NÖHÜ Müh. Bilim. Derg. / NOHU J. Eng. Sci., 2023; 12(4), 1271-1281

Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi
Niğde Ömer Halisdemir University Journal of Engineering Sciences

Araştırma makalesi / Research article

www.dergipark.org.tr/tr/pub/ngumuh / www.dergipark.org.tr/en/pub/ngumuh

Distributed coverage control with mobile robots: A potential game approach

Mobil robotlar ile dağıtık kapsama kontrolü: Bir potansiyel oyun yaklaşımı

Samet Güler[1,*]

[1] *Abdullah Gül Üniversitesi, Elektrik-Elektronik Mühendisliği Bölümü, 38080, Kayseri Türkiye*

Abstract

The use of mobile robots in industrial applications has led to a demand for autonomous multi-robot systems with robust and distributed algorithms. A critical objective in such systems is coverage control, where a team of mobile robots need to respond to spatiotemporal events in a bounded region. Here, we address a specific coverage problem, where a group of mobile robots are tasked with responding to events by covering specific locations on two sides of a linear workstation. We formulate the problem as a game played by the mobile robots with well-designed player strategies, and we demonstrate that the resulting framework is a potential game based on equally shared utilities among the robots. The proposed framework is distributed and decentralized, allowing for anonymous identities and constrained sensing capabilities in the robots. A set of simulation studies verify our approach.

Keywords: Coverage control, Multi-robot systems, Learning algorithms

Öz

Endüstriyel uygulamalarda mobil robotların kullanımı, gürbüz ve dağıtık algoritma içeren otonom çoklu-robot sistemlerine bir gereksinim oluşturmuştur. Bir robot takımının sınırlı bir alanda uzaysal-zamansal olaylara cevap vermesi anlamına gelen kapsama kontrolü bu tür sistemlerde kritik bir hedeftir. Bu çalışmada, bir grup mobil robotun doğrusal bir iş istasyonunun iki tarafında belirli lokasyonları kapsamakla görevli olduğu özel bir kapsama problemini ele alıyoruz. Problemi iyi kurgulanmış oyuncu stratejileri ile mobil robotlar arasında oynanan bir oyun olarak formalize ediyor ve ortaya çıkan yapının eşit paylaşılan fayda temelli bir potansiyel oyun olduğunu gösteriyoruz. Sunulan yapı, robotlarda anonim kimlikler ve kısıtlı algılama yeteneklerine izin veren dağıtık ve merkezi olmayan bir yapıdır. Bir grup simülasyon çalışması yaklaşımımızı doğrulamaktadır.

Anahtar kelimeler: Kapsama kontrolü, Çoklu-robot sistemleri, Öğrenme algoritmaları.

1 Introduction

The integration of mobile robots in industrial environments has the potential to greatly improve the efficiency and productivity of many industrial processes. In an industrial workspace, the use of mobile robots along with robot manipulators can enhance the flexibility, robustness, and resiliency of routine operations [1-3]. Particularly, mobile manipulators can combine the advantages of ground robots and manipulators and extend the configuration space of manipulators by moving their bases in a Cartesian environment. This additional freedom of motion is being explored across many applications. In some Industry 4.0 settings, a mobile manipulator is usually tasked with picking objects from designated locations and placing them at a given target location autonomously. Additionally, mobile manipulators can be used for monitoring a predetermined workspace, allowing remote operators to detect and respond to events in real time. Mobile manipulators have demonstrated advantages in terms of precision in attaining these tasks compared to human workers [3].

An important task of service robots in industrial environments is to cover a desired workspace with some performance guarantees. Coverage control involves the spatial allocation of a set of robots in a workspace to achieve some desired level of coverage. Many distributed coverage control schemes have been derived for multi-robot systems; see for instance [4-6] and the references therein. If the workspace consists of dynamic entities, such as varying event locations and dynamic obstacles, the robots need to respond to events reactively, which require a well-designed perception and decision mechanism on the robots. In such cases distributed and decentralized algorithms possess critical benefits over the centralized approaches [4], [7,8]. Unlike centralized approaches, decentralized approaches do not require all-to-all robot communication or complete sensing mechanism, resulting in a stable real-time implementation and seamless integration with fewer sensing units. Therefore, distributed and decentralized algorithms can be utilized to achieve coverage control in dynamic settings efficiently.

Game theory provides a powerful framework for addressing coverage control problems in a distributed and decentralized fashion. Game theoretical models allow us to formally analyze the interactions between robots, and to design control algorithms that can balance conflicting objectives such as maximizing coverage while minimizing the number of robots used [7-9]. Recently, there has been significant interest in using game theoretical approaches to address coverage control problems in service robots. One major challenge in applying game theoretical approaches to coverage control in service robots is the need to model the robots' behavior in a way that reflects their physical

* Sorumlu yazar / Corresponding author, e-posta / e-mail: samet.guler@agu.edu.tr (S. Güler)
Geliş / Recieved: 05.05.2023 Kabul / Accepted: 17.09.2023 Yayımlanma / Published: 15.10.2023
doi: 10.28948/ngmuh.1293191

capabilities and limitations. Service robots have a wide range of capabilities, ranging from simple robotic arms that perform repetitive tasks to highly sophisticated autonomous systems that can navigate complex environments. To design effective coverage control algorithms for these systems, it is important to accurately model their capabilities and limitations, including their sensing, communication, and mobility capabilities [4-9]. Another important challenge in game theoretical coverage control for service robots is to design algorithms that can effectively handle dynamic environments. To be effective, coverage control algorithms must be able to adapt to these changing environments, considering the changing distribution of tasks and the changing availability of resources [7,8]. Additionally, it is important to consider the scalability of coverage control algorithms, as they will often be applied to large-scale systems with many robots operating in parallel.

Potential game refers to a type of game where the collective behavior of the players can be related to a potential function which is aligned with the unilateral deviation of a player's utility when the other players maintain their actions [9]. In a multi-agent system, since the potential function can be associated with the collective objective of the system, each agent's utility can be designed in such a way that favors increase of the potential, resulting in the emergence of the desired collective behavior [10]. Potential games have been successfully applied to several engineering problems, including unmanned aerial vehicle (UAV) search-and-rescue [10], power control in wireless networks [11], and distributed optimization problems [12]. Recently, state-based potential games have introduced an additional degree-of-freedom in the design process, thus enabling us to solve challenging engineering constraints at the design stage [13]. For instance, state-based potential games are designed for 2D area coverage in [14-17], where the trade-off between the energy consumption of the agents and the covered area is formulated with a game state. Several modifications to the original coverage algorithms aim to improve the convergence rate in unknown environments [18].

Many learning algorithms have been introduced to complete the design process of multi-player potential games, such as the binary log-linear learning (BLLL) algorithm and the better reply processes [6], [13]. Remarkably, some learning algorithms ensure the attainment of the maxima of the potential function in the steady state, i.e., as time approaches infinity [19]. Although the short-term behavior of these algorithms lacks theoretical justifications, this asymptotic behavior can be well-applied to solve many engineering problems. It is worth emphasizing that most learning algorithms allow designing of player strategies in a distributed and decentralized fashion, enabling real-world implementations.

We consider the coverage control of service robots (e.g., mobile manipulators) in a typical industrial application scenario. The robots are tasked with responding to events which can occur on a workstation sporadically, by arriving at certain locations on both sides of the event location. In particular, the workstation consists of a linear track with finite event locations. This scenario is motivated by a variety of application examples, ranging from the object loading/unloading task by collaborating mobile manipulators and drones to the task of event monitoring by mobile ground robots. We tackle the problem by formulating it as a game played by mobile robots, where the utilities are determined based on the achievement of the responses to the events. We show that the designed game constitutes a potential game, where the potential function corresponds to the total success of the robots and is aligned with the individual robot utilities. Subsequently, we design a BLLL algorithm among the robots. Finally, we discuss possible practical constraints that can appear in real-world application of the proposed algorithm and modifications to the algorithm to solve these challenges. Several simulation results demonstrate the effectiveness of the approach and the effect of using different parameter values on the performance.

The paper is organized as follows. In Section 2, we provide the system formulation and the method, providing a brief background about game theory and notations. In Section 3, we give the simulation results. Finally, Section 4 is on conclusions.

2 Material and method

In this section, we present the problem formulation and the solution method. We start by defining a generic multi-robot system (MRS) considered in abstract terms. Then, we give the main objective of the paper. Finally, the game theory-based solution method is introduced.

2.1 System modeling

We consider a workspace W, a subset of the two-dimensional (2D) Cartesian plane $\Re^2$, with a fixed global frame Σ_G (Figure 1). Assume that there exist three linear parallel tracks with finite lengths on the workspace W, named W_C, W_L, and W_R, where the subscripts C, L, R denote the center, the left, and the right tracks. Each track forms a line segment lying along the Σ_G^y-axis of the frame Σ_G at a certain location (Figure 1). We denote the Σ_G^x-axis and the Σ_G^y-axis by the lateral axis and the longitudinal axis, respectively.

On the center workspace W_C, a set of events may occur at certain locations which are located on a set of finite number of waypoints $\{w_i^C\}, i \in \{1, \dots, K\}, K \geq 3$, where $w_i^C = [w_{i,x}^C, w_{i,y}^C]^\top \in \Re^2$. For instance, in an industrial application scenario, these events can include workstations requiring loading/unloading operations, or human operators waiting for mobile robot service. The events are detected by a set of sensors, such as cameras and LIDARs, which are mounted underneath a set of vertical-take-off-and-landing (VTOL) drones. Therefore, we consider N number of drones $\mathcal{D}_i, i = \{1, \dots, N\}, 1 < N < K$. In essence, a VTOL drone agent can move linearly in and rotate around the three Cartesian axes (x, y, z) in space, which results in six degree-of-freedom (6-DOF) motion model. For convenience, we assume that the low-level controller of a drone stabilizes its altitude, x-axis motion, and the roll, pitch, and yaw angles. Hence, a drone can move linearly on the waypoints $w_i^C, i \in \{1, \dots, K\}$ along the workspace W_C at a constant altitude. We assume that the drones can move fast and precisely between

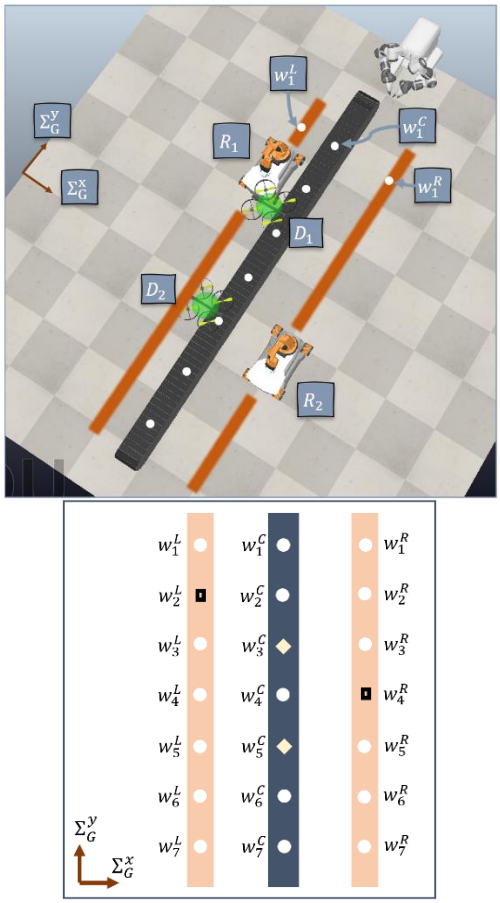

Figure 1. (Top) A sample workplace environment for $N = 2, K = 7$. The drones represent the events that occur on the conveyor belt by moving along the indicated axis on the belt. The UGVs aim to spend most of their operation time around these locations by moving on the orange-colored tracks on both sides of the belt. The waypoints w_1^C, w_1^L, w_1^R are shown as example. Drone $\mathcal{D}_2$ is covered by UGV R_2, while drone $\mathcal{D}_1$ is not covered by any UGV. (Bottom) The 2D representation of the workspace given. White dots denote the waypoints, black rectangles denote UGVs, and the yellow squares denote the drones.

the waypoints, and their motion dynamics can be ignored, i.e., the presence of a drone at a waypoint w_i^C at a time step t indicates that an event occurred at w_i at that time step. Denote the position of drone $\mathcal{D}_i$ on the longitudinal axis by $y_i^D, i \in \{1, \dots, N\}$.

A set of unmanned ground vehicles (UGVs) $R_i, i = \{1, \dots, N\}$, aim to serve the events that occur on the workspace W_C by moving along the workspaces W_L, W_R. For convenience, we impose that $\lceil N/2 \rceil$ number of UGVs lie on the track W_L, and the rest of the UGVs lie on the track W_R, where $\lceil \cdot \rceil$ is the ceiling function. Denote the sets which contain the UGVs on the track W_L and the UGVs on the track W_R by $\mathcal{S}_L$ and $\mathcal{S}_R$, respectively. On the workspaces W_L, W_R, define the sets of the waypoints $\{w_k^L\}$ and $\{w_k^R\}, k \in \{1, \dots, K\}$, where $w_k^L = \left[w_{k,x}^L, w_{k,y}^L\right]^\top \in \Re^2$ and $w_k^R = \left[w_{k,x}^R, w_{k,y}^R\right]^\top \in \Re^2$. The waypoints have the property that $w_{k,y}^C = w_{k,y}^L = w_{k,y}^R, k \in \{1, \dots, K\}$, i.e., the waypoints w_k^C, w_k^L, and w_k^R reside on the same lateral axis for each $k \in \{1, \dots, K\}$ (Figure 1). We say that at any time step t, UGV R_i engages with (or serves) drone $\mathcal{D}_j$ if drone $\mathcal{D}_j$ is at the waypoint w_k^C, and UGV R_i is at one of the waypoints w_k^L, w_k^R. Evidently, a drone (or event) can be covered by at most two UGVs, one on W_L and one on W_R.

We denote by y^i the y-axis position of $R_i, i = \{1, \dots, N\}$. We assume that the UGVs can move on their longitudinal axes precisely, i.e., their low-level control mechanisms always keep them on the linear tracks W_L, W_R by controlling their lateral motions and heading angles. Define $t \in \{0,1, \dots, T_f\}$ as the discrete time index, where T_f is the final time. We assume that the UGV motions on the longitudinal axes take place between two consecutive time steps. That is, if a UGV R_i starts moving from a waypoint at a time step t_0, then it can reach its destination waypoint until the next time step $t_0 + 1$.

We assume that the UGVs and the drones are equipped with ultrawideband (UWB) sensors for distance measurement and inter-robot communication. For localization purposes, each follower UGV is also equipped with other sensors, such as monocular/depth cameras. In this work, assuming that the UGV localization is achieved by another control layer with sufficient precision, we focus on the task allocation of the UGVs in the following part.

2.2 *Objective*

The main goal is to design a path planning algorithm for the UGV team so that they spend most of their operation time next to the drones' locations. This goal is relevant to several industrial applications; for instance, the UGVs can collaborate with other robot manipulators in achieving a common task at the event location, such as object loading/unloading operation. As an illustration, consider Figure 1-top, where two drones are hovering on top of detected events to broadcast event locations, and two UGVs are aiming to cover the drones by moving to the neighborhoods of the drones on both sides. Here, the term coverage is interpreted as moving the UGVs to the drones' locations laterally. Notably, the UGVs are restricted to move on the workspaces W_L, W_R along the longitudinal axis of the global frame Σ_G (the orange tracks in Figure 1).

It is desired that the drones $\mathcal{D}_i, i = \{1, \dots, N\}$, are served by the UGVs $R_j, j = \{1, \dots, N\}$, where the horizontal location of a UGV does not affect its efficiency, i.e., the UGVs $R_i \in \mathcal{S}_L$ and the UGVs $R_i \in \mathcal{S}_R$ can serve at the drone event with the same efficiency. However, if a drone is covered by a UGV from either the left track W_L or the right track W_R, the involvement of a second UGV with the same drone contributes to the total efficiency less than the reward gained by the involvement of the first UGV. Accordingly, define the utility $E_i(\mathrm{Y}[t], \mathrm{Y}^\mathrm{D})$ at time step t for drone $\mathcal{D}_i$, where $\mathrm{Y}^\mathrm{D} = [y_1^D, \dots, y_N^D]^\top, \mathrm{Y} = [y_1, \dots, y_N]^\top$, with the following rules:

i) $E_i(\mathrm{Y}[t], \mathrm{Y}^\mathrm{D}) = 0$ if drone $\mathcal{D}_i$ is not covered by any UGV,

ii) $E_i(\mathrm{Y}[t], \mathrm{Y}^\mathrm{D}) = p$ if drone $\mathcal{D}_i$ is covered by two UGVs,

iii) $E_i(\mathrm{Y}[t], \mathrm{Y}^{\mathrm{D}}) = 2p$ if drone $\mathcal{D}_i$ is covered by one UGV.

Here, $p > 0$ denotes the reward gained by engagement of a UGV with a drone and can take any value.

Furthermore, UGV motions should be scheduled so that no two UGVs collide with each other, i.e., at any time step t it must be satisfied that $y_i[t] \neq y_j[t]$ for all $i, j \in \{1, \dots, N\}, i \neq j$. Considering these requirements, the main goal is summarized as follows:

Objective 1: Given the MRS defined thus far, design a reactive path planning algorithm for each UGV $R_i, i = \{1, \dots, N\}$, to solve the following optimization problem:

$$\max_{\mathrm{Y}} \sum_{i=1}^{N} E_i(\mathrm{Y}, \mathrm{Y}^{\mathrm{D}}) \quad \text{s.t. } y_i \neq y_j \text{ for all } i, j \in \{1, \dots, N\}, i \neq j. \tag{1}$$

The objective function in Equation (1) aims to maximize the coverage of the drones such that each drone is desired to be covered by one UGV only while avoiding collisions among the UGVs. Thus, the maximum value corresponds to the case that N drones are covered by N distinct UGVs. An important property of this optimization objective is that the UGV identities remain anonymous, i.e., identities of the UGVs do not affect the objective function's value. Accordingly, the UGV allocations that lead to the maximum value in Equation (1) is not unique. We examine the optimality of the UGV allocations in Section 2.3 in detail.

2.3 The proposed approach

A common approach for addressing Objective 1 involves designing an optimization-based path planning framework among the UGVs. This method can be achieved by having the UGVs create an optimal path planning algorithm when the drones are at rest, and then adjusting the optimization process as the drones begin to move. This two-step process can be repeated continuously to accommodate ongoing operations. However, such methods usually require a centralized computational unit which acquires the real-time information from all agents, performs the computation, and broadcasts the solution to all agents. This structure would require a very large communication bandwidth and long-range sensory devices.

To address the challenges that can be faced with centralized approaches, we opt for a distributed and decentralized solution. Our approach is composed of two layers: Game design and learning algorithm design.

2.3.1 Game design

To overcome the deficiencies imposed by a centralized approach, we model Objective 1 as a game played by the UGVs. We design a game-based planning method because it allows the UGVs to strategically position themselves around the drones through a well-designed game model. As a distributed approach, the designed game has the potential to efficiently allocate the UGVs at event locations.

A hypothetical game consists of three components: players (who make decisions), actions (from a defined set of actions), and utilities (rewards). In a repeated game, each player selects an action from its action set at each time step based on its utility evaluation. At the end of the selection procedure, each player receives a reward, referred to as utility. The repetition of this process over time constitutes a repeated game.

Potential games refer to a particular type of game where the strategies of each player correspond to a potential function that aligns with the changes in every player's strategy. The potential function represents the overall satisfaction of the players and can be used as a design guideline in several multi-agent system objectives, such as coverage and resource allocation. We start by defining these concepts formally.

Definition 1 (Game model): A game consists of a set of players ($\mathcal{P}_i$), a set of actions ($a_i \in \mathcal{A}_i$), and the utilities (U_i) where $i \in \{1, \dots, N\}$.

Let $a_{-\mathrm{i}} = (a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N)$ denote the actions of all players except player i, where $a_i \in \mathcal{A}_i$, with $\mathcal{A}_i$ denoting the action set of player $a_i, i \in \{1, \dots, N\}$. Also, let $(a_i, a_{-i}) \in \mathcal{A}$ represent the joint actions of the players, and $\mathcal{A} = \mathcal{A}_1 \times \dots \times \mathcal{A}_{\mathrm{N}}$ denote the set of joint actions. A fictitious potential game is defined as follows:

Definition 2 (Potential Game): Define $\phi(\mathrm{a}): \mathcal{A} \to \Re$ as the potential function assigned for the action set $\mathrm{a} = \{a_1, \dots, a_N\}$. An exact potential game satisfies the following condition:

$$\phi(a_i', a_{-i}) - \phi(a_i'', a_{-i}) = U_i(a_i', a_{-i}) - U_i(a_i'', a_{-i}) \tag{2}$$

for all $a_i', a_i'' \in \mathcal{A}_i,\ a_i' \neq a_i''$, and for all $i \in \{1, \dots, N\}$.

An important connection between potential games and coordination control of MRS is Nash equilibrium, which is defined as follows:

Definition 3 (Nash Equilibrium): Consider a game with players ($\mathcal{P}_i$), a set of actions ($a_i \in \mathcal{A}_i$), and the utilities (U_i) where $i \in \{1, \dots, N\}$. An action profile a^* is called a pure Nash equilibrium if

$$U_i(a_i^*, a_{-i}^*) = \max_{a_i \in \mathcal{A}_i} U_i(a_i, a_{-i}^*), \tag{3}$$

for all $i \in \{1, \dots, N\}$.

Potential games have notable advantages in formulating MRS objectives. The goal of MRS can be expressed through a potential function, allowing for player strategy profiles to be optimized for maximizing this function in a decentralized fashion. Therefore, we aim to design a potential game to solve Objective 1 in the following.

We design a game with the following elements:

- Players: The UGVs $R_i, i = \{1, \dots, N\}$.
- Actions: The coordinates of the waypoints $w_k^L \in \Re^2, k \in \{1, \dots, K\}$, for the UGVs in the set $\mathcal{S}_L$, and the coordinates of the waypoints $w_k^R \in \Re^2, k \in \{1, \dots, K\}$, for the UGVs in the set $\mathcal{S}_R$.
- Utilities: In the following, the utility for a UGV R_i is designed to align with the ultimate objective, which is the maximization of the summation of $E_i(\mathrm{Y}, \mathrm{Y}^{\mathrm{D}})$.

We now design the utilities of the players. Denoting the difference in the longitudinal positions of UGV R_i and drone $\mathcal{D}_j$ by $\zeta_{ij} = \left|y_i - y_j^D\right|, i,j \in \{1,\dots,N\}$, we propose to use the following utility for UGV R_i, in compliance with Objective 1:

$$U_i(a[t]) = \sum_{j=1}^{N} U_i^j(\mathrm{a}[t]), \tag{4}$$

where $U_i^j(\mathrm{a}[t])$ denotes the share of UGV R_i for engaging with drone $\mathcal{D}_j$ for the action set $\mathrm{a}[t]$ at time step t, defined by

$$U_i^j(\mathrm{a}[t]) = \begin{cases} \bar{\mathrm{u}}, \text{if } \zeta_{ij}[t] = 0 \text{ and } \eta_j^i[t] = 0, \\ \dfrac{\bar{\mathrm{u}}}{2}, \text{ if } \zeta_{ij}[t] = 0 \text{ and } \eta_j^i[t] = 1, \\ 0, \qquad \text{if } \zeta_{ij}[t] \neq 0, \end{cases} \tag{5}$$

for $i,j \in \{1,\dots,N\}$. Here, $\eta_j^i[t] \in \{0,1\}$, denotes whether drone $\mathcal{D}_j$ is already covered by a UGV and is defined by:

$$\eta_j^i[t] = \begin{cases} 1, \text{ if } \exists k \in \{1,\dots,N\}, k \neq i, k \neq j, \zeta_{kj}[t] = 0 \\ 0, \qquad \text{o. w.} \end{cases} \tag{6}$$

where $|\mathrm{S}|$ denotes the cardinality of set S. Therefore, the utility $U_i^j(\mathrm{a}[\mathrm{t}])$ denotes the share of a UGV R_i from covering a target $\mathcal{D}_j$. More precisely, if a drone $\mathcal{D}_j$ is not covered by a UGV except R_i, then R_i, which satisfies $\zeta_{ij} = 0$, will get the reward $\bar{\mathrm{u}}$; otherwise, UGV R_i's reward will be divided by half to reduce the incentive to choosing a target which was already occupied by another UGV. The following proposition summarizes the properties of the game designed above.

Proposition 1: The game structure defined by the players R_i, the actions $a_i[t] \in \mathcal{A}_i$, and the utilities U_i as described in this section constitutes a potential game with the potential function:

$$\phi(\mathrm{a}) = \sum_{\substack{j=1 \\ \mathcal{D}_j \in \mathcal{D}^*}}^{N} \sum_{i=1}^{n_j} \frac{\bar{\mathrm{u}}}{i}, \tag{7}$$

where $\mathcal{D}^*$ is the set of all drones covered by at least one UGV at time t, and $n_j \in \{1,2\}$ is the number of the UGVs which engage with drone $\mathcal{D}_j$. Therefore, the designed game has at least one pure Nash equilibrium.

Proof: To see that Equation (2) holds with $\phi(\mathrm{a})$ of Equation (7) for all $a_i', a_i'' \in \mathcal{A}_i, i \in \{1,\dots,n\}$ and all $\mathcal{D}_j, j \in \{1,\dots,N\}$, consider a drone $\mathcal{D}_j$ which is already covered by a UGV R_i whose utility is $\bar{\mathrm{u}}$ at time t. In this case, the potential due to drone $\mathcal{D}_j$ is $\phi_j(\mathrm{a}[t]) = \bar{\mathrm{u}}$.

If a new UGV $R_k, k \neq i$, attempts to cover drone $\mathcal{D}_j$ as well at the next time step $t+1$, then there are two cases. The first is that UGV R_k was not engaged with another drone at time t. In this case, since drone $\mathcal{D}_j$ was already covered by UGV R_i, the shares of UGVs R_i and R_k would be $U_i(a[t+1]) = U_k(a[t+1]) = \frac{\bar{\mathrm{u}}}{2}$, while the potential becomes $\phi(\mathrm{a}[t+1]) = \frac{3\bar{\mathrm{u}}}{2}$. Thus,

$$\begin{aligned} \phi(\mathrm{a}[t+1]) - \phi(\mathrm{a}[t]) &= \frac{\bar{\mathrm{u}}}{2} \\ &= U_k(\mathrm{a}[t+1]) - U_k(\mathrm{a}[t]). \end{aligned} \tag{8}$$

The second case is that UGV R_k was engaged with another drone $\mathcal{D}_l$ at time t. In this case, there are two chances: (i) Drone $\mathcal{D}_l$ was covered by another UGV and (ii) drone $\mathcal{D}_l$ was not covered by another UGV. In the first case, $U_k(\mathrm{a}[t+1]) = U_k(\mathrm{a}[t])$, and the potential does not change, $\phi(\mathrm{a}[t+1]) = \phi(\mathrm{a}[t])$. In the second case, $U_k(\mathrm{a}[t+1]) - U_k(\mathrm{a}[t]) = \frac{-\bar{\mathrm{u}}}{2}$, and it can be shown to be equal to the change in the potential, i.e., $\phi(\mathrm{a}[t+1]) - \phi(\mathrm{a}[t]) = \frac{-\bar{\mathrm{u}}}{2}$. Thus, the conditions of the potential game are satisfied for the cases considered above. It can be shown for the other scenarios that the change in the utility U_k is always the same with the change in the potential ϕ. Therefore, the game is a potential game with the potential function in Equation (7). Since a potential game has at least one pure Nash equilibrium as stated in [4], the designed game has this property, as well. This completes the proof.

Remark 1: Since the UGV identity does not make any difference in the calculation of the target utilities U^j and the potential ϕ, the game allows anonymous allocation of the UGVs. The utility design in Equation (4) is called equally shared utility (ESU).

It is established in potential game theory that the actions that maximize the potential function lead to Nash equilibrium, and the players (UGVs) tend to choose those actions in steady state if suitable learning algorithms are used. In our problem setup, this means that the UGV team operate most of the time at equilibrium actions. The following result characterizes the Nash equilibria for the designed game.

Proposition 2: A pure Nash equilibrium maximizes the potential $\phi(\mathrm{a})$ and corresponds to a distinct allocation of the UGVs, i.e., a Nash equilibrium is formed when N drones are covered by N distinct UGVs. The potential corresponding to a pure Nash equilibrium is $\phi(\mathrm{a}^*) = N\bar{\mathrm{u}}$. Furthermore, multiple Nash equilibria exist.

Proof: A Nash equilibrium is an equilibrium state where no UGV wants to change its action unilaterally. It is stated in [4], [19] that the actions that maximize the potential $\phi(\mathrm{a})$ are the pure Nash equilibria of the game. It is evident that $\phi(\mathrm{a}^*) = N\bar{\mathrm{u}}$, and the corresponding actions are such that each UGV covers a drone because in any other case the potential $\phi(\mathrm{a}) < N\bar{\mathrm{u}}$ due to the definition of the utilities in Equation (4). For instance, if two UGVs are covering one drone while the other $(N-2)$ UGVs are covering the remaining $(N-2)$ drones, then $\phi(\mathrm{a}) = (N-0.5)\bar{\mathrm{u}}$.

Since $N \geq 2$, the UGVs can be positioned around the drones in several distinct configurations. As an example,

consider the $N = 2$ case in Figure 2, where two drones are located at the waypoints w_2^C and w_4^C, while the waypoint w_3^C is empty. The UGVs R_1, R_2 can be positioned at the waypoints w_2^L and w_4^R as a Nash equilibrium. Alternatively, they can choose the waypoints w_4^L and w_2^R as a Nash equilibrium. Both allocations maximize the potential, resulting in $\phi(\mathrm{a}^*) = 2\bar{\mathrm{u}}$. Therefore, multiple Nash equilibria exist based on the UGV allocations. Obviously, the number of distinct Nash equilibrium action profiles a^* increases as N increases. This completes the proof.

In the following part, we design a learning algorithm that allocates the UGVs in real time.

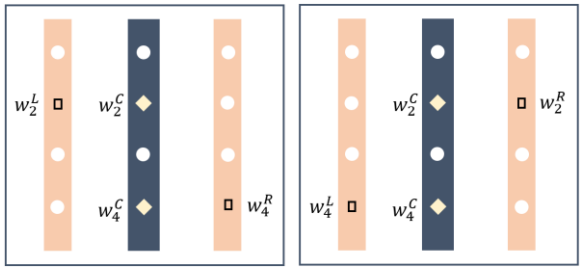

Figure 2. Illustration of two separate UGV allocations corresponding to two Nash equilibria for the same drone setting. White dots denote the waypoints, black rectangles denote UGVs, and the yellow squares denote the drones.

2.3.2 *Learning algorithm design*

In an exact potential game where player synchrony is satisfied, the Binary Log-Linear Learning (BLLL) algorithm can be used in a decentralized manner. In this part, we describe an application of the BLLL algorithm for our objective.

At each time step t, a UGV R_i is selected uniformly randomly from the players set. Then, UGV R_i decides on its action from its action set $\mathcal{A}_i$. To operate the game within a defined workspace by satisfying the collision avoidance requirement of Objective 1, we employ the concept of the constrained action sets $\mathcal{C}_i(a_i(t-1))$ for the one-step motion primitives of the UGVs.

At any time t, the constrained action set $\mathcal{C}_i(a_i(t-1))$ for a UGV R_i defines the allowable locations which both reside within the workspace and are collision-free. Assume that UGV $R_i \in \mathcal{S}_L$ resides on the waypoint $w_k^L, k \in \{1, \dots, K\}$ at time step $t-1$. We construct the constrained action set $\mathcal{C}_i(a_i(t-1))$ of UGV R_i as follows. Since R_i is allowed to move to only its neighbor waypoints w_s^L, $s \in \{k-1, k, k+1\}$, it can choose among three motion primitives $\{w_{k-1}^L, w_k^L, w_{k+1}^L\}$. If one of the waypoints w_{k-1}^L, w_{k+1}^L is occluded by another UGV or remains outside of the workspace W_L, then that waypoint is discarded from the constrained action set of R_i. Thus, $\mathcal{C}_i(a_i(t-1)) \subseteq \{w_{k-1}^L, w_k^L, w_{k+1}^L\}$. The same design procedure is used for the UGVs in the set $\mathcal{S}_R$ by replacing the subscripts and superscripts L by R. As an example, consider the setting in Figure 3. The constrained action set of UGV R_1 is $\mathcal{C}_1(a_1(t-1)) = \{w_1^L, w_2^L, w_3^L\}$ because UGV R_1 does not have any neighbor UGV, and it can move to one of its neighbor waypoints or stay at its current location at the next time step. The constrained action set of UGV R_2 is $\mathcal{C}_2(a_2(t-1)) = \{w_1^R\}$ because it resides at the edge of the workspace W_L, and its neighbor waypoint w_2^R is occluded by UGV R_3. Thus, the only feasible action for UGV R_2 at time step t is its current location. Similarly, $\mathcal{C}_3(a_3(t-1)) = \{w_2^R, w_3^R\}$.

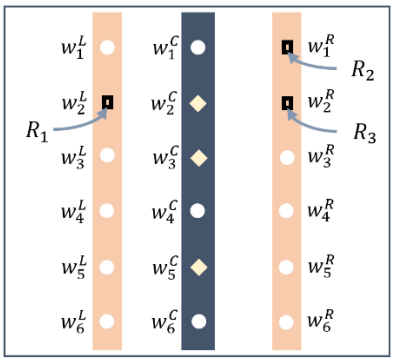

Figure 3. A sample workspace for the explanation of the constrained action sets.

After establishing the action set, robot R_i chooses its tentative action $\hat{a}_\mathrm{i}$ from the set $\mathcal{C}_i(a_i(t-1))$ using the following strategy:

1. $\pi(\hat{a}_\mathrm{i}) = 1/3$ for $\hat{a}_\mathrm{i} \in \mathcal{C}_i(a_i(t-1))$,
2. $\pi(a_i(t-1)) = 1 - \left(\left|\mathcal{C}_i(a_i(t-1))\right| - 1\right)/3$,

where $\pi(a_\mathrm{i})$ denotes the probability of choosing the action a_i. Notably, if all neighbor locations are discarded (due to occlusion by another UGV or violation of the workspace boundary condition), then $\hat{a}_i(t) = a_\mathrm{i}(t-1)$. Afterwards, R_i moves to $\hat{a}_\mathrm{i}(t)$ with the following strategy:

$$\begin{aligned}
\pi(\hat{a}_i(t)) &= \frac{e^{\beta U_i(\hat{a}_i(t), a_{-i}(t-1))}}{e^{\beta U_i(\hat{a}_i(t), a_{-i}(t-1))} + e^{\beta U_i(\mathrm{a}(t-1))}}, \\
\pi(a_\mathrm{i}(t-1)) &= \frac{e^{\beta U_\mathrm{i}(\mathrm{a}(t-1))}}{e^{\beta U_\mathrm{i}(\hat{a}_\mathrm{i}(t), a_{-\mathrm{i}}(t-1))} + e^{\beta U_\mathrm{i}(\mathrm{a}(t-1))}}, \\
\pi(a_\mathrm{i}) &= 0 \quad \forall a_\mathrm{i} \in \mathcal{C}_\mathrm{i}(a_\mathrm{i}(t-1)) \backslash \{a_\mathrm{i}(t-1), \hat{a}_\mathrm{i}(t)\},
\end{aligned} \tag{9}$$

where $\beta \geq 0$ is the so-called forgetting factor which adjusts UGV R_i's tendency to choose a suboptimal solution. Notably, as β approaches 0, the UGV tends to select its tentative action $\hat{a}_i(t)$ or its current action $a_i(t-1)$ with equal probability, i.e., $\pi(\hat{a}_i(t)) = \pi(a_i(t-1)) = 1/2$. As β goes to infinity, the UGVs tend to select the optimal action a^* with arbitrarily high probability.

Remark 2: The structure of the players' strategy in Equation (9) is designed by observing the form of the stochastically stable actions of the game. Particularly, if certain conditions are satisfied by the constrained action sets $\mathcal{C}_i(a_i(t-1)), i \in \{1, \dots, N\}$, then the evolution of the game induces a Markov chain with a unique stationary distribution characterized by the potential function $\phi(\mathrm{a})$ [4]. Notably, the

form of Equation (9) is common for finite-player distributed resource allocation games; see e.g., [4], [7], [8], [13-19].

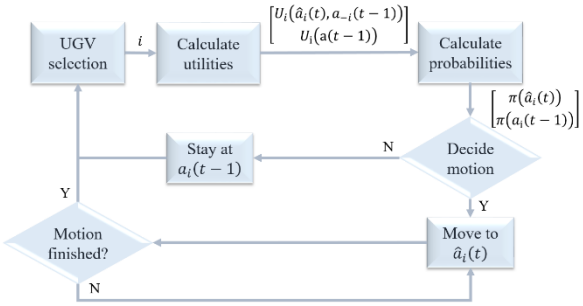

Figure 4. Flow diagram of the proposed BLLL algorithm

A flow diagram of the proposed BLLL algorithm is given in Figure 4. To start the process at each time step t, a UGV $R_i, i = \{1, \dots, N\}$, is chosen uniformly randomly. Then, UGV R_i chooses an action $\hat{a}_i$ from its constrained action set $\mathcal{C}_i(a_i(t-1))$ and calculates the probabilities $\pi(\hat{a}_i)$, $\pi(a_i(t-1))$ based on the utilities $U_i(\hat{a}_i(t), a_{-i}(t-1))$, $U_i(a(t-1))$. Finally, it decides on its action, i.e., decides whether to move to $\hat{a}_i$ or stay at $a_i(t-1)$ based on the probability distribution defined by $\pi(\hat{a}_i)$, $\pi(a_i(t-1))$. If R_i chooses to move to $\hat{a}_i$, then the decision process stops and waits for the UGV to finish its motion. This process is repeated for each time step, leading to a repetitive game.

The main advantage of the proposed game theoretical framework lies in its practical simplicity. The entire framework is comprised of two steps: Player selection and probability distribution calculation by the selected player. The player selection part is a decision mechanism from a uniform distribution, and thus it possesses no computational burden. In the second step, the selected player first calculates the probability distribution based on the utilities and then chooses to whether it should move to $\hat{a}_i(t)$ or stay at the current location. Notably, the required information for this step is the constrained action set $\mathcal{C}_i(a_i(t-1))$ and parameter $\eta_j^i[t]$, which can be obtained from the neighbors of the players. Thus, the required information can be acquired by sensing/communication modules with limited range onboard of the UGVs. This property makes the proposed distributed framework favorable compared with centralized approaches.

If the constrained action sets of the players satisfy the feasibility and reversibility conditions defined in [4] for all time steps, then the BLLL algorithm enables the UGVs to choose actions so that they spend most of their operation time at optimal locations. This behavior is also referred to as asymptotic behavior because the UGVs tend to choose optimal actions for sufficiently large β as time goes to infinity. However, since the constrained action sets are designed to satisfy the collision avoidance requirement of Objective 1, the conditions in [4] may not be satisfied for all time steps. Nevertheless, the BLLL algorithm showed near optimal performance in our simulations and resulted in significant increase in the potential value $\phi(a)$ in the first few time steps.

3 Results and discussion

This section presents the evaluation results of the proposed approach and a detailed discussion about the results.

3.1 Simulation setup

Several MATLAB simulations were performed for the derived fictitious game. The simulations used ideal conditions where the UGV kinematics are ignored, and the UGVs were assumed to move from a waypoint to a destination waypoint between two consecutive time steps. A fictitious game consisting of ten drones and ten UGVs ($N = 10$) was simulated, and the utilities in Equation (4) and (5) were used with $\bar{u} = 2$. Notably, the steady-state properties of the algorithm are independent from the number of robots, but the transient characteristics may alter based on the number of UGVs and their initial conditions. The boundary conditions were taken as $y_{\min} = -10$, $y_{\max} = 12$ m, i.e., the UGVs moved within the boundary $y_i[t] \in [-10,12]$ m for all $t \in \{0,1, \dots, T_f\}$. The x-axis separation of the drones and the UGVs is taken as $|x^i[t] - x_D^i[t]| = 2$ m on both sides of the drones, noting that this parameter does not affect the algorithm's performance. The drones were kept stationary at the positions $\mathrm{p}_i^D[k] = [0 \quad y_i^D]^\mathsf{T}$ m, where y_i^D ranged from -8 to 10 m with two-meter separation between two consecutive drones. A sample simulation configuration is given Figure 5.

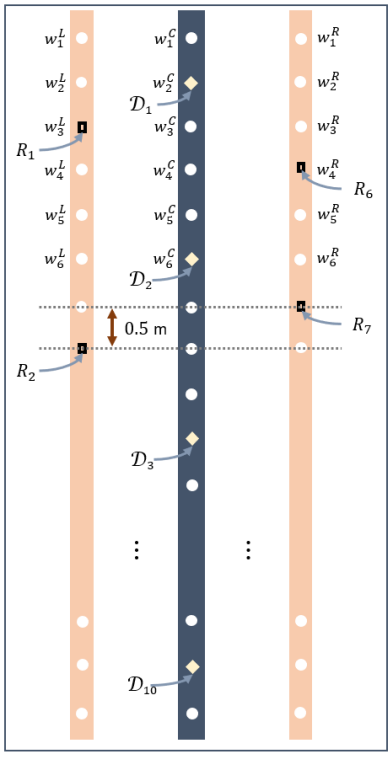

Figure 5. A sample initial configuration used in simulations. No drone was covered by a UGV at the initial time step, resulting in $\phi(a(0)) = 0$.

The UGVs were initiated at random locations with $x_i[0] = \pm 2$ m and $y_i[0]$ varying between $y_{\min}$ and $y_{\max}$. Remarkably, the UGVs were not engaged with any drone initially, as seen in Figure 5. The pure Nash equilibria for this setting results in the maximum potential $\phi_{\text{opt}} = 10\bar{u} = 20$, which corresponds to the case that each UGV engages with different drones. If two UGVs engage with one drone, then we obtain $\phi(a) < 10\bar{u}$. For instance, if nine UGVs engage with nine separate events, and the remaining UGV engages with an event that was already covered by another UGV, then the potential would be $\phi_{\text{sub}} = 9.5\bar{u}$. Particularly, such cases occur when some UGVs engage with the same event, and at least one waypoint exists between two consecutive drones. For instance, consider the drone placement given in Figure 5, where drone $\mathcal{D}_1$ is at waypoint w_2^C, and the neighbor waypoints w_1^C, w_3^C do not include any drone. If two UGVs, say R_1 and R_6 cover drone $\mathcal{D}_1$, at a future time step, they would not want to leave the waypoint w_2^C because the tentative actions in their constrained action sets will not include another drone (and thus the utility of moving will be less than the utility of staying).

3.2 Results

The forgetting factor β significantly affects the UGVs' motion behavior. Thus, several experiments were conducted to analyze the effect of β on the overall performance. Remarkably, as β approaches zero, the UGVs tend to choose to move to the selected action $\hat{a}_i(t)$ or stay at current location $a_i(t-1)$ with equal probability. On the other hand, as β goes to infinity, the UGVs tend to find and stay on the actions which maximize their utilities, at the expense of leaving some events explored less. We observed that choosing $\beta > 0.9$ does not change the behavior significantly compared with $\beta = 0.9$. Thus, ten simulations were conducted for each of the β values from the set $\beta = \{0.2, 0.4, 0.6, 0.8, 0.9\}$, where each simulation was run for $T_f = 1000$ time steps. The potential function ϕ is recorded for each experiment.

Figure 6 presents the distribution of ϕ values over time such that the shaded region shows the interval between the minimum and maximum ϕ values, and the red curve shows the mean value at that time step t. It can be observed from this figure that, in all simulations, the potential ϕ increases rapidly after initialization which means that the UGVs engage with the event locations around their initial locations, as desired. Another observation is that as β approaches zero, to search for more events that are not in the vicinity of the UGVs' initial locations, the UGVs tend to choose the actions that have small utilities more frequently. This result is reflected in Figure 6 in the sense that the average potential remained around $\phi = 11$ for $\beta = 0.2$, and $\phi = 15$ for $\beta = 0.4$, whereas it could reach its maximum (i.e., $\phi = 20$) for $\beta = 0.6$ and $\beta = 0.8$ in some simulation runs. On the other hand, it was observed that the maximum ϕ value was not achieved in any simulation for $\beta = 0.9$. This result mainly stems from the fact that as β increases, a UGV tends to remain engaged to an event which maximizes its utility because the utility $\pi(\hat{a}_i(t))$ of choosing action $\hat{a}_i$ becomes quite small.

The distributions of the UGV locations in a sample run for each of the β values in the set $\beta = \{0.2, 0.4, 0.6, 0.8, 0.9\}$ are presented in Figure 7. It can be observed that when $\beta = 0.2$, to explore the maximum possible area, the UGVs move almost uniformly randomly across the y-axis.

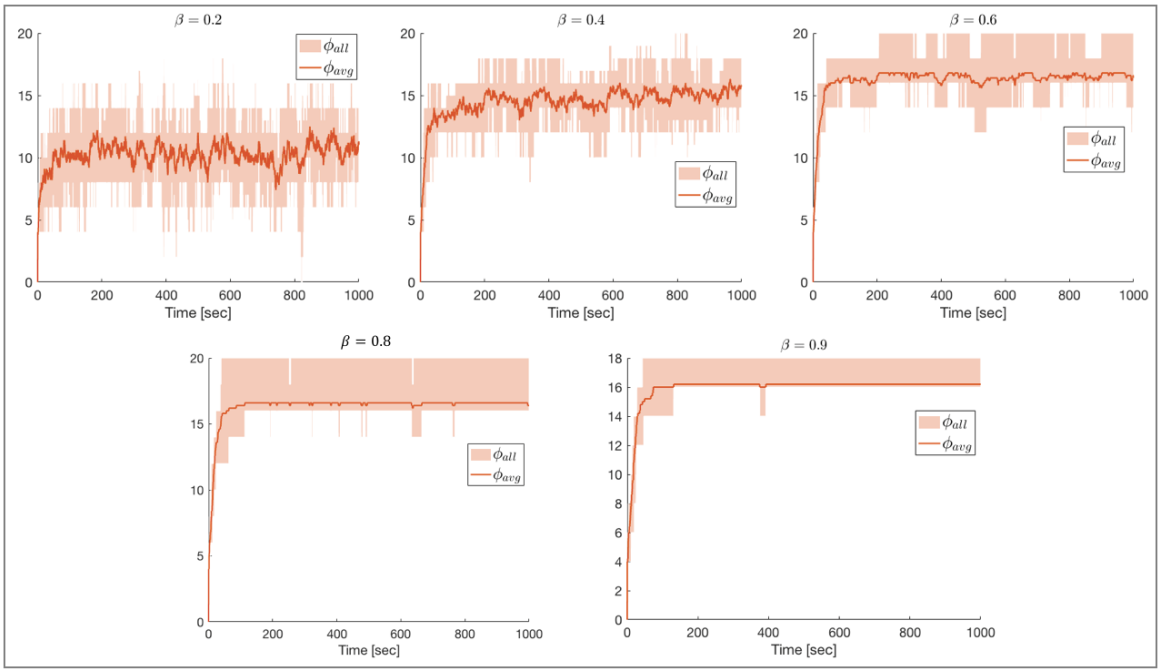

Figure 6. Potential function ϕ values over time for five β values. Each graph presents the results of 10 simulations. In each graph, the orange-colored shaded region shows the interval within the maximum and minimum ϕ values, and the red curves denote the mean ϕ values calculated at that time step.

Thus, because of the tendency to exploration, all drone locations are visited at least once by at least one UGV. As β value increases, the UGVs tend to stay at the drone locations once they are found. Therefore, the UGVs spend more time at the desired locations for high β values as compared to low β values. However, the behavior observed for high β values may also lead to missing some drones because the UGVs tend to show exploitation behavior rather than exploration. This fact can be seen in the bottom three graphs (i.e., $\beta \geq 0.6$): Although at least eight drones are covered during most of the simulation time (yellow-colored bars), some drones were covered for a short duration (dark blue colored bars). Higher β values are not illustrated for brevity because they show quite similar behavior with $\beta = 0.9$.

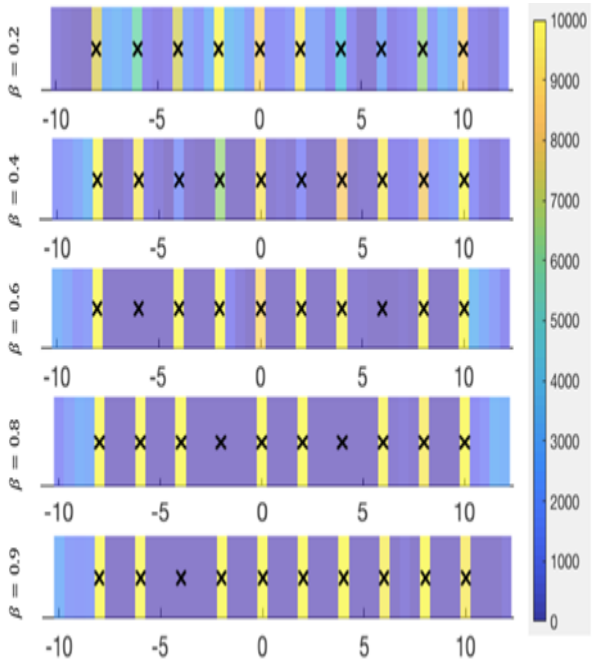

Figure 7. The positional distributions of the UGVs for five different β values. The black crosses indicate the drones' locations. Each vertical bar corresponds to a region of length of 0.5 m. The color bar on the right shows the color code which indicates the coverage density of a specific location by a UGV.

Finally, to demonstrate the adaptability of the UGVs to changing drone positions, all drones are moved to new locations at specific time steps during the simulation (Figure 8). In this test with $\beta = 0.6$, the UGVs first found a near optimal allocation ($\phi(\mathrm{a}) \geq 16$) in around 100 steps. Then, the drones were repositioned at time steps $t = 334$ and $t = 667$, where the potential diminished ($\phi(\mathrm{a}) = 0$) because no drone was covered by a UGV in the new configurations. It can be observed that the UGVs moved to the new configurations and increased the potential function swiftly right after the drones were reconfigured. Although the maximum potential was not achieved in this test ($\phi(\mathrm{a}^*) = 20$), it is worth noting that the proposed algorithm can react to the varying drone configurations swiftly.

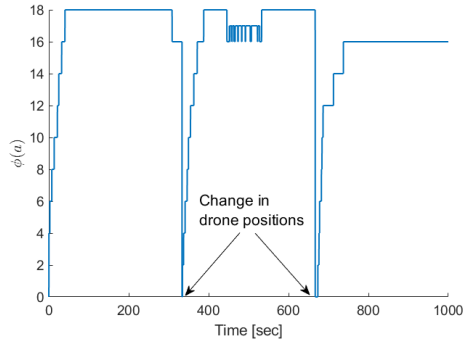

Figure 8. The potential $\phi(\mathrm{a})$ for changing drone configurations. UGVs position themselves to increase swiftly the potential for the new drone configurations.

3.3 *Practical considerations and remedies*

The original BLLL algorithm with the players' actions satisfying feasibility and reversibility conditions leads to convergence to pure Nash equilibria and potential maximization. However, as stated in Section 2, due to the modification for collision avoidance, the proposed BLLL algorithm with the proposed constrained action sets lacks this property. Furthermore, due to the physical constraints on the motion of the UGVs and possible asynchrony among the players, one may need to modify the BLLL algorithm. We discuss such modifications and their effects in this section.

While the UGVs are assumed to be synchronized in the fictitious game, and a UGV runs the game algorithm at each time step, this assumption may be difficult to satisfy in real-world applications because it requires to have a continuous and flawless communication mechanism among the UGV team. In other words, to comply with the requirement that only one player can make decision at a time step, the UGVs must communicate. As a modification, one can assume that a UGV can decide marginally without considering the other UGVs, turning the fictitious game design to a real-time distributed application. In this modified framework, each UGV $R_i, i = \{1, \dots, N\}$ can repeat its game loop once its previous action (moving to $\hat{a}_i$ or staying at current location) is completed, without obeying a common synchronized clock. In this case, the designer must modify the function in Equation (1), e.g., by increasing the collision avoidance radius of the UGVs.

Secondly, although the UGVs are assumed to move between the designated waypoints $w_k, k \in \{1, \dots, K\}$, it may not be feasible to control the UGV motion accordingly. Evidently, precise UGV motion between waypoints requires having a low-level motion control mechanism integrated with a localization module among the robots. Particularly, the nonholonomic UGV case needs special attention because the UGVs may be desired to satisfy a certain heading angle and a docking mechanism at the waypoints. Therefore, one can assume that the UGVs can choose their actions and stop when a certain condition on the distance to the new action satisfies a certain condition. This modification avoids undesired extra time required to position the UGVs at the exact action location and enables fast response to the change

in the targets' locations. We aim at addressing such modifications in our future works.

Finally, when transitioning from simulation world to real-world applications, one needs to consider the sensing and communication units onboard the UGVs. As a viable option, ultrawideband (UWB) modules can be used on drones and UGVs for both distance sensing and inter-robot communication. However, UWB modules with time-of-flight mechanism produce additive bounded noise on distance measurements, and integrating these noise effects into a repeated game changes the game structure and requires particular attention. Notably, the design of the utilities U_i^j needs to be modified to handle measurement noises. For instance, a small threshold around zero can be used in place of the condition $\zeta_{kj}[t] = 0$.

4 Conclusion

We have addressed a particular coverage problem that can arise in several industrial applications utilizing mobile unmanned ground vehicles (UGVs). Specifically, we consider a scenario where a group of UGVs are tasked with responding to sporadic events at a workplace by covering specific locations on two sides of the workplace. To address this challenge, we have formulated the objective as a coverage game with carefully designed agent utilities. Our analysis has revealed that the proposed approach constitutes a potential game with an equally shared utility design, which enables the use of common learning algorithms, such as the BLLL. Importantly, the framework has been designed to be distributed and decentralized, allowing for anonymous agent identities. Simulation results have demonstrated that the UGVs effectively operate at optimal locations where the potential function is maximized, resulting in efficient coverage of the desired areas.

Acknowledgment

This paper has been produced benefiting from the 2232 International Fellowship for Outstanding Researchers Program of TÜBİTAK (Project No: 118C348). However, the entire responsibility of the paper belongs to the owner of the paper. The financial support received from TÜBİTAK does not mean that the content of the publication is approved in a scientific sense by TÜBİTAK.

Conflict of interest

The authors declare no conflict of interest.

Similarity rate (iThenticate): 10%

References

[1] F. Chen, M. Selvaggio and D. G. Caldwell, Dexterous grasping by manipulability selection for mobile manipulator with visual guidance, IEEE Transactions on Industrial Informatics. 15 (2), 1202-1210,2019, https://doi.org/10.1109/TII.2018.2879426.

[2] G.B. Dai and Y.C. Liu, Distributed coordination and cooperation control for networked mobile manipulators, IEEE Transactions on Industrial Electronics, 64 (6), 5065-5074, 2017, https://doi.org/10.1109/TIE.2016.2642880.

[3] H. Engemann, S. Du, S. Kallweit, P. Cönen, and H. Dawar, OMNIVIL—An Autonomous mobile manipulator for flexible production. Sensors, 20 (24), 7249, https://doi.org/10.3390/s20247249.

[4] J. R. Marden, G. Arslan and J. S. Shamma, Cooperative control and potential games. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39 (6), 1393-1407, 2009, https://doi.org/10.1109/TSMCB.2009.2017273.

[5] J. Cortes, S. Martinez, T. Karatas and F. Bullo, Coverage control for mobile sensing networks. IEEE Transactions on Robotics and Automation, 20 (2), 243-255, 2004, https://doi.org/10.1109/TRA.2004.824698.

[6] M. R. Olfati-Saber, J. A. Fax and R. M. Murray, Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE, 95 (1), 215-233, 2007, https://doi.org/10.1109/JPROC.2006.887293.

[7] J. R. Marden and J. S. Shamma, Game theory and control, annual review of control. Robotics, and Autonomous Systems, 1(1), 105-134, 2018, https://doi.org/10.1146/annurev-control-060117105102.

[8] G. Arslan, J. R. Marden and J. S. Shamma, Autonomous vehicle-target assignment: a game-theoretical formulation, ASME. J. Dyn. Sys., Meas., Control, 129 (5), 584–596, 2007. https://doi.org/10.1115/1.2766722.

[9] D. Monderer, L.S. Shapley, Potential games. Games and Economic Behavior, 14 (1), 124-143, 1996, https://doi.org/10.1006/game.1996.0044.

[10] P. Li, H. Duan, A potential game approach to multiple UAV cooperative search and surveillance. Aerospace Science and Technology, 68 (1), 403-415, 2017, https://doi.org/10.1016/j.ast.2017.05.031.

[11] U.O. Candogan, I. Menache, A. Ozdaglar and P.A. Parrilo, Near-optimal power control in wireless networks: a potential game approach. 2010 Proceedings IEEE INFOCOM, 2010,1-9, San Diego, CA, USA, https://doi.org/10.1109/INFCOM.2010.5462017.

[12] P. Yi, Y. Zhang and Y. Hong, Potential game design for a class of distributed optimization problems. Journal of Control and Decision, 1 (2), 166-179, 2014, https://doi.org/10.1080/23307706.2014.899111.

[13] J. R. Marden, State based potential games. Automatica, 48 (12), 3075-3088, 2012, https://doi.org/10.1016/j.automatica.2012.08.037.

[14] S. Rahili and W. Ren, Game theory control solution for sensor coverage problem in unknown environment. 53rd IEEE Conference on Decision and Control, 1173-1178, Los Angeles, CA, USA, 2014, https://doi.org/10.1109/CDC.2014.7039540.

[15] M. Zhu and S. Martinez, Distributed coverage games for energy-aware mobile sensor networks. SIAM Journal on Control and Optimization, 51 (1), 1-27, 2013, https://doi.org/10.1137/100784163.

[16] S. Rahili, J. Lu, W. Ren and U. M. Al-Saggaf, Distributed coverage control of mobile sensor networks in unknown environment using game theory:

algorithms and experiments. IEEE Transactions on Mobile Computing, 17 (6), 1303-1313, June 2018, https://doi.org/10.1109/TMC.2017.2761351.

[17] E. Paraskevas, D. Maity and J. S. Baras, Distributed energy-aware mobile sensor coverage: A game theoretic approach. 2016 IEEE American Control Conference (ACC), 6259-6264, Boston, MA, USA, 2016, https://doi.org/10.1109/ACC.2016.7526653.

[18] J. Ni, G. Tang, Z. Mo, W. Cao and S. X. Yang, An improved potential game theory based method for multi-uav cooperative search. IEEE Access, 8, 47787-47796, 2020, https://doi.org/10.1109/access.2020.2978853.

[19] J. R. Marden and J. S. Shamma, Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation. Games and Economic Behavior, 75 (2), 788-808, 2012, https://doi.org/10.1016/j.geb.2012.03.006.