An online learning framework for UAV search mission in adversarial environments

Khial, Noor; Mhaisen, Naram; Mabrok, Mohamed; Mohamed, Amr

DOI
10.1016/j.eswa.2024.126136

Publication date
2025

Document Version
Final published version

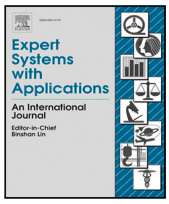
Published in
Expert Systems with Applications

Expert Systems With Applications 267 (2025) 126136

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



An online learning framework for UAV search mission in adversarial environments

Noor Khial [a,*], Naram Mhaisen [b], Mohamed Mabrok [c], Amr Mohamed [a]

[a] College of Engineering, Qatar University, Qatar
[b] College of Electrical Engineering, Mathematics, and Computer Science, TU Delft, The Netherlands
[c] College of Arts and Sciences, Qatar University, Qatar

ARTICLE INFO

Keywords:
UAV
Search mission
Online learning
Multi-armed bandits
Experts
Human-in-the-loop

ABSTRACT

The rapid evolution of Unmanned Aerial Vehicles (UAVs) has revolutionized target search operations in various fields, including military applications, search and rescue missions, and post-disaster management. This paper presents the application of a multi-armed bandit algorithm for UAV search mission. The UAV's mission is to locate a mobile target formation, operating under the assumption of an unknown and potentially non-stationary probability distribution, by learning the formation's strategy over time. To achieve this, we formulate an optimization problem and leverage the Exp3 algorithm (exponential-weighted exploration and exploitation) for its solution. To enhance the learning process, we integrate environment observations as context, resulting in a variant referred to as C-Exp3. However, C-Exp3 is not designed for scenarios where the target formation strategy changes over time. Therefore, AC-Exp3 is proposed as an adaptive solution, featuring a human-centric drift detection mechanism to detect the changes in the formation strategy and adjust the learning process accordingly. Furthermore, the Exp4 algorithm is proposed as a self-adjustment meta-learner to address changes in the formation's strategy. We evaluate the performance of C-Exp3, AC-Exp3, and Exp4 through a series of experiments with a focus on non-stationary environments. Our primary objective is reaching the unknown optimal-in-hindsight policy as the time t approaches the horizon T, thereby reflecting the UAV's capacity to learn formation's strategy. AC-Exp3 demonstrates enhanced adaptability compared to C-Exp3. Meanwhile, Exp4 emerges as a robust performer, swiftly adapting to new strategies.

1. Introduction

1.1. Motivation and background

Unmanned aerial vehicles (UAVs) have become essential tools across various civilian sectors and critical domains, including military operations, post-disaster wireless service restoration, and search and rescue(SAR) operations (Mozaffari, Saad, Bennis, Nam, & Debbah, 2019; Xiaoning, 2020). Equipped with off-the-shelf sensors and advanced imaging technologies, UAVs provide real-time and precise information regarding the locations and conditions of both individuals and infrastructure (Gu, Su, Wang, Du, & Guizani, 2018).

SAR missions, as well as surveillance tasks, frequently entail the challenge of locating targets over expansive areas. In military operations, this can involve finding enemy positions or tracking potential threats. During these operations, sensors and imaging technologies are systematically employed to explore unknown areas and allocate targets. UAVs strategically determine flight paths to observe and locate these targets. The primary objective is to identify a formation of targets while learning from their strategies. Consequently, the UAV dynamically fine-tunes its flight trajectory based on real-time observations, optimizing coverage, and adapting adapting to targets' evolving strategies. This necessitates the UAV's ability to adapt to such changes.

In recent studies such as (Soliman et al., 2023a; Wei, Huang, Lu, & Song, 2019; Yue, Guan, & Wang, 2019; Yue, Guan, & Xi, 2019), reinforcement learning (RL) techniques have been employed to optimize UAV search processes. However, these models are typically trained in ideal virtual environments, which do not account for the pervasive noise found in real-world applications, especially in safety-critical fields like UAVs and robotics (Goodfellow, Shlens, & Szegedy, 2014). Agents trained under such conditions can adopt sub-optimal policies when exposed to adversarial noise (Kos & Song, 2017), even from minor perturbations. To train more robust models to noise, (Wan, Gao, Hu, &

* Corresponding author.
E-mail addresses: nk1703044@qu.edu.qa (N. Khial), N.Mhaisen@tudelft.nl (N. Mhaisen), m.a.mabrok@qu.edu.qa (M. Mabrok), amrm@qu.edu.qa (A. Mohamed).

https://doi.org/10.1016/j.eswa.2024.126136
Received 1 February 2024; Received in revised form 30 November 2024; Accepted 9 December 2024
Available online 19 December 2024

Wu, 2020) introduced an adversarial attack technique with the DDPG algorithm that models the environmental noise as Gaussian noise added to the current state.

Our approach assumes that the noise is not governed by a specific distribution. The algorithm handles worst-case scenarios, where the adversary's distribution changes at each time step. We focus on locating dynamic targets in uncertain environments without prior knowledge of target formation distributions, addressing challenges from adversarial environments where targets exhibit non-stationary mobility patterns and may intentionally mislead the UAV.

To solve this problem, we apply online learning algorithms for the UAV search mission, which provide performance guarantees without incorporating stochastic assumptions about target probability distributions. While our approach employs a single agent, it can be expanded into a multi-agent online learning framework. The concept of multi-agent multi-armed bandit learning has been explored in offloading scheduling in edge computing (Wu, Chen, Ni, & Wang, 2021), where users collaboratively provide feedback based on their observations.

1.2. Methodology and contributions

We leverage the adversarial multi-armed bandits (MAB) technique to address challenges in navigating adversarial environments. Our approach utilizes the Exp3 algorithm for exponential-weighted exploration and exploitation, as well as the Exp4 algorithm, which incorporates experts for exploration and exploitation. The algorithms, Exp3 and Exp4, were originally introduced in Auer, Cesa-Bianchi, Freund, and Schapire (1995) and have since been the subject of many enhancements and modifications, as detailed in the bibliographic remarks in ((Lattimore & Szepesvári, 2020), Ch. 11 and 18). The bandit problem represents a simple RL formulation, where an agent interacts with the environment by performing actions at discrete time steps (Baccour et al., 2022). It is modeled as a sequential game between the learner (agent) and the environment over a time horizon T. In each round t, the learner selects an action k from a predefined set $\mathcal{K}$ and receives feedback in the form of a penalty associated with the chosen action. The learner aims to improve performance by learning from this feedback to reach the optimal policy (target formation strategy) through ongoing interaction, assuming the targets exhibit adversarial behavior.

In this paper, we propose a new searching techniques that is based on Exp3 and Exp4 algorithms, to deal with the non-stationary behavior in the unknown mobility of the targets. Therefore, the main contributions of this paper are as follows:

1. Formulate the search mission problem as a game theoretic approach using the MAB technique, with the objective of learning the unknown policy of the targets. Leverage the contextual Exp3 algorithm to address the challenge of a UAV's search mission in an unknown and adversarial environment, which ultimately converges to the unknown optimal policy.
2. Propose an adaptive solution based on the contextual Exp3 algorithm to address periodic changes in the unknown policy by incorporating a drift detection mechanism to identify changes in the distribution of the targets.
3. Propose a meta-learner based of Exp4 algorithm to deal with the changes in the optimal policy without the incorporation of drift detector. The goal of the meta-learner is to attain the optimal policy by selecting the best expert from a set of distinct experts, each representing a unique approach to action selection.
4. Evaluate and compare the performance of Exp3, adaptive Exp3, and Exp4, within the context of a UAV search mission. We focus on showing the rate of convergence for the MAB-based algorithms in non-stationary environment, and compare their performance with RL-based model.

The paper is structured as follows: Section 2 presents a comprehensive review of related literature. In Section 3, we establish the system model and articulate the problem formulation as an optimization problem. Section 4 propose the online learning framework under the assumption of a fixed formation strategy. Section 5 further discusses our solutions, modeling the problem within an online learning framework to address changes in the target formation strategy. Finally, Section 6 evaluates the algorithms in reaching the optimal policy across various scenarios.

2. Related work

In this section, we first review the broader context of UAV SAR missions, followed by the technical approaches used to address these challenges. We organize our discussion around two main themes: (1) UAV SAR missions and their associated tasks, and (2) methodological approaches for solving these problems.

2.1. UAV search and rescue missions

UAVs have been utilized in SAR missions across a variety of applications, including disaster response efforts (Alotaibi, Alqefari, & Koubaa, 2019; Silvagni, Tonoli, Zenerino, & Chiaberge, 2017; Zhang, Li, Wang, Li, & Li, 2022), human–UAV collaborations for finding escaped criminals (Zheng, Du, Ling, Sheng, & Chen, 2019), searches for missing tourists in nature reserves (Zheng, Du, Sheng & Ling, 2019), and sea rescue operations (Wang et al., 2018). These applications tackled different mission objectives, which we discuss and compare with our approach.

Mission Objectives. For example, (Wang et al., 2018) enhanced UAV-based SAR operations at sea using deep learning techniques but did not account for uncertainties in target locations caused by ocean drifts. Other research has explored multi-objective approaches to address challenges like minimizing mission duration (Alotaibi et al., 2019; Soliman et al., 2023b; Zhang, Zhao, Liu, & Li, 2023; Zheng et al., 2019; Zheng, Du, Sheng & Ling, 2019), improving target identification accuracy (Alotaibi et al., 2019; Hong, Wang, Du, Chen, & Zheng, 2021; Zhang et al., 2023; Zheng, Du, Sheng & Ling, 2019), optimizing UAV communication (Alotaibi et al., 2019; Zheng et al., 2019), enhancing task allocation (Alotaibi et al., 2019; Hong et al., 2021; Zheng, Du, Sheng & Ling, 2019), and managing environmental dynamics and uncertainties (Alotaibi et al., 2019; Soliman et al., 2023b; Zhang et al., 2023; Zheng, Du, Sheng & Ling, 2019). In this work, we specifically focus on addressing the challenges of managing environment uncertainties.

The Layered SAR (LSAR) framework proposed by Alotaibi et al. (2019) aimed to address these objectives while handling uncertainty. However, its implementation is hindered by challenges such as computationally expensive real-time processing, integration complexities with various UAV systems, and the need for robust statistical modeling. Similarly, the Distributed Autonomous Collaborative Mission Planning (DACMP) approach by Zhang et al. (2023) tackled environmental uncertainty with objectives like mission duration minimization and maximizing target identification probability. However, DACMP faces scalability issues as the state space grows exponentially with more UAVs, targets, and mission constraints. Meanwhile, (Zheng, Du, Sheng & Ling, 2019) introduced a biogeography-inspired evolutionary algorithm to tackle the high computational complexity of collaborative human–UAV SAR missions. This approach effectively solves multi-objective problems but suffers from long preparation times, limiting its real-time adaptability in dynamic environment.

Despite these advancements, existing methods do not adequately address the non-stationary behavior of the mobile targets. Our proposed algorithm focuses on this gap by providing a computationally efficient solution specifically designed to handle non-stationarity in SAR missions.

2.2. Methodological approaches

To address the aforementioned challenges in UAV SAR missions, researchers have developed various technical approaches, each offering distinct advantages for specific aspects of the mission. We review the main categories of these approaches.

Optimization Approaches. Path planning, an important task in search operations, poses an NP-hard problem that significantly impacts mission effectiveness (Lin & Goodrich, 2014a). Recent optimization techniques have focused on maximizing target detection probability while considering real-world constraints (Gan & Sukkarieh, 2011; Li, Xu, Nie, Mao, & Yu, 2021; Pérez-Carabaza, Scherer, Rinner, López-Orozco, & Besada-Portas, 2019). Notable advances include the heuristic crossing SAR optimization algorithm (HC-SAR) (Zhang, Zhou, Qin & Tang, 2023), which improves convergence speed while maintaining target formation diversity. Some approaches have incorporated partial observation using heuristics to enhance performance (Lin & Goodrich, 2014b). However, these methods often face limitations in dynamic environments where target formations exhibit real-time movement changes. Soliman et al. (2023b) have specifically addressed the path planning problem considering target mobility uncertainty through stochastic distribution modeling.

Reinforcement Learning Approaches. RL has been applied for UAV search missions by formulating them as Markov Decision Processes (MDPs) (Xu et al., 2022). Multiple works have applied different RL models for target search missions involving single or multiple UAVs. The majority of these studies (Blais & Akhloufi, 2024; Hu & Li, 2021; Shurrab, Mizouni, Singh, & Otrok, 2023; Venturini et al., 2021; Zhang, Zheng, & Lambotharan, 2020) applied the Deep Q-Network (DQN) model to optimize the search process. Another work (Wu et al., 2023) have utilized the Q-Table technique in for SAR operation, while assuming an unknown environments. Additionally, (Qi, Yang, & Xia, 2024; Qi, Zhao, Li, & Jia, 2024; Soliman et al., 2023b) employed the Proximal Policy Optimization (PPO) algorithm to enhance the strategic approach of agents for visiting multiple targets in vast areas. The existing RL works for UAV search missions have not adequately considered the potential impact of non-stationary behavior on target mobility in SAR environments, which can significantly influence the learning process. In this paper, we develop a more robust search algorithm that addresses the adversarial nature of targets and compare its performance against DQN, Q-Table, and PPO algorithms.

Multi-Armed Bandit Approaches. MAB frameworks have emerged as a promising approach for addressing uncertainty and optimization in search missions, gaining prominence across various domains (Bouneffouf, Rish, & Aggarwal, 2020). Recent applications in search mission optimization (Zheng et al., 2022) have demonstrated their potential for balancing exploration and exploitation in dynamic environments. Studies have shown MAB algorithms' effectiveness in improving system performance during search missions (Balafrej, Bessiere, & Paparrizou, 2015; Cherif, Habet, & Terrioux, 2020), particularly in scenarios requiring adaptive decision-making. However, the specific application of MAB algorithms for UAV-based detection of target formations remains relatively unexplored, especially in contexts involving non-stationary target behavior. This gap presents an opportunity for novel contributions in developing more efficient and adaptive search strategies, which we address in this work.

3. System model and problem formulation

In this section, we present the system model for a mobile UAV(agent) with a search mission in an unknown and adversarial environment. The unknown parameter in the environment is the mobility of the targets. Specifically, the mobility or, equivalently, the observation of the targets is assumed to be unknown a priori.

We focus on highlighting the key components of the environment that facilitates solving the problem using bandit algorithms. Next, we formulate our problem as an optimization problem. The objective is to maximize the overall performance of the agent by minimizing the cumulative *regret* of the agent. The notion of regret measures the performance gap between the agent's performance and the optimal-in-hindsight unknown policy. Table 1 shows the key notations.

Table 1
Key notations

Notation	Description
$P_{k_t}^t$	Sampling distribution at round t for selecting action k_t
$\mathbf{L}_t^{k_t}$	Loss associated with the observed action k_t at round t
$\hat{S}_{t,k}$	Estimated weights for each action at time slot t
η_{opt}	Optimal learning rate
T	Time horizon
$\mathcal{N}$	Set of possible cells for the UAV's current location
$\mathcal{C}$	Set of contexts
$\mathcal{K}$	Action set for context c
$\mathbf{L}_{c_t}^{k_t}$	Penalty received by the learner for action k_t^c in context c at time slot t
$\mathbf{E}_t(m)$	The loss vector of the of m expert with respect to action k at time t.
$\mathcal{R}_T$	Cumulative regret
k^*	Best action in hindsight
$\mathcal{P}$	Set of RPG-based mobility pattern
$\mathcal{M}$	Set of experts

Fig. 1. The system model describes a possible scenario for the environment during one time slot t. The targets are moving as a formation. The agent takes discrete actions $k \in \mathcal{K}$ to move from one cell to another.

3.1. Architecture

We define the coverage area and the movement of the agent within this area. The coverage area is represented as a grid consisting of $|\mathcal{N}|$ cells, each labeled with $n \in \{1, 2, \ldots, |\mathcal{N}|\}$. These cells correspond to potential actions k within the action space $\mathcal{K}$, as shown in Fig. 1. Furthermore, each cell represents a unique location that can either be empty or contain one or more targets. This grid structure serves as a framework for the agent to systematically explore the area.

Agent's Movement. The agent's movement is defined by discrete steps, ensuring that the UAV's trajectory across the predefined grid-based coverage area consists of discrete actions. At any time slot t, the agent selects an action k from its available action space $\mathcal{K}$, determining its transition to a neighboring cell within the grid or choosing to remain in the current cell. The time slot t represents the duration required for the agent to move between cells, depending on the cell size and the UAV's speed. The non-stationary nature reflecting the movements

of the targets occurs between time slots, but the targets may require multiple time slots or just one, depending on their speed. All algorithms discussed in this paper are agnostic to the actual duration of the abstract concept of the *time slot* and remain applicable to different configurations with the same guarantees.

Observation and Decision. The agent's information is constrained to its current cell n. It has the capability to observe the presence or absence of targets within the cell it currently occupies, leading to a loss $\mathbf{L}_t$ associated with the observation of targets at time slot t. Based on the accumulated observations or losses, the agent then decides the next movement (action).

Exploration Strategy. The primary objective of the agent is to search for targets within a fixed coverage area. By systematically moving from cell to cell, the agent observes each cell to gather information about target presence, enabling it to learn the strategy of the target formation across the grid, assuming their movement resembles adversarial behavior. The targets can observe the agent's actions and gain insight into its policy, potentially leading them to attempt to deceive the UAV in an adversarial environment.

Interactive Modeling and Observed Loss $\mathbf{L}_t$. Our interactive model for the search mission comprises two components: (1) The target behavior model, representing the non-stationary movement of targets, and (2) The agent, utilizing the MAB algorithm, responsible for the search mission. In this model, the targets and the agent dynamically interact at each time step t. If the agent fails to detect targets after taking an action k_t, a loss of $\mathbf{L}_t^{k_t}$ is assigned as 1; otherwise, the loss is set to 0. The optimal action k^* is estimated based on the full knowledge of future losses, with the benchmark being the cumulative loss incurred by the agent, assuming it knows the presence of targets across all $t \in \mathcal{T}$. However, the optimal action remains unknown to the agent since it cannot observe the time-varying cost associated with action k_t without first sampling k_t and then observing the cost at time slot t. The loss $L_t^{k_t}$ can be expressed as:

$$\mathbf{L}_t^{k_t} = \begin{cases} 0 & \text{if a target is observed with } k_t \\ 1 & \text{if no target is observed with } k_t \end{cases} \tag{1}$$

3.2. Problem statement

Our main objective is to maximize the agent's performance by formulating the problem to minimize the cumulative losses, thus, minimizing the cumulative regret $\mathcal{R}_\mathcal{T}$, with the ultimate goal of identifying the optimal action k^*. The optimal policy k^* describes the underlying strategy of the target formation. However, the policy k^* is unknown to the agent. Therefore, our aim is to find a policy k^* that minimizes the cumulative regret $\mathcal{R}_\mathcal{T}$. This can be formulated as the following optimization problem:

$$\mathbf{P}: \quad \min_k \sum_{t=1}^{\mathcal{T}} \mathbf{L}_t^k \tag{2}$$

According to $\mathbf{P}$, the agent determines the action k to be sampled at each time slot t in the time horizon $\mathcal{T}$. The main challenge arises at time slot t when the agent needs to decide the next movement. At this point, the loss $\mathbf{L}_{t+1}^k$ associated with the next movement is inaccessible, which make it hard to solve it as an optimization problem. However, the loss will indeed be revealed in the next time slot once the agent makes a decision to take an action for the next destination cell n_{t+1} to be visited and subsequently makes an observation of the existing targets in cell n_{t+1}. Then, the $\mathbf{L}_{t+1}^k$ is revealed to the agent. Therefore, we apply the MAB algorithms (Lattimore & Szepesvári, 2020), which learns from the continuous interaction with the environment and reaches the optimal policy k^* as $t \to \mathcal{T}$.

3.3. Optimal policy and cumulative regret.

In this subsection, we present the concept of **regret** in the context of online convex optimization. Regret serves as a metric for assessing the UAV's decision-making performance during the mission, measuring how effectively the agent's decisions align with optimal action in hindsight. In online convex optimization, a player, in our case the UAV, repeatedly makes decisions without knowledge of future outcomes, incurring costs based on its chosen actions, all with the ultimate objective of achieving the optimal policy k^*. This concept holds particular relevance in our UAV search mission scenario, where the UAV strives to optimize its actions while adapting to the non-stationary behavior of the target formation.

Regret. To accomplish this, we begin by establishing the components of the system. We adopt the standard online setup, where at each time slot $t \in \{1, 2, \ldots, \mathcal{T}\}$, the agent selects an action k_t from a set $\mathcal{K}$ at time slot t. This set is characterized as closed and bounded. The outcome of action k_t is reflected in the loss $\mathbf{L}_t^{k_t}$ associated with action k at time t. The regret, with respect to the best fixed action k^*, is defined as the sequence of actions k_t every time slot t in terms of their cumulative losses:

$$\mathcal{R}_\mathcal{T} = \sum_{t=1}^{\mathcal{T}} (\mathbf{L}_t^{k_t} - \mathbf{L}_t^{k^*}) \tag{3}$$

Optimal Policy. Here, $k^* \in \arg\min_{k \in \mathcal{K}} \sum_{t=1}^{\mathcal{T}} \mathbf{L}_t^k$ represents the action k^* that minimizes the accumulated loss between all actions in the action space $\mathcal{K}$. In essence, k^* serves as a benchmark for comparison, representing the optimal action based on perfect knowledge of the outcomes. Over time, we expect to observe a saturation in regret $R_\mathcal{T}$ as t approaches $\mathcal{T}$, or in mathematical terms, $\lim_{t \to \infty} \frac{\mathcal{R}_\mathcal{T}}{\mathcal{T}} = 0$. Specifically, we aim to achieve $\mathcal{R}_\mathcal{T} = O(\sqrt{\mathcal{T}})$, demonstrating that the agent achieves the optimal policy as t approaches the time horizon $\mathcal{T}$.

4. Contextual Exp3

In this section, we leverage Contextual Exp3 (C-Exp3) for the search mission with a fixed optimal policy. Furthermore, we present a new approach called Adaptive Contextual Exp3 (AC-Exp3), which detect the change in the optimal policy by incorporating a drift detection mechanism. The change in the optimal policy describes a change in the underlying distribution of the target formation.

4.1. Exp3 algorithm

The Exp3 (Exponential-weights for Exploration and Exploitation) algorithm is a well-known approach to tackle the multi-armed bandit problem, where the goal is to make a series of decisions (selecting actions) in adversarial environment while balancing the exploration/exploitation trade-off.

Contextual Bandits. In many bandit problems, the agent has access to additional information (context) that could aid in predicting the quality of actions (Lattimore & Szepesvári, 2020). In our scenario, we incorporate context into the Exp3 algorithm (C-Exp3). Here, we utilize the UAV's current location, which can be any cell $n \in \mathcal{N}$, as a context c where $c \in \mathcal{C}$ with $\mathcal{C}$ representing the set of available contexts. The action space for each context c is denoted as $\mathcal{K}$ and remains consistent across all contexts. It is defined as $\mathcal{K}$ = Current, North (N), South (S), East (E), West (W), Northeast (NE), Northwest (NW), Southeast (SE), Southwest (SW). This action space offers the agent a choice among eight possible directions for movement within a 2D grid at each time slot, corresponding to the agent's current position and its neighboring cells. Additionally, this definition of the action space imposes a constraint on the agent's movement, allowing only one step in any direction. Consequently, the agent moves to one of the adjacent cells during each time slot, promoting smaller and more localized actions.

Algorithm 1 Contextual Exp3 (C-Exp3)

1: **Input:** Time horizon $\mathcal{T}$, Action space $\mathcal{K}$, Learning rate η, Context set C
2: Initialize $\hat{S}_{0,k} = 0$ for all $k \in \mathcal{K}$
3: Initialize weights for actions
4: **for** $t = 1$ to $\mathcal{T}$ **do**
5: Observe context $c_t \in C$
6: Compute the sampling distribution $P_{t,k}$ for all $k \in \mathcal{K}$:
7: $P_{t,k} = \frac{\exp(\eta \hat{S}_{t-1,k})}{\sum_{j=1}^{|\mathcal{K}|} \exp(\eta \hat{S}_{t-1,j})}$
8: Sample action k_t according to $P_{t,k}$ and observe the loss $\mathbf{L}^{k_t}$
9: **for** each $k \in \mathcal{K}$ **do**
10: Update the estimate: $\hat{S}_{t,k} = \hat{S}_{t-1,k} + 1 - \frac{\mathbb{I}\{k=k_t\} \cdot \mathbf{L}_{c_t}^{k_t}}{P_{t,k}}$
11: **end for**
12: **end for**

Collectively, the action space for all contexts C defines the entire grid. In each context, each cell n represents a potential direction relative to the current context c_t. For instance, cell $n = 9$ can be associated with context $c = 5$ when moving in the S direction or context $c = 10$ when moving in the W direction (see Fig. 3).

Algorithm. The algorithmic steps are shown in Algorithm 1, and detailed next. For simplicity, assume that we have a single context and thus we drop c_t in the next paragraph. The discussion still holds in the general case since the steps (and the discussed quantities) are simply replicated for each context $c_t \in \mathcal{X}$. We begin by initializing the actions weights uniformly, $P_t = (P_t^k, \forall k \in \mathcal{K}), P_t^k = (1/|\mathcal{K}|)$. which defines the probability of the possible destination cells for the agent within the grid. Then, at each time step t, this sampling distribution is updated via the rule in line 6. Intuitively, the updated weights are made proportional to the accumulated (estimated) reward for their corresponding action $\hat{S}_{t-1,k}$, then normalized by dividing on the sum to ensure that P_t is still a distribution. The parameter η controls how strong our update is set as minimiax optimal of $\eta = \sqrt{\log K / TK}$ (Lattimore & Szepesvári, 2020, Thm 11.1). After updating the sampling distribution, we select and action $k \sim P_t$ (line 7), observe a reward, and update our estimate for k's rewards (line 9). Note that the fraction we accumulate in line 9 is an unbiased estimator for the loss ($\mathbb{E}[1 - (\mathbb{I}\{k=k_t\}\mathbf{L}_{c_t}^{k_t})/P_{t,k}] = \mathbf{L}_{c_t}^{k_t}$), and thus we will indeed move in the direction of actions with low losses at the following weights update P_{t+1} (see Fig. 2).

Regret. To evaluate the agent's performance, we use a regret measure for each c, which quantifies the agent's total loss compared to the optimal context-dependent policy, k_c^*, in hindsight in reference to Eq. (18.1) in Lattimore and Szepesvári (2020):

$$\mathcal{R}_{\mathcal{T},c} = \sum_{t=1}^{\mathcal{T}} \left[(\mathbf{L}_{c_t}^{k_t} - \mathbf{L}_{c_t}^{k_c^*}) \right] \mathbb{I}\{c = c_t\} \tag{4}$$

Hence, the regret $\mathcal{R}_{\mathcal{T}}$ for agent is:

$$\mathcal{R}_{\mathcal{T}} = \sum_{c \in C} \mathcal{R}_{\mathcal{T},c} \tag{5}$$

Here, $k_c^* \in \arg\min_{k \in \mathcal{K}} \sum_{t=1}^{\mathcal{T}} \mathbf{L}_{c_t}^{k} \mathbb{I}\{c = c_t\}$. We measure the difference in loss between the agent's decision k_t and the optimal context-dependent best action k_c^* for a given context c at each time slot t. By summing these differences across the time horizon $\mathcal{T}$ and potential contexts from the set C, we derive the total regret $\mathcal{R}_{\mathcal{T}}$. The identity function $\mathbb{I}\{c = c_t\}$ acts as a filter, equaling 1 when the condition $c = c_t$ is true, indicating that the regret calculation applies only when c matches c_t. If the condition is false, the identity function equals 0, effectively excluding those instances from the sum.

Furthermore, we define a possible cell n as optimal if it reflects a k_c^* for multiple contexts that align with the strategy of the formation. Our goal is to learn the mapping between contexts and optimal actions.

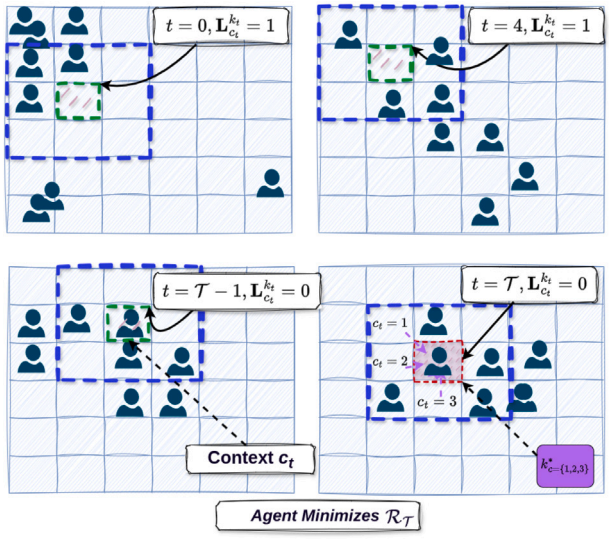


Fig. 2. Performance of C-Exp3 with respect to c at time t assuming non-stationary movements of target formation. The Agent aims to reach k_c^* (minimize $\mathcal{R}_{\mathcal{T}}$) as $t \to \mathcal{T}$. The k_c^* represents the different movements with respect to context $c = \{1, 2, 3\}$. For instance, $k_{c=1}^*$ is a movement to E with respect to $c = 2$.

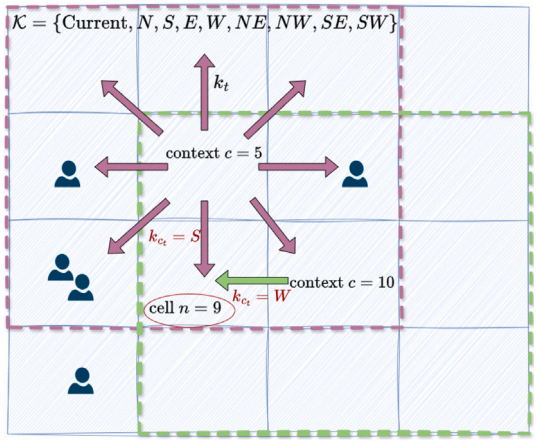


Fig. 3. Example of a context dependent action selection.

The Exp3 algorithm provides an upper bound for worst-case scenarios, accounting for the observed context distribution. This bound is valid in adversarial environments for the entire system, as detailed in Eq. (18.3) in Lattimore and Szepesvári (2020):

$$\mathcal{R}_{\mathcal{T}} \leq \sum_{c \in C} \mathcal{R}_{\mathcal{T},c} \leq 2 \sum_{c \in C} \sqrt{\log(|\mathcal{K}|) \sum_{t=1}^{\mathcal{T}} |\mathcal{K}| \mathbb{I}\{c = c_t\}} \tag{6}$$

This concept has a practical implications that can be explored further. In the current approach, the agent's movement is discretized due to the discretization of the action space. However, if we increase the number of cells in the grid, we can reduce the level of discretization and allow for more precise movement for the agent. Nevertheless, increasing the number of cells poses challenges. The location n serves as a context, leading to a large context space, which in turn, introduces additional overhead for the agent. However, these challenges can be addressed by employing alternative versions of contextual bandits, such as Exp4. In this paper, our focus is on a small context space size, but the potential for broader exploration exists.

State–Context Correlation. States in RL, for example, encompass complete information about the environment and involve temporal

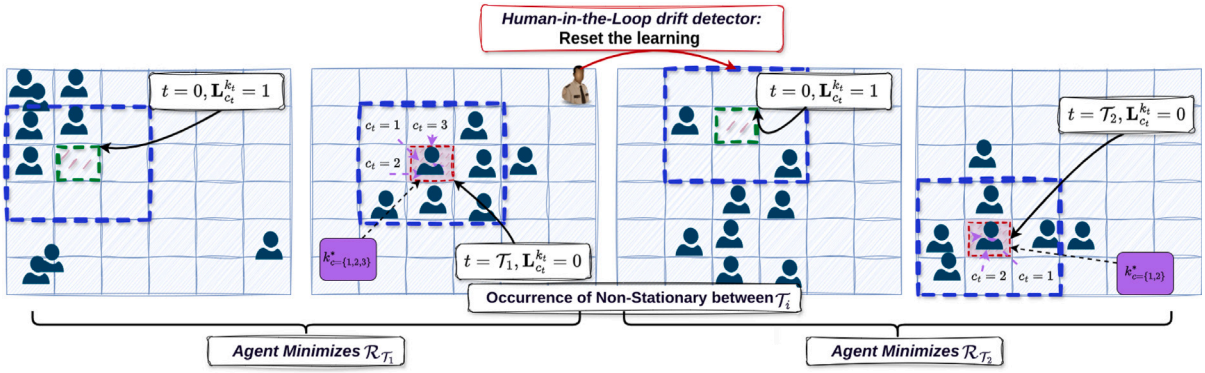


Fig. 4. Performance of AC-Exp3 with respect to c_t assuming non-stationary movements of target formation. The agent resets the algorithm according to the drift detector. At each time interval $\mathcal{T}_i$, the agent learns the optimal action k_c^* which minimizes the $\mathcal{R}_{\mathcal{T}_i}$.

dependencies, while contexts in contextual bandits capture relevant information for action selection in an episodic setting. Both concepts play a role in decision-making and action selection, although in different settings and with different considerations.

To summarize this, in each time slot t, the agent observes c_t, chooses an action $k_t \in \mathcal{K}$, and receives a penalty $\mathbf{L}_{c_t}^{k_t}$. In this section, we specifically focus on the **One Bandit per Context** approach, wherein we define a separate instance of the Exp3 algorithm for each context c. Algorithm 1 shows the steps of contextual Exp3. Our goal is to learn the mapping between contexts and optimal actions.

4.2. Adaptive contextual Exp3

The limitation of Exp3 stems from its focus on learning a fixed optimal action for the entire duration, which proves too restrictive for our problem. It fails to account for the possibility of target formation changing their strategies, leading to changes in the optimal action k^*. Exp3 aims to minimize regret against the best action by using an unbiased estimation to compute choice probabilities for each action assuming non-stationary movement every time slot (worst case scenario). Non-stationary also occurs between intervals of time (*e.g.*, periodically every $\mathcal{T} \in T$). If an action performs well over $\mathcal{T}_1$ (Exp3 converges to k^*) but performs poorly in $\mathcal{T}_2$, the Exp3 algorithm may need a significant number of trials, equal to the first interval's length, to switch to a different k^*.

We introduce the adaptive contextual Exp3 algorithm (AC-Exp3), designed to learn multiple optimal actions k^*, each within a time interval $\mathcal{T}_i$. The change in k^* is influenced by variations in the strategy of the formation. At each time interval $T = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_i, \ldots\}$, where $t \in \mathcal{T}$, the best action k^* is fixed for each interval $\mathcal{T}_i$ but changes from one interval to another, following a different strategy, denoted by $p \in \mathcal{P}$. Here, $\mathcal{P}$ represents a set of different mobility patterns. We deal with each interval as a new problem with a new C-Exp3 instance. Within each time interval $\mathcal{T}$, the regret definition aligns with the contextual Exp3 regret $\mathcal{R}_{\mathcal{T}}$ defined in Section 4.1, Eq. (5), along with an upper bound guarantee $\mathcal{R}_{\mathcal{T}}$ as defined in Eq. (6). Consequently, we define the cumulative regret of the adaptive contextual Exp3 as in Eq. (5) for each interval of time. Furthermore, the context-dependent best action k_c^* is fixed in each interval of time $\mathcal{T}$. On Each interval $\mathcal{T}$, We apply the static regret definition and each of the time interval $\mathcal{T}$ maintains its own upper bound guarantee.

The challenge here for the agent is to identify the changing in the mobility pattern. Once a change in the mobility pattern is detected, the agent can reset the C-Exp3 instance where it can be seen as a new instance of the algorithm. To solve this challenge, we leverage the concept of a drift detection to identify the change in the mobility pattern, then reset the algorithm. There are different techniques that can be used to design the drift detection, for example (Almarzoqi, Yahya, Matar, & Gomaa, 2022; Kanoun & van der Schaar, 2015). Here, we introduce the concept of human-in-the-loop as a drift detection mechanism.

Human-centric drift identifier. The human-in-the-loop (HITL) approach enhances UAV adaptability in complex environments by integrating human knowledge with machine learning (Mosqueira-Rey, Hernández-Pereira, Alonso-Ríos, Bobes-Bascarán, & Fernández-Leal, 2023). Human operators provide feedback to the learning system during operation, which includes insights and strategies that algorithms might miss, such as sudden environmental changes. When such changes occur, operators can issue an alert signal to the UAV system. This signal triggers a reset of the learning process to prevent the UAV from struggling to find successful paths on its own. For example, an alert may be triggered in response to unexpected obstacles or other real-time factors that the UAV's onboard systems might not fully account for. An example of introducing HITL feedback with RL is provided by Zhang et al. (2023), where the algorithm learns from both human demonstrations and self-exploration to accelerate the training process.

Algorithm. The algorithmic steps are shown in Algorithm 2, and detailed next. The HITL-based drift detector operates by leveraging HITL feedback to adjust a contextual bandit system in real-time. Initially, the algorithm takes HITL feedback as input. For each time interval $\mathcal{T}$, the agent uses the C-Exp3 algorithm (as detailed in Algorithm 1). If an alert indicating drift is detected, the C-Exp3 algorithm is reset to adapt to the new conditions. This approach ensures that the contextual bandit system remains responsive to changes in dynamic environments. Refer to Fig. 4 for a visual representation for the agent during the learning process with AC-Exp3.

Algorithm 2 HITL-based Drift Detector Algorithm

```
Input: HITL feedback
for 𝒯 ∈ T where i = {1, 2, ..} do
    Run Contextual Exp3 for t ∈ 𝒯
    if alert is received then
        Reset Contextual Exp3
    end if
end for
```

5. Meta learner in non-stationary environments

Regret minimization algorithms have demonstrated their effectiveness in adapting to unknown distribution environments and converging towards optimal policy. Nevertheless, they face challenges in environments where the best action k^* changes due to periodic changes in distribution of the formation. Particularly, the contextual Exp3 algorithm tends to converge to a fixed optimal policy even when the optimal action varies. In the this section, we utilize the Exp4 algorithm which serves as a meta-learner to handle changes in the k^*.

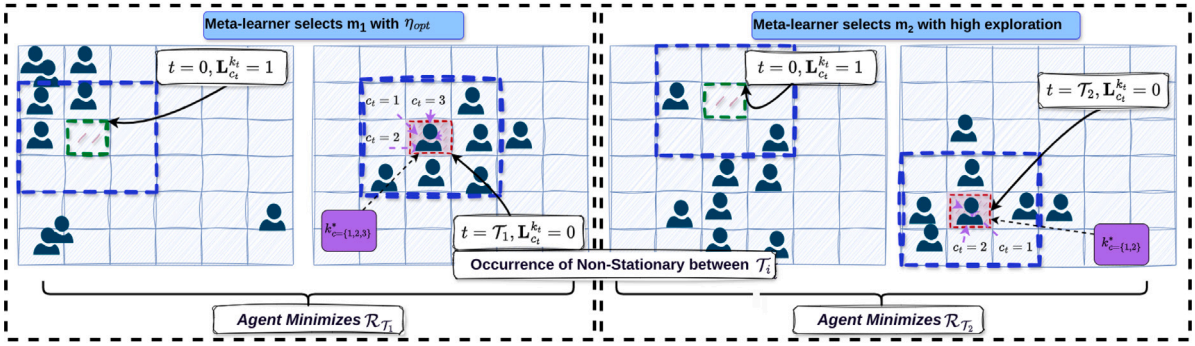


Fig. 5. Performance of Exp4 with respect to c_t assuming non-stationary movements of target formation. The meta-learner selects the expert m_1 during $\mathcal{T}_1$, which minimizes the regret of the system $\mathcal{R}_T$. Furthermore, the learner selects m_2 during $\mathcal{T}_2$ (drift in the environment), which is tuned towards exploration.

5.1. Exp4

We follow a distinct approach in learning the optimal performance, eliminating the necessity of detecting drifts in the environment. Instead of relying on AC-Exp3, which was previously discussed in Section 4.2, we embrace the conventional learning paradigm involving experts, with a particular focus on the Exp4 algorithm. Exp4 falls under the category of contextual multi-armed bandit algorithms (Lattimore & Szepesvári, 2020). In this framework, multiple experts provide their recommendations to the master learner (the agent), each offering their unique perspective on the environment and predictions for action k_t. The master learner evaluates a given set of experts $\mathcal{M}$ and selects the best expert from this set to minimize the regret. The primary objective of the master learner is to compete with the best expert in hindsight. Therefore, the focus is not solely on the actions themselves, but rather on the evaluation of the experts. We introduce the concept of experts and presents the regret analysis of the Exp4 algorithm within the context of the search mission problem.

Experts. We model the experts, denoted as $m \in \mathcal{M}$, where $\mathcal{M}$ represents a predefined set of experts, each deploying a C-Exp3 instance. Specifically, expert m_2 is configured with a high exploration rate, while expert m_1 is tuned with an optimal learning rate η_{opt} to achieve a balance between exploration and exploitation. This setup enables the Exp4 agent to handle changes in the best action more effectively compared to C-Exp3.

In the context of adversarial bandits, fine-tuning the learning rate is crucial. A learning rate that is excessively high can make the agent vulnerable to noisy environmental changes, potentially leading to instability and sub-optimal performance. Conversely, a learning rate that is too low may result in slow adaptation. Our goal is to achieve a well-balanced equilibrium that optimizes the trade-off between exploration and exploitation in a non-stationary environment.

To illustrate how this configuration helps in reaching the optimal performance in a non-stationary environment, we consider a use case scenario where the best action changes between two time intervals $\mathcal{T}_1$ and $\mathcal{T}_2$. By changing the best action periodically, the best expert is expected to change as well to balance between a high and a low exploration rate according to this change. However, it is important to note that this approach can be consistently employed across the entire time horizon T with any given ensemble of experts $\mathcal{M}$. We will use this scenario as a consistent reference point throughout our paper.

During $\mathcal{T}_1$: The agent determines that the best expert is m_1 as it employs C-Exp3 with η_{opt}. In contrast, m_2 persists in exclusively exploring the optimal action, as it is specifically configured for exploration.

During $\mathcal{T}_2$: The performance of expert m_2 begins to fall short of the expected level as the best action changes. m_2 takes multiple rounds to adapt and eliminate the influence of the previous time interval $\mathcal{T}_1$ while learning the new best action. In contrast, expert m_2 performs better since it continues to explore the changing mobility pattern. As a result, we observe a crossover in performance between m_1 and m_2 during $\mathcal{T}_2$.

The primary idea behind the agent learning the best expert in hindsight is to provide the best action observed during each time interval $\mathcal{T}$ (see Fig. 5).

We have fine-tuned Exp4 to handle changes in the formation of target strategies, which is reflected in periodic changes in the best action. This approach empowers the agent to conduct extensive exploration in the environment and learn new strategies, which could take longer for the Exp3 agent to accomplish by integrating two experts with distinct adjustments. Experts can show in various forms; for instance, an expert can be a human-in-the-loop or another algorithm. However, it is important to note that the algorithm incurs $O(|\mathcal{M}|)$ memory and $O(|\mathcal{M}|+|\mathcal{K}|)$ computation per time slot (Lattimore & Szepesvári, 2020). Consequently, its practicality is viable only when both $\mathcal{M}$ and $\mathcal{K}$ are reasonably small.

Regret. The regret metric of Exp4 quantifies the cumulative loss incurred by the master learner compared to the losses of the best expert in hindsight m^*. The predictions of the $|\mathcal{M}|$ experts in time slot $t \in T$ are represented by a matrix $\mathbf{E}_t \in [0,1]^{(|\mathcal{M}|\times|\mathcal{K}|)}$, where each row $\mathbf{E}_t(m)$ (mth row) corresponds to a probability vector over $\mathcal{K}$ indicating the recommendations of the mth expert for time slot t. Furthermore, the expression $\mathbf{E}_t(m)\mathbf{L}_{c_t}^{k_t}$ is the loss of the m expert with respect to action k at time t. We use the static regret definition as described in Eq. (18.6) (Lattimore & Szepesvári, 2020):

$$\tilde{\mathcal{R}_T} = \sum_{t=1}^{T} \left[\mathbf{L}_{c_t}^{k_t} - \mathbf{E}_t(m^*)\mathbf{L}_{c_t}^{k_t} \right] \tag{7}$$

$$m^* = \arg\min_{m\in[\mathcal{M}]} \mathbf{E}_t(m)\mathbf{L}_{c_t}^{k_t}$$

In this context, $\tilde{\mathcal{R}_T}$ corresponds to the regret within the time horizon $T = \{\mathcal{T}_1, \mathcal{T}_2\}$. Regret evaluate the agent's performance by measuring the gape between the loss associated with the selected expert m at time t by the master learner and the performance of the best expert $\mathbf{E}_t(m^*)$ in hindsight based on its associated loss. The expert m^* is chosen to minimize loss, and the selection of the m varies with each time slot. As a result, each expert during the time interval $\mathcal{T}$ possesses its own bound, utilizing the C-Exp3 bound specified in Eq. (6), along with its corresponding best actions k_c^*. The regret bound of the master learner is estimated using Eq. (18.7) as outlined in Lattimore and Szepesvári (2020).

$$\tilde{\mathcal{R}_T} \leq \sqrt{2T|\mathcal{K}|\log(|\mathcal{M}|)} \tag{8}$$

Algorithm. The algorithmic steps are shown in Algorithm 3 and detailed below. The meta-learner begins by initializing the weights Q_1 for all experts equally. At each time slot t, the learner receives recommendations from the experts in the form of a matrix $\mathbf{E}_t$, which contains the possible actions from each expert.

The learner selects an action k_t by sampling from a probability distribution P_t, derived from the current weights vector Q_t and the expert advice matrix $\mathbf{E}_t$. The ith row of $\mathbf{E}_t$ represents the ith expert's

Algorithm 3 Exp4

1: **Input:** $T, \mathcal{K}, \mathcal{M}, \eta$
2: $Q_1 \leftarrow \left(\frac{1}{|\mathcal{M}|}, \ldots, \frac{1}{|\mathcal{M}|}\right) \in [0,1]^{1\times|\mathcal{M}|}$ (a row vector)
3: **for** $t = 1$ to T **do**
4: Receive advice $\mathbf{E}_t$
5: Choose the action $k_t \sim P_t$, where $P_t = Q_t\mathbf{E}_t$
6: Receive the Cost $Y_t = \mathbf{L}_{c_t}^{k_t}$
7: Estimate the action Cost: $\hat{Y}_{ti} = \frac{\mathbb{I}\{k_t=i\}}{P_{ti}}(Y_t)$
8: Propagate the Cost to the experts: $\tilde{Y}_t = \mathbf{E}_t\hat{Y}_t$
9: Update the distribution Q_{t+1} using exponential weighting: $Q_{t+1,m} \leftarrow \frac{\exp(\eta\tilde{Y}_{t,m})Q_{t,m}}{\sum_m \exp(\eta\tilde{Y}_{t,m})Q_{t,m}}$ for all $m \in [\mathcal{M}]$
10: **end for**

recommendation, and the probability distribution is computed as $P_t = Q_t \cdot \mathbf{E}_t$. Sampling from P_t is equivalent to first sampling an expert and then sampling an action recommended by that expert. After executing the selected action k_t, the learner receives a cost Y_t associated with this action. The learner then estimates the cost for each action using the observed cost and the probability distribution P_t. The estimated cost for action i is given by $\hat{Y}_{ti} = \frac{\mathbb{I}\{k_t=i\}}{P_{ti}} \cdot Y_t$, where $\hat{Y}_{ti}$ serves as an unbiased estimator for Y_t. The estimated costs $\hat{Y}_t$ are propagated to the experts to update the cost vector $\tilde{Y}_t = \mathbf{E}_t \cdot \hat{Y}_t$, reflecting the aggregated performance of the experts based on their recommendations and the observed costs. Finally, the learner updates the weight distribution Q_{t+1} for the next time slot using an exponential gradient weighting mechanism (line 9), where η is the learning rate parameter.

Combined Policy. In each time interval $\mathcal{T}$, the master learner identifies the best expert that minimizes regret within that specific interval. This process is consistently applied across all time intervals $\mathcal{T} \in T$, resulting in the formation of a combined policy which defines a meta-learning policy that entails a decision-making combining predictions from the experts based on their priority weights, which are learned from observations at each time slot t. This approach underscores the fundamental concept of meta-learning, where knowledge is derived from multiple experts. Within this framework, the Exp4 agent systematically learns a combined policy over the entire duration T, encompassing both $\mathcal{T}_1$ and $\mathcal{T}_2$. Each interval within this time horizon possesses its own distinct best actions k_c^*, tailored to the formation's strategy, which, in turn, represents the mobility pattern of the formation.

Regret of the System. Now we derive the regret of the system which is composed of a master learner (Exp4) agent and the experts (C-Exp3). As previously explained in Section 4.1, the regret for each individual expert $\tilde{\mathcal{R}}_{m,\mathcal{T}}$ during a time interval $\mathcal{T}$ is defined as:

$$\tilde{\mathcal{R}}_{m,\mathcal{T}} = \sum_{t=1}^{\mathcal{T}} \left[(\mathbf{L}_{c_t}^{k_t} - \mathbf{L}_{c_t}^{k_c^*})\right] \mathbb{I}\{c = c_t\} \tag{9}$$

The objective of the meta-learner is to learn the best expert at each time interval $\mathcal{T}$, which will minimize the regret from the available set of experts $\mathcal{M}$ according to:

$$\mathcal{R}_T = \min\{\tilde{\mathcal{R}}_{1,\mathcal{T}}, \tilde{\mathcal{R}}_{2,\mathcal{T}}\} + \tilde{\mathcal{R}}_T \tag{10}$$

Where $\tilde{\mathcal{R}}_T$ represents the regret of the meta-learner and $\mathcal{R}_T$ is the regret of the system. The meta-learner selects the best expert with the minimum regret $\tilde{\mathcal{R}}_{m,\mathcal{T}}$, leading to a minimum regret for the overall system $\mathcal{R}_T$ at each time interval $\mathcal{T} \in T$. As the best actions k_c^* changes within the range $T = \{\mathcal{T}_1, \mathcal{T}_2\}$, the meta-learner adjusts its learning process to accommodate these variations in the best action which is reflected in the performance of the experts, following the use case scenario that described earlier:

$$\mathcal{R}_T = \begin{cases} \tilde{\mathcal{R}}_{1,\mathcal{T}} + \tilde{\mathcal{R}}_T & \text{if } t \in \mathcal{T}_1 \\ \tilde{\mathcal{R}}_{2,\mathcal{T}} + \tilde{\mathcal{R}}_T & \text{if } t \in \mathcal{T}_2 \end{cases} \tag{11}$$

In Eq. (11), the meta-learner selects $\tilde{\mathcal{R}}_{1,\mathcal{T}}$ since the best action is fixed during $\mathcal{T}_1$. However, the best action changes during $\mathcal{T}_2$, therefore, the meta-learner switches to $\tilde{\mathcal{R}}_{2,\mathcal{T}}$. Finally, the regret of the system follows:

$$\mathcal{R}_T = \sum_{t=1}^{T} \left[\mathbf{L}_{c_t}^{k_t} - \mathbf{E}_t(m^*)\mathbf{L}_{c_t}^{k_t}\right] + \sum_{t=1}^{\mathcal{T}} \left[(\mathbf{L}_{c_t}^{k_t} - \mathbf{L}_{c_t}^{k_c^*})\right] \mathbb{I}\{c = c_t\} \tag{12}$$

In summary, the master learner (Exp4) employs two experts, m_1 and m_2, which are specifically adapted to handle an environment where the best action changes periodically due to shifts in the formation's strategy. The master learner continuously learns the best expert for each time interval $\mathcal{T}$ by maintaining a probability distribution Q_t over experts and updating it at each time slot t. As a result, experts m_1 and m_2 exhibit crossover performance when the best action changes, this is how the agent learns the combined policy. The regret is estimated using Eq. (7), and an upper bound is maintained as in Eq. (8).

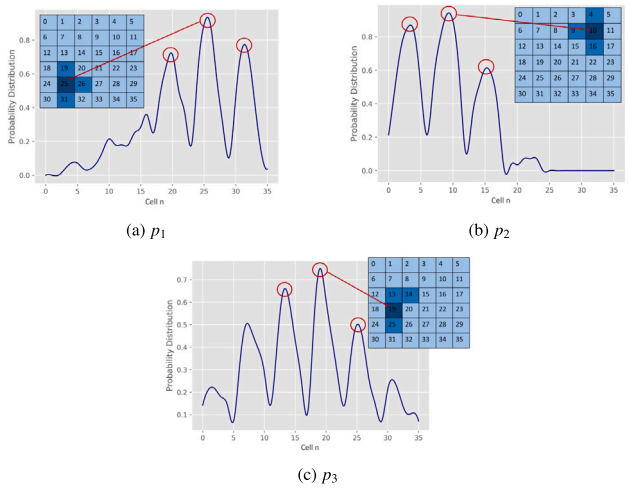
6. Performance evaluation

In this section, we start with a detailed description of the environment, followed by an overview of the model parameters. Subsequently, we will outline the performance metrics, and then proceed to evaluate the performance of the three algorithms (contextual Exp3, adaptive contextual Exp3, and Exp4) in the context of a search mission featuring a single UAV agent. Our environment assumes an absence of obstacles and focuses exclusively on the mobility of targets, disregarding any external factors that might influence their movement. It is worth noting that the environment remains unknown, and the targets adhere to the mobility pattern denoted as the reference point group (RPG) within a time horizon T. In this scenario, we assume that the agent operates at a fixed altitude and is equipped with imaging sensors, enabling it to observe the environment. The code used to conduct the experiments is publicly available at the following GitHub repository.[1]

Environment. The RPG model serves as a potent tool for emulating the collective dynamics of target formations. The RPG model is used in Khan, Heurtefeux, Mohamed, Harras, and Hassan (2017) for the problem of low complexity target tracking to cover and follow moving targets using UAV. Within this model, each target is affiliated with a logical center, known as the group leader, which governs the collective motion characteristics of the group. The targets comprising a group are distributed in an adversarial manner around the reference point. By employing their distinct mobility models, these targets are moving with random magnitude ν and angle direction θ assimilated into the reference point, which steers their trajectories following the group's direction.

At any given moment in the time horizon T, each target possesses unique values of θ and ν, which deviate randomly from those of the group leader. However, the target remains within the leader's boundary. In our RPG model, the movement of the group leader involves selecting a destination point within the deployment region in a stochastic manner. The leader then moves towards this destination with corresponding values of θ^* and ν^*. This motion profile establishes the leader's distinct trajectory and sets the overall motion trend for the entire group. As a result, each group member exhibits variations from this predominant motion vector, introducing individualistic dynamics into the collective formation. Therefore, the mobility pattern of the targets, is influenced by the movement of a leader, which follows a probability distribution over the $|\mathcal{N}|$ cells within the grid during time interval $\mathcal{T} \in T$. The movements of the followers are randomized, with $l_{t+1} = l_t + (\theta\nu)$. It is crucial to highlight that l_{t+1} always remains within the boundaries defined by the leader's center point. Fig. 7 shows the movement of the RPG-based mobility pattern.

[1] https://github.com/Noor-Khial/Online-Learning-UAV-Search.git



(a) p_1 (b) p_2

(c) p_3

Fig. 6. Probability Distribution of $p \in \mathcal{P}$ of the leader. The peaks of the distribution showed in the red circles represent a frequently visited cells.

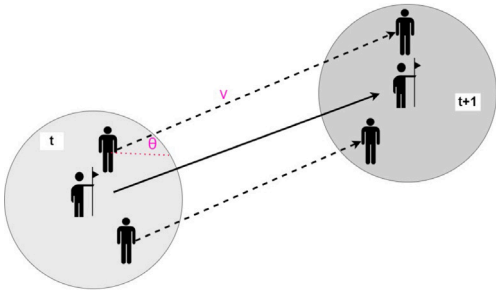


Fig. 7. RPG-based Mobility Pattern.

Mobility Pattern. We consider three distinct RPG-based mobility patterns denoted as $p \in \mathcal{P}$, each associated with a unique probability distribution for the leader. Notably, each mobility pattern corresponds to its specific optimal policy. Fig. 6 illustrates the probability distribution of each p.

Model Parameters. In all experiments for the contextual Exp3 algorithm, we apply the optimal learning rate η_{opt} as discussed in Section 4.1. The action space $\mathcal{K}$ and the context space $\mathcal{C}$ are predefined. Given that each location cell $n \in \mathcal{N} = 36$ represents a context $c \in \mathcal{C}$, we have $\mathcal{C} = 36$. In this paper, we specifically focus on analyzing the algorithms' performance, which is why we opted for a relatively small context space. Furthermore, the interval of time $\mathcal{T} \in T$ is predefined and varies according to each experiment. Table 2 summarizes the remaining parameters used in each experiment.

Performance Metric. We evaluate the algorithm's performance using cumulative regret and we show the utility and cell probabilities. The goal is to minimize cumulative regret, which converges over time. The utility measures the algorithm's success in accumulating rewards relative to the maximum potential rewards achievable with the best action k^*. The utility at each time step is expressed as a percentage of the benchmark. Furthermore, we average each data point with the preceding 100 points. Additionally, cell probabilities indicate the assigned weights to cell, with the highest weight signifying the agent's effective policy to minimize losses.

We conduct a thorough evaluation of algorithms in diverse non-stationary environments. First, we evaluate the performance of the contextual Exp3 algorithm in scenarios with both fixed and changing best actions. Next, we assess the adaptive contextual Exp3 algorithm, which incorporates a human-based drift detector (as discussed in Section 4.2), along with the performance of Exp4 as a meta-learner in the presence of changing best actions.

Table 2
Parameters summary.

Parameter	Experiment			
	1	2	3	4
$\mathcal{T}_1 \times 10^3$		[0, 40]		
$\mathcal{T}_2 \times 10^3$	[40, 100]	[40, 200]	[40, 100]	[40, 100]
Optimal policy	10	10,25	10,25	25,19
η_{opt}		0.0018		

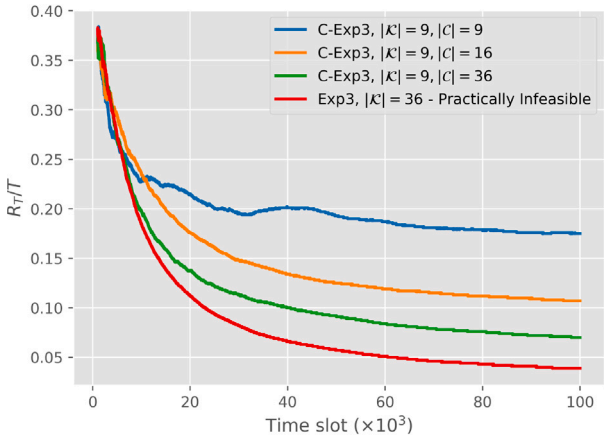


Fig. 8. $\mathcal{R}_T/T$ for C-Exp3 with $|C| = \{9, 16, 36\}$.

6.1. Performance of the agent with C-Exp3 algorithm

We first analyze the performance of C-Exp3 with varying context numbers, comparing it to the Exp3 version. Next, we evaluate C-Exp3 under optimal context tuning with both fixed and dynamic best actions, and benchmark it against state-of-the-art RL models. The C-Exp3 algorithm, detailed in Algorithm 1, computes regret as per Eq. (5) and upper bounds following the approach in Section 4.1 (Eq. (6)). We use the optimal learning rate η_{opt} listed in Table 2.

6.1.1. Impact of context scaling on C-Exp3 performance

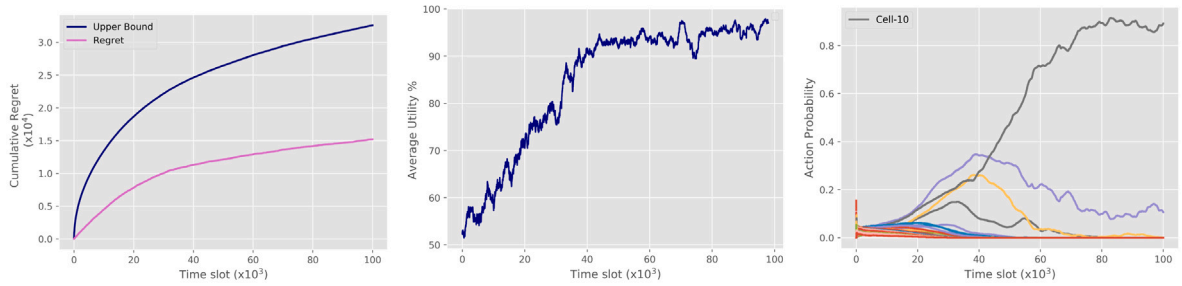
We study the performance of the C-Exp3, if we considered different sizes of the context space $|C|$. This is important to select the optimal tuning for the number of contexts. Additionally, we test the Exp3 algorithm, without the addition of the context, which we assume to be practically infeasible, and we explain next in details the configuration for C-Exp3 and Exp3.

In our experimental setup, the environment consists of a 6×6 grid, with a total of 36 cells. We define the context space by dividing the environment into smaller sub-grids, each with dimensions of $(d \times d)$ cells. The size of these sub-grids is carefully adjusted to create specific numbers of contexts, where the context set sizes $|C|$ are 9, 16, and 36. As we decrease the size of the sub-grids (*i.e.*, reduce the value of d), the number of distinct contexts increases. This continues until each sub-grid is reduced to just one cell.

In C-Exp3, the action space consists of directional movements, enabling the agent to navigate through the environment. We also tested an Exp3 version where actions involve selecting specific cells directly, rather than moving directionally. However, this setup is impractical, as it assumes the agent can instantly jump between cells (*e.g.*, $0 \rightarrow 36$ cells) without intermediate steps, ignoring the agent's need for stepwise navigation.

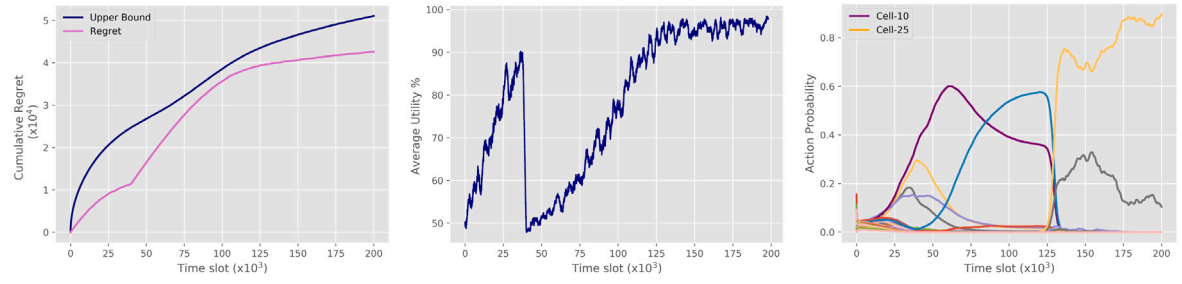
Despite this impracticality, this version can serve as a benchmark, since the exploration time for Exp3 would be shorter compared to C-Exp3. In the C-Exp3 setup, the agent requires more time to navigate the environment, moving from one context to another as it selects actions.

Discussion. Fig. 8 illustrates the performance comparison. It shows that as the context space size increases to match the total cell count, the



(a) Cumulative Regret (b) Average Utility (c) Cell Probability

Fig. 9. Performance of Contextual Exp3 in Adversarial Environment with Fixed Best Action.



(a) Cumulative Regret (b) Average Utility (c) Cell Probability

Fig. 10. Performance of Contextual Exp3 in Adversarial Environment with Changing Best Action.

average regret $\mathcal{R}_{\mathcal{T}}/\mathcal{T}$ converges faster to zero. When each context is a single cell, the agent can consistently learn the optimal action for each cell across action–context pairs until reaching the targets. Meanwhile, with larger sub-grid contexts, the agent encounters uncertainty among several sub-optimal directions that slow the convergence.

6.1.2. Fixed best action

Results of experiment 1 is illustrated in Fig. 9. The leader adheres to the mobility pattern $p = 1$ as depicted in Fig. 6(a), with the optimal policy $n = 10$. To determine this optimal policy (benchmark), we use the probability distribution based on the leader's mobility pattern $p = 1$ during $\mathcal{T}_1$.

Discussion. In Fig. 9(a), the agent's cumulative regret consistently remains lower than the upper bound. This convergence pattern suggests a steady relationship between the cumulative regret of the optimal policy $n = 10$ and the agent's cumulative regret. The utility in Fig. 9(b) further emphasizes this trend by exhibiting a continuous increase over time, ultimately matching the performance of the optimal policy $n = 10$. Additionally, the alignment seen in Fig. 9(c) in the highest cell probability $n = 10$ adds further support to these observations.

6.1.3. Comparison with state-of-the-art solutions

According to the current work, DQN, Q-Table, and PPO are the commonly used approaches in this problem. Therefore, we have implemented these algorithms and compare them with C-Exp3. Furthermore, we use the same action space $\mathcal{K}$ in all the RL models. The exploration–exploitation trade-off is handled through the epsilon-greedy policy for DQN and Q-Table, and through Gaussian noise for PPO. The hyperparameters used for these approaches are summarized in Table 3.

Table 4 compares the average regret ($\mathcal{R}_{\mathcal{T}}/\mathcal{T}$) of three reinforcement learning algorithms (Q-Table, DQN, and PPO) against our proposed C-Exp3 algorithm across different learning rates η. The results indicate that C-Exp3 achieves the lowest regret of 0.08 at $\eta = 0.001$, significantly outperforming all RL approaches, while PPO maintains a high regret of 0.6.

The superior performance of C-Exp3 is due to its distinct learning approach. In C-Exp3, the agent leverages prior knowledge about the context space and treats each context independently, focusing on learning the optimal action per context. Unlike RL methods, which attempt to learn state transition probabilities and build value functions, C-Exp3 makes no assumptions about probabilistic correlations between contexts. This characteristic is particularly advantageous in adversarial settings, where RL algorithms struggle to model transitions in arbitrary and potentially misleading environments. The results demonstrate that C-Exp3's simpler modeling assumptions lead to more effective learning in adversarial conditions compared to traditional RL approaches that rely on learning state transition probabilities.

Table 3
Hyperparameters and configurations for PPO, DQN, and Q-Table.

Parameter	PPO	DQN	Q-Table
Discount Factor	0.99	0.9	0.9
Exploration Strategy	Gaussian Noise	ϵ-greedy	ϵ-greedy
Exploration Rate (ϵ)	–	0.2	0.2
Entropy Coefficient	0.01	–	–
Neural Network Layers	(64, 64)	(64, 64)	–
Optimizer	Adam	Adam	–
Grid Size	6 × 6	6 × 6	6 × 6
Maximum Steps	1 per episode	1 per episode	1 per episode

Table 4
Comparison of RL Models and C-Exp3 in terms of $\mathcal{R}_{\mathcal{T}}/\mathcal{T}$.

Algorithm	Q-Table	DQN	PPO	C-Exp3
$\eta = 0.0001$	0.20	0.25	0.6	0.18
$\eta = 0.001$	0.22	0.17	0.6	**0.08**
$\eta = 0.01$	0.27	0.21	0.6	0.19

6.1.4. Changing best action

Results of experiment 1 is illustrated in Fig. 10. Initially, the leader adheres to the mobility pattern $p = 1$ as described in Fig. 6(a), with the optimal policy set at $n = 10$ during the time interval $t \in \mathcal{T}_1$, as specified in Table 2. Subsequently, the leader transitions to the mobility pattern $p = 2$ as described in Fig. 6(b), where the optimal policy becomes $n = 25$ during the time interval $t \in \mathcal{T}_2$. The optimal policies of $n = 10$ and $n = 25$

Table 5
Performance of AC-Exp3 with different initialization methods.

Initialization Method	$\mathcal{R}/\mathcal{T}$		Utility %		k^* Probability	
	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_1$	$\mathcal{T}_2$
Uniform Weights	0.27	0.23	89	97	0.45	0.62
Biased Weights	0.25	**0.20**	90	**99**	0.45	**0.99**

Table 6
Performance of Exp4 in adversarial environment with changing k^*.

Learner	$\mathcal{R}/\mathcal{T}$		Utility %		k^* Probability	
	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_1$	$\mathcal{T}_2$
Expert 1	0.25	0.5	91	60	0.67	0.3
Expert 2	0.5	0.21	61	96	0.13	0.71
Meta-Learner	**0.25**	**0.22**	**90**	**96**	0.53	0.48

are determined based on the leader distribution corresponding to the mobility patterns $p = 1$ and $p = 2$, respectively.

Discussion. In Fig. 10(a), during $\mathcal{T}_1$, the cumulative regret converges, and the agent's performance stays below the upper bound. However, in $\mathcal{T}_2$, a shift in target formations to $p = 2$ leads to an increase in regret as the optimal policy changes to $n = 25$. The regret eventually converges, indicating the agent's adaptation to the new optimal action.

Fig. 10(b) further highlights the difference between the agent's performance and the optimal policy. Initially, the agent's performance aligns with the optimal policy of $n = 10$ but declines with the change in the best action, eventually recovering. Similarly, Fig. 10(c) shows that during $\mathcal{T}_1$, the highest action probability corresponds to $n = 10$. As the target formation changes, the agent needs time to adapt to $n = 25$ in $\mathcal{T}_2$. Fig. 10(a) shows that the DQN model, with a higher η_{DQN}, performs better during $\mathcal{T}_1$ but struggles more during $\mathcal{T}_2$ compared to C-Exp3.

6.2. Performance of the agent with AC-Exp3 algorithm

In this section, we present the AC-Exp3 algorithm implementation as per Algorithm 2. For each interval $\mathcal{T}_i$, we calculate cumulative regret using Eq. (5), with upper bounds determined via Section 4.1 and Eq. (6). We examine two scenarios: AC-Exp3 with uniform initial weights and with biased initial weights.

Uniform weights. In experiment 3, we have used the same setup for testing the AC-Exp3 algorithm in a non-stationary setting and a changing best action as in experiment 2. The initial weight for all actions represents a uniform distribution, $\hat{S}_{0,i} = 0$ on each $\mathcal{T}$.

Biased Weights. In experiment 4, we apply the same setup as in experiment 2 to test the AC-Exp3 algorithm in an adversarial scenario with a changing best action. However, instead of initializing the actions with uniform weights during the algorithm's resetting, we adopted biased weights based on the concept of aligning the leader's mobility pattern during $\mathcal{T}_1$ with the new mobility pattern during the interval $\mathcal{T}_2$. To elaborate further, during $\mathcal{T}_1$, the optimal policy is $n = 25$, while during $\mathcal{T}_2$, the optimal policy is $n = 19$, where cell $n = 19$ is located just above cell $n = 25$. Consequently, rather than starting with uniform actions, the agent prioritizes the neighboring cells to the one representing the best action in $\mathcal{T}_1$.

Comparing AC-Exp3 with Uniform and Biased Weights. The experimental results in Table 5 demonstrate the superiority of AC-Exp3 with biased weights over uniform initialization. During $\mathcal{T}_2$, biased weights achieve 99% utility and 0.20 average regret, compared to 97% utility and 0.23 regret with uniform weights. Most significantly, the optimal action probability reaches 0.99 with biased weights versus 0.62 with uniform weights. These findings confirm that leveraging relationships between consecutive optimal policies through biased initialization substantially improves the algorithm's efficiency by reducing unnecessary exploration.

Comparison Between AC-Exp3 and C-Exp3 with a Change in Best Action. AC-Exp3 exhibits superior regret performance compared to C-Exp3 because it considers the potential change in the best action and subsequently resets the Exp3 algorithm. In the worst-case scenario where the HITL-based drift detector fails to predict a change in the leader's mobility pattern, the agent's performance will be the same as C-Exp3.

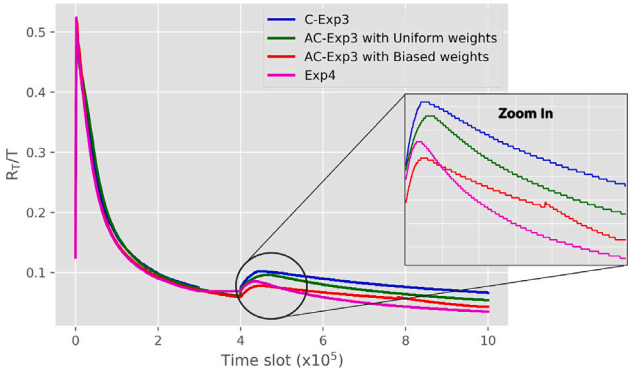


Fig. 11. $\mathcal{R}_T/T$ for C-Exp3, AC-Exp3, and Exp4.

6.3. Performance of the agent with Exp4 algorithm

We demonstrate how Exp4 adapts to changes in the best action k^* by transitioning between experts, without the necessity of a drift detector. For experiment 5, we utilize the Exp4 algorithm, as described in Algorithm 3. Within each time interval $\mathcal{T}_i$ as in Table 2, the regret is calculated using Eq. (7), and the upper bound is determined following the method explained in Section 5.1 in Eq. (8).

6.3.1. Changing best action

We investigate the performance of the agent in a search mission using the Exp4 algorithm. Table 6 shows the agent's performance in an adversarial environment where the optimal action changes periodically over time. Initially, the leader adheres to mobility pattern $p = 1$, as depicted in Fig. 6(a). In this phase, the optimal policy is $n = 10$ within the time interval $\mathcal{T}_1$. Subsequently, the leader transitions to mobility pattern $p = 2$, as described in Fig. 6(b). In this phase, the optimal policy becomes $n = 25$ within the time interval $\mathcal{T}_2$.

Discussion. The average regret $\bar{\mathcal{R}}_T$ for the meta-learner convergence is evident during two distinct intervals: $t \in \mathcal{T}_1$ and $t \in \mathcal{T}_2$. In the first interval, $\mathcal{T}_1$, the meta-learner relies on expert m_1, identified as optimal with policy $n = 10$. In the second interval, $\mathcal{T}_2$, the focus shifts to expert m_2 with policy $n = 25$. Showing m_1 as the best expert during $\mathcal{T}_1$ and m_2 during $\mathcal{T}_2$, leading to a combined policy and crossover in performance. The utility increases steadily over time within each interval $\mathcal{T}$, reaching 90% of the optimal benchmark.

Experts. The performance of expert m_1 mirrors the C-Exp3 algorithm, while expert m_2 focuses on exploration, with an optimal policy of $n = 25$. The average utility of expert m_2 remains stable despite changes in the best action due to its exploratory nature. During $\mathcal{T}_2$, expert m_2 outperforms expert m_1 in utility, leading the master to switch to m_2. However, the master later reverts to m_1 after it retains the best action, resulting in a performance crossover. This highlights the meta-learner's ability to adapt by strategically switching between experts.

Comparing Exp4 and AC-Exp3 with Uniform Weights. The system's regret $\mathcal{R}_T$ converges as t approaches the time horizon T. Fig. 11 shows that $\lim_{t\to\infty} \frac{\mathcal{R}_T}{T} = 0$, indicating sub-linear regret convergence. For C-Exp3, the $\mathcal{R}_T/T$ ratio exhibits a less favorable trend due to a longer adjustment period, with a noticeable fluctuation during $t \in \mathcal{T}_2 = [40, 10] \times 10^5$. In contrast, AC-Exp3 shows reduced fluctuation in $\mathcal{R}_T/T$ during $\mathcal{T}_2$, indicating better adaptation to the new strategy. The biased

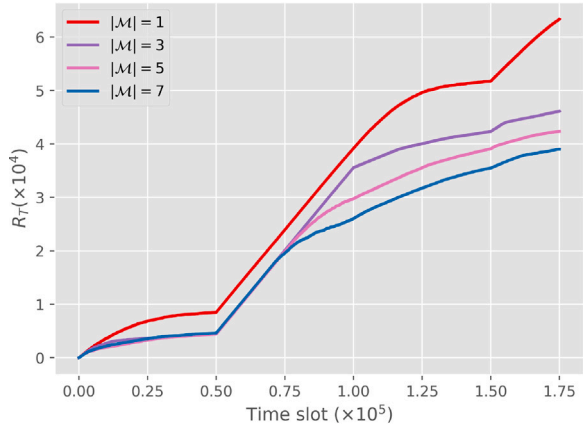


Fig. 12. $\mathcal{R}_T$ for Exp4 with different number of experts.

weights version of AC-Exp3 outperforms the uniform weight version by leveraging prior knowledge and minimizing exploration. Exp4 outperforms all other methods, rapidly adapting to the new strategy with minimal fluctuation during $\mathcal{T}_2$.

6.3.2. Impact of the number of experts on Exp4

To test the adaptivity of the Exp4 algorithm under multiple shifts in target mobility, which result in changes to the optimal policy, we conducted experiments with $|\mathcal{M}|$ set to 1, 3, 5, and 7. Each expert was assigned a different learning rate η, tuned using a logarithmic search grid around the minmax optimal learning rate η_{opt}. Specifically, for 3 experts, one expert uses η_{opt}, while the other two use η values that are $\eta_{\text{opt}}/10$ and $10 \times \eta_{\text{opt}}$. For 5 experts, we included the same 3 experts and added two more with η values that are $\eta_{\text{opt}}/100$ and $100 \times \eta_{\text{opt}}$. The same approach was applied to 7 experts. We introduced 4 shifts in this experiment. The target distribution was p_1 during $\mathcal{T}_1 = [0, 0.5]$, p_2 during $\mathcal{T}_2 = [0.5, 1]$, p_3 during $\mathcal{T}_3 = [1, 1.5]$, and returned to p_1 during $\mathcal{T}_4 = [1.5, 1.75]$, where $\mathcal{T}$ is scaled by 10^5.

Discussion. Fig. 12 presents the performance of the meta-learner in terms of cumulative regret as the number of experts varies. The results show that the rate of convergence improves with an increasing number of experts. This improvement occurs because having more experts enables the meta-learner to explore more effectively, given they use logarithmic search grid around the minmax optimal learning rate. Consequently, the algorithm adapts more quickly to changes in the environment. Furthermore, the figure shows that the impact of the shift is almost negligible with time as the peak of the cumulative regret in each shift interval is flattened as in $\mathcal{T}_3$&$\mathcal{T}_4$. While adding more experts enhances the rate of convergence, it also increases computational complexity, as the meta-learner must simultaneously manage and execute more experts.

7. Conclusion

In conclusion, we use MAB algorithms for UAV search mission in unknown and adversarial environments. The C-Exp3 algorithm is applied with a fixed target formation strategy. To handle periodic changes in this strategy, we introduce AC-Exp3, and use Exp4 as a meta-learner. Our experiments show that AC-Exp3 and Exp4 perform better than C-Exp3 in non-stationary environments.

7.1. Future work

We discuss several potential future directions for this work. The cluttered and obstacle-filled nature of environments like military operations or SAR missions can be Incorporated as part of the problem. Furthermore, we introduce our approach to operate online. However, we have not yet considered some critical aspects of UAV operations, such as energy limitations. The objective would be to learn the targets' strategies while exploring the environment and to manage energy consumption at each time step. Additionally, while we have demonstrated the performance of C-Exp3 in comparison to DQN, it operates under the assumption of the worst-case scenario at all times, which may not always be realistic. The environment could vary between stochastic i.i.d. (independent and identically distributed) conditions and fully adversarial losses. This raises the question: *How can we design an agent capable of simultaneously considering these different scenarios?* A possible solution for this in introduced by Sachs, Hadiji, van Erven, and Guzmán (2022).

CRediT authorship contribution statement

Noor Khial: Conceptualization, Methodology, Software, Formal analysis, Writing – original draft. **Naram Mhaisen:** Conceptualization, Methodology, Formal analysis, Writing – review & editing, Supervision. **Mohamed Mabrok:** Conceptualization, Writing – review & editing, Supervision. **Amr Mohamed:** Conceptualization, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Research reported in this publication was supported by the Qatar Research Development and Innovation Council ARG01-0527-230356. The content is solely the responsibility of the authors and does not necessarily represent the official views of Qatar Research Development and Innovation Council.

Data availability

No data was used for the research described in the article.

References

Almarzoqi, S. A., Yahya, A., Matar, Z., & Gomaa, I. (2022). Re-learning exp3 multi-armed bandit algorithm for enhancing the massive iot-lorawan network performance. *Sensors*, *22*(4), 1603.

Alotaibi, E. T., Alqefari, S. S., & Koubaa, A. (2019). Lsar: Multi-uav collaboration for search and rescue missions. *IEEE Access*, *7*, 55817–55832.

Auer, P., Cesa-Bianchi, N., Freund, Y., & Schapire, R. E. (1995). Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th annual foundations of computer science* (pp. 322–331). IEEE.

Baccour, E., Mhaisen, N., Abdellatif, A. A., Erbad, A., Mohamed, A., Hamdi, M., et al. (2022). Pervasive AI for IoT applications: A survey on resource-efficient distributed artificial intelligence. *IEEE Communications Surveys & Tutorials*.

Balafrej, A., Bessiere, C., & Paparrizou, A. (2015). Multi-armed bandits for adaptive constraint propagation. In *IJCAI: international joint conference on artificial intelligence* (pp. 290–296). AAAI Press.

Blais, M.-A., & Akhloufi, M. A. (2024). Drone swarm coordination using reinforcement learning for efficient wildfires fighting. *SN Computer Science*, *5*(3), 314.

Bouneffouf, D., Rish, I., & Aggarwal, C. (2020). Survey on applications of multi-armed and contextual bandits. In *2020 IEEE congress on evolutionary computation* (pp. 1–8). IEEE.

Cherif, M. S., Habet, D., & Terrioux, C. (2020). On the refinement of conflict history search through multi-armed bandit. In *2020 IEEE 32nd international conference on tools with artificial intelligence* (pp. 264–271). IEEE.

Gan, S. K., & Sukkarieh, S. (2011). Multi-UAV target search using explicit decentralized gradient-based negotiation. In *2011 IEEE international conference on robotics and automation* (pp. 751–756). IEEE.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

Gu, J., Su, T., Wang, Q., Du, X., & Guizani, M. (2018). Multiple moving targets surveillance based on a cooperative network for multi-UAV. *IEEE Communications Magazine*, *56*(4), 82–89.

Hong, L., Wang, Y., Du, Y., Chen, X., & Zheng, Y. (2021). UAV search-and-rescue planning using an adaptive memetic algorithm. *Frontiers of Information Technology & Electronic Engineering*, *22*(11), 1477–1491.

Hu, B., & Li, J. (2021). Shifting deep reinforcement learning algorithm toward training directly in transient real-world environment: A case study in powertrain control. *IEEE Transactions on Industrial Informatics*, *17*(12), 8198–8206.

Kanoun, K., & van der Schaar, M. (2015). Big-data streaming applications scheduling with online learning and concept drift detection. In *2015 design, automation & test in europe conference & exhibition* (pp. 1547–1550). IEEE.

Khan, M., Heurtefeux, K., Mohamed, A., Harras, K. A., & Hassan, M. M. (2017). Mobile target coverage and tracking on drone-be-gone UAV cyber-physical testbed. *IEEE Systems Journal*, *12*(4), 3485–3496.

Kos, J., & Song, D. (2017). Delving into adversarial attacks on deep policies. arXiv preprint arXiv:1705.06452.

Lattimore, T., & Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press.

Li, L., Xu, S., Nie, H., Mao, Y., & Yu, S. (2021). Collaborative target search algorithm for UAV based on chaotic disturbance pigeon-inspired optimization. *Applied Sciences*, *11*(16), 7358.

Lin, L., & Goodrich, M. A. (2014a). Hierarchical heuristic search using a Gaussian mixture model for UAV coverage planning. *IEEE Transactions on Cybernetics*, *44*(12), 2532–2544.

Lin, L., & Goodrich, M. A. (2014b). Hierarchical heuristic search using a Gaussian mixture model for UAV coverage planning. *IEEE Transactions on Cybernetics*, *44*(12), 2532–2544.

Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D., Bobes-Bascarán, J., & Fernández-Leal, Á. (2023). Human-in-the-loop machine learning: A state of the art. *Artificial Intelligence Review*, *56*(4), 3005–3054.

Mozaffari, M., Saad, W., Bennis, M., Nam, Y.-H., & Debbah, M. (2019). A tutorial on UAVs for wireless networks: Applications, challenges, and open problems. *IEEE Communications Surveys & Tutorials*, *21*(3), 2334–2360.

Pérez-Carabaza, S., Scherer, J., Rinner, B., López-Orozco, J. A., & Besada-Portas, E. (2019). UAV trajectory optimization for minimum time search with communication constraints and collision avoidance. *Engineering Applications of Artificial Intelligence*, *85*, 357–371.

Qi, Q., Yang, X., & Xia, G. (2024). Safe reinforcement learning filter for multicopter collision-free tracking under disturbances. arXiv preprint arXiv:2410.06852.

Qi, D., Zhao, Y., Li, L., & Jia, Z. (2024). Optimization of predefined-time agent-scheduling strategy based on PPO. *Mathematics*, *12*(15), 2387.

Sachs, S., Hadiji, H., van Erven, T., & Guzmán, C. (2022). Between stochastic and adversarial online convex optimization: Improved regret bounds via smoothness. *Advances in Neural Information Processing Systems*, *35*, 691–702.

Shurrab, M., Mizouni, R., Singh, S., & Otrok, H. (2023). Reinforcement learning framework for UAV-based target localization applications. *Internet of Things*, *23*, Article 100867.

Silvagni, M., Tonoli, A., Zenerino, E., & Chiaberge, M. (2017). Multipurpose UAV for search and rescue operations in mountain avalanche events. *Geomatics, Natural Hazards and Risk*, *8*(1), 18–33.

Soliman, A., Al-Ali, A., Mohamed, A., Gedawy, H., Izham, D., Bahri, M., et al. (2023a). AI-based UAV navigation framework with digital twin technology for mobile target visitation. *Engineering Applications of Artificial Intelligence*, *123*, Article 106318.

Soliman, A., Al-Ali, A., Mohamed, A., Gedawy, H., Izham, D., Bahri, M., et al. (2023b). AI-based UAV navigation framework with digital twin technology for mobile target visitation. *Engineering Applications of Artificial Intelligence*, *123*, Article 106318.

Venturini, F., Mason, F., Pase, F., Chiariotti, F., Testolin, A., Zanella, A., et al. (2021). Distributed reinforcement learning for flexible and efficient UAV swarm control. *IEEE Transactions on Cognitive Communications and Networking*, *7*(3), 955–969.

Wan, K., Gao, X., Hu, Z., & Wu, G. (2020). Robust motion control for UAV in dynamic uncertain environments using deep reinforcement learning. *Remote Sensing*, *12*(4), 640.

Wang, S., Han, Y., Chen, J., Zhang, Z., Wang, G., & Du, N. (2018). A deep-learning-based sea search and rescue algorithm by UAV remote sensing. In *2018 IEEE cSAA guidance, navigation and control conference* (pp. 1–5). IEEE.

Wei, X. L., Huang, X. L., Lu, T., & Song, G. G. (2019). An improved method based on deep reinforcement learning for target searching. In *2019 4th international conference on robotics and automation engineering (icrae)* (pp. 130–134). IEEE.

Wu, B., Chen, T., Ni, W., & Wang, X. (2021). Multi-agent multi-armed bandit learning for online management of edge-assisted computing. *IEEE Transactions on Communications*, *69*(12), 8188–8199.

Wu, J., Li, D., Shi, J., Li, X., Gao, L., Yu, L., et al. (2023). An adaptive conversion speed Q-learning algorithm for search and rescue UAV path planning in unknown environments. *IEEE Transactions on Vehicular Technology*.

Xiaoning, Z. (2020). Analysis of military application of UAV swarm technology. In *2020 3rd international conference on unmanned systems* (pp. 1200–1204). IEEE.

Xu, Z., Wang, Q., Kong, F., Yu, H., Gao, S., & Pan, D. (2022). Ga-DQN: A gravity-aware DQN based UAV path p@onlinetravelwiseway. In *2022 IEEE international conference on unmanned systems* (pp. 1215–1220). IEEE.

Yue, W., Guan, X., & Wang, L. (2019). A novel searching method using reinforcement learning scheme for multi-uavs in unknown environments. *Applied Sciences*, *9*(22), 4964.

Yue, W., Guan, X., & Xi, Y. (2019). Reinforcement learning based approach for multi-UAV cooperative searching in unknown environments. In *2019 Chinese automation congress* (pp. 2018–2023). IEEE.

Zhang, M., Li, W., Wang, M., Li, S., & Li, B. (2022). Helicopter–UAVs search and rescue task allocation considering UAVs operating environment and performance. *Computers & Industrial Engineering*, *167*, Article 107994.

Zhang, S., Li, Y., Ye, F., Geng, X., Zhou, Z., & Shi, T. (2023). A hybrid human-in-the-loop deep reinforcement learning method for UAV motion planning for long trajectories with unpredictable obstacles. *Drones*, *7*(5), 311.

Zhang, X., Zhao, W., Liu, C., & Li, J. (2023). Distributed multi-target search and surveillance mission planning for unmanned aerial vehicles in uncertain environments. *Drones*, *7*(6), 355.

Zhang, X., Zheng, G., & Lambotharan, S. (2020). Trajectory design for UAV-assisted emergency communications: A transfer learning approach. In *IEEE global communications conference, Taipei, Taiwan*.

Zhang, C., Zhou, W., Qin, W., & Tang, W. (2023). A novel UAV path planning approach: Heuristic crossing search and rescue optimization algorithm. *Expert Systems with Applications*, *215*, Article 119243.

Zheng, Y.-J., Du, Y.-C., Ling, H.-F., Sheng, W.-G., & Chen, S.-Y. (2019). Evolutionary collaborative human-UAV search for escaped criminals. *IEEE Transactions on Evolutionary Computation*, *24*(2), 217–231.

Zheng, Y.-J., Du, Y.-C., Sheng, W.-G., & Ling, H.-F. (2019). Collaborative human–UAV search and rescue for missing tourists in nature reserves. *INFORMS Journal on Applied Analytics*, *49*(5), 371–383.

Zheng, J., He, K., Zhou, J., Jin, Y., Li, C.-M., & Manyà, F. (2022). Bandmaxsat: A local search maxsat solver with multi-armed bandit. arXiv preprint arXiv:2201.05544.