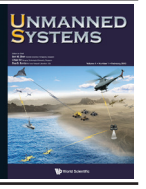
Unmanned Systems, Vol. 14, No. 2 (2026) 341–356

DOI: 10.1142/S2301385026500032

UNMANNED SYSTEMS
https://www.worldscientific.com/worldscinet/us



A Multi-Agent Visibility-Based Persistent Monitoring Method Using a KAN-Mix Network

Jun Luo, Xi Chen, Junye Wu, Wenbo Hui, Bowen Yang, Yangmin Xie*

Shanghai Key Laboratory of Intelligent Manufacturing and Robotics
Shanghai University, No. 99 Shangda Road, Shanghai 200444, P. R. China

To address the Multi-agent Visibility-based Persistent Monitoring (MVPM) problem using clustered unmanned systems, we propose an enhanced Q-value mixing network named KAN-Mix, which incorporates Kolmogorov–Arnold networks. Additionally, we design the MVPM reward based on information entropy, introducing information dynamics and probability theory into the framework. Comprehensive experiments validate the performance of KAN-Mix, demonstrating significant improvements in both coverage rate and intruder detection rate compared to the original QMIX method. Our algorithm demonstrates a performance improvement of 6–12.3% in the coverage rate of maps compared to the QMIX algorithm across three distinct maps. Additionally, it exhibits a significantly higher catch rate than both learning-based and traditional algorithms, with some maps achieving success rates of up to 100%. The average number of catch steps required to capture intruders ranks among the top two on all maps. The entropy-based reward outperforms the traditional coverage-based reward by 6.4–58.3% in coverage and by 6–61.9% in the number of catch steps. The entropy-based reward is shown to be more effective than the traditional coverage-based rewards. Combining these advantages, KAN-Mix delivers superior results compared to all standard Q-value strategies.

Keywords: Multi-agent visibility-based persistent monitoring (MVPM); multi-agent reinforcement learning (MARL); Q-value mixing network enhanced by KANs (KAN-Mix).

US

1. Introduction

Standard Persistent Monitoring (PM) problem traditionally involves patrolling agents visiting a predetermined sequence of locations. This approach is well-suited for scenarios with fixed monitoring stations. However, in cases where full coverage of an area is required, yet visibility is limited for the agents, a variant named Visibility-based Persistent Monitoring (VPM) is proposed [1]. With the increasing integration of unmanned systems into various autonomous tasks, such as coordinated regional lockdown control, the opportunity arises to leverage these systems for executing organized and strategic VPM missions. This is the Multi-agent Visibility-based Persistent Monitoring (MVPM) problem discussed in this paper. The relationship between PM, VPM, and MVPM can be illustrated in Fig. 1. VPM builds upon the concepts of PM and extends them to accommodate the challenges posed by limited visibility. MVPM allows a cooperative working manner for VPM, thus offering a more efficient solution for monitoring large areas by unmanned systems.

In the past few decades, researchers have extensively explored the solutions for MVPM [2, 3]. The primary goal of MVPM is to maximize the possibility of discovering any targets that invaded a specified area by planning the trajectories of multiple robots cooperatively. The area under monitoring is usually considered with distributed obstacles, and is modeled in two ways, the topology graph or grid map. The topology map encodes environmental locations within maps using nodes and edges, with directed connections between nodes reflecting the sequential order of access to locations by agents.

Researchers formulated the MVPM as a closed-loop Traveling Salesman Problem (TSP) problem for multi-travelers and

Received 17 July 2024; Accepted 28 December 2024; Published 18 March 2025. This paper was recommended for publication in its revised form by editorial board member, Yi Dong.
Email Address: *xieym@shu.edu.cn
*Corresponding author.

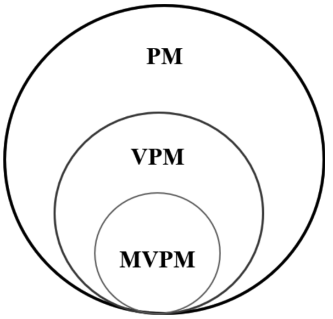


Fig. 1. MVPM, VPM, PM relationship diagram.

solved it with tools including linear optimization control [4], Kalman filtering [5], and Markov chains [6, 7]. However, they usually did not emphasize the node distribution with visibility issues, and due to the inherent NP-hard complexity, the number of nodes is limited. The second way to use a grid map addresses the visibility by relating it to the grid size. In this case, the environment data is more structured, and MVPM is turned to be a Multi-agent Coverage Path Planning (MCPP) problem. There are mainly two MCPP solutions. The first one is the Divide Areas Based on Robots Initial Positions (DARP) algorithm, which distributes areas among multiple agents iteratively and uses Spanning Tree Coverage (STC) to construct a minimum spanning tree and the corresponding path within each section[8]. The second method is the Glasius Bio-inspired Neural Network (GBNN) algorithm, which is a biologically inspired neural network that generates coverage paths by adjusting neuron activity values[9].

In recent years, machine learning techniques have gained significant traction across a wide array of practical applications (such as the neural network [10–12], Gaussian process regression [13–15] and graphical technique [16]), with reinforcement learning emerging as a key subfield. Reinforcement learning, in particular, plays a crucial role in decision-making and planning tasks, where an agent learns to optimize its behavior through interactions with an environment (such as MVPM problem). The aforementioned works employ centralized approaches, requiring decision-making and data management at a central location. However, as the number of nodes increases, the complexity of these centralized approaches escalates, making rapid solutions challenging, particularly for multi-objective optimization problems. Consequently, decentralized reinforcement learning methods have garnered research interest in recent years. The dominant solution for decentralized implementation is Multi-Agent Reinforcement Learning (MARL). Various MARL frameworks, such as MAPPO [1], IQL [17], QMIX [18], have been proposed to address the MVPM problem. MAPPO is a typical policy optimization method. The latter two, IQL and QMIX, are Q-value-based methods. IQL disregards the coupled relationships among agents, while QMIX improves performance by training the agent Q-value networks using a hyper-mixing network that captures these relationships. Generally speaking, Q-value-based methods are more sample-efficient and more suitable with discrete state space [19], which are two common features in MVPM problems.

The application of MARL to MVPM problems presents three key challenges: (1) The investigation into the impact of recent advancements in novel neural network architectures on the performance of MARL in the past two years is insufficiently explored. (2) There is a lack of appropriate indicators used for reward function design to assess MVPM problem effectively. Designs such as GBNN and DARP, which utilize coverage as an evaluation metric, are limited in their ability to detect intruders effectively. (3) There is an absence of comprehensive data comparisons between traditional algorithms and learning-based algorithms in addressing MVPM problems.

In this paper, to solve an MVPM problem in a complex environment we propose an enhanced QMIX method, named KAN-Mix, to improve dynamic patrolling performance in MVPM problems. This work makes three key contributions:

- QMIX network enhanced by Kolmogorov–Arnold networks (KANs): This is the first work to apply the KAN network to reinforcement learning tasks and evaluate its performance. The Multi-Layer Perceptron (MLP) in the action-value network of traditional Q-Mix is replaced with KAN which possesses fewer parameters and exhibits superior fitting performance [20]. Some researchers have initiated the application of KAN within the realm of Deep Learning (DL), such as KCN [21], and GKAN [22]. However, the utilization of KAN in reinforcement learning is still unexplored. We integrate KAN in QMIX to capture the nonlinear relationship between state, action, and Q-value. The experimental results demonstrate that the resulting KAN-Mix framework outperforms the original QMIX across all test metrics.
- Reward function formulated by information entropy: Inspired by Shannon's information entropy theory, we have formulated a reward function specifically for the MVPM problem. This design tackles the issues of reward sparsity and information dynamics. By nonlinearly adapting to variations in environmental information during monitoring, it effectively enhances area coverage and improves intruder catch rates.
- Comparison of traditional methods and various Q-value-based MARL methods: We conducted comparisons among traditional methods, such as GBNN and DARP, and typical Q-value-based methods, including IQL, VDN, QMIX, and KAN-Mix, to analyze the adaptability and

effectiveness of different Q-value architectures in the MVPM task under the same reward function and scenario settings. To achieve this, we designed three types of maps with varying levels of complexity and performed comprehensive analyses and evaluations with varying numbers of agents.

This paper is arranged as follows. Section 2 provides an overview of the research background regarding the MVPM problem. In Sec. 3, we introduce the modeling work of the MVPM problem, and Sec. 4 elaborates on the network details of the KAN-Mix. Section 5 presents experimental designs conducted in different maps, alongside comparative and ablation experiments. Finally, Sec. 6 offers a summary of the entire paper.

2. Related Works

The problem of MVPM has been investigated significantly in recent years, and its literature presents a wide range of methods. In [2, 3, 23], surveys of MVPM strategies are studied where strategies are evaluated based on agent perception, communication, coordination, and decision-making capabilities at different times. In this research, numerous scholars have concentrated on MVPM problems, establishing diverse evaluation metrics to assess performance across various dimensions. These metrics significantly influence the research emphasis of each investigator when addressing MVPM problems. Following the selection of suitable evaluation criteria, researchers typically explore algorithms for MVPM problem-solving through three principal methodologies: planning-based algorithms, heuristic algorithms, and learning-based algorithms.

2.1. *Evaluation metrics*

As is shown in Fig. 2, evaluation metrics for MVPM problem are mainly categorized into intrinsic and extrinsic evaluations. Intrinsic metrics assess the internal state of a multi-agent system during task execution, emphasizing the quality, efficiency. Extrinsic metrics evaluate the system's external objectives or outcomes, focusing on the interactions between the system and its environment. Intrinsic metrics gage the internal characteristics of the system's patrol strategy, such as idleness, node visitation rates, and environmental uncertainty. For instance, the authors of [2, 24–33] employ worst-case/average idleness as a metric, aiming to reduce the time agents spend returning to the same state or agent, thus enhancing patrol efficiency and quality. The authors of [26, 32, 34–39] consider node visitation frequency to enhance the multi-agent system's utilization rate by increasing the average visitation frequency per node or the frequency of specific nodes, thereby obtaining efficient patrol strategies. The authors of [40] optimize Mean First-Passage Time (MFPT) to balance the overall time consumption of different agents' corresponding strategies, thereby improving the efficiency of the strategies. Another approach uses environmental uncertainty as a metric, measured by the entropy or entropy rate of node return times [6, 7]. In terms of extrinsic metrics, they typically pertain to indicators related to the detection of intruders (maximizing the probability of detecting intruders [41, 42] or maximizing the number of intruders detected [43]). Since our MVPM task scenario utilizes griding maps and rewards designed based on information entropy, we employ coverage rate and total rewards as intrinsic metrics. To reflect the capability of unmanned systems in capturing

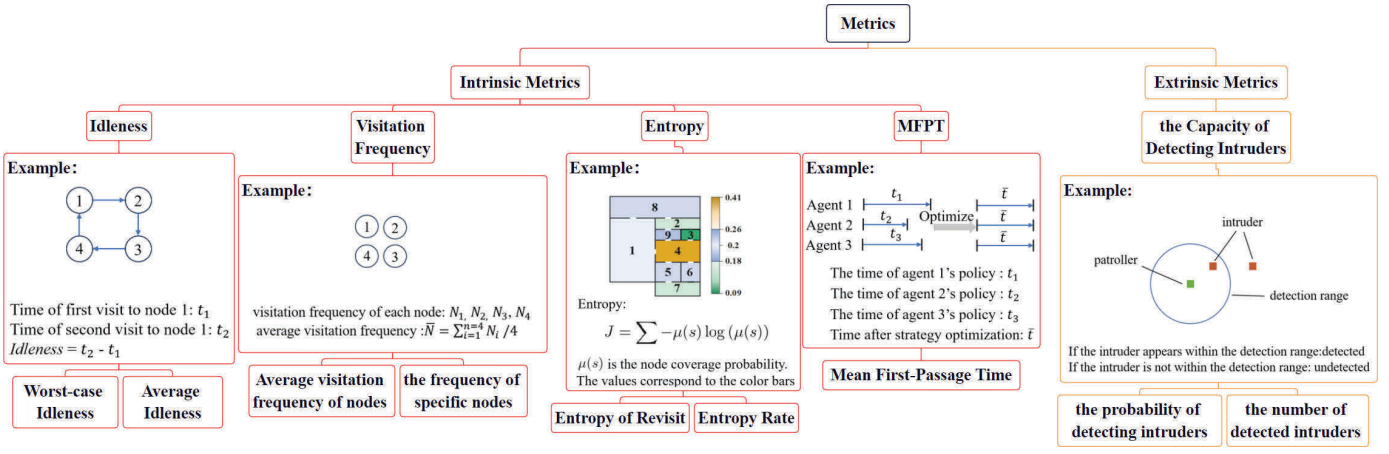


Fig. 2. A mind map on metrics classification. The second level involves the two primary categories of metrics, while the third level delineates the four existing intrinsic and extrinsic indicators. The fourth level provides illustrative examples to explain these indicators, and the fifth level offers a more detailed classification of the indicators discussed in the third level.

intruders, we use the catch rate and the catch step of intruders as external metrics of MVPM.

2.2. *Planning-based algorithms*

Planning-based algorithms systematically explore all possible states to find the optimal solution, typically solving only once. This category generally encompasses two research directions: offline or online planning. Offline planning based on specific rules or specified sequence orders [29], often modeled as a multi-agent optimization problem based on global goals [24, 37] or decomposed into multiple discrete single-agent goal optimization problems [27, 28, 33, 34, 39, 40, 44–46] for subsequent resolution. It formulates the MVPM problem into the mathematical optimization framework [47]. Online planning based on real-time environments usually requires large computing power. The authors of [24, 48] attempt to achieve large-scale real-time decision-making with smaller computing power around receding horizon optimization tools. The authors of [24] show that partitioning the MVPM area and applying the tree traversal approach can achieve similar performance to the unpartitioned case up to a certain number of robots but requires less optimization time for online planning in the MVPM problem. The authors of [48] use the matroid constraint approach to design its suboptimal submodular-based policy. This overcomes the challenge that the assigned policy per each mobile agent over the receding horizon is a vector rather than a point.

2.3. *Heuristic algorithms*

Heuristic algorithms [26, 32, 38, 43, 49–51] are constructed based on intuition or experience and do not guarantee an optimal solution. However, they provide an MVPM solution within acceptable time and space costs, often applied to NP-hard problems or those lacking known efficient algorithms. The authors of [26] present four strategies that consider important aspects of the patrolling movement, such as time, uncertainty, and communication. These strategies improve the centralized version of the NC-Drone by considering the uniform distribution of visits and drastically reducing the standard deviation, thereby making the algorithm more stable. The authors of [32] propose three decentralized techniques that use only local information written on the nodes and can be implemented in simple robots.

2.4. *Learning-based algorithms*

Learning-based algorithms, which typically translate multiple optimization objectives into rewards and optimize using Markov chains or various neural network structures. Depending on the solution logic, these algorithms are divided into reinforcement learning [17] and multi-agent learning [1]. The authors of [18] achieved relatively good results by specifically designing MVPM-related rewards and using QMIX to learn the monitoring strategy. However, when the complexity of MVPM increases, the effectiveness of this design and method remains to be verified. To enhance the performance of QMIX, we introduced KAN and designed task scenarios with varying levels of difficulty. Through extensive comparative algorithm experiments, we verified that the incorporation of KAN significantly improves the capabilities of QMIX.

Existing intrinsic metrics mainly focus on improving performance by increasing coverage or balancing execution time. However, the MVPM problem requires quick identification of potential intruders through effective strategies, which necessitates accurate modeling of intruder behavior. In the MVPM context, agents are unaware of intruder locations. Planning-based algorithms are limited by offline planning and poor robustness, while heuristic algorithms are constrained to small subsets of nodes for online planning. The KAN-Mix algorithm presented here overcomes these challenges by using intruder presence probability to calculate information entropy. Additionally, it leverages learning-based algorithms for improved robustness in solving the MVPM problem. A detailed description of the algorithm follows.

3. Problem Statement and Settings

3.1. *Problem statement for MVPM*

This section defines the MVPM problem in a reinforcement learning manner. We propose a patrolling environment G and subject it to grid-based processing $G = \{G_{(1,1)}, G_{(1,2)}, G_{(1,3)}, \ldots, G_{(H,H)}\}$, where H represents the side length of the map. There are obstacles distributed on the map, and intruders are set to appear randomly within the area.

The MVPM problem in this paper is established as a partially observable Markov game model represented by the tuple $\langle M, A, U, R, S, O, \pi, \gamma, T\rangle$, where $M = \{m_1, m_2, m_3, \ldots, m_n\}$ represents n monitoring agents, $S = \{S_1, S_2, S_3, \ldots, S_T\}$ describes the true state of the environment, $O = \{O^1, O^2, O^3, \ldots, O^N\}$ describes the individual observations of each agent and T represents the duration of monitoring. At each time step, each agent receives an observation O_t^i, and gets available action directions u_t^i, which constitutes all available action sets at all time, $U^i = \{u_1^i, u_2^i, u_3^i, \ldots, u_T^i\}$. Then available action sets of N agents $U = \{U^1, U^2, U^3, \ldots, U^N\}$. Then each agent selects an action a_t^i based on the information within the monitoring range, where $a_t \in u_t$,

and all agents' actions at the same time constitute a joint action $A_t = \{a_t^1, a_t^2, a_t^3, \ldots, a_t^n\}$, where $A_t \in A$, and A is the set of actions within N time steps, $A = \{A_1, A_2, A_3, \ldots, A_T\}$. All agents execute the joint action A_t and extend their paths, which leads to a change in the global state $S_{t-1} \times A_t \to S_t$. The state transition function $Z : S \times U \to P(S)$ defines the transition probabilities of possible states. All agents receive a monitoring reward $R_t : S \times \mathbf{U} \to \mathbb{R}, R_t \in \mathbb{R}$ after executing the joint action A_t. Each agent executes a monitoring strategy $\pi^i : O^i \to P(U^i)$, which maps the observations of each agent to a distribution over its available action set. Each agent has an action-observation history $\tau^i \in J \equiv (O \times U)^*$. After numbers of iterations, each agent learns a monitoring strategy π^i to maximize its expected discounted return, $Q^i(\pi^i) = E_{(a^1 \sim \pi^1, \ldots, a^N \sim \pi^N, S \sim Z)}\ [\sum_{t=0}^{\infty} \gamma^t r_t^i (S_t, a_t^1, \ldots, a_t^N)]$, where $\gamma \in [0, 1]$. Since all agents share the same monitoring reward computation function, we consider the area monitoring as a fully cooperative partially observable Markov game.

Although training is centralized, execution is decentralized, and the learning algorithm has access to all local action observation histories τ and global state S, each agent's learned policy can condition only on its action observation history τ^i.

3.2. *Environmental setup and action space design*

We discretize the environment using an occupancy grid map of patrolling areas. Taking the diameter of the agent detection as a reference, calculate the unit length, discretely represent the patrol scene, and grid-model the spatial coordinate system. As shown in Fig. 3, the side length a of a grid is set to be one-third of the diameter of the agent's detection circle:

$$a = \frac{\sqrt{2}}{6} d, \tag{1}$$

where d represents the detection diameter. If a grid length greater than the detection diameter is employed, it becomes challenging to determine whether the agent has successfully detected all relevant information within that range. Conversely, while reducing the grid size could improve detection accuracy, it may lead to a decrease in the overall detection efficiency of the agent.

Without considering motion constraints, we discretize the action space of the agent, where each agent can move in nine directions (forward, backward, left, right, left front, right front, left back, right back, and stay put), as shown in Fig. 3. We define a_t^i to represent the action taken by the agent i at time t.

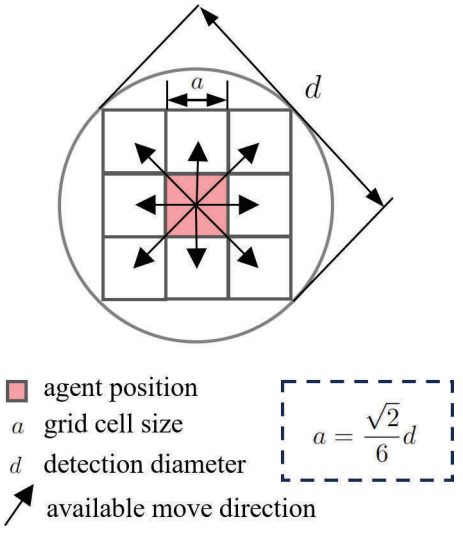


Fig. 3. Action space definition.

3.3. *State space design*

The global state space S is crucial for the cluster to understand the information of the map. We design coverage information maps and monitoring information maps, which together constitute the cluster's global state space. That is, the global state at time t, denoted as S_t, is composed of the coverage state S_c and the monitoring information state S_{info}.

$$S_t = \langle S_c, S_{\text{info}} \rangle. \tag{2}$$

3.3.1. *Coverage state*

The coverage state is $S_c = \{s_{(1,1)}, s_{(1,2)}, s_{(1,3)}, \ldots, s_{(H,H)}\}$, where H represents the side length of the map, consists of grids of size $H \times H$. Figure 4 is an example of the coverage state map at time t. The state of each grid is denoted as $s_{(j,k)}$, representing the state of the grid at coordinates (j, k):

$$s_{(j,k)} = \begin{cases} 0 & \text{grid } (j,k) \text{ if the grid has not been detected,} \\ 0.5 & \text{grid } (j,k) \text{ is detected and not an obstacle,} \\ 1 & \text{grid } (j,k) \text{ is detected and is an obstacle.} \end{cases} \tag{3}$$

When the patrol task has not yet begun, as the area information is unknown, the state of all grids $s_{(j,k)}$ is 0. After the patrol task commences, the information state of grids within the detection range will change.

3.3.2. *Monitoring information state*

The monitoring information state is defined as $S_{\text{info}} = \{\lambda_{(1,1)}(t), \lambda_{(1,2)}(t), \lambda_{(1,3)}(t), \ldots, \lambda_{(H,H)}(t)\}$, used for calculating information entropy. The monitoring

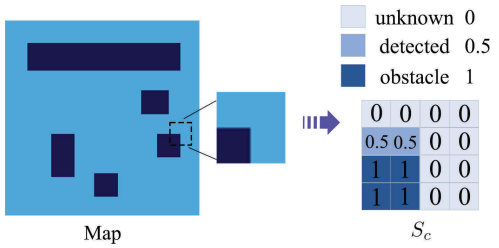


Fig. 4. The illustration of the coverage state.

information state of each grid is denoted as $\lambda_{(j,k)}(t)$, representing the intruder's existence probability at the grid with coordinates (j,k):

$$\lambda_{(j,k)}(t) = \begin{cases} 0 & \text{if grid } (j,k) \in \text{Agent}(t), \\ \lambda_{(j,k)}(t-1) + \dfrac{1}{T} & \text{if grid } (j,k) \notin \text{Agent}(t), \end{cases} \tag{4}$$

where $\lambda_{(j,k)}(0)$ is set to be 0.5 so that the probability of each grid containing or not containing an intruder is initialized to be equal. T represents the maximum patrol time. grid (j,k) denotes the grid corresponding to coordinates (j,k), Agent (t) represents the detection range of the agent. Figure 5 is an example of the monitoring information map at time $t = 0$.

3.4. *Observation space design*

The observation space consists of two parts: the patrol path L_t^i of the ith agent, denoted as $L_t^i = \{l_1^i, l_2^i, \ldots, l_t^i, \ldots, l_T^i\}$, and the historical turning angles θ_t^i of the ith agent, denoted as $\theta_t^i = \{\theta_1^i, \theta_2^i, \ldots, \theta_t^i, \ldots, \theta_T^i\}$. Accordingly, $o_t^i = \langle L_t^i, \theta_i^t, i\rangle$ represents the individual observation space of the ith agent at time t. Additionally, each agent has access to the observation space of other agents, collectively forming a combined observation space $O_t^i = \langle o_t^i, o_t^{\text{other}}\rangle$, where o_t^{other} represents the collection of observation states of other agents.

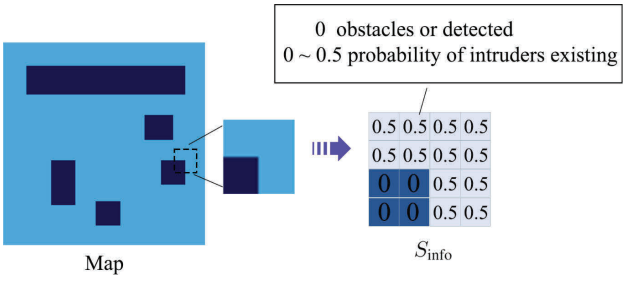


Fig. 5. The illustration of the monitoring information state. "0" denotes the detection or presence of an obstacle, while values ranging from "0" to "0.5" represent the probability of an intruder potentially being present. Given the inherent uncertainty regarding the intruder's presence, the highest probability is limited to 0.5, indicating that the likelihood of the intruder's presence is equally uncertain, analogous to the binary presence or absence of the intruder.

3.5. *Reward function design*

MVPM is a problem that involves inferring the potential location of an intelligent agent through environmental monitoring. This process leads to dynamic updates in the agent's known information of the environment. Inspired by Shannon's information theory, we designed the reward function based on information entropy. The function of information entropy evaluates uncertainties, and we utilize it to describe the uncertainty regarding the existence condition of the intruder in a certain grid.

The reward calculation process of a single intelligent agent is shown in Fig. 6. With $\lambda_{(j,k)}(t)$ being treated as the probability mass function, the amount of information present in a single grid is denoted as $I_{(j,k)}(t)$:

$$I_{(j,k)}(t) = \begin{cases} 0 & \text{if } \lambda_{(j,k)}(t) = 0, \\ -\lambda_{(j,k)}(t)\ \log \lambda_{(j,k)}(t) & \text{if } \lambda_{(j,k)}(t) \neq 0. \end{cases} \tag{5}$$

Assuming the presence of intruders in each grid is a Bernoulli distribution, the information entropy corresponding to each grid is

$$e_{(j,k)}(t) = I_{(j,k)}(t). \tag{6}$$

The single-agent reward is defined as the information gain at time t:

$$r_i(t) = -\sum [e_{j,k}(t) - e_{j,k}(t-1)], \tag{7}$$

where grid(j,k) have been searched by agent i.

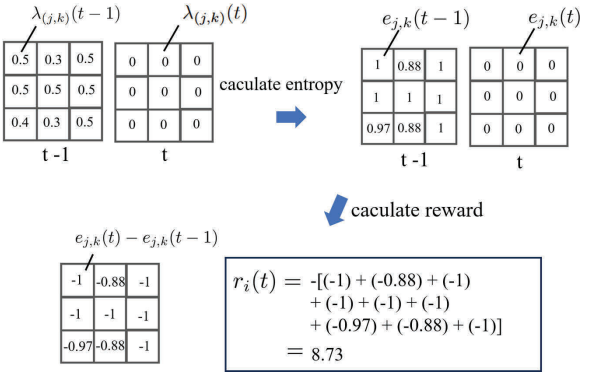


Fig. 6. The illustration of reward calculate. Each agent calculates the probability of potential intruders within each grid cell of its monitoring domain. Based on these probabilities, the agent then calculates the information entropy for each grid. Finally, the agent evaluates its performance by calculating a reward functions.

The multi-agent monitoring reward is the sum of all single-agent rewards at time t:

$$R_t = \sum_{i=1}^{N} r_i(t), \tag{8}$$

where N is the total number of agents.

4. QMIX Network Enhanced by KANs

This section details the Q-value mixing network enhanced by KANs (KAN-Mix). KAN-Mix adheres to the QMIX framework, employing Centralized Training and Decentralized Execution (CTDE). The KAN-Mix consists of a mixing network and KAN action-value networks, as shown in Fig. 7. The mixing network serves the purpose of training the KAN networks during the training phase. Once trained, only the KAN networks are required for task execution in subsequent phases. Appendix A describes the pseudocode of this algorithm and the specific details of the algorithm process.

4.1. *KAN action-value network*

As shown in Fig. 7(c), we employ KANs as the agents' action-value network to accommodate a vast state space.

KAN, inspired by the Kolmogorov–Arnold representation theorem, is an advanced type of neural network featuring learnable activation functions on edges of the neurons, as opposed to the fixed activations on nodes in traditional MLPs [20]. These activation functions are parameterized by B-splines, which are piecewise polynomial functions defined by control points and knots. The Kolmogorov–Arnold representation theorem states that any multivariate continuous function can be represented as a composition of univariate functions and the addition operation.

$$f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right), \tag{9}$$

where $\phi_{q,p}$ are univariates' functions that map each input variable (x_p) such $\phi_{q,p} : [0, 1] \to \mathbb{R}$ and $\phi_q : \mathbb{R} \to \mathbb{R}$. Since all functions to be learned are univariate functions, we can parametrize each 1D function as a B-spline curve, with learnable coefficients of local B-spline basis functions. The key insight comes when we see the similarities between MLPs and KANs. In MLPs, a layer includes a linear transformation followed by nonlinear operations, and you can make the network deeper by adding more layers. Let us define what a KAN layer is

$$\Phi = \{\phi_{q,p}\}, \quad p = 1, 2, \ldots, n_{\text{in}}, \quad q = 1, 2 \ldots, n_{\text{out}}, \tag{10}$$

where $\phi_{q,p}$ are parametrized function of learnable parameters. In the Kolmogorov–Arnold theorem, the inner functions form a KAN layer with $n_{\text{in}} = n$ and $n_{\text{out}} = 2n + 1$, and the outer functions form a KAN layer with $n_{\text{in}} = 2n + 1$ and $n_{\text{out}} = 1$. So, the Kolmogorov–Arnold representations in Eq. (10) are simply compositions of two KAN layers. Now it becomes clear what it means to have deep Kolmogorov–Arnold representation. Taking the notation from [20] let us define a shape of KAN $[n_0, n_1, \ldots, n_L]$, where n_i is the number of nodes in the ith layer of the computational graph. We denote the ith neuron in the lth layer by (l, i), and

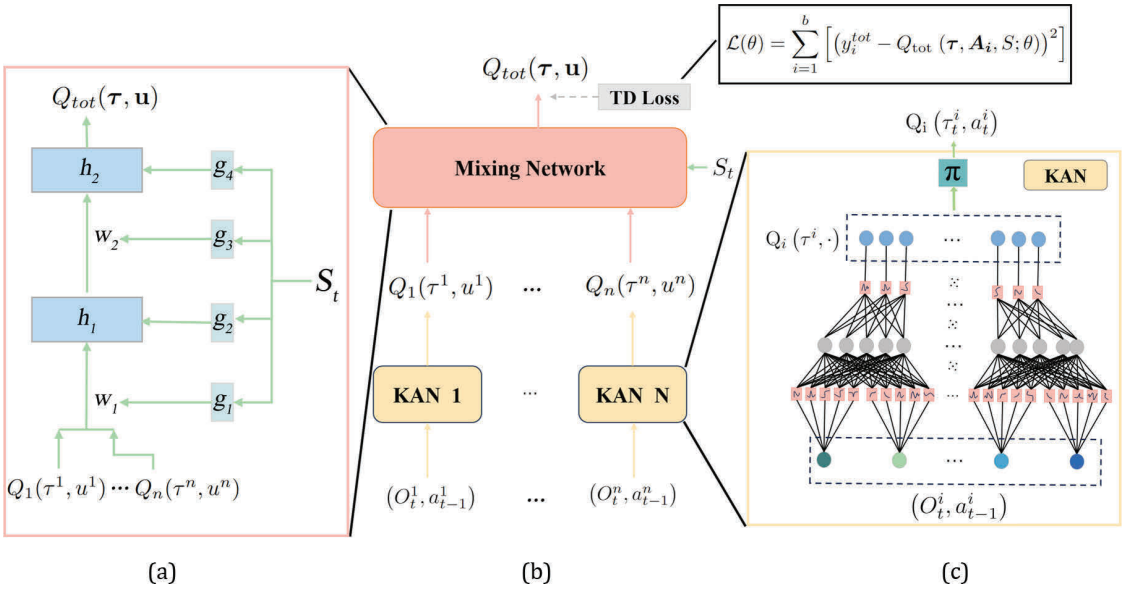


Fig. 7. (a) Mixing network, (b) the main framework of KAN-Mix, (c) KAN action-value network.

the activation value of the (l,i) neuron by $x_{l,i}$. Between layer l and layer $l+1$, there are $n_l n_{l+1}$activation functions: the activation function that connects (l,i) and $(l+1,j)$ is denoted by

$$\phi_{l,j,i}, \quad l=0,\ldots,L-1, \quad i=1,\ldots,n_l, \quad j=1,\ldots,n_{l+1}. \tag{11}$$

The pre-activation of $\phi_{l,j,i}$ is simply $x_{l,i}$; the post-activation of $\phi_{l,j,i}$ is denoted by $\tilde{x}_{l,j,i} \equiv \phi_{l,j,i}(x_{l,i})$. The activation value of the $(l+1,j)$ neuron is simply the sum of all incoming post-activations:

$$x_{l+1,j} = \sum_{i=1}^{n_l} \tilde{x}_{l,j,i} = \sum_{i=1}^{n_l} \phi_{l,j,i}(x_{l,i}), \quad j=1,\ldots,n_{l+1}. \tag{12}$$

Rewriting it under the matrix form will give

$$\mathbf{x}_{l+1} = \underbrace{\begin{pmatrix} \phi_{l,1,1}(\cdot) & \phi_{l,1,2}(\cdot) & \cdots & \phi_{l,1,n_l}(\cdot) \\ \phi_{l,2,1}(\cdot) & \phi_{l,2,2}(\cdot) & \cdots & \phi_{l,2,n_l}(\cdot) \\ \vdots & \vdots & & \vdots \\ \phi_{l,n_{l+1},1}(\cdot) & \phi_{l,n_{l+1},2}(\cdot) & \cdots & \phi_{l,n_{l+1},n_l}(\cdot) \end{pmatrix}}_{\mathbf{\Phi}_l} \mathbf{x}_l, \tag{13}$$

where Φ_l is the function matrix corresponding to the lth KAN layer. A general KAN network is a composition of L layers: given an input vector $\mathbf{x}_0 \in \mathbb{R}^{n_0}$, the output of KAN is

$$\mathrm{KAN}(\mathbf{x}) = (\mathbf{\Phi}_{L-1} \circ \mathbf{\Phi}_{L-2} \circ \cdots \circ \mathbf{\Phi}_1 \circ \mathbf{\Phi}_0)\mathbf{x}. \tag{14}$$

Concatenate O_t^i and a_{t-1}^i into $\mathbf{x}$:

$$\mathbf{x} = \langle O_t^i, a_{t-1}^i \rangle. \tag{15}$$

At this point, x serves as the input of the KAN network and the output obtained is the q of each agent. The Q-value of each agent is

$$\mathrm{Q}_i(\tau_t^i, a_t^i) = \mathrm{KAN}(O_t^i, a_{t-1}^i) \circ \pi_i, \tag{16}$$

where O_t^i represents the observation of agent i at time t, a_{t-1}^i represents the action at $t-1$, π_i represents the set of optional actions for agent i in the current time, and O_t^i is the value set of movable directions in the current time. Each agent's learned policy can condition only on its action observation history τ^i.

As illustrated in Fig. 8, during task execution, $\langle O_t^i, a_{t-1}^i \rangle$ serves as the input to the action value network of each agent, which computes the action value for each possible direction. The agent then selects the direction with the highest action value for its subsequent move.

4.2. *Mixing network*

As shown in Fig. 7(a), we employ a hypernetwork as a mixing network. The operational principle of this network adheres to Eq. (17). The hypernetwork utilizes the global

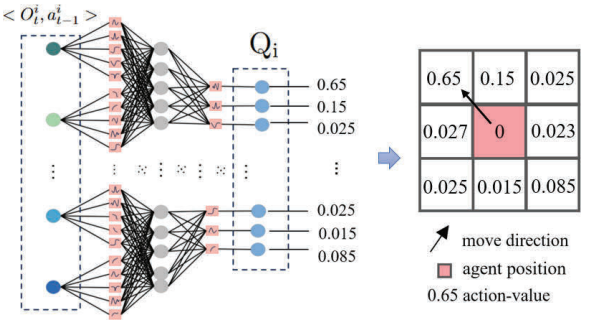


Fig. 8. Explanation of action selection in KAN.

state and the action value of individual agents as inputs, and it nonlinearly fits the joint action value Q_{tot} through the hypernetwork.

$$\underset{\mathbf{A_t}}{\mathrm{argmax}}\, Q_{\mathrm{tot}}(\boldsymbol{\tau}, \mathbf{A_t}) = \begin{pmatrix} \mathrm{argmax}_{a_t^1} Q_1(\tau_t^1, a_t^1) \\ \vdots \\ \mathrm{argmax}_{a_t^n} Q_n(\tau_t^n, a_t^n) \end{pmatrix}, \tag{17}$$

and the mixing network employs two sets of identical network structures, namely, estimation network and target network, to respectively, obtain Q_{tot}^* and y_{target}^*. KAN-Mix is trained end-to-end to minimize the following loss:

$$\mathcal{L}(\theta) = \sum_{i=1}^{b} [(y_i^{\mathrm{tot}} - Q_{\mathrm{tot}}(\boldsymbol{\tau}, \mathbf{A_i}, S; \theta))^2], \tag{18}$$

where b is the batch size sampled from the replay buffer, and θ represents the parameters of the network. $y_{\mathrm{target}} = r + \gamma \max_{\mathbf{A_i}'} Q_{\mathrm{tot}}(\boldsymbol{\tau}', \mathbf{A_i}', S'; \theta^-)$ and θ^- are the parameters of a target network as in KANs. Since Eq. (18) holds, we can perform the maximization of Q_{tot} in time linear in the number of agents (as opposed to scaling exponentially in the worst case).

5. Experiments

5.1. *Experiment settings*

We designed three 50×50 maps for algorithm validation, each with distinct features: a barrier-free map, a complex obstacle map, and a connected six-room map (as shown in Fig. 9). We tested scenarios with 3, 6, and 9 agents on these maps for the MVPM task. Intruders randomly appear at any position, and detection by monitoring is considered as being caught.

The parameter settings related to KAN-Mix can be found in Table 1. The KAN-Mix and comparison algorithms are trained under a distributed training framework with 12 CPU cores and 2 NVIDIA GeForce RTX 3090 GPUs.

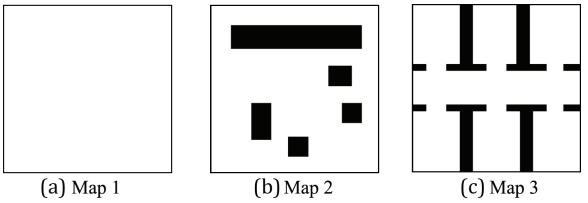


Fig. 9. Three different maps: (a) Map 1: barrier-free, (b) Map 2: complex obstacle, (c) Map 3: connected six-room. The black region denotes the impassable zones and obstacles, whereas the white region represents the traversable areas.

Table 1. The parameter settings of KAN-Mix.

Parameters	Settings
Monitoring step	1000,1000,400
Learning rate	0.001
Batch size	128
Training iterations	25K
Testing iterations	10K
KAN Layer number	3
Mixing network layer number	4

5.2. *The output trajectories of KAN-Mix*

KAN-Mix is designed to train multi-agent systems to generate cooperative PM trajectories. To demonstrate its effectiveness in improving monitoring coverage, we compared the initial trajectories (before training the KAN) with the stable trajectories (after training the KAN). The visualizations of these trajectories are shown in Fig. 10. To compare the effectiveness of the patrol strategy, we also presented the results of the well-trained agents in Fig. 10.

First, we analyzed the impact of KAN-Mix before and after training by depicting the trajectories of nine agents on each map, as shown in Fig. 10. The untrained agents exhibit numerous repetitive paths and often get stuck in place on maps with obstacles. In contrast, the trained agents show reduced path repetition, employ proactive obstacle avoidance strategies, and maximize the MVPM effect. Finally, we visualized the trajectory of the agent's continuous monitoring process. The results demonstrate that the nine intelligent agents have achieved near-complete coverage of the three maps. This demonstrates the significant improvement in monitoring coverage and efficiency achieved by training the agents with KAN-Mix.

5.3. *Comparison with traditional algorithms and Q-value-based MARL algorithms*

To more effectively highlight the advantages of the KAN-Mix algorithm, we conducted a comparative analysis with

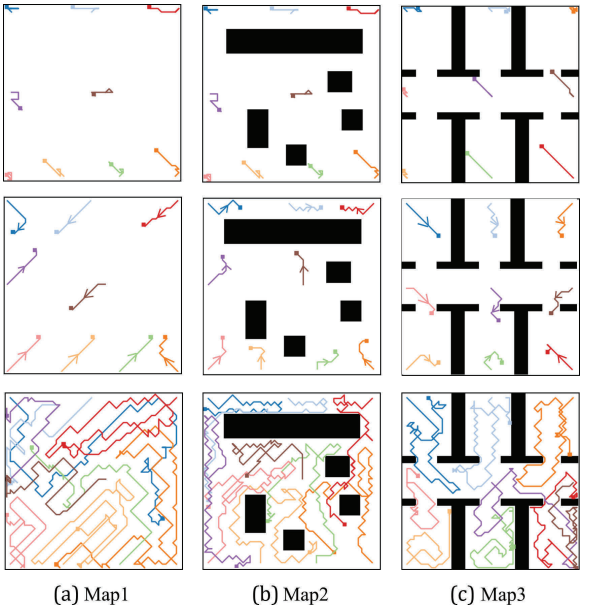


Fig. 10. The output trajectories of KAN-Mix (nine agent). The colored square represents the current position of the agent, and the arrow indicates the direction of movement. The first line shows the trajectories of the untrained agents (the first 10 step), and the second line shows the trajectories of the trained agents (the first 10 step), and the third line shows the trajectories of the well-trained agents.

traditional algorithms, such as Divide Areas based on Robots Initial Positions (DARP) and Glasius bio-inspired neural networks (GBNN), as well as conventional MARL algorithms based on Q-value optimization, including Independent Q-Learning (IQL), VDN, and QMIX. The training iterations for the conventional MARL algorithms are set to 25K, while the testing iterations are configured to 10K for all algorithms. The training results of conventional MARL algorithms algorithm are recorded in Fig. 11, and the testing results are recorded in Tables 2–4.

In all scenarios shown in Fig. 11, VDN, which merely sums the Q-values, did not outperform IQL which does not sum Q-values in experimental outcomes. In fact, on most maps, it converged more slowly and the rewards of the trained strategies were notably lower than those achieved by IQL. The QMIX and KAN-Mix, which engage in the nonlinear aggregation of Q-values, demonstrate notably superior training performance compared to IQL, and VDN. Within the initial 1K training iterations, they surpassed the performance of the aforementioned three algorithms. Furthermore, in the subsequent iterations, they yielded cumulative rewards that were significantly greater than those achieved by the other algorithms.

To evaluate the performance of the models trained by each algorithm, we selected the top 20% of the MARL models based on their total reward as the test models,

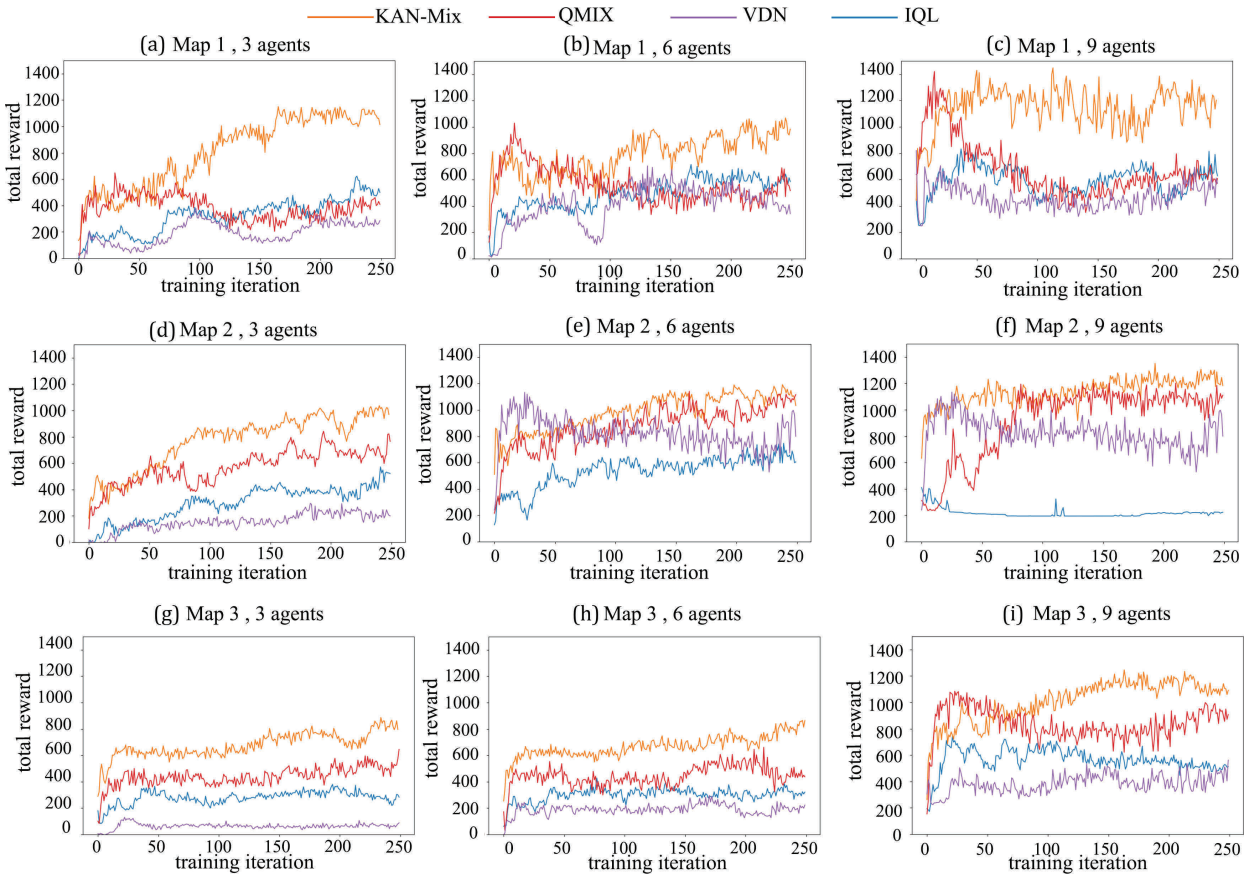


Fig. 11. Training results across various maps, algorithms, and agent numbers (averaged every 100 episodes).

constituting a 10K test dataset. The test results are presented in Tables 2–4. We have placed the monitoring information entropy results of the final step of nine agents on different maps and algorithms in Fig. 12. Due to the computation dead loop of GBNN on MAP 3, the information entropy visualization of GBNN is not shown in Fig. 12.

As shown in Fig. 12, the random distribution of policy information entropy is disordered, whereas MARL leverages learned policies to render the information entropy of the environment more ordered. Although the information entropy of traditional algorithms has been significantly reduced, DARP's limitation of not revisiting previously explored areas results in an increase in the information entropy of regions explored earlier. Nevertheless, the overall environmental knowledge acquired by DARP is substantially superior to that of random, IQL and VDN. The information entropy of the strategies learned by QMIX across the entire map is significantly lower than that of IQL and VDN (lighter blue indicates higher environmental knowledge). In contrast, KAN-Mix exhibits only a small number of grids in dark blue, indicating that the strategies learned by KAN-Mix possess a higher level of environmental knowledge compared to other algorithms.

Since the GBNN and DARP algorithms do not require rewards for iteration, this item is crossed out by a "/" in the table. First, we analyze the performance of each algorithm on the different maps individually. As shown in Table 2, for the accessible MAP 1, both GBNN and DARP exhibit high coverage. However, the probability of successfully capturing intruders is approximately 10% lower for these two algorithms compared to other methods. While MARL-based approaches may not perform as well in GBNN and DARP, they maintain a capture probability exceeding 95%. In this accessible map, traditional algorithms are insufficient for efficiently detecting intruders, suggesting that a higher coverage rate correlates with improved performance in solving the MVPM problem. This result further confirms that rewards based on information entropy are more effective than rewards based on coverage.

In Table 3, for MAP 2, which contains several significant obstacles, GBNN and DARP still achieve 100% coverage, but the probability of intruder detection has decreased, particularly for the GBNN algorithm. Although GBNN demonstrates a faster intruder capture rate in the data, this is primarily due to a lower number of intruder captures.

Table 2. Comparison across different algorithms on Map 1.

*N*_agent	Algorithm	Total reward	Coverage rate	Catch rate	Catch step
3	Random	623.14	0.622	0.995	100
	GBNN	\	**1**	0.904	**78**
	DARP	\	**1**	0.865	263
	IQL	829.96	0.767	0.989	106
	VDN	594	0.465	0.989	130
	QMIX	918.21	0.741	0.99	122
	KAN-Mix	**1299.9**	0.929	**1**	127
6	Random	959.67	0.794	0.981	**48**
	GBNN	\	**1**	0.817	69
	DARP	\	**1**	0.87	131
	IQL	1029.52	0.631	0.983	58
	VDN	818.77	0.596	0.984	51
	QMIX	859.7	0.67	0.986	54
	KAN-Mix	**1249.1**	0.88	**0.988**	53
9	Random	1252.18	0.721	0.973	**32**
	GBNN	\	**1**	0.989	106
	DARP	\	**1**	0.87	89
	IQL	801	0.677	0.973	58
	VDN	677.35	0.493	0.975	46
	QMIX	1273	0.812	0.975	33
	KAN-Mix	**1471**	0.915	**0.98**	40

Table 3. Comparison across different algorithms on Map 2.

*N*_agent	Algorithm	Total reward	Coverage rate	Catch rate	Catch step
3	Random	466.6	0.547	0.99	99
	GBNN	\	**1**	0.817	**69**
	DARP	\	**1**	0.793	232
	IQL	550.42	0.681	0.989	117
	VDN	384	0.491	0.989	142
	QMIX	790.59	0.856	0.99	102
	KAN-Mix	**1028.5**	0.938	**1**	103
6	Random	740.34	0.742	0.982	51
	GBNN	\	**1**	0.74	69
	DARP	\	**1**	0.795	114
	IQL	853.32	0.657	0.982	59
	VDN	1225.7	0.802	0.983	**36**
	QMIX	1176.25	0.824	0.993	50
	KAN-Mix	**1245.4**	0.876	**1**	48
9	Random	989.4	0.696	0.972	33
	GBNN	\	**1**	0.465	**11**
	DARP	\	**1**	0.788	75
	IQL	448	0.466	0.949	45
	VDN	1184	0.838	0.972	39
	QMIX	1206.43	0.877	0.974	39
	KAN-Mix	**1299**	0.94	**0.981**	25

Table 4 highlights the performance on a connected six-room map, where GBNN encounters a computational deadlock, leading to the exclusion of its results, marked with a "-". In contrast, QMIX and KAN-MIX consistently maintain a capture probability exceeding 97% while ensuring high coverage and faster intruder detection. The results of these two maps also indicate that GBNN and DARP ensure high coverage, but are prone to missing intruders in the maps.

Overall, as shown in Tables 2–4, in completely accessible Map 1 and connectivity-rich Map 3, while both GBNN and DARP are capable of ensuring complete coverage during the exploration of the entire map, their trajectories exhibit a high degree of regularity, making them susceptible to evasion by intruders. Consequently, their probability of capturing intruders is lower than that of other methods. Notably, DARP requires considerably more time to capture intruders compared to alternative algorithms. As the map's complexity increases, particularly with more intricate obstacles, the performance of both methods declines significantly. Specifically, the probability of intruder capture for GBNN decreases by approximately 48.57%, while DARP's decreases by 10%.

The random strategy does not perform worse than IQL and VDN. On the contrary, due to its complete randomness, it demonstrates a faster intruder capture capability in certain map scenarios (several scenarios in Map 1, achieving approximately 10% faster catch step). However, as maps grow more complex, such as with the appearance of larger

Table 4. Comparison across different algorithms on Map 3.

*N*_agent	Algorithm	Total reward	Coverage rate	Catch rate	Catch step
3	Random	775.92	0.77	0.983	49
	GBNN	—	—	—	—
	DARP	\	**1**	0.782	219
	IQL	382.15	0.548	0.982	50
	VDN	121.75	0.267	0.991	97
	QMIX	623.52	0.751	0.983	49
	KAN-Mix	**869.54**	0.958	**0.991**	**48**
6	Random	775.75	0.773	0.982	50
	GBNN	—	—	—	—
	DARP	\	1	0.774	111
	IQL	391.54	0.539	0.982	46
	VDN	191	0.39	0.98	**35**
	QMIX	652.85	0.828	0.992	42
	KAN-Mix	**823**	**0.94**	**0.993**	39
9	Random	1036.73	0.773	0.973	32
	GBNN	—	—	—	—
	DARP	\	**1**	0.783	74
	IQL	741.88	0.577	0.968	41
	VDN	454.79	0.459	0.973	35
	QMIX	1154.12	0.827	0.976	33
	KAN-Mix	**1182**	0.902	**0.977**	**31**

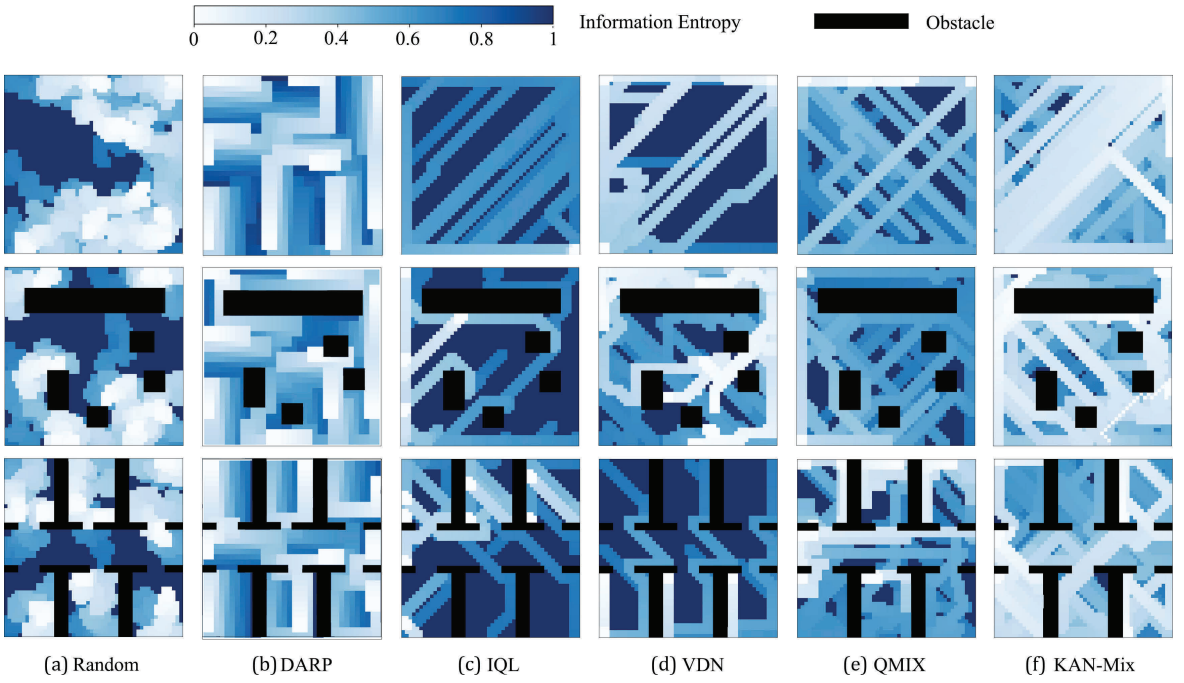


Fig. 12. Testing results about information entropy across various map, and algorithms (nine agents).

obstacles in Map 1 that obstruct feasible paths, their effectiveness noticeably diminishes. In the test results, the simple summation of Q-values in VDN does not outperform IQL, which does not aggregate Q-values. Across most map scenarios, IQL exhibits superior coverage and catch step performance compared to VDN by approximately 20%. Only in the 6-agent scenario of Map 2 does VDN demonstrate better effectiveness than IQL. The QMIX method, which employs nonlinear aggregation of Q-values, significantly outperforms random, IQL, and VDN strategies. QMIX achieves rewards and coverage rates that are at least 30% higher compared to these methods. In various testing scenarios, KAN-Mix exhibits the highest rewards and the highest coverage, along with the fastest intrusion detection capabilities in most environments. In a few specific scenarios (e.g. Map 1), it slightly underperforms compared to random within the catch step metric (within 10 steps). This highlights the strong comprehensive capability of KAN-Mix, especially when considering its reward effectiveness designed in conjunction with information entropy.

To further evaluate the effectiveness of QMIX and KAN-Mix, which both utilize nonlinear additive Q-values, we initially compared their parameter counts and memory usage, as illustrated in Fig. 13. KAN-Mix exhibits a decision-making speed that is approximately 16.32% slower than that of QMIX. KAN-Mix has 2.49% fewer parameters but 1.65% greater memory usage compared to QMIX, suggesting that the activation function in KAN consumes more memory. Despite the increased memory usage during

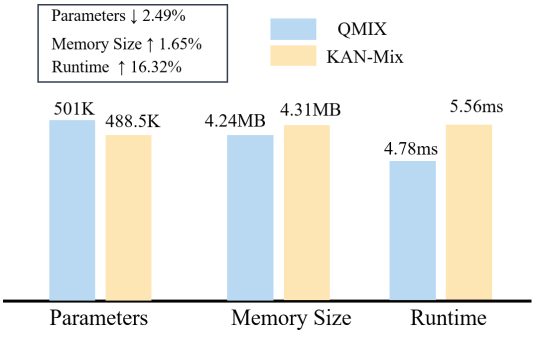


Fig. 13. Comparison of parameters, memory size, runtime.

training, KAN-Mix converged faster than QMIX and achieved significantly higher rewards across all scenarios, justifying the use of KAN to optimize QMIX.

During the testing phase, KAN-Mix demonstrated superior performance over QMIX, achieving coverage improvements of at least 10% and securing cumulative rewards that were 10–30% higher across specific scenarios. Notably, in extensive testing scenarios encompassing 10,000 testing iterations on varied maps, KAN-Mix consistently achieved a 100% capture rate, a feat unattained by QMIX.

5.4. *Ablation study on reward function*

To verify the impact of reward design, we compared the information-entropy-based reward (named reward type 1)

Table 5. Performance comparison between reward designs.

Map	*N*_agent	Reward type	Coverage rate	Catch rate	Catch step
1	3	1	**0.929**	**1**	**127**
		2	0.616	0.991	131
	6	**1**	**0.88**	**0.988**	**53**
		2	0.813	0.991	60
	9	**1**	**0.915**	**0.98**	**40**
		2	0.88	0.978	43
2	3	**1**	**0.938**	**1**	**103**
		2	0.692	0.982	121
	6	1	**0.876**	**1**	**48**
		2	0.819	0.988	59
	9	**1**	**0.94**	**0.981**	**25**
		2	0.883	0.976	37
3	3	1	**0.958**	**0.991**	**48**
		2	0.607	0.989	126
	6	1	**0.94**	**0.993**	**39**
		2	0.815	0.978	52
	9	1	**0.902**	**0.977**	**31**
		2	0.856	0.987	33

with the traditional coverage-based reward (named reward type 2) on KAN-Mix in ablation experiments. We conducted a comprehensive testing regimen comprising 10,000 iterations on KAN-Mix models selected from the top 20% in terms of coverage achieved during training based on reward type 1 and reward type 2. The results, shown in Table 5, indicate that the KAN-Mix network designed with information-entropy-based rewards achieves higher coverage, cumulative rewards, and faster intruder detection compared to the traditional coverage-based reward.

6. Conclusion

In this paper, we introduce the KAN-Mix network, an enhanced version of the QMIX network that incorporates KANs to address the MVPM problem. Our approach includes an information entropy-based reward system designed to improve the efficiency of multi-agent systems in promptly detecting intruders while ensuring comprehensive regional monitoring coverage. The KAN component in KAN-Mix demonstrates superior fitting capabilities compared to the traditional MLP used in QMIX, enabling sustained high coverage and detection rates even in complex environments. Ablation studies confirm that rewards based on information entropy are well-suited for MVPM problems.

The results across all experimental scenarios indicate that KAN-Mix, characterized by enhanced Q-value fitting and nonlinear aggregation capabilities, surpasses alternative algorithms in overall performance when guided by rewards derived from information entropy principles. This study presents a practical implementation of KAN within MARL frameworks, suggesting novel avenues for optimizing MARL methodologies. Furthermore, the design of rewards based on information entropy theory introduces a more theoretically robust optimization criterion for investigating MVPM challenges.

While KAN-Mix has demonstrated strong performance in addressing the MVPM problem, its decision-making time is slower compared to QMIX. Additionally, this study explores the MVPM problem without accounting for agent dynamics and energy consumption constraints. Future research could focus on integrating capability constraints and energy consumption considerations into the reward design to further optimize performance.

Acknowledgments

This research was funded by the National Key Research and Development Program of China, grant number 2021YFC2803003, and the National Natural Science Foundation of China, grant number 62173220.

Appendix A. Pseudocode

The details of our algorithm are given in Algorithm A.1.

Algorithm A.1. KAN-Mix.

Input: Environment E, Number of agents N, Discount factor γ, Learning rate α, Number of episodes M, Replay buffer $\mathcal{D}$, Target network update frequency T
Output: Policy π_θ
Initialize KAN networks Q_i for each agent i, with parameters θ_i
Initialize mixing network Q_{tot}, with parameters θ_{tot}
Initialize target networks $\hat{Q}_i$ and $\hat{Q}_{\text{tot}}$
for episode $m = 1, 2, \ldots, M$ **do**
Initialize state S_0 and agent states $O_0^i = \varnothing$
for time step $t = 0, 1, 2, \ldots$ **do**
for each agent i **do**
Select action $a_i^t \sim \pi_{\theta_i}(s^t)$ using epsilon-greedy policy
end for
Execute actions $\{a_i^t\}$, observe next state s_{t+1}, and rewards r_i^{t+1}
Store transition $(s_t, \{a_i^t\}, r_t, s_{t+1})$ in replay buffer $\mathcal{D}$
Sample a mini-batch of transitions $\{(s_t, \{a_i^t\}, r_t, s_{t+1})\}$ from $\mathcal{D}$
Compute values of each KAN network $\hat{Q}_i$
Compute values of mixing network:

$$(Q_{\text{tot}}) = \text{QMIX}(Q_1, Q_2, \ldots, Q_N)$$

Compute target value for mixing network:

$$(y_{\text{tot}} = R_t + \gamma \max_{a'} Q_i(s_{t+1}, a'))$$

Update parameters θ_i and θ_{tot} by minimizing the loss:

$$\mathcal{L} = \|Q_{\text{tot}} - y_{\text{tot}}\|^2$$

if $t \mod T == 0$ **then**
Update target networks:

$$\hat{Q}_i \leftarrow Q_i, \hat{Q}_{\text{tot}} \leftarrow Q_{\text{tot}}$$

end if
end for
end for

ORCID

Yangmin Xie https://orcid.org/0000-0002-6517-2917

References

[1] J. Chen, A. Baskaran, Z. Zhang and P. Tokekar, Multi-agent reinforcement learning for visibility-based persistent monitoring, in *2021 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)* (IEEE, 2021), pp. 2563–2570.

[2] A. Almeida, G. Ramalho, H. Santana, P. Tedesco, T. Menezes, V. Corruble and Y. Chevaleyre, Recent advances on multi-agent patrolling, in *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symp. Artificial Intelligence*, Sao Luis, Maranhao, Brazil, September 29-October 1, 2004. Proc. 17. (Springer, 2004), pp. 474–483.

[3] N. Basilico, Recent trends in robotic patrolling, *Curr. Robot. Rep.* **3**(2) (2022) 65–76.

[4] S. L. Smith, M. Schwager and D. Rus, Persistent monitoring of changing environments using a robot with limited range sensing, in *2011 IEEE Int. Conf. Robotics and Automation* (IEEE, 2011), pp. 5448–5455.

[5] F. Zhang and N. E. Leonard, Cooperative filters and control for cooperative exploration, *IEEE Trans. Autom. Control* **55**(3) (2010) 650–663.

[6] X. Duan, M. George and F. Bullo, Markov chains with maximum return time entropy for robotic surveillance, *IEEE Trans. Autom. Control* **65** (1) (2019) 72–86.

[7] M. George, S. Jafarpour and F. Bullo, Markov chains with maximum entropy for robotic surveillance, *IEEE Trans. Autom. Control* **64**(4) (2018) 1566–1580.

[8] A. C. Kapoutsis, S. A. Chatzichristofis and E. B. Kosmatopoulos, DARP: Divide areas algorithm for optimal multi-robot coverage path planning, *J. Intell. Robot. Syst.* **86** (2017) 663–680.

[9] D. Zhu, C. Tian, B. Sun and C. Luo, Complete coverage path planning of autonomous underwater vehicle based on GBNN algorithm, *J. Intell. Robot. Syst.* **94** (2019) 237–249.

[10] B. Jin and X. Xu, Wholesale price forecasts of green grams using the neural network, *Asian J. Econ. Bank.* ahead-of-print (2024).

[11] B. Jin and X. Xu, Price forecasting through neural networks for crude oil, heating oil, and natural gas, *Measurement: Energy* **1** (2024) 100001.

[12] B. Jin and X. Xu, Carbon emission allowance price forecasting for China Guangdong carbon emission exchange via the neural network, *Glob. Finance Rev.* **6**(1) (2024) 3491–3491.

[13] B. Jin and X. Xu, Forecasting wholesale prices of yellow corn through the Gaussian process regression, *Neural Comput. Appl.* **36**(15) (2024) 8693–8710.

[14] B. Jin and X. Xu, Pre-owned housing price index forecasts using Gaussian process regressions, *J. Model. Manag.* **19**(6) (2024) 1927–1958.

[15] B. Jin and X. Xu, Machine learning predictions of regional steel price indices for East China, *Ironmak. Steelmak.* (2024) 03019233241254891.

[16] B. Jin and X. Xu, Contemporaneous causality among price indices of ten major steel products, *Ironmak. Steelmak.* **51**(6) (2024) 515–526, 03019233241249361.

[17] S. Yanes Luis, M. Perales Esteve, D. Gutiérrez Reina and S. Toral Marín, Deep reinforcement learning applied to multi-agent informative path planning in environmental missions, in *Mobile Robot: Motion Control and Path Planning* (Springer, 2023), pp. 31–61.

[18] Y. Sun, R. Zhang, W. Liang and C. Xu, Multi-agent cooperative search based on reinforcement learning, in *2020 3rd Int. Conf. Unmanned Systems (ICUS)* (IEEE, 2020), pp. 891–896.

[19] N. Vieillard, M. Andrychowicz, A. Raichuk, O. Pietquin and M. Geist, Implicitly regularized RL with implicit Q-values, arXiv:2108.07041.

[20] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou and M. Tegmark, KAN: Kolmogorov-Arnold networks, arXiv:2404.19756 (2024).

[21] M. Cheon, Kolmogorov-Arnold network for satellite image classification in remote sensing, arXiv:2406.00600 (2024).

[22] M. Kiamari, M. Kiamari and B. Krishnamachari, GKAN: Graph Kolmogorov-Arnold Networks, arXiv:2406.06470.

[23] D. Portugal and R. Rocha, A survey on multi-robot patrolling algorithms, in *Technological Innovation for Sustainability: Second IFIP WG 5.5/SOCOLNET Doctoral Conf. Computing, Electrical and Industrial Systems, DoCEIS 2011*, Costa de Caparica, Portugal, February 21–23, 2011. Proc. 2 (Springer, 2011), pp. 139–146.

[24] J. Scherer and B. Rinner, Multi-robot persistent surveillance with connectivity constraints, *IEEE Access* **8** (2020) 15093–15109.

[25] S. Y. Luis, D. G. Reina and S. L. T. Marín, A multiagent deep reinforcement learning approach for path planning in autonomous surface vehicles: The Ypacaraí lake patrolling case, *IEEE Access* **9** (2021) 17084–17099.

[26] K. S. Kappel, T. M. Cabreira, J. L. Marins, L. B. de Brisolara and P. R. Ferreira, Strategies for patrolling missions with multiple UAVs, *J. Intell. Robot. Syst.* **99** (2020) 499–515.

[27] L. Collins, P. Ghassemi, E. T. Esfahani, D. Doermann, K. Dantu and S. Chowdhury, Scalable coverage path planning of multi-robot teams for monitoring non-convex areas, in *2021 IEEE Int. Conf. Robotics and Automation (ICRA)* (IEEE, 2021), pp. 7393–7399.

[28] Y. Hong, Y. Kyung and S.-L. Kim, A multi-robot cooperative patrolling algorithm with sharing multiple cycles, in *2019 European Conf. Networks and Communications (EuCNC)* (IEEE, 2019), pp. 300–304.

[29] Y. Chevaleyre, Theoretical analysis of the multi-agent patrolling problem," in *Proc. IEEE/WIC/ACM Int. Conf. Intelligent Agent Technology, 2004.(IAT 2004)* (IEEE, 2004), pp. 302–308.

[30] H. Santana, G. Ramalho, V. Corruble and B. Ratitch, Multi-agent patrolling with reinforcement learning, in *Int. Joint Conf. Autonomous Agents and Multiagent Systems*, Vol. 4. (IEEE Computer Society, 2004), pp. 1122–1129.

[31] D. Portugal and R. P. Rocha, Cooperative multi-robot patrol with Bayesian learning, *Auton. Robots* **40**(5) (2016) 929–953.

[32] P. A. Sampaio and K. F. d. S. da Silva, Decentralized strategies based on node marks for multi-robot patrolling on weighted graphs, in *2019 Latin American Robotics Symp. (LARS), 2019 Brazilian Symp. Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)* (IEEE, 2019), pp. 317–322.

[33] F. Pasqualetti, J. W. Durham and F. Bullo, Cooperative patrolling via weighted tours: Performance analysis and distributed algorithms, *IEEE Trans. Robot.* **28**(5) (2012) 1181–1188.

[34] V. Sea, A. Sugiyama and T. Sugawara, Frequency-based multi-agent patrolling model and its area partitioning solution method for balanced workload, in *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 15th Int. Conf., CPAIOR 2018*, Delft, The Netherlands, June 26–29, 2018, Proc. 15 (Springer, 2018), pp. 530–545.

[35] Y. Elmaliach, N. Agmon and G. A. Kaminka, Multi-robot area patrol under frequency constraints, *Ann. Math. Artif. Intell.* **57** (2009) 293–320.

[36] T. Mao and L. Ray, Frequency-based patrolling with heterogeneous agents and limited communication, arXiv:1402.1757.

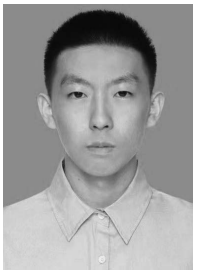
[37] G. Cannata and A. Sgorbissa, A minimalist algorithm for multirobot continuous coverage, *IEEE Trans. Robot.* **27**(2) (2011) 297–312.

[38] G. Cannata and Sgorbissa, A distributed, real-time approach to multi robot uniform frequency coverage, in *Distributed Autonomous*

Robotic Systems: The 10th Int. Symp. (Springer, Berlin, Heidelberg, 2013), pp. 19–32.
[39] V. Mersheeva and G. Friedrich, Multi-UAV monitoring with priorities and limited energy resources, in *Proc. Int. Conf. Automated Planning and Scheduling*, Vol. 25 (Jerusalem, Israel, 2015), pp. 347–355.
[40] T. Alam, M. M. Rahman, P. Carrillo, L. Bobadilla and B. Rapp, Stochastic multi-robot patrolling with limited visibility, *J. Intell. Robot. Syst.* **97** (2020) 411–429.
[41] A. B. Asghar and S. L. Smith, Stochastic patrolling in adversarial settings, in *2016 American Control Conf. (ACC)* (IEEE, 2016), pp. 6435–6440.
[42] L. Guo, H. Pan, X. Duan and J. He, Balancing efficiency and unpredictability in multi-robot patrolling: A marl-based approach, in *2023 IEEE Int. Conf. Robotics and Automation (ICRA)* (IEEE, 2023), pp. 3504–3509.
[43] T. Sak, J. Wainer and S. K. Goldenstein, Probabilistic multiagent patrolling, in *Advances in Artificial Intelligence-SBIA 2008: 19th Brazilian Symp. Artificial Intelligence Savador*, Brazil, October 26-30, 2008. Proc. 19 (Springer, 2008), pp. 124–133.
[44] D. Mitchell, M. Corah, N. Chakraborty, K. Sycara and N. Michael, Multi-robot long-term persistent coverage with fuel constrained robots, in *2015 IEEE Int. Conf. Robotics and Automation (ICRA)* (IEEE, 2015), pp. 1093–1099.
[45] J. J. Acevedo, B. Arrue, J. M. Diaz-Banez, I. Ventura, I. Maza and A. Ollero, Decentralized strategy to ensure information propagation in area monitoring missions with a team of UAVs under limited communications, in *2013 Int. Conf. Unmanned Aircraft Systems (ICUAS)* (IEEE, 2013), pp. 565–574.
[46] F. Pasqualetti, A. Franchi and F. Bullo, On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms, *IEEE Trans. Robot.* **28**(3) (2012) 592–606.
[47] H. Guo, Q. Kang, W.-Y. Yau, M. H. Ang and D. Rus, EM-patroller: Entropy maximized multi-robot patrolling with steady state distribution approximation, *IEEE Robot. Autom. Lett.* **8**(9) (2023) 5712–5719.
[48] N. Rezazadeh and S. S. Kia, A sub-modular receding horizon approach to persistent monitoring for a group of mobile agents over an urban area, *IFAC-PapersOnLine* **52**(20) (2019) 217–222.
[49] A. Machado, A. Almeida, G. Ramalho, J.-D. Zucker and A. Drogoul, Multi-agent movement coordination in patrolling, in *Proc. 3rd Int. Conf. Computer and Game* (Edmonton, Canada, 2002), pp. 155–170.
[50] N. Nigam, S. Bieniawski, I. Kroo and J. Vian, Control of multiple UAVs for persistent surveillance: Algorithm and flight test results, *IEEE Trans. Control Syst. Technol.* **20**(5) (2011) 1236–1251.
[51] M. Baglietto, G. Cannata, F. Capezio and A. Sgorbissa, Multi-robot uniform frequency coverage of significant locations in the environment, *Distributed Autonomous Robotic Systems 8* (Springer, 2009), pp. 3–14.

Jun Luo received his Bachelor's degree and Master's degree in Mechanical Engineering from the Henan Polytechnic University, China, and Ph.D. degree in Mechanical Engineering, the Shanghai Jiao Tong University, Shanghai, China. He is currently Professor at the Shanghai University. He mainly works on the design, sensing, and control techniques of various robots, with applications on UUVs, UGVs, and USVs.

Xi Chen received his B.S. degree from the East China Jiaotong University, China. He is currently pursuing an M.S. degree at the Shanghai University, China. His main search interests include cooperative control of multi-agent and machine vision.



Junye Wu is studying at the Shanghai University and currently pursuing a bachelor's degree. His main search interests include the construction and simulation of Autonomous Underwater Vehicle by ROS2.

Wenbo Hui received his B.S. degree from the Shanghai University, China. He is currently pursuing M.S. degree at the Shanghai University, China. His main search interests include multi-robot coverage path planning and cooperative control of multi-agent.

Bowen Yang is studying at the Shanghai University and currently pursuing an undergraduate degree. His main research interest is Autonomous Underwater Vehicle simulation and control.

Yangmin Xie received the B.S. and M.S. degrees from the School of Mechanical Engineering, Beihang University, Beijing, China, and the Ph.D. degree in Mechanical Engineering from the University of Illinois at Urbana-Champaign, Illinois, U.S. She is currently Full Professor at the School of Mechatronics Engineering and Automation, the Shanghai University, Shanghai, China. Her current research interests include intelligent sensing and control of mobile robotics. She mainly focuses on environmental modeling, machine vision, and auto-navigation techniques for unmanned systems. Her recent work involves applications on planetary rovers, autonomous surveillance, and undersea mining.