, the sensing decision of each SU at every time step serves as the decision variable. The sequence of decision-making steps taken by the SUs, including their interactions with the environment and the subsequent actions, aligns with the Markov game framework, as it involves multiple agents (SUs) making sequential decisions (i.e., sensing decisions) based on the current state of the environment. These sensing decisions represent the agents' actions, which directly influence future states and rewards. Consequently, the problem fits within the Markov game framework, defined by its key components—agents, states, actions, transitions, and rewards [44], [45]. In this framework, M SUs act as agents within a cooperative environment. Each SU obtains sensing results from itself and from temporary neighbors, which cover the same PUs. As a result, each SU makes decisions based solely on this localized information, which means that the environment is partially observable from the perspective of each SU. Given this partial observability, we then model the problem as a Dec-POMDP [42], defined as the tuple: $(\mathcal{M}, \mathcal{S}, \{\mathcal{A}^m\}_{m\in\mathcal{M}}, \mathcal{P}, \{r^m\}_{m\in\mathcal{M}}, \{\mathcal{O}^m\}_{m\in\mathcal{M}}, \{\mathcal{Y}^m\}_{m\in\mathcal{M}}, \gamma)$, where $\mathcal{M} = \{1, ..., M\}$ is the set of all agents, $\mathcal{S}$ represents the actual state of the environment, $\mathcal{A}^m$ is the action space of the mth agent, $\mathcal{P}(\mathbf{s}'|\mathbf{s}, a) : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ denotes the state transition probability, $r^m(\mathbf{s}, a, \mathbf{s}') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ represents the reward function for agent mth when it takes action a from state $\mathbf{s}$ to $\mathbf{s}'$, $\mathcal{O}^m$ signifies the partial observation space available to agent mth, $\mathcal{Y}^m : \mathcal{S} \to \mathcal{P}(\mathcal{O}_m)$ defines the observation channel that the mth agent uses to perceive its local knowledge, and $\gamma \in [0, 1)$ is the discount factor. At every time step t, the mth agent utilizes its partial observation $\mathbf{o}_t^m$ to execute an action a_t^m. The joint actions of all agents, $\{a_t^1, ..., a_t^M\}$, cause the environment to transition from state $\mathbf{s}_t$ to $\mathbf{s}_{t+1}$, while each agent receives a reward $r_t^m(\mathbf{s}_t, a_t, \mathbf{s}_{t+1})$. The object of the mth agent is to determine an optimal policy $\pi^m : \mathcal{O}^m \to \Delta(\mathcal{A}^m)$ that maximizes its expected long-term discounted reward [46]–[48].

1) Actual State of the Environment: The actual state of the environment includes all relevant information within the considered area, such as the sensing collision state of all SUs, their interconnections, and the coverage information of PUs. Note that this actual environmental state is not observable by the SUs during execution. It is only available during the training phase, where it can be used for learning purposes.

We denote the environment state at time t as $\mathbf{s}_t$, defined as:

$$\begin{aligned} \mathbf{s}_t &= \{\boldsymbol{\chi}_t, \boldsymbol{\mathcal{E}}_t, \boldsymbol{\rho}_t\}, \\ \text{where } \boldsymbol{\chi}_t &= [\chi_t^m], \quad \chi_t^m \in \{0, 1\}, \\ \boldsymbol{\mathcal{E}}_t &= [\mathbf{e}_t^m], \quad \mathbf{e}_t^m = [e_t^{m,n}], \quad e_t^{m,n} \in \{0, 1\}, \\ \boldsymbol{\rho}_t &= [\rho_t^u], \quad \rho_t^u \in [0, M], \\ &\quad \text{for } 0 \leq m, n < M, \quad 0 \leq u < U. \end{aligned} \tag{8}$$

The components of the state are defined as follows:

- **Actual Sensing Collision State $\boldsymbol{\chi}_t$**: A vector representing the actual sensing collision state of SU_m at the previous time step. Each element of the vector, χ_t^m, is a binary variable which equals 1 if SU_m experienced a sensing collision, and 0 otherwise. As a true environmental signal, χ_t^m is unobservable during execution and is available only during training for learning purposes.

$$\chi_t^m = \begin{cases} 1, & \text{if SU}_m \text{ experienced a sensing collision,} \\ 0, & \text{otherwise.} \end{cases}$$

- **SU Connection State $\boldsymbol{\mathcal{E}}_t$**: An $M \times M$ connectivity matrix indicating the relationship among SUs. If SU_m and SU_n are connected (i.e., if their current groups share at least one common member), then $e_t^{m,n} = 1$; otherwise, $e_t^{m,n} = 0$.

- **PU Coverage $\boldsymbol{\rho}_t$**: A vector where each element ρ_t^u represents the number of SUs currently covering a specific PU_u. The value ranges from 0 to M.

2) Partial Observations by Secondary Users: After executing an action, each SU relocates and receives a partial observation composed of the following information:

i. The observered sensing collision states of all SUs within its previous group,
ii. The identities of SUs in its current group (i.e., current neighborhood),
iii. The number of group members covering each PU that the current SU can observe based on estimated PU locations.

These observations are based on SUs' own estimations of PU locations, not the ground truth PU positions as defined in the system's actual state. For a given SU_m, the observation at time t is defined as:

$$\begin{aligned} &\mathbf{o}_t^m = \{\tilde{\boldsymbol{\chi}}_t^m, \tilde{\boldsymbol{\eta}}_t^m, \tilde{\boldsymbol{\rho}}_t^m\}, \\ \text{where} \quad &\tilde{\boldsymbol{\chi}}_t^m = [\tilde{\chi}_t^{m,n}], \quad \tilde{\chi}_t^{m,n} \in [-1, 1], \\ &\tilde{\boldsymbol{\eta}}_t^m = [\tilde{\eta}_t^{m,n}], \quad \tilde{\eta}_t^{m,n} \in \{0, 1\}, \\ &\tilde{\boldsymbol{\rho}}_t^m = [\tilde{\rho}_t^{m,u}], \quad \tilde{\rho}_t^{m,u} \in [0, |\mathcal{G}_t^m|], \\ &\text{with } 0 \le m, n < M, \quad 0 \le u < U. \end{aligned} \tag{9}$$

The components of the observation are defined as follows:

- **Observerd Sensing Collision State $\tilde{\boldsymbol{\chi}}_t^m$**: A vector where each element represents SU_m's observation of the collision state of another SU, SU_n at the previous time step. This observed variable can take one of three values: 1, 0, or –1. The value –1 is used to explicitly capture cases where SU_m lacks observability or information sharing with SU_n, particularly when they do not belong to the same group.
  - If SU_m was in event I at the previous time step:
    - $\tilde{\chi}_t^{m,n} = -1$ if SU_n did not belong to the previous group of SU_m (i.e., $n \notin \mathcal{G}_{t-1}^m$).
    - $\tilde{\chi}_t^{m,n} = 1$ if SU_n either experienced a sensing collision or chose not to sense incase some channels remain unsensed.
    - $\tilde{\chi}_t^{m,n} = 0$ if no sensing collision occurred for SU_n.
  - If SU_m was in event O at the previous time step:
    - $\tilde{\chi}_t^{m,n} = -1$ if $n \neq m$.
    - $\tilde{\chi}_t^{m,n} = 0$ if $n = m$ and SU_m chose not to sense; otherwise, $\tilde{\chi}_t^{m,n} = 1$.
- **Current Group Membership $\tilde{\boldsymbol{\eta}}_t^m$**: A vector where each element indicates whether SU_n is currently in the same group as SU_m.

$$\tilde{\eta}_t^{m,n} = \begin{cases} 1, & \text{if } n \in \mathcal{G}_t^m \text{ or } n = m, \\ 0, & \text{otherwise.} \end{cases}$$

- **Estimated PU Coverage $\tilde{\boldsymbol{\rho}}_t^m$**: A vector where each element represents SU_m's estimate of how many of its current group members are covering each PU that SU_m can also currently observe.

$$\tilde{\rho}_t^{m,u} = \begin{cases} 0, & \text{if } u \notin \mathcal{U}_t^m, \\ \sum_{m \in \mathcal{G}_t^m} \mathbf{1}(u \in \mathcal{U}_t^m), & \text{otherwise.} \end{cases}$$

3) Action Space of Secondary Users: Each agent can sense k channel-PU pairs or skip sensing at each time step. Thus, the action space has a size of: $\binom{U \times K}{k} + 1$. Accordingly, the action taken by the mth agent at time step t is defined as follows:

$$a_t^m = [c_t^{m,i,j}]_{0 \le i < K, 0 \le j < U}, \tag{10}$$

where $c_t^{m,i,j} \in \{0, 1\}$, and i, j represent the channel and PU indices, respectively. Note that, for convenience, we now use a_t^m to denote the action taken by SU_m at time step tth, instead of $\mathbf{c}_t^m$ as defined in Section IV.

4) Reward Function: At any time step, an SU may encounter one of the following four scenarios based on its sensing decision:

#1. The SU is in event O and chooses not to sense.
#2. The SU is in event O and chooses to sense.
#3. The SU is in event I and chooses to sense PUs that are not in its sensing range.
#4. The SU is in event I and either (i) chooses not to sense, or (ii) senses a PU within its sensing range.

In our work, the original objective function (i.e., Eq. (5)) aims to maximize the comprehensive awareness rate of the SUs while minimizing unnecessary sensing, especially under event O (where no PU is within sensing range). The first objective can be achieved by reducing sensing collisions among SUs within the same group. Minimizing sensing collisions among SUs increases the number of unique channel-PU states each SU can access after sensing and sharing results within its group. As defined in Eq. (4), increasing the number of unique channel-PU states available to each SU leads to a higher comprehensive awareness rate. Therefore, the Dec-POMDP formulation, with its objective of maximizing the expected cumulative reward is valid to capture the original goal of maximizing the comprehensive awareness rate. Meanwhile the second objective can be supported using the estimated locations of PUs. Motivated by these goals and the four possible SU scenarios above, we define the reward function with the following key components:

- A positive reward is assigned when an SU correctly chooses not to sense under event O (Scenario #1) – which aligns with reducing unnecessary sensing.
- A penalty is imposed when an SU senses in event O (Scenario #2) – discouraging redundant sensing.
- A penalty is imposed when an SU senses a PU outside its sensing range in event I (Scenario #3) – promoting spatially accurate sensing.
- A dynamic reward is used in event I for Scenario #4 – incentivizing collaboration among group members by minimizing sensing collision. This, in turn improves the collective channel-PU state awareness.

In this way, each agent receives a reward for selecting an appropriate action and a penalty for making an incorrect choice. Eventually, by optimizing this reward function through multi-agent reinforcement learning, each SU learns to select actions that: (i) increase the probability of covering all relevant PU channels within the group by reducing sensing collisions, and (ii) reduce sensing redundancy. It is important to note that the reward is assigned based on the actual state of the

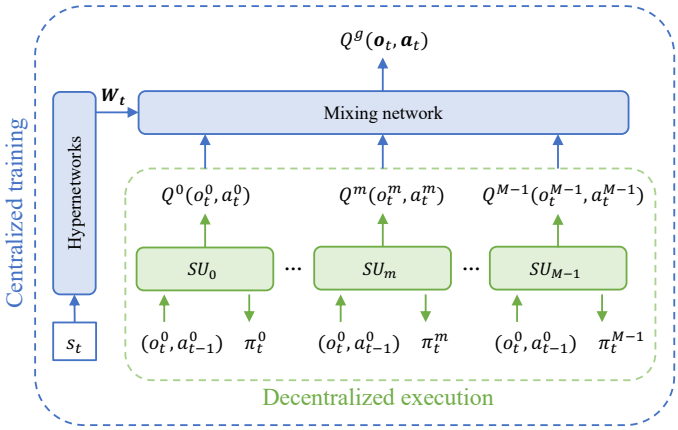

Fig. 3: QMIX architechture.

environment. Over time, through the learning process, the agent is expected to converge towards an effective policy, which can then be deployed during the execution phase. Hence, the reward function is given as follows:

$$r_t^m = \begin{cases} 0, & \text{if \#1 happens,} \\ -b & \text{if \#2 or \#3 happens,} \\ -b\chi_t^m, & \text{if \#4 happens,} \end{cases} \tag{11}$$

where $b > 0$ denotes the penalty score, and χ_t^m is the true sensing collision state of SU_m at the previous time step which is 1 if SU_m experienced a sensing collision, and 0 otherwise. Note that the reward value we choose here is a neutral value (i.e., 0), this is soly for the purpose of easy mornitoring when developing the aglorithm, it has no affect to the performance of the solution.

C. Multiagent Reinforcement Learning-based Sensing Collision Mitigation

To solve the Dec-POMDP problem, we first adopt the CTDE architecture, where the system Q-value is learned in a centralized manner using action observations acknowledged by all agents. During the execution phase, once agents have gained sufficient knowledge of the environment, they utilize their policy network to determine the optimal action. In CTDE, centralized training is applied only during the training phase to enhance cooperative learning and learning efficiency, while execution remains fully decentralized, allowing each agent to make independent decisions. Since finding an optimal solution for Dec-POMDPs is often intractable, we use QMIX [29], a cooperative MARL algorithm, to approximate a near-optimal solution that remains consistent with the objective of the original problem formulation. We employ the QMIX algorithm in this study , which establishes a monotonic relationship between the total Q-value of the system and the individual Q-value of each agent in cooperative environments. Thus, a mixing network is trained centrally to learn the global action-value function. Additionally, each agent is equipped with its Q-network, which takes the local observations of the agent as input and produces a corresponding optimal decision. We define Q^g as the global action-value function of the entire system and Q^m as the individual agent action-value function. To ensure the monotonic relationship between these values, the following constraint must be satisfied:

$$\frac{\partial Q^g}{\partial Q^m} \geq 0 \quad \forall m \in \mathcal{M}, \tag{12}$$

This constraint is enforced by how the algorithm controls the weight of the mixing network. In the QMIX architecture, in addition to the mixing network (Q^g) and the agent network (Q^m), a set of hypernetworks exist, responsible for learning nonnegative weights $\mathbf{W}$ for the mixing network. The number of hypernetworks equals the number of layers in the mixing network. The input for these hypernetworks is the global state $\mathbf{s}_t$ of the system. This global state serves as extra information indirectly fed into the mixing network. Fig. 3 illustrates the overall QMIX architecture.

Similar to the conventional deep Q-learning algorithm, the loss function for updating these networks is based on the temporal difference error, expressed as follows:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_B (y_t^g - Q^g(\mathbf{o}_t, \mathbf{a}_t|\boldsymbol{\theta}))^2, \tag{13}$$

where $\boldsymbol{\theta}$ represents the parameters for the evaluation agent network and hypernetworks, B denotes the batch data sampled from the replay buffer, and y_t^g is defined as follows:

$$y_t^g = \sum_{m \in \mathcal{M}} r_t^m + \gamma \max_{\mathbf{a}_{t+1}} Q^g(\mathbf{o}_{t+1}, \mathbf{a}_{t+1}|\boldsymbol{\theta}'), \tag{14}$$

where $\boldsymbol{\theta}'$ denotes the parameter for the target agent network and hypernetworks. The evaluation network and hypernetwork parameters are updated as follows:

$$\boldsymbol{\theta}_t := \boldsymbol{\theta}_{t-1} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}), \tag{15}$$

where α represents the learning rate. The step-by-step SCM procedure is detailed in Algorithm 2.

D. Computational Complexity Analysis

In the training phase, each agent is equipped with a Q-network, while a centralized Q-network is accompanied by a set of hypernetworks. Let V denote the number of layers for in all Q-networks. Consequently, there are also V hypernetworks, each consisting of Z layers.

The computational complexity for a single episode of a single agent's Q-mix network is given by: $O\Big(T_s\big(X_1H_1 + (V-2)H_1^2 + Y_1H_1\big)\Big)$ where H_1 represents the hidden layer size, and X_1 and Y_1 denote the dimensions of the input and output layers, respectively. As illustrated in Fig. 3, the agent network takes as input the current partial observation and the last action, producing the Q-value as output. Thus, we have: $X_1 = 2M + U + KU, \quad Y_1 = 1.$

Similarly, the computational complexity of the centralized Q-network is given by: $O\Big(T_s\big(X_2H_2 + (V-2)H_2^2 + Y_2H_2\big)\Big)$ where $X_2 = M$ represents the joint state dimension (i.e., the number of agents in the system), $Y_2 = 1$ is the output

Algorithm 2 Sensing collision mitigation (SCM) algorithm

Initialize the cognitive environment
Initialize the replay buffer and all hyperparameters
Initialize weight parameters for the hypernetworks, evaluation agent networks, and target agent networks
repeat
  Obtain the initial state $\mathbf{s}_0$ and initial observation $\mathbf{o}_0$
  for $t := 1...T_s$ **do**
    for $SU_m \in \mathcal{M}$ **do**
      Select action a_t^m from π^m according to ϵ-greedy
    end for
    All SUs perform sensing based on the chosen actions
    SUs obtain reward $\mathbf{r}_t$ based on (11)
    The environment transitions to the next state $\mathbf{s}_{t+1}$
    SUs observe the next observation $\mathbf{o}_{t+1}$
    Store $(\mathbf{s}_t, \mathbf{o}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{o}_{t+1}, \mathbf{r}_t)$ to the replay buffer
    $\mathbf{s}_t \leftarrow \mathbf{s}_{t+1}, \mathbf{o}_t \leftarrow \mathbf{o}_{t+1}$
    if enough experiences **then**
      Sample batch B from the replay buffer
      Obtain all Q_t^m and Q_{t+1}^m from the evaluation and target agent networks
      Obtain $\mathbf{W}$ from the hypernetworks
      Obtain Q_t^g and Q_{t+1}^g from the mixing network with inputs of Q_t^m and Q_{t+1}^m
      Calculate the loss according to (14)
      Update the evaluation agent and hypernetwork weights according to (15)
      if update == true **then**
        Update the target agent networks from the evaluation networks
      end if
    end if
    if $\epsilon > \epsilon_{min}$ **then**
      $\epsilon \leftarrow \epsilon - \epsilon_d$
    end if
  end for
until convergence or aborted

dimension of the centralized Q-network, and H_2 denotes the hidden layer size used in the centralized Q-network.

Finally, the computational complexity of the set of hypernetworks is given by: $O\Big(T_s V\big(X_3 H_3 + (Z-2)H_3^2 + Y_3 H_3\big)\Big)$ where $X_3 = M + M^2 + U$ represents the combined input size, $Y_3 = H_2$ is the output size of the hypernetwork, which matches the hidden layer size of the centralized Q-network, and H_3 denotes the hidden layer size used in the hypernetwork.

VI. Numerical Results

This section presents simulations to validate the performance of the SCM algorithm. Table II details all essential hyperparameters for the wireless environment and the proposed algorithm. The first part of this section evaluates the performance of the PLE algorithm, measuring its effectiveness in terms of the PU identification rate (PIR) and Hausdorff distance. Following that, we illustrate the overall effectiveness of the proposed solution, highlighting its superior convergence performance, reduced sensing collision rate, enhanced channel state awareness rates, and lower sensing overhead compared with baseline schemes. We compare the proposed algorithm with five baseline schemes, described as follows.

1) Central-V [49]: A multiagent extension of the temporal difference actor-critic algorithm, a policy gradient-based method.
2) Value-decomposition networks (VDN) [50]: A MARL algorithm that follows the CTDE approach. Unlike QMIX, VDN employs a simpler decomposition, representing the system Q-value as the sum of all individual agents' Q-values.
3) Centralized CSS (C-CSS): A centralized node that collects all information regarding all SU locations within the network at each time step and determines the optimal sensing channel-PU indices for all SUs based on the estimated PU locations, ensuring no sensing collisions. This represents a near-ideal scenario, demonstrating close-to-optimal coordination.
4) Random sensing with PLE (R-PLE): SUs decide whether to sense at each time instant based on PU location estimation output from PLE. If sensing is selected, the k sensing channel-PU pairs are chosen randomly.
5) Random sensing without PLE (R-nPLE): Agents lack information since PLE is not performed. At each time step, agents choose k channel-PU pairs to sense, making their selections randomly.

TABLE II: Simulation parameters.

Parameter	Value
Number of PUs U	3, 7, 11
Number of channel bands K	4
Number of SUs M	7, 10
Number of sectors L	4, 6, 8
Zone width z	1 km
SU sensing range s	100, 200, 300 m
Number of sensing channel-PU pairs k	1, 2
Time steps per episode (PLE) T_f	100, 200
No. of episodes (PLE) E_f	30
Area threshold (PLE) μ	1e-4
Time steps per episode (QMIX) T_s	100
Optimizer	Adam
Learning rate	1e-3
Discount factor γ	0.99
Agent network layers	3
Mixing network layers	3
Initial ϵ	1
Epsilon decay value ϵ_d	19e-6
Minimum ϵ_{min}	0.05
Time steps to update target networks	200
Batch size	32
Buffer size	5e3
Reward penalty score b	20

A. PU Location Estimation (PLE)

1) Effect of Different Numbers of PUs: This simulation evaluates the PLE performance under varying numbers of PUs, ranging from three to 11. The algorithm is executed 10 times for each PU quantity, and the results are averaged. The applied performance metric is the PIR, defined as the frequency with which SUs correctly select the sector covering any PU over the total event occurrences I. PU locations are randomly positioned within the considered zone width in each runtime. Fig. 4 illustrates the results for each episode. After several episodes, the algorithm converges to a PIR of 99% when the number of PUs is three and 95% when the number of PUs is 11. Additionally, as the number of PUs increases, the PIR decreases. This occurs because higher density makes it more challenging to distinguish nearby PUs, especially when they frequently fall within the same sector area.

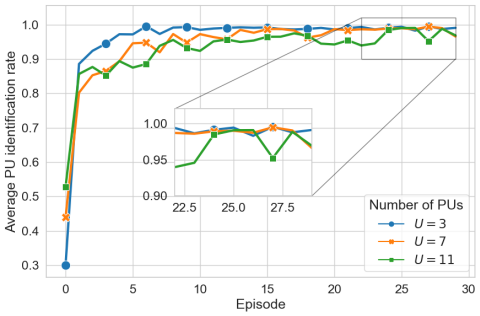

Fig. 4: Average PU identification rate (PIR) with different numbers of PUs, $L = 4, s = 300\ m, T_f = 100$.

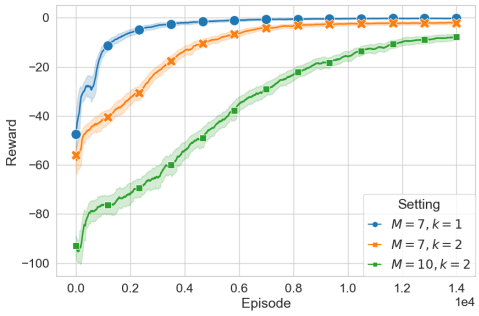

Fig. 6: Convergence with different settings, $U = 3$.

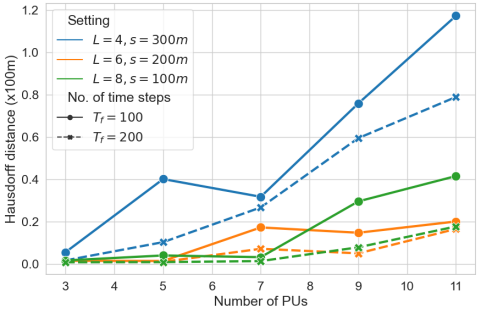

Fig. 5: Average Hausdorff distance with different settings.

2) Effect of Different Sector Settings and Number of Time Steps for Each Episode: This experiment employs the Hausdorff distance [51] as the performance metric, measuring the discrepancy between the actual and estimated PU positions. The Hausdorff distance between two sets of points, $\mathcal{X}$ and $\mathcal{Y}$, in a two-dimensional space, quantifies their similarity and is computed as follows:

$$d_H(\mathcal{X}, \mathcal{Y}) = \max(\sup_{p_x \in \mathcal{X}} \inf_{p_y \in \mathcal{Y}} d(p_x, p_y), \sup_{p_y \in \mathcal{Y}} \inf_{p_x \in \mathcal{X}} d(p_y, p_x)), \tag{16}$$

where sup and inf are the supremum (maximum) and infimum (minimum) operators, and $d(p_x, p_y)$ denotes the Euclidean distance from point p_x to p_y.

Fig. 5 illustrates the algorithm's performance under different sector settings, including the number of sectors L and the sensing range s. A larger Hausdorff distance indicates greater dissimilarity between the actual and estimated PU locations. As shown in the figure, all settings exhibit an upward trend, reflecting that as the number of PUs increases, the algorithm encounters greater challenges in accurately estimating their locations. Conversely, a higher number of sectors and a shorter sensing range result in a smaller sensing area per sector, leading to better PLE performance with a smaller Hausdorff distance. Additionally, increased time steps per episode allow the SU to accumulate more historical data, further reducing the Hausdorff distance. Generally, a configuration of six sectors, a sensing range of 200 m, and 100 time steps per episode demonstrates commendable performance, with a maximum Hausdorff distance of approximately 20 m when the number of PUs is 11.

B. Multiagent Reinforcement Learning-based

This section presents comprehensive simulations to evaluate the performance of the proposed SCM algorithm, considering the convergence behavior, sensing collision rate, channel state awareness rate, and sensing overhead.

1) Convergence Analysis: This simulation initially executes the algorithm under various settings concerning the number of SUs M and the sensing channel-PU pairs k. Fig. 6 illustrates the sum system reward convergence curves over episodes for different values of M and k with a fixed number of PUs ($U = 3$). Notably, the algorithm demonstrates stable convergence after sufficient episodes, regardless of these parameters. Specifically, a faster convergence speed is observed with fewer agents (i.e., $M = 7$), and a similar trend occurs with a smaller k (i.e., $k = 1$). This result is expected as agents learn to avoid sensing collisions and having fewer M and k simplifies this goal.

Fig. 7 presents a convergence comparison between the proposed SCM algorithm and two baseline approaches, VDN and Central-V. The simulation environment is identical for all three approaches: The number of SUs M is set to $M = 7$, the number of sensing channel-PU pairs is $k = 2$, the number of PUs is $U = 3$. All three approaches exhibit convergence after several episodes. The proposed SCM algorithm demonstrates superior convergence performance, achieving the highest reward (close to zero) with a more stable convergence process. VDN and Central-V converged earlier (around episode 1000) but only achieved approximately -30 and -42 of reward values, respectively. This is because our objective is to achieve fine-grained interactions between agents' actions, such as avoiding conflicts. QMIX, with its nonlinear and state-dependent mixing network, enables complex inter-agent coordination. This allows proposed solution to effectively capture the intricate relationships between agents' actions and their collective impact on the team reward. In contrast, Central-V learns a joint value function for the entire system but does not explicitly model individual agents' contributions or the joint action space. As a result, it is less effective at capturing the detailed interactions between agents' actions. Similarly, VDN simplifies the problem by decomposing the joint Q-value as a linear sum of individual agents' Q-values. While easier to implement, this linear approach struggles to handle the complexities of multi-agent coordination.

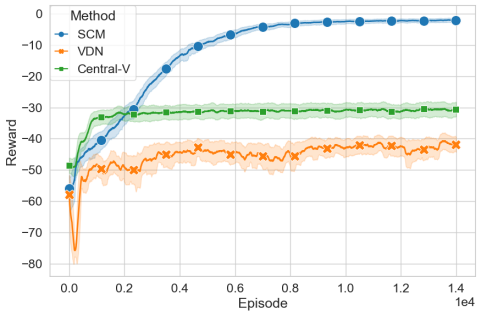

Fig. 7: Convergence comparison, $M = 7$, $k = 2$, and $U = 3$.

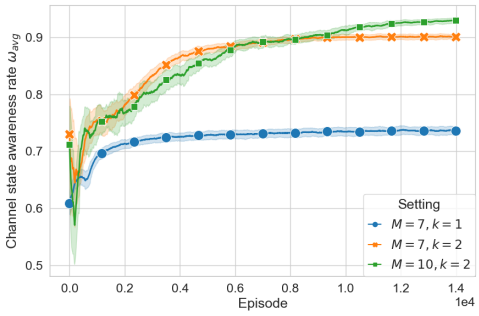

Fig. 9: Awareness rate with different settings, $U = 3$.

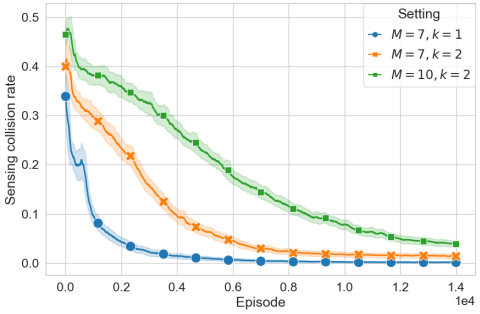

Fig. 8: Sensing collision rate with different settings, $U = 3$.

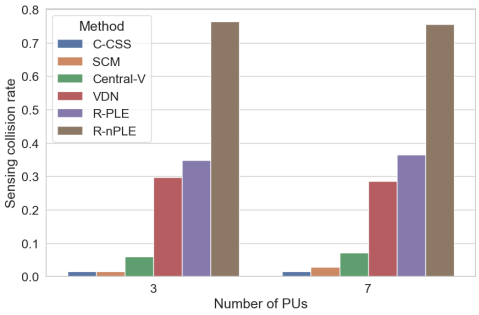

Fig. 10: Sensing collision rate comparison, $M = 7$ and $k = 2$.

2) Sensing Collision Rate with Different Settings: Fig. 8 illustrates the impact of M (number of SUs) and k (sensing channel-PU pairs per SU) on the average sensing collision rate of all SUs. A lower collision rate (close to zero) is observed when fewer SUs and sensing channel-PU pairs are present (i.e., $M = 7$, $k = 1$). This indicates that, at each time slot, individual SUs can effectively select sensing channel-PU pairs that do not overlap with those chosen by nearby SUs. In scenarios such as $M = 7$, $k = 2$ or $M = 10$, $k = 2$, the observed sensing collision rates are approximately 0.01 and 0.04, respectively. These results demonstrate the algorithm's capability to minimize collisions, even as the number of SUs increases.

3) Channel State Awareness Rate with Different Settings: Fig. 9 compares different M (number of SUs) and k (sensing channel-PU pairs per SU) settings w.r.t. the average channel state awareness rate per episode, defined as follows:

$$\omega_{avg} = \frac{1}{T_s}\frac{1}{M}\sum_{t=0}^{T_s-1}\sum_{m=0}^{M-1}\omega_t^m. \tag{17}$$

This experiment revealed a reverse trend compared to the sensing collision rate, where a higher number of SUs M and sensing channel-PU pairs k resulted in a higher channel state awareness rate. With $M = 7$ and $k = 1$, the algorithm performed least effectively, achieving approximately 74% channel state awareness rate. Meanwhile, a configuration of $M = 7$ and $k = 2$ exhibits a faster increase in ω_{avg} than $M = 10$ and $k = 2$ but ultimately reaches a lower channel state awareness rate (approximately 90%). This outcome indicates that, although higher M and k values make it more challenging for agents to learn the optimal policy, the final performance can still exceed configurations with smaller M and k. This is because a greater number of agents and more sensing channel-PU pairs promote higher cooperation, ultimately improving overall system efficiency.

4) Sensing Collision Rate Comparison with Baseline Schemes: Fig. 10 compares sensing collision rates between the proposed scheme and various baseline methods under different PU scenarios. For this comparison, we set the number of SUs at $M = 7$ and the number of sensing channel-PU pairs at $k = 2$. R-nPLE, which lacks environmental learning, exhibited the highest collision rate (approximately 75.5%). R-PLE, relying solely on PLE, reducing the collision rate to approximately 36%. VDN and Central-V further lowered the collision rates of 29% and 6.5%, respectively. The proposed SCM scheme outperformed all decentralized counterparts, achieving 1% sensing collision rate $U = 3$ and 1% sensing collision rate with $U = 7$. Notably, the SCM scheme's performance is comparable to the nearly optimal collision rate achieved by the C-CSS scheme.

5) Channel state awareness rate comparison with baseline schemes: Fig. 11 compares the SCM algorithm with baseline schemes in terms of the average channel state awareness rate with $U = 3$ and $U = 7$. Across scenarios with varying numbers of PUs, the R-nPLE scheme consistently exhibited the lowest performance. VDN and R-PLE performed better, achieving ω_{avg} approximately 73% and 84% for $U = 3$, and 64% and 67% for $U = 7$. Central-V demonstrated commendable performance, with only marginal degradation compared to the proposed SCM scheme. The proposed SCM algorithm outperformed all decentralized methods, achieving ω_{avg} approximately 91% for $U = 3$ and 72% for $U = 7$. This performance is nearly comparable to the C-CSS scheme. Moreover, an increase in the number of PUs does not signifi-

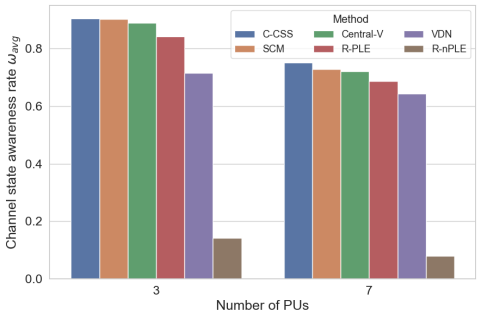

Fig. 11: Awareness rate comparison, $M = 7$ and $k = 2$.

cantly affect the sensing collision rate. However, the channel state awareness rate decreased because each SU covers a higher number of channel-PU pairs per time step on average with a higher number of PUs. Consequently, maintaining a fixed number of sensing channel-PU pairs at $k = 2$ decreases the channel state awareness rate.

6) Sensing Overhead Comparison with Baseline Schemes: Fig. 12 presents a performance comparison of sensing overhead versus average channel state awareness between the proposed solution and baseline methods. For this experiment, the number of sensing channel-PU pairs is set at ($k = 2$), and the number of PUs is set at ($U = 3$). The bar chart represents the sensing overhead, measured as the average k that one SU chooses to sense in a time slot, expressed as follows:

$$k_{avg} = \frac{1}{T_s}\frac{1}{M}\sum_{t=0}^{T_s-1}\sum_{m=0}^{M-1} k_t^m, \tag{18}$$

where $k_t^m \in \{0, k\}$.

The line chart illustrates the average channel state awareness rate corresponding to each of the comparison schemes. First, we analyzed the results of all schemes with seven SUs ($M = 7$). Both the proposed scheme and the optimal case of the C-CSS scheme exhibited similar average channel state awareness rates and required sensing overhead. Moreover, compared to the highest overhead scenario of $k = 2$ (observed in the R-nPLE scheme), the proposed solution reduces sensing overhead by approximately 53.2%. When comparing with the R-PLE scheme, where SUs greedily sense in every time slot when they move near any PU, the proposed scheme can save approximately 13.4% of the sensing overhead while maintaining commendable performance. Although the VDN-based scheme achieved the lowest average sensing overhead, it resulted in the second-worst ω_{avg} (outperforming only the R-nPLE scheme). Furthermore, when comparing the proposed scheme with different numbers of SUs ($M = 7$ and $M = 10$), the algorithm demonstrates improved performance with reduced sensing overhead (approximately 0.83 channel-PU pairs per time step) as more agents are incorporated. This outcome underscores the cooperative nature of the scheme and indicates that overall system performance can be enhanced by introducing additional SUs into the network.

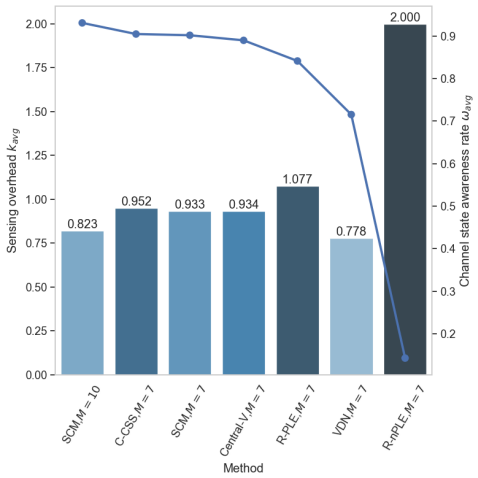

Fig. 12: Sensing overhead comparison, $k = 2, U = 3$.

VII. Conclusion and Future Works

This study investigates a decentralized CSS scheme for mobile directional CRNs. We formulated a stochastic sequential optimization problem to enhance channel state awareness while reducing unnecessary overhead through dynamic sensing decisions for channel-PU pairs. To address this problem, we proposed a two-stage learning approach that comprises a PLE algorithm and an SCM method. In the second stage, the problem was transformed into a Dec-POMDP, for which we applied QMIX, a CTDE-based MARL algorithm. This CTDE architecture allows SUs to learn centrally and operate decentralized with limited communication after sufficient training. Numerical results demonstrate that our solution outperforms alternative schemes regarding convergence, sensing collision rate, channel state awareness, and sensing overhead. Moreover, the findings highlight the cooperative nature of the problem, indicating that increased SU participation can further enhance overall performance. While this study offers valuable insights into enhancing channel state awareness through SU cooperation, some limitations present opportunities for further research. Our model assumes ideal conditions for applying directional antennas to mobile SUs; however, real-world challenges—such as multipath effects, non-line-of-sight conditions, and handover issues—must be addressed. Future work could integrate advanced beamforming techniques, such as massive MIMO, or hybrid beamforming to mitigate these challenges while preserving the benefits of directional antennas. Although our findings provide a strong foundation, further research is needed to generalize the approach across diverse environments.

References

[1] Lionel Sujay Vailshery. Number of internet of things (iot) connected devices worldwide from 2019 to 2023, with forecasts from 2022 to 2030 (in billions) [graph], July 1 2023.

[2] Ian F Akyildiz, Won-Yeol Lee, Mehmet C Vuran, and Shantidev Mohanty. Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey. *Computer networks*, 50(13):2127–2159, 2006.

[3] Ian F. Akyildiz, Brandon F. Lo, and Ravikumar Balakrishnan. Cooperative spectrum sensing in cognitive radio networks: A survey. *Physical Communication*, 4(1):40–62, 2011.

[4] Woongsoo Na, Jongha Yoon, Sungrae Cho, David Griffith, and Nada Golmie. Centralized cooperative directional spectrum sensing for cognitive radio networks. *IEEE Transactions on Mobile Computing*, 17(6):1260–1274, 2018.

[5] Woongsup Lee, Minhoe Kim, and Dong-Ho Cho. Deep cooperative sensing: Cooperative spectrum sensing based on convolutional neural networks. *IEEE Transactions on Vehicular Technology*, 68(3):3005–3009, 2019.

[6] Rahil Sarikhani and Farshid Keynia. Cooperative spectrum sensing meets machine learning: Deep reinforcement learning approach. *IEEE Communications Letters*, 24(7):1459–1462, 2020.

[7] Jun Wu, Cong Wang, Yue Yu, Tiecheng Song, and Jing Hu. Performance optimisation of cooperative spectrum sensing in mobile cognitive radio networks. *IET Communications*, 14(6):1028–1036, 2020.

[8] Xiaoying Liu, Kechen Zheng, Kaikai Chi, and Yi-Hua Zhu. Cooperative spectrum sensing optimization in energy-harvesting cognitive radio networks. *IEEE Transactions on Wireless Communications*, 19(11):7663–7676, 2020.

[9] Sadia Khaf, Georges Kaddoum, and Joao Victor de Carvalho Evangelista. Partially cooperative rl for hybrid action crns with imperfect csi. *IEEE Open Journal of the Communications Society*, 5:3762–3774, 2024.

[10] Jun Wu, Mingkun Su, Lei Qiao, Xiaorong Xu, Xuesong Liang, Hao Wang, Jianrong Bao, and Weiwei Cao. Quick parallel cooperative spectrum sensing in cognitive wireless sensor networks. *IEEE Sensors Letters*, 8(8):1–4, 2024.

[11] Lingyun Zhou, Wenqiang Pu, Yixin Jiang, Ming-Yi You, Rongqing Zhang, and Qingjiang Shi. Joint optimization of uav deployment and directional antenna orientation for multi-uav cooperative sensing system. *IEEE Transactions on Wireless Communications*, 23(10):14052–14065, 2024.

[12] Qing Hu, Chensong Zhao, Shahzad Bashir, Shuaiheng Huai, Yanpeng Dai, Qing Zhang, Yuchen Wang, and Mingming Li. Cooperative spectrum sensing for the offshore cognitive ship ad hoc network. *IEEE Sensors Journal*, 25(2):3199–3211, 2025.

[13] Jie Shi, Jiaxin Wu, Jingzheng Chong, and Zhihua Yang. Collaborative spectrum sensing for multi-uav system: a u-net approach with uncertainty awareness. *IEEE Transactions on Vehicular Technology*, pages 1–16, 2025.

[14] Ang Gao, Chengyuan Du, Soon Ng, and Wei Liang. A cooperative spectrum sensing with multi-agent reinforcement learning approach in cognitive radio networks. *IEEE Communications Letters*, PP:1–1, 05 2021.

[15] Jarmo Lunden, Sanjeev R Kulkarni, Visa Koivunen, and H Vincent Poor. Multiagent reinforcement learning based spectrum sensing policies for cognitive radio networks. *IEEE journal of selected topics in signal processing*, 7(5):858–868, 2013.

[16] Wenli Ning, Xiaoyan Huang, Kun Yang, Fan Wu, and Supeng Leng. Reinforcement learning enabled cooperative spectrum sensing in cognitive radio networks. *Journal of Communications and Networks*, 22(1):12–22, 2020.

[17] Chunghyun Lee, Junsuk Oh, Woongsoo Na, Jongha Yoon, Wonjong Noh, and Sungrae Cho. Directional-antenna-based spatial and energy-efficient semi-distributed spectrum sensing in cognitive internet-of-things networks. *Journal of Network and Computer Applications*, page 103687, 2023.

[18] Xuanheng Li, Yulong Zhang, Haichuan Ding, and Yuguang Fang. Intelligent spectrum sensing and access with partial observation based on hierarchical multi-agent deep reinforcement learning. *IEEE Transactions on Wireless Communications*, 23(4):3131–3145, 2024.

[19] Ang Gao, Qinyu Wang, Yongze Wang, Chengyuan Du, Yansu Hu, Wei Liang, and Soon Xin Ng. Attention enhanced multi-agent reinforcement learning for cooperative spectrum sensing in cognitive radio networks. *IEEE Transactions on Vehicular Technology*, 73(7):10464–10477, 2024.

[20] Sungwook Kim. Multi-agent learning and bargaining scheme for cooperative spectrum sharing process. *IEEE Access*, 11:47863–47872, 2023.

[21] Jiawen Li, Yonghua Wang, Yanqing Chen, Guanhai Xu, and Bingfeng Zheng. Local cooperative sensing in 3d spectrum availability-heterogeneous uav networks. *IEEE Transactions on Vehicular Technology*, pages 1–14, 2024.

[22] Xiang Tan, Li Zhou, Haijun Wang, Yuli Sun, Haitao Zhao, Boon-Chong Seet, Jibo Wei, and Victor C. M. Leung. Cooperative multi-agent reinforcement-learning-based distributed dynamic spectrum access in cognitive radio networks. *IEEE Internet of Things Journal*, 9(19):19477–19488, 2022.

[23] Weishan Zhang, Yue Wang, Xiang Chen, Lingjia Liu, and Zhi Tian. Collaborative learning-based spectrum sensing under partial observations. *IEEE Transactions on Cognitive Communications and Networking*, 10(5):1843–1855, 2024.

[24] Qiu Wang, Hong-Ning Dai, Orestis Georgiou, Zhiguo Shi, and Wei Zhang. Connectivity of underlay cognitive radio networks with directional antennas. *IEEE Transactions on Vehicular Technology*, 67(8):7003–7017, 2018.

[25] Shuchi Tripathi, Abhishek K Gupta, and SaiDhiraj Amuru. Coverage analysis of cognitive mmwave networks with directional sensing. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 125–129. IEEE, 2021.

[26] Yanxiao Zhao, Prosanta Paul, ChunSheng Xin, and Min Song. Performance analysis of spectrum sensing with mobile sus in cognitive radio networks. In *2014 IEEE International Conference on Communications (ICC)*, pages 2761–2766, 2014.

[27] Xinyu Wang, Min Jia, Qing Guo, and Xuemai Gu. Performance analysis of spectrum sensing for mobile cognitive radio networks. In *2015 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2015.

[28] Alexander W Min and Kang G Shin. Impact of mobility on spectrum sensing in cognitive radio networks. In *Proceedings of the 2009 ACM workshop on Cognitive radio networks*, pages 13–18, 2009.

[29] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1):7234–7284, 2020.

[30] Christian Bettstetter, Hannes Hartenstein, and Xavier Pérez-Costa. Stochastic properties of the random waypoint mobility model. *Wireless networks*, 10(5):555–567, 2004.

[31] Palanisamy Vigneshwaran and Satkunarajah Suthaharan. Dynamic sectorized routing in mobile ad hoc networks using reconfigurable directional antenna. In *Proceedings of International Conference on Advanced Computing Applications: ICACA 2021*, pages 165–175. Springer, 2022.

[32] Zhe Chu, Fei Hu, Elizabeth Bentley, and Sunil Kumar. Intelligent routing in directional ad hoc networks through predictive directional heat map from spatio-temporal deep learning. *IEEE Transactions on Mobile Computing*, 23(4):2639–2656, 2024.

[33] Tianyi Zhang, Hongwei Zhang, and Zhibo Meng. Interference and coverage analysis of mmwave inter-vehicle broadcast with directional antennas. In *ICC 2022 - IEEE International Conference on Communications*, pages 273–278, 2022.

[34] Yiming Huo, Franklin Lu, Felix Wu, and Xiaodai Dong. Multi-beam multi-stream communications for 5g and beyond mobile user equipment and uav proof of concept designs. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pages 1–5, 2019.

[35] Jing Zhang, Huan Xu, Lin Xiang, and Jun Yang. On the application of directional antennas in multi-tier unmanned aerial vehicle networks. *IEEE Access*, 7:132095–132110, 2019.

[36] Yue Xu, Jianyuan Yu, and R Michael Buehrer. The application of deep reinforcement learning to distributed spectrum access in dynamic heterogeneous environments with partial observations. *IEEE Transactions on Wireless Communications*, 19(7):4494–4506, 2020.

[37] Yoel Bokobza, Ron Dabora, and Kobi Cohen. Deep reinforcement learning for simultaneous sensing and channel access in cognitive networks. *IEEE Transactions on Wireless Communications*, 2023.

[38] Yi Song and Jiang Xie. Performance analysis of spectrum handoff for cognitive radio ad hoc networks without common control channel under homogeneous primary traffic. In *2011 Proceedings IEEE INFOCOM*, pages 3011–3019. IEEE, 2011.

[39] Qihui Wu, Guoru Ding, Jinlong Wang, and Yu-Dong Yao. Spatial-temporal opportunity detection for spectrum-heterogeneous cognitive radio networks: Two-dimensional sensing. *IEEE Transactions on Wireless Communications*, 12(2):516–526, 2013.

[40] Feng Shen, Guoru Ding, Zheng Wang, and Qihui Wu. Uav-based 3d spectrum sensing in spectrum-heterogeneous networks. *IEEE Transactions on Vehicular Technology*, 68(6):5711–5722, 2019.

[41] Hrishikesh Gossain, Tarun Joshi, C De Morais Cordeiro, and Dharma P Agrawal. Drp: An efficient directional routing protocol for mobile ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 17(12):1438–1541, 2006.

[42] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

[43] Ronald L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Info. Proc. Lett.*, 1:132–133, 1972.

[44] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[45] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[46] Won Joon Yun, Soohyun Park, Joongheon Kim, MyungJae Shin, Soyi Jung, David A Mohaisen, and Jae-Hyun Kim. Cooperative multiagent deep reinforcement learning for reliable surveillance via autonomous multi-uav control. *IEEE Transactions on Industrial Informatics*, 18(10):7086–7096, 2022.

[47] Dohyun Kwon, Joohyung Jeon, Soohyun Park, Joongheon Kim, and Sungrae Cho. Multiagent ddpg-based deep learning for smart ocean federated learning iot networks. *IEEE Internet of Things Journal*, 7(10):9895–9903, 2020.

[48] Soohyun Park, Chanyoung Park, and Joongheon Kim. Learning-based cooperative mobility control for autonomous drone-delivery. *IEEE Transactions on Vehicular Technology*, 73(4):4870–4885, 2024.

[49] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[50] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

[51] Abdel Aziz Taha and Allan Hanbury. An efficient algorithm for calculating the exact hausdorff distance. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2153–2163, 2015.

Thi Thu Hien Pham rreceived the B.S. degree in Data Communications and Computer Networks from the Hanoi University of Science and Technology, Vietnam, in 2018, and the M.S. degree in Computer Science and engineering from Chung-Ang University, South Korea, in 2024.

Arooj Masood received the B.S. and M.S. degrees in computer science from the Lahore College for Women University, Lahore, Pakistan, in 2011 and 2013, respectively. She is currently pursuing the Ph.D. degree with Chung-Ang University, Seoul, South Korea. She was a Lecturer with the Department of Computer Science, Government College for Women Gulberg, Lahore. Her current research interests include vehicular cloud computing, ubiquitous computing, wireless networks, deep space networks, and deep reinforcement learning.

Thi My Tuyen Nguyen received the B.S. degree in Mathematics from University of Science, Ho Chi Minh City, Viet Nam in 2016, and M.S. degree in Computer Science and Engineering from Soongsil University, South Korea in 2021. She is currently pursuing Ph.D. in Big Data at Chung-Ang University, South Korea. Her research interests include machine learning, optimization, and their applications in wireless communications.

Chunghyun Lee received B.S. and M.S degree in the Department of Computer Science and Engineering from Chung-Ang University, Seoul, South Korea, in 2018. He is currently pursuing his Ph.D. degree in the Department of Computer Science and Engineering at Chung-Ang University, Seoul, Korea. His research interests include 5G and 6G wireless communications and networks, spectrum sensing, demand response, game theory, internet of things, microgrids, and smart grids.

Sungrae Cho is a professor with the School of Computer Sciences and Engineering, Chung-Ang University (CAU), Seoul. Before joining CAU, he served as an Assistant Professor at the Department of Computer Sciences, Georgia Southern University, Statesboro, GA, USA, from 2003 to 2006, and a senior member of the technical staff at the Samsung Advanced Institute of Technology (SAIT), Kiheung, South Korea, in 2003. From 1994 to 1996, he was a research staff member at the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea. From 2012 to 2013, he served as a visiting professor at the National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA. He received B.S. and M.S. degrees in Electronics Engineering from Korea University, Seoul, South Korea, in 1992 and 1994, respectively, and a Ph.D. degree in Electrical and Computer Engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2002. His research interests include wireless networking, ubiquitous computing, and ICT convergence. He has been a subject editor of IET Electronics Letters since 2018 and was an area editor of Ad Hoc Networks (Elsevier) from 2012 to 2017. He has served as an organizing committee chair for numerous international conferences such as IEEE SECON, ICOIN, ICTC, ICUFN, TridentCom, and IEEE MASS and as a program committee member for conferences such as IEEE ICC, GLOBECOM, VTC, MobiApps, SENSORNETS, and WINSYS.