2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

A Potential Game with a Dynamic Penalization Map for Multi-robot Cooperative Search Missions

David Hurtado-Barreto[1] and Nicanor Quijano[2]

Abstract— **This paper addresses the problem of multi-robot coordinated navigation in target localization missions. We employ a potential game with an added penalization term that is dynamically updated to improve the performance of the multi-robot system by decreasing the number of movements needed to localize the targets and therefore, save time and energy. Then, we give conditions on this penalization map to guarantee low probabilities of revisiting explored zones on consecutive turns. By employing binary log-linear learning (BLLL) we solve the game for different simulated scenarios and compare them to recently developed strategies. Afterwards, we implement a decentralized controller on a robot simulator and illustrate the penalization map on a physical robot in a simple scenario.**

I. INTRODUCTION

Multiple efforts have been made in the last decade for the development of robotic multi-agent systems in complex scenarios. In the context of target localization missions, which involve applications such as search and rescue, surveillance, or inspection, there is a notable focus on these missions due to their ability to parallelize tasks and maintain resilience in the face of faults. By treating the ensemble of robotic agents as a network, several distributed algorithms and models of their interactions have been developed to guarantee coordinated actions when confronting challenges such as consensus or deployment problems [1]. However, as it has been highlighted on a recent review [2], several technical barriers exist in many of the areas that surround these types of missions: the balance between cost and functionality, human-robot interactions, and the need for proper planning and communication between agents. Both planning and communication relate to the way each agent decides when and how it will perform a movement or share information to perform tasks in a coordinated and effective manner. Therefore, efforts must be made to develop strategies that guarantee high levels of coordination for fast, safe and efficient exploration of space. Since game theory analysis is one of the best approaches for local task distribution, it can be used to model agents' decisions to achieve a global objective [3].

The collaborative implementation of game-theoretical tools in search-and-rescue, surveillance and navigation contexts has been investigated using a variety of approaches, which in general, can be divided into two main types. The first one aims to give a complete answer to the problem, modeling the whole scenario as a potential game where the objective function of each agent is perfectly aligned with the global function of the system, and therefore each action of an agent, within a possible fixed set of actions, has a direct impact on the global function. In this way, the existence of a Nash equilibrium is guaranteed and the problem can be solved. The study presented in [4] proposes a scenario for multi-robot exploration, where the global function is maximized by exploring the points of a discretely divided bounded and unbounded space. A different approach is taken in [5], where the authors study a coverage area problem for surveillance missions of unmanned aerial vehicle (UAV) troops. Through a modular framework, consisting on local sensing and information fusion through a consensus-based filter, the UAVs are able to coordinately move to search on a previously unknown space and survey a specific area portion. In a similar scenario with UAV troops, in [6] several modifications on the potential-game solving algorithm binary log-linear learning (BLLL), and an introduction of strategies for escaping zero-utility area are used to increase convergence speed for collaborative search problems in complex scenarios. Similarly, in [7] a potential game is used as a base structure for a distributed algorithm that aims to respond to robot failures when a system of multiple robots perform area coverage tasks in unknown environments. Another approach is developed in [8], posing a non-zero sum game in a changing environment where the utility function is estimated based on a probability map of the location of different targets, travel costs and current decisions of each robot. With this utility function and different strategies, the agents make calculated decisions to approach the different targets.

The second approach involves incorporating game theory as a supplementary tool to enhance the fundamental control techniques, such as genetic algorithms, swarm algorithms or fuzzy control schemes.The purpose of this is to enhance the synchronized actions of the robots as to alleviate any disadvantages that may develop in such situations. For example, in [9] the authors employ two different motion planners based on genetic algorithms and pulse frequency modulation (PFM) along with strategic coordination to solve coordinated robot navigation. These procedures are activated when there is a conflict between two robots that make decisions according to three different strategies of different levels of cooperation. The results show a reduction in the elapsed time to reach the targets with respect to exclusively heuristic methods. On the other hand, in [10] a control based on particle swarm optimization (PSO) is employed in a scenario where a swarm of robots tries to progressively increase its fitness function by

[1]David Hurtado-Barreto is with School of Engineering, Universidad de los Andes, Bogota, Colombia `dr.hurtado@uniandes.edu.co`
[2]Nicanor Quijano is with School of Engineering, Universidad de los Andes, Bogota, Colombia `nquijano@uniandes.edu.co`

overcoming mazes and obstacles. In this case, a sequential game is implemented between the swarm and each agent to avoid fast convergence to local minima and to have a better balance between exploration and exploitation. In such manner, the multi-robot system is better distributed in space, and the desired target is found more efficiently.

In this paper, we present a target localization scenario modelled as an area-coverage problem of multi-agent robotic systems as illustrated in Figure 1. This is done through potential games formulation, which gives a robust solution to the global problem. The main contribution of this paper is the introduction of a novel term in the potential function of the game, which minimizes the probability of revisiting explored regions in zero-utility areas, thus reducing the number of movements per agent until convergence. Additionally, we state the necessary conditions on this penalization map to guarantee a low probability of re-exploring zones, guide the agents, and efficiently scatter the group towards non-explored regions. To illustrate the game, as in [5][6], we employ BLLL to solve the new potential game on a 2D simulated scenario with different numbers of agents and objectives. As opposed to previous works where the main utilized metric is the number of steps to convergence, we employ a new metric that accounts for energy consumption and gives insight on the real time employed to reach the objective. Moreover, the same game is implemented on a 3D robot simulator under different scenarios in a decentralized manner. This involves taking into account critical elements that are indispensable for practical applications, including the management of turns, protocols for information exchange, robot hardware, and dynamics. Finally, in order to illustrate the penalization map, a one-agent scenario is developed on a real robotic platform.

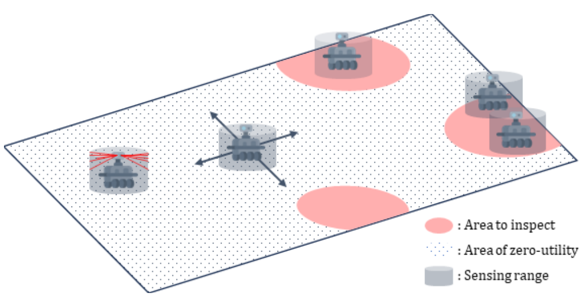


Fig. 1. Scenario of cooperative search for selective area inspection

The paper is organized as follows. In Section II, we give a theoretical background on potential games and the algorithm that is used to solve the game. In Section III, we present the modified game formulation of multi-robot target search as an area-coverage problem and state the necessary conditions for the added penalization term. Next, in Section IV we discuss the implemented decentralized controller in both the robot simulator and real-life implementation. Section V presents the testing scenarios and results for the simulations, and the real-life implementation. The results are discussed and compared to previously studied formulations in Section VI. Lastly, in Section VII we expose some final remarks.

II. Theoretical Framework

A. Potential Games

Potential games are a subclass of weakly-acyclic games characterized by the existence of a global utility function that is perfectly aligned with the utility function of each agent. Specifically, a game consisting of a set S of m players where each player i has a set of actions $A_i = \{a_i^1, a_i^2, ..., a_i^n\}$, and a utility function $U_i : A_i \to \mathbb{R}$, is an exact potential game if:

$$\begin{aligned} &U_i\left(a_i', a_{-i}\right) - U_i\left(a_i, a_{-i}\right) = \Phi\left(a_i', a_{-i}\right) - \Phi\left(a_i, a_{-i}\right) \\ &\quad \forall i \in S,\ \forall a_{-i} \in A_{-i},\ \forall a_i', a_i \in A_i, \end{aligned} \tag{1}$$

where $\Phi(\cdot)$ is the global utility function or potential function, and A_{-i} represents the action set of all the players except for the ith player. The terms a_i and a_i' correspond to a pair of single actions of the ith player [11]. This type of games guarantee the existence of at least one pure Nash equilibrium, a state in which no unilateral action by a player will cause an increase in the global utility function. Therefore, by maximizing each agent's utility function we can simultaneously maximize the global utility and attain a global objective in a coordinated manner.

B. Binary log-learning learning algorithm

The BLLL was first introduced in [11] as a learning algorithm suitable for real-time applications that guarantees convergence to a suboptimal Nash equilibrium. This algorithm takes into account the case where the actions of an agent at every time is restricted, i.e., there exists a constrained set $C_{a_i(t)}$ that is a subset of the set of actions A_i and is a function of $a_i(t)$, which is the action currently being played by the agent i.

The algorithm consists of the following steps. First, an agent is randomly selected to play with equal probability, while the others repeat their current action. Secondly, the agent selects a trial action a_i' with the following probability:

$$\begin{cases} P(a_i' = a(t)) = \frac{1}{z_i}, & \text{for any } a(t) \in C_{a_i(t)} \setminus a_i(t) \\ P(a_i' = a_i(t)) = 1 - \frac{(|C_{a_i(t)} - 1|)}{z_i}, & \text{otherwise.} \end{cases}$$

The variable z_i is the maximum number of actions for an agent in any restricted action set, $a(t)$ is an action inside the constrained action set different to the current action, and $|C_{a_i(t)}|$ is the number of actions in the current agent's restricted set.

The third step of the algorithm is the calculation of the agent's current utility function $U_i(a_i(t))$, and the expected utility function $U_i(a_i', a_{-i}(t))$ if the trial action was implemented, while the other agents repeat their current action. Finally, the probability of implementing the trial action or keeping the current action is calculated according to:

$$\begin{cases} P\left(a_i(t+1) = a_i(t)\right) = \dfrac{e^{\beta U_i(a_i(t))}}{e^{\beta U_i(a_i(t))} + e^{\beta U_i\left(a_i', a_{-i}(t)\right)}} \\ P\left(a_i(t+1) = a_i'\right) = \dfrac{e^{\beta U_i\left(a_i', a_{-i}(t)\right)}}{e^{\beta U_i(a_i(t))} + e^{\beta U_i\left(a_i', a_{-i}(t)\right)}}. \end{cases} \tag{2}$$

The parameter β, is usually referred to as the exploration parameter since decreasing it will encourage the agent to have a more exploratory behavior.

III. Game formulation

A. Potential search game

Following the notation in [5][6], the potential search game is set up as an area-coverage problem. The scenario consists of a grid that subdivides a bounded space Ω of known width and height $W \times L$ into equal square cells of side length Δx. Such a space has an associated probability density function $\eta(g), g \in \Omega$ that results from a Gaussian mixed model: $\eta(g) = \sum_{j=1}^{K} \eta(g)\eta(g \mid j) = \sum_{j=1}^{K} w_j \Psi(g \mid \mu_j, \Gamma_j)$, where Ψ is the density function of a multivariate Gaussian distribution, μ_j the mean vector, and Γ_j is the covariance matrix of the jth component. Thus, every point on the grid will have an associated probability with the existence of a target where μ_j represents the target centers. For the agents, we treat the vehicle system as a dynamic network where each vehicle has certain associated characteristics, such as a position on the grid $\pi_{i,t}$, a restricted action set $C_{a_{i(t)}}$ that depends on its current location, a communication radius for information transmission R_c, and a sensing radius for target or signal detection R_s, which always lies within the space Ω. Then, the set of vertices of the graph corresponds to the vehicles, and the set of edges will depend on the distance between the agents and their communication radii. This is shown on the following set of equations: $Net = (E(t), V), V = \{v_1, v_2, ..., v_m\}, E(t) = \{\{v_i, v_j\} : v_i, v_j \in V; ||\pi_{i,t} - \pi_{j,t}|| \leq R_{c_i}\}$.

For the game implementation, several assumptions are made: (1) the team is homogeneous, i.e., all the agents have the same sensing, communication and motion capabilities; (2) they can only move in nine discrete positions as depicted by the constrained action set in Figure 2; and (3) they have full information about the density and penalization map, and the position of the rest of the team. Figure 2 depicts all of the aforementioned elements in an example scenario.

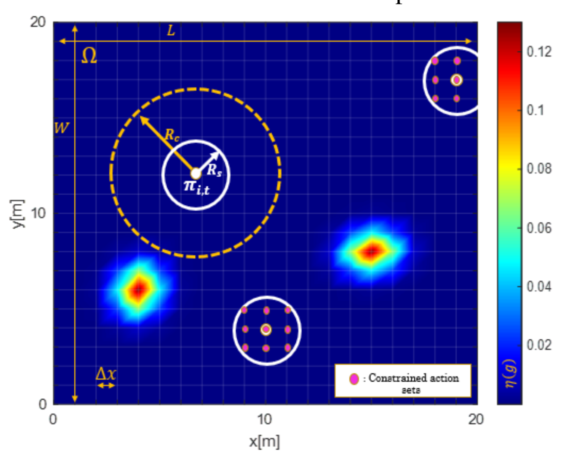


Fig. 2. Example scenario depicting the basic elements of the game.

The potential function that governs the game is given by the following expression:

$$\Phi(a_i, a_{-i}) = \sum_{g \in \Omega} f\left(\min_{j \in \{1,2,\dots,m\}} \|g - \pi_j\|\right)(\eta(g) + \rho(g))\Delta x^2, \quad (3)$$

where:

$$f\left(\|g - \pi_j\|\right) = \begin{cases} 1 & \|g - \pi_j\| \leq R_s \\ 0 & \text{otherwise} \end{cases}$$

$$\rho(g) = \begin{cases} \xi & \|g - \pi_k\| \leq R_s \quad \forall \pi_k \in \mathrm{M}, \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

in which M is the set of positions π_k that have already been visited by some agent, and ξ is the value of penalization. The potential function measures the accumulated values of probability and penalization under the sensing radii of the agents over the entire search area.

Following the proof on [6], by using the concept of *Wonderful Life Utility (WLU)*, the individual utility function $U_i(a_i, a_{-i})$ can be obtained from the agent's marginal contribution as shown below:

$$\begin{aligned} U_i(a_i, a_{-i}) &= \Phi(a_i, a_{-i}) \; - \; \Phi(a_{-i}) \\ &= \Phi(a_i, a_{-i}) - \; \dots \\ &- \sum_{g \in \Omega} f\left(\min_{j \in \{1,2,\dots,i-1,i+1,\dots,m\}} \|g - \pi_j\|\right)(\eta(g) + \rho(g))\Delta x^2. \end{aligned} \quad (5)$$

Lemma 1: Consider a target search problem modeled as an area-coverage cooperative game with $V = \{v_1, v_2, ..., v_m\}$ representing the players' set and $A = \{a_1, a_2, ..., a_n\}$ representing the set of joint actions. If every player's utility in the set is given by (5) then the game is a potential game with (3) being the potential function.

Proof: For a given agent i the change in individual utility when changing from an action a_i to a_i' with $a_i, a_i' \in A_i$, while the rest of players maintain their actions is expressed in terms of the potential function as :

$$\begin{aligned} &U_i\left(a_i', a_{-i}\right) - U_i\left(a_i, a_{-i}\right) = \\ &\Phi\left(a_i', a_{-i}\right) - \Phi\left(a_{-i}\right) - \left(\Phi\left(a_i, a_{-i}\right) - \Phi\left(a_{-i}\right)\right) = \\ &\Phi\left(a_i', a_{-i}\right) - \Phi\left(a_i, a_{-i}\right). \end{aligned}$$

Resulting on the potential game definition given by (1).

■

B. Map penalization

Due to the nature of the learning algorithm, in zero-utility areas the agents do not have any particular incentive that prevents them to revisit explored areas. This means that between consecutive turns of the same agent, it can repeat the same positions. This results in an increasing number of turns that do not contribute to the target localization, and imply time and energy waste. For this reason, $\rho(g)$ can be designed to minimize the probability of revisiting explored zones in zero-utility areas by dynamically updating a shared memory map every time an agent decides to make a move in these types of zones. Zero-utility areas are defined as positions where every action leads to an individual utility value of zero, i.e.,

$$\max\{U_i(a_i, a_i(t)\} = 0, \; \forall a_i \in C_{a_i(t)} \setminus a_i(t). \quad (6)$$

Assumption 1: $\rho(g)$ is only updated when an agent meets the condition given by (6) at a given time step.

Assumption 2: The sensing radii between the agent taking the action and the rest of the network agents do not overlap in the agent's current or trial position.

Assumption 3: There is at least one unexplored grid point in the agent's current position.

Theorem 1: Under the conditions stated by Assumptions 1, 2 and 3, the penalization map $\rho(g)$ given by (4) with $\xi \leq -\frac{1}{\Delta x^2 \beta(\alpha-1)}, \alpha \in (1,2)$ guarantees a minimal probability of returning to already explored zones in the map as $\alpha \to 1$. The proof of this theorem has been omitted due to page limitations.

IV. The Decentralized Controller

In order to execute the game in a more realistic scenario, we implement a decentralized scheme on a 3D robot simulator. By taking into account the dynamics of the robot, we approach the real convergence time of the algorithm, since communication and movement times are considered. To do this, we have to manage both the turns timing and the position sharing. For the first, following the proposed guidelines on [12], each agent of the vehicle network contains a Poisson clock with $\lambda = 1$ to randomly activate its turn. In addition, we introduce the broadcast of indicators to notify the rest of the players and block their turn. In this way, we guarantee that only one robot "plays" while the rest maintain their positions.

On the other hand, for position tracking and sharing, we equip each agent with a perfect resolution GPS to accurately obtain the robot's position. Then, to share the information we model the inter-agent communication to send and receive the positions of the rest of the agents. In this manner, each robot can build the shared memory map $\rho(g)$ and calculate their utility to execute the algorithm. Additionally, the proximity sensors from the robot model are used to calculate the constrained set of actions and the motor commands are calibrated to perform the discrete movements.

For both the 3D robot simulator and pyhsical implementations we employ the second version of the educational robots e-puck[1]. However, in the case of the physical implementation the GPS is replaced by a coordinates server that contains the agents' positions by extracting them from image analysis of the robot's color tag. The connection setup used is illustrated in Figure 3.

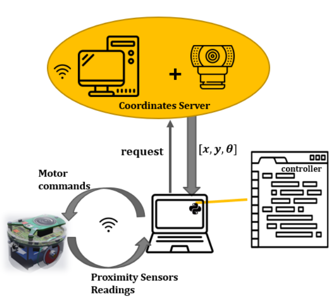


Fig. 3. Communication scheme for physical implementation.

[1] https://www.gctronic.com/doc/index.php/e-puck2.

V. Simulated Scenarios and Results

This section presents the series of experiments used to illustrate the potential game in search scenarios starting from 2D scenarios, where the dynamics of the agents are not taken into account, until the use of the decentralized controller discussed in the previous section for the robotic simulator and a simple real robot implementation. In order to compare our results to previous formulations, we employ the metric NNM (normalized number of movements) that is defined as:

$$\text{NNM} = \frac{\text{Number of movements of all agents until convergence}}{\text{Total number of agents}}. \tag{7}$$

Due to the fact that in many of the iterations the agents do not change position, the proposed metric accounts for the real time and energy costs that result from moving. In addition, we employ the mission success percentage to represent the percentage of missions where all of the targets are found.

A. 2D simulations

The first simulation consists of a 20×20 grid scenario with 2 agents and 2 targets. The communication radius is large enough to guarantee connectivity with all the agents during the entire mission, while the sensing radius assures the coverage over all the maximum trial actions of their constrained set. We execute the game over five different initial agent configurations for ten iterations per configuration. This is done for both the game with and without the penalization map term. Figure 4 shows an example of the resulting scenario and penalization map after 1000 iterations. Figure 5 shows the comparison between the games for each configuration.

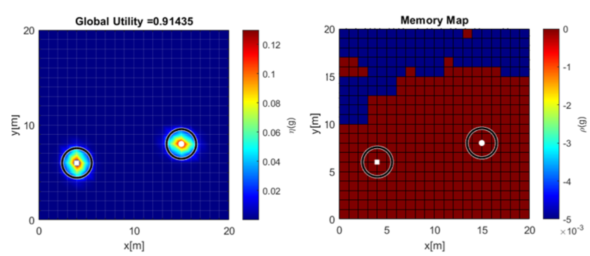


Fig. 4. Example of the final positions at the end of the mission and the penalized memory map $\rho(g)$.

The second set of simulations deals with the case of 4 agents in a 30×30 scenario and 3 targets to localize. For this simulation, the initial position of the agents and the size and position of the targets are randomized resulting in 8 different scenarios. We test each scenario three times on each of the following: the original potential game, the original game including a zero-utility escaping strategy proposed in [6], and our proposed potential game with the penalization term. Figure 8 shows the comparative results in terms of the normalized number of movements.

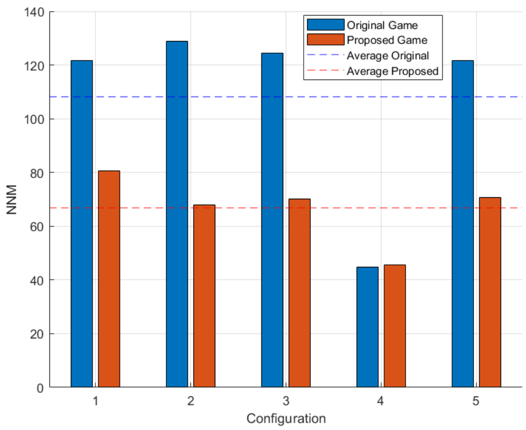


Fig. 5. Performance of the game in terms of the proposed metric (7). (10 averaged trials of 5 different initial configurations).

B. Robot simulator

The game is set up on the *Webots* simulator, on a 1.5 by 0.9 m scenario, equivalent to the robotic platform dimensions used in the physical implementation described next. The area is divided into cells of 7.5 cm as shown in Figure 6. Three tests per configuration are performed on 4 different initial configurations for both one and two agents. The results in terms of total time to convergence, normalized number of movements and mission success rate are shown in Table I.

TABLE I
AVERAGE RESULTS OVER 12 MISSIONS

	Time to reach the objective [min]		#Movements/agent		%success	
# of agents	1	2	1	2	1	2
Original	18.2	39.3	138.8	109.9	100%	25%
Proposed	5.7	17.2	37.5	42.6	100%	50%

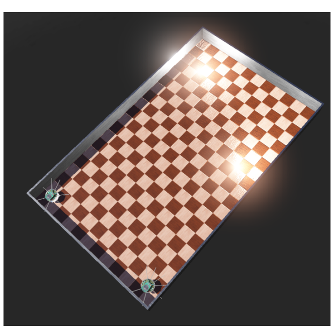
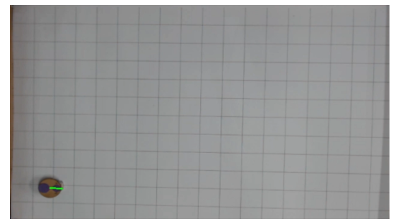


Fig. 6. Example of initial setup for the robot simulator tests with two agents.

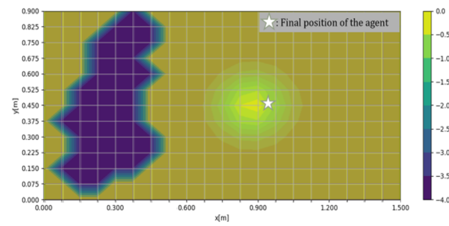
C. Physical implementation

To illustrate the proposed penalization map under the stated conditions, a simple setup of 1 agent and 1 target to localize is implemented on a real robotic platform and two trials are performed[2]. Figure 7a shows the initial robot scenario for one of the trials, while 7b illustrates the navigated map with the penalized areas showing the explored zero-utility zones. The average of the performed trials corresponds to 23.5 movements and 11.8 minutes to find the desired target.

[2]A sample video of a trial can be seen in: https://youtu.be/UhyegwAkP8U.



(a)



(b)

Fig. 7. (a) Camera view of the scene of the physical implementation; (b) Penalization map and final agent position.

VI. DISCUSSION

The results obtained in the series of conducted experiments reflect a significant improvement in terms of movements per agent. For the 2D simulation case with two agents, there is an average reduction of 38% in the number of movements when using the proposed approach. This is evident on four out of the five initial configurations (Figure 5), as in the fourth configuration there is not a considerable difference between the results. This comes from having the agents too close to the targets, which implies few zero-utility zones to penalize between the initial robots' position and the targets' location.

For the case of 4 agents in the randomized scenarios, the proposed modification on the game shows an improved performance not only compared to the original game, which did not include a zero-utility escape mechanism, but also to the other game that implemented it (Figure 8). With an average value of 83.5 movements per agent over, the addition of a penalization map shows an improvement of 60% and 45% with respect to the original and the alternative zero-utility escape strategy implementations, respectively. Again, this supports that dynamically penalizing the map as agents traverse zero-utility areas is beneficial for increasing the cooperative search, as already sensed areas are avoided and marked to guide the rest of the team, ultimately increasing the efficiency in exploration.

In the robot simulator experiments, there is approximately a 70% reduction on both time and number of movement per agent for the case of 1 agent, and a reduction of around 60% in the case of two agents, when implementing the proposed dynamic penalization. Another interesting result is

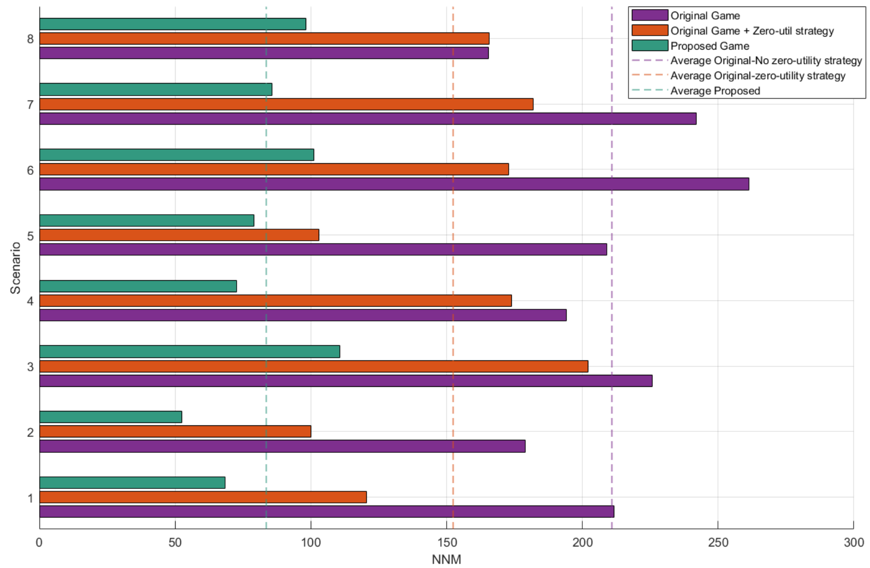


Fig. 8. Comparison results of the proposed metric for 3 trials of 8 random scenarios.

the fact that more missions were successful with the proposed approach, suggesting that the map not only helps to reduce unnecessary movements, but also to better disperse the agents in space. Finally, the physical implementation results are consistent with the NNM metric and the total time previously found. The final position of the agent as seen in Figure 7b indicates a maximization of the covered area, implying a successful target localization.

VII. Conclusions

Based on an area-coverage potential game, we have added a penalization term for previously explored zero-utility zones. By analyzing the BLLL algorithm we have managed to give certain conditions on this term to minimize the probability of returning to already explored areas. Specifically, by assigning a value of $\xi \leq -\frac{1}{\Delta x^2 \beta(\alpha-1)}, \alpha \in (1,2)$ in already explored zero utility points of the partitioned space, the penalization map induces a behaviour where fewer movements are needed to reach the targets as the agents avoid revisiting the same areas, skipping their turn to wait for a better action. The inclusion of this novel term is tested in various 2D, 3D simulated missions, and in a physical implementation, resulting in a significant improvement in the number of movements per agent required to localize the targets, reaching reductions of 40-60% in the 2D case and 60-70% in the robot simulator case with respect to recent models. In other words, we have managed to increase the efficiency of exploration to perform target search missions in a coordinated manner in terms of time and energy. In addition, we have implemented a decentralized controller that takes into account the real hardware and dynamics of the robot, obtaining more accurate results of the real time response when implementing this framework on search missions.

Future work should address more complex scenarios and the adoption of new strategies to avoid local minima and increase robustness. In this way, these types of games could be implemented at full-scale on real scenarios for search and rescue, inspection, or surveillance missions.

References

[1] Francesco Bullo, Jorge Cortés, and Sonia Martinez. *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*, volume 27. Princeton University Press, 2009.

[2] Daniel S Drew. Multi-agent systems for search and rescue applications. *Current Robotics Reports*, 2:189–200, 2021.

[3] Rinat Galin, Roman Meshcheryakov, and Saniya Kamesheva. Distributing tasks in multi-agent robotic system for human-robot interaction applications. In *Proceedings of the International Conference on Interactive Collaborative Robotics*, pages 99–106. Springer, 2020.

[4] George Philip, Sidney N Givigi, and Howard M Schwartz. Multi-robot exploration using potential games. In *Proceedings of 2013 IEEE International Conference on Mechatronics and Automation*, pages 1203–1210. IEEE, 2013.

[5] Pei Li and Haibin Duan. A potential game approach to multiple uav cooperative search and surveillance. *Aerospace Science and Technology*, 68:403–415, 2017.

[6] Jianjun Ni, Guangyi Tang, Zhengpei Mo, Weidong Cao, and Simon X Yang. An improved potential game theory based method for multi-uav cooperative search. *IEEE Access*, 8:47787–47796, 2020.

[7] Junnan Song and Shalabh Gupta. Care: Cooperative autonomy for resilience and efficiency of robot teams for complete coverage of unknown environments under robot failures. *Autonomous Robots*, 44(3):647–671, 2020.

[8] Yan Meng. Multi-robot searching using game-theory based approach. *International Journal of Advanced Robotic Systems*, 5(4):44, 2008.

[9] Buddhadeb Pradhan, Nirmal Baran Hui, Diptendu Sinha Roy, Gautam Srivastava, and Jerry Chun-Wei Lin. Game-theoretic strategic coordination and navigation of multiple wheeled robots. *ACM Transactions on Internet Technology (TOIT)*, 21(4):1–15, 2021.

[10] Khalil Al-Rahman Youssefi, Modjtaba Rouhani, Habib Rajabi Mashhadi, and Wilfried Elmenreich. A swarm intelligence-based robotic search algorithm integrated with game theory. *Applied Soft Computing*, 122:108873, 2022.

[11] Jason R Marden, Gürdal Arslan, and Jeff S Shamma. Connections between cooperative control and potential games illustrated on the consensus problem. In *Proceedings of 2007 European Control Conference (ECC)*, pages 4604–4611. IEEE, 2007.

[12] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530, 2006.