Multi-Robot Heterogeneous Adversarial Coverage

Yair Korngut[1] and Noa Agmon[2]

***Abstract*— Robotic coverage is one of the canonical problems in robotics research, seeking to find a path that visits each point in an area while optimizing some criteria, usually minimizing the time to complete the coverage. This paper considers a variant of the robotic coverage problem, *multi-robot adversarial coverage*, in which a team of robots is required to cover an area containing threats that might stop the robots with some probability. Motivated by the advantages of using heterogeneous robots for this mission, we formulate the problem while accounting for the trade-off between the coverage time and the expected number of covered cells, considering also the different (heterogeneous) characteristics of the robots involved in the mission. We formulate the problem as a Dec-POMDP and use multi-agent reinforcement algorithms to compute an optimal policy. We have implemented our RL-based methods along with an enhanced heuristic algorithm, and show their superiority compared to the state of the art. Finally, we discuss the possible limitations of learning-based algorithms in different settings.**

I. Introduction

Robotics coverage path planning is one of the fundamental problems in robotics. The goal of coverage path planning is to find a sequence of world locations that allows the robot(s) to visit every part of a target area while optimizing some criteria, usually minimizing travel time or cost while avoiding obstacles. The problem has many real-world applications, from mapping and surveillance [1] to forest fire monitoring [2] and search and rescue in disaster areas [3], [4], etc.

Adversarial coverage path planning [5] is a variant of the canonical coverage problem, which refers to a robot, or a team of robots, that needs to cover an area containing *threats*, that might disable the robots, and thus stop them during their coverage task. Adversarial coverage is widely applicable, from military applications in which the enemy may attack the robots, to search and rescue in disaster areas, where debris may harm the robots and prevent them from completing their mission. The objective of the robot in adversarial coverage path planning includes maximizing the survivability of the robots or the covered area, along with minimizing the travel time or cost.

When performing coverage in adversarial environments, it is highly beneficial to use a team of robots rather than a single robot, because redundancy and survivability are very important, especially compared to the non-adversarial setting. The robots in the team may be homogeneous, but often they are heterogeneous, such that robots differ in their velocity, capabilities, or immunity to threats in the environment.

[1]Yair Korngut is with the Department of Computer Science, Bar-Ilan University, 5290002 Ramat-Gan, Israel. kornguy@biu.ac.il

[2]Noa Agmon is with the Department of Computer Science, Bar-Ilan University, 5290002 Ramat-Gan, Israel. agmon@cs.biu.ac.il

Previous work of adversarial coverage dealt mainly with single-robot coverage [6], optimizing both the number of covered cells and the time to complete the coverage, and a simplified, multi-robot problem that considers only maximization of the number of covered cells, while ignoring the time it takes for covering them [7], [8]. In many adversarial environments, ignoring the coverage time results in highly inefficient coverage paths, as robots tend to travel back and forth in the environment, trying to avoid threats.

In this work, we therefore examine the heterogeneous multi-robot adversarial coverage (HMRAC) problem, in which a team of possibly heterogeneous robots covers a given area containing threats.

We formally define the HMRAC problem and formulate an objective function that accounts for both the number of covered cells and the coverage time, allowing a more complex trade-off between the covered area and the coverage time. We approach the problem using both a heuristic algorithm and a learning system and prove that the learning system converges to an optimal solution. Following, we model the HMRAC problem as a Dec-POMDP and use multi-agent reinforcement learning (MARL) methods to find an optimal solution (i.e., policy) for the problem. We show that using RL straightforwardly in some environments containing threats may be unstable and converge to a sub-optimal solution for single-robot coverage, thus we suggest an alternative process for training, leading to a faster, more stable convergence.

Finally, we performed a rigorous empirical evaluation, comparing our suggested methods with the state-of-the-art. We show the superiority of our methods in terms of coverage time and covered area and discuss the limitations of using reinforcement learning to solve the HMRAC problem. An open-source implementation of all the algorithms and tasks is available for full reproducibility[1].

II. Related Work

The multi-robot coverage is a canonical robotics problem, in which one or more robots are required to visit each point in a given area at least once, usually while trying to minimize the coverage time or cost. A comprehensive review of robotics coverage research can be found in [9].

Yehoshua et al. [6] defined the problem of robotic *adversarial* coverage in which the area contains threats that may stop the covering robots, thus the goal is to either maximize the survivability of the coverage path and minimize the coverage time. The problem was proven to be hard, thus three approaches were presented for the single-agent variant

[1]https://github.com/YairKorn/MAPS

[5], [10] - a greedy algorithm, an algorithm that utilizes Spanning Tree Coverage and an algorithm that models the problem as MDP and uses real-time dynamic programming to find a solution. The first two approaches were combined into a heuristic algorithm for the homogeneous multi-robot case [7], called MRAC.

A variation of the multi-robot adversarial coverage problem was defined by Jorgensen et al. [8], in which each robot is required to survive with a minimal probability ρ. They presented a greedy algorithm similar to the greedy algorithm [5] for the multi-robot case and also considered the heterogeneous case by computing the greedy heuristic regarding the threats that affect each robot.

Note that adversarial coverage is more a general problem than the resilient coverage problem (RCP) [11], [12], [13], [14], since the RCP problem assumes an equal probability of failure in the whole environment, while the adversarial coverage problem is able the take advantage of the distribution of threat in the environment for optimizing a given criterion.

The use of reinforcement learning for generating single- and multi-robot coverage paths is found in the literature. Lakshmanan et al. [15] used experience-replay actor-critic (ACER) to plan a complete coverage path and Boufous [16] used DQN for single-robot complete coverage path planning.

Xiao et al. [17] used distributed cooperative Q-learning for multi-agent coverage under communication restrictions. Hu et al. [18] used GANs to allocate subareas to the robots. Rückin [19] combined MCTS and CNN for coverage.

In our work, we implemented multi-agent reinforcement learning methods with frequent rewards, allowing the robots to learn to coordinate optimally and fast. In the adversarial problem, we focused on, robots can be disabled, therefore good coordination is essential, and the methods mentioned above cannot learn to coordinate well, due to sub-optimal assignments or sparse rewards. In section IV we describe the methods that allow us to find optimal coverage paths for the heterogeneous multi-agent adversarial coverage.

III. Problem Definition

In the heterogeneous multi-robot adversarial coverage (HMRAC) problem, a team of heterogeneous robots is required to cover an environment that contains threats, such that a robot that enters a threatened cell might be disabled with some probability, and will thus not be able to continue to cover the environment. We consider the coverage problem in its *offline* setting, in which the environment is known in advance and it is fully observable, i.e., the robots know the location of the obstacles and the location and probability distribution of the threats in the environment, and know their absolute location at any time.

A group of m robots $R = \{r_1, \dots r_m\}$ is required to cover an area S. The robots are heterogeneous, such that each robot is associated with one of K types $\{R_1, \dots R_K\}$, $R = \oplus_{1 \le k \le K} R_k$. The heterogeneity of the robots is reflected by their immunity to threats, that is, the probability of being disabled by a threat might differ between robots.

Let S, the area needed to be covered, be represented by a grid of n cells $S = \{c_1, \dots c_n\}$. Each cell c_i in the grid is associated with a risk vector $[p_i^1, p_i^2, \dots, p_i^K]$, that is static and known in advance, such that p_i^k is the probability of a robot of type k that enters cell c_i to be disabled. The risk profile $P(c_i)$ is the function that maps for each cell c_i its corresponding risk vector $[p_i^1, p_i^2, \dots, p_i^K]$. A cell is considered to be *covered* when a robot enters the cell (thus a cell is considered covered even if the robot was disabled by the threat associated with that cell).

Note that the threat distribution in the area, as viewed from different robots' eyes, may differ not only by scalar factor. Some areas may be of more danger to one robot, while other areas may be more dangerous to another. This property is demonstrated in Figure 1.

A path for a robot $r_j \in R$, denoted by T_j, is a sequence of cells, $T_j = \{c_j^1, c_j^2, \dots\}$ such that $c_j^l \in S$ and is obstacle-free, and $c_j^l, c_j^{l+1} \in T_j$ are adjacent cells in S. A path may contain a cell more than once (that is, cells may be revisited). Paths are dynamic, i.e., a robot is able to modify its path to deal with changes in the environment (e.g., another robot that has been disabled). We assume that the robots cover one cell per time step, therefore the *coverage time* of a robot r_j is the number of cells in T_j (denoted by $|T_j|$), and the coverage time of the team of robots R is defined as the maximal coverage path of its members, that is, $\max_{r_j \in R} |T_j|$.

The goal of this work is to find an algorithm that determines a set of paths, $T = T_1, T_2, \dots, T_m$ such that $\bigcup_{j=1}^{m} T_j = S$, that maximize optimization criterion (described below) for the given group of m robots, the area S, and the risk profile P in the area.

A. Optimization Criterion

As mentioned above, there is a trade-off between the number of covered cells and the coverage time of each member, and the team. One path may induce higher risk but a shorter time, while another may induce lower risk but a longer time.

For determining the relative importance of coverage success (i.e., the number of covered cells) and coverage time, we define the optimization criterion as follows:

$$\mathbb{C}(T) = E[\mathbb{A}(T)] - \alpha E[max_i |T_i|] \tag{1}$$

the left expression ($E[\mathbb{A}(T)]$) is the expectation of the number of covered cells, and the right expression ($E[max_i |T_i|]$) is the expectation of the maximal length of the robots' paths. Our purpose is to **maximize** this criterion.

The constant α determines the relative importance of the coverage success and coverage time. For $\alpha = 0$, the robots do not care about coverage time but only about coverage success. In this scenario, the robots will try to avoid risk as long as they can, thus we expect them to visit threatened cells starting from lower-risk cells to higher-risk cells.

For $\alpha \to \infty$, the problem is not a coverage problem and the optimization criteria is $\mathbb{C}(T) \approx -\alpha E[\max_i |T_i|]$, ignoring the covered area. An optimal solution for this scenario might be a path that disables the robots as soon

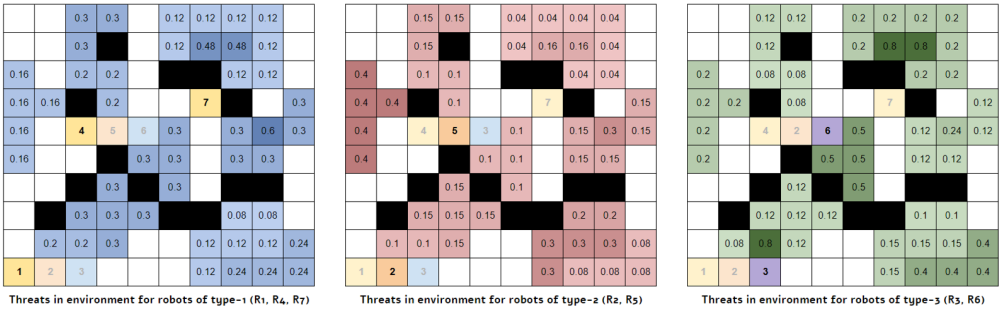

Fig. 1. The threat distribution in the area as viewed by different robot types. Black cells are obstacles; the darker the cell, the higher the risk associated with it.

as possible and minimizes the path length (equivalent to the coverage time). Hence, we focus on reasonable values of α, limiting it by $0 < \alpha \leq 1$.

The effective value of α depends on the number of robots. That is, in an environment with m robots, in one time step the team can cover up to m cells, hence effectively the behavior of the robots depends on α/m rather than α.

B. Modeling HMRAC as a Dec-POMDP

For solving the HMRAC problem using multi-agent RL methods, we modeled the problem as a Dec-POMDP. Though we focused on the fully-observable case, we followed the convention in MARL to model the general, partial-observable case. A Dec-POMDP is a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{Z}, \mathcal{O}, \mathcal{R}, \gamma)$, such that $\mathcal{N}$ is the number of agents, $\mathcal{S}$ is the state space, $\mathcal{A} = (a_1, \ldots, a_{\mathcal{N}})$ is the joint-action space, such that $a_i \in \mathcal{A}_i$ is the i-th agent's action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to P(\mathcal{S})$ is the probabilistic transition function, Z is the observation space, $\mathcal{O} : \mathcal{S} \times \mathcal{N} \to \mathcal{Z}$ is the observation function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the joint reward function, and $0 < \gamma \leq 1$ is the discount factor.

In our problem:

- **States**: A state $s \in \mathcal{S}$ is defined by the locations of the active robots (that is, that were not yet disabled) and the coverage status of each cell (covered or not).
- **Observations**: Since our problem is fully-observable, an observation $o \in \mathcal{O}$ is similar to the state, and also indicates for each robot its own location in the environment.
- **Actions**: At each step, every active robot selects an action $a_i \in \{\text{Up, Down, Left, Right, Stay}\}$. Actions that cause collisions with other robots or obstacles are masked and cannot be selected.
  Since Dec-POMDP models cannot express a change in the number of agents, we model this change by removing the robots from states and observations and considering them to select $a_i = \text{Stay}$ until the end of the episode.
- **Transitions**: Movements are deterministic, such that an action is always carried out successfully. When a robot of type k enters a cell with an associated risk of p_i^k, the robot gets disabled with probability p_i^k, and remains active with probability $1 - p_i^k$. These probabilities are independent, i.e., the result of the joint action is a multiplication of the results of the single-robot actions. If more than one agent tries to enter the same cell, one of them (selected arbitrarily) succeeds, while the others stay in their cells.
- **Reward**: When a robot enters an uncovered cell, a reward of $+1$ is given. At each time step, a reward of $-\alpha$ is given. In addition, if the robots have successfully covered the whole area, a high reward (e.g., 100) is given, which helps the robots converge faster to a solution.
  We defined a heuristic reward function for the single robot case that improves the learning for that case, described in Section IV-A.
- **Discount factor**: $\gamma = 0.99$. However, we evaluated different values, $0.93 \leq \gamma \leq 0.99$, and observed that this value has very little effect on the solution quality and the convergence properties.

Based on this model, one can use MARL methods to find an optimal policy π that maximizes the optimization criterion, as described above.

IV. Solving HMRAC

In this section, we describe the algorithms we used to solve the HMRAC problem, including the reward function for the RL-based solutions, and the heuristic algorithm HMRAC*.

A. Single-Agent Reward Function

A major challenge in utilizing reinforcement learning to solve the adversarial coverage problem is the accessibility of the state space. During the training phase, it is important to explore the environment, but threats in the environment might disable the robot, preventing it from continuing the exploration and making the whole exploration process inefficient (i.e., very slow) or insufficient (i.e., focusing only on safe areas of the environment).

To overcome this problem, the agent was not disabled by threats but rather got a negative reward when entering a threatened cell that estimates the effect of the threat in that cell on the expected reward. We refer to this as *simulated mode* and for the problem with the reward function that is defined in Section III and active threats we refer to it as *real*

mode. Accordingly, we defined the following reward for the single-robot case:

$$\mathcal{R}_t = N_C - \alpha - (1-\alpha)p_i^k C \tag{2}$$

Such that N_C is 1 if a new cell was covered, otherwise $N_C = 0$. α is the optimization constant, as described above; p_i^k is the threat in the cell of the agent, and C is the number of remaining uncovered cells in the environment. Intuitively, $(1-\alpha)p_i^k C$ is an approximation of the effect of the possible robot disablement on the reward, since in simulation mode it is no longer represented by the transition function.

Let us examine the reward function in more detail. For simplicity, we first consider an environment with only one threatened cell. Denote the robot's path as T, such that its length $|T| = t$; the number of uncovered cells when the robot enters the threat as C; the time it enters the cell as t_C; and the number of obstacle-free cells in the environment as n. The total expected reward using simulated mode is:

$$\begin{aligned} E_T^{sim}[\mathcal{R}] &= n - \alpha t - (1-\alpha)p_i^k C = \\ &= (n - p_i^k C) - \alpha(p_i^k(t-C) + (1-p_i^k)t) \end{aligned} \tag{3}$$

while the expected reward using real mode is:

$$E_T^{real}[\mathcal{R}] = (n - p_i^k C) - \alpha(p_i^k t_C + (1-p_i^k)t) \tag{4}$$

Therefore, $\Delta E_T[\mathcal{R}] = \alpha p_i^k((t-C) - t_C)$. The expression $((t-C)-t_C)$ is the difference between the remaining length of the path $(t - t_C)$ and the number of new cells in it (C). At high values of α, this difference is usually small because paths tend to be time-efficient, while at low values of α, the whole expression is small because it is multiplied by α.

For paths containing more than one threatened cell, these expressions can be easily expanded, such that $\Delta E_T[\mathcal{R}] = \alpha \sum_j p_{i_j}^k((t-C_j) - t_{C_j})$ for C_j, t_{C_j} the number of uncovered cells and the time when the robot enters the j-th threatened cell in the path, respectively.

In practice, despite the potential bias, the simulated mode stabilizes the learning process and performs much better than the real mode, as shown in Section V-A.

Note that the described mechanism fits well with the single-robot problem, but not necessarily with the multi-robot problem because when trying to learn a multi-robot policy, robots need to learn how to behave when some robots are disabled. Using a "simulated mode" that does not disable robots, does not allow the robots to learn a policy for these states. Therefore, we used this mechanism to enhance single-robot learning, but not multi-robot learning.

B. Multi-Agent Reward Function

In this subsection, we prove the optimality of the reward function, as described in Section III, for the multi-agent settings. Therefore, given sufficient run-time and exploration, the learning will converge to the optimal solution.

Lemma 4.1: A team of robots with a policy π that maximizes the expected reward $\mathcal{R}$, maximizes the optimization criterion $\mathbb{C}$ (eq. 1).
The proof is in the supplementary[2].

[2]https://u.cs.biu.ac.il/~agmon/HMRAC-Sup.pdf

C. Heuristic Approach

Yehoshua et al. [7] presented a heuristic algorithm for solving the multi-robot adversarial coverage problem for *homogeneous* robotic teams. In the following subsection, we briefly describe it, then present two novel extensions for their algorithm - the first one allows the algorithm to heuristically solve the problem regarding the aforementioned optimization criterion, rather than the simpler criterion offered in [7] that takes into account only the coverage success. The second extension allows the algorithm to solve the problem for heterogeneous teams.

The basic algorithm (MRAC) [7], works as follows:

1) Create a list of connected areas in the graph that have the same level of risk (e.g., using BFS).
2) Create a graph representing the cost of entering a cell, defined as:
$$c_i = \left\{ \begin{array}{ll} 1/N, & \text{for } p_i = 0 \\ p_i/p_{min}, & \text{for } p_i > 0 \end{array} \right\} \tag{5}$$
For N the number of cells in the area and p_{min} the value of the minimal non-zeroed threat.
3) Assign each robot to an area with the lowest threat level available. If there is more than one area available, select the area with the lowest cost of the path to the area. If more than one robot is assigned to an area, split it between the robots.
4) For covering an area, at every step select the uncovered cell that the path to (from the robot's location) is the lowest, based on the aforementioned graph.
5) While there is at least one active robot and the environment is not completely covered:
   - A robot that finishes covering its area, is assigned to another one.
   - If a robot gets disabled, return its area to the pool.

This algorithm has some drawbacks that lead to a sub-optimal solution even when regarding only the number of covered cells. In particular, the assignment mechanism prioritizes areas with low threat levels, even if the optimal path to these areas is more dangerous than covering another area. That increases the probability of a robot getting disabled, resulting in a lower expected number of covered cells.

Based on this algorithm, we present HMRAC*, which is similar to MRAC but contains two significant extensions:

1) Optimizing Coverage and Time: The first extension allows the algorithm to heuristically solve the more general MRAC problem, which takes into account not only the number of covered cells but also the coverage time. For that, we modified the graph used to assign areas to the robots and to build paths towards and within areas. We define the modified graph as follows:

$$c_i = (1 - \alpha/N_A)p_i C + \alpha \tag{6}$$

Such that p_i is the threat in that cell (note that if $p_i = 0$, we get $c_i = \alpha$, the cost of a single time step), C is the number of remaining uncovered cells in the environment, and N_A is the number of active robots in the environment

(since, as explained in section III, the effective value of α is α/N). This expression is inspired by the reward function for the single-agent problem, as described above.

This heuristic cost function is not a good approximation for the multi-agent case as it is to the single-agent case because the effect of a disabled robot on the obtained reward highly depends on the exact threats and obstacles distribution and the number and location of the other robots. Therefore, using this heuristic function would present a bias in the learning process. Accordingly, we didn't use this heuristic during the learning.

In addition, instead of assigning the safest available areas to the robot, we assigned areas to the robots based on the cost of the path only, since an optimal solution for higher values of α may require the robots to cover closer, more dangerous areas before far, safer areas, hence it is unreasonable to assign more dangerous areas earlier.

2) Heterogeneous Extension: The second extension allows the algorithm to handle environments that contain robots of different types. In this case, one cannot consider connected areas in the environment as areas with the same level of risk, because different robots may experience different threats in the same area. For that reason, we incorporated two modifications to the algorithm:

First, the definition of a connected area is an area that has the same level of threats for all the robot types in the environment, i.e., two adjacent cells i, j are contained in the same area iff $\forall k : p_i^k = p_j^k$. In practice, similar to the base algorithm, before comparing the level of threats, we round up the threats to the closest quarter, creating larger areas, which score better results.

Second, instead of building one graph for all the robots, each robot has a graph that considers the threats from its perspective, i.e.:

$$c_i = (1 - \alpha/N_A)p_i^k C + \alpha \tag{7}$$

Because decisions are made locally, each robot tries to maximize the optimization criterion as it experiences it. That way, the heterogeneous robots work together to maximize the optimization criterion in a similar fashion to the homogeneous case.

V. Empirical Analysis

In this section, we compare the performance of the classical algorithms MRAC, the novel extension HMRAC*, and a learning-based method for the HMRAC problem described above. Additionally, we compare the performance of single-agent adversarial coverage with and without simulated mode.

For learning a policy for the HMRAC problem, we used Deep Coordination Graphs (DCG), a multi-agent reinforcement learning algorithm described in [20].

We evaluated the algorithms in various environments, and classified environments into four groups, based on the distribution of threats and obstacles in the environments (see illustration in Figure 2). First, we distinguished between environments with **connected** threats (i.e., the threats in the environments are clustered, creating few big connected areas of threats) and **scattered** threats (i.e., the threats are scattered across the environment, creating many small connected areas of threats). Based on their mechanism, we expect the heuristics algorithms to perform better in connected environments.

Second, we distinguished between environments with **dividing** and **flank** threats. Dividing threats are threats that the robots must cross to reach a substantial part of the environment, while flank threats are threats that are not. Dividing threats makes the exploration process harder, therefore we expect the learning to perform worse in *divided* environments.

The exact encoding of the observation, as it was fed to the neural networks used for reinforcement is presented in Figure 3. The experiment's configuration, additional results, and further analysis can be found in the supplementary.

A. Single-Robot Adversarial Coverage

We utilized single-agent reinforcement learning for comparing the results of single-agent adversarial coverage in simulated and real mode as described in section III-B.

We evaluated Q-learning in several environments, including environments with dividing and flank threats. We set $\alpha = 0.2, 0.05$. The results are presented in figure 4. As expected, the simulated mode is superior compared to the real mode, which improves the results and stabilizes the learning process consistently.

B. Heterogeneous Multi-Robot Adversarial Coverage

We evaluated the HMRAC algorithms (MRAC, HMRAC*, Learning) in multiple environments from different environment types as detailed above. Figure 5 presents the results of the optimization criterion for $\alpha = 0.1, 0.5, 1.0$. Due to space limitations, we present here one representing an environment for every class. In heterogeneous environments, we compared HMRAC* and learning with a version of MRAC that includes the heterogeneous extension, since the basic algorithm cannot run in heterogeneous environments.

Note that as α increases, the maximum possible value of the optimization criteria decreases, because the coverage time's importance increases and time is a non-negative value. In addition, the maximal value of the optimization criterion varies between environments. Hence, the results should be evaluated by the difference between the results, rather than the absolute value.

As expected, HMRAC* outperforms MRAC for all environments and α values, with confidence $p \geq 0.95$ in most experiments. Since MRAC does not consider the coverage time, the coverage paths remain the same for all values of α while the relative importance of the time changes. For that reason, the results of the MRAC significantly drop as α increases - the coverage time remains the same but the cost of the time increases. On the other hand, HMRAC* which considers both the coverage time and the number of covered cells, successfully balances time and coverage success.

Furthermore, we found that learning does not work well for lower values of α (e.g., $\alpha = 0.1$), apparently because the difference in the cumulative reward is not significant

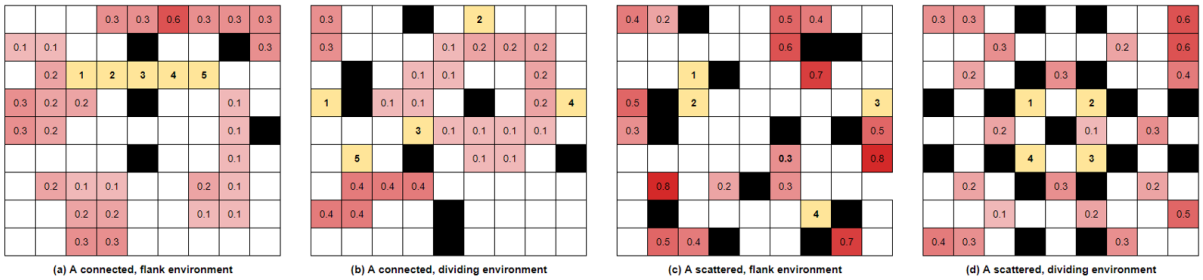

Fig. 2. An illustration of the different types of environments. As demonstrated in (a, b), in connected environments the threats are clustered, while in scattered environments (c, d) they are spread all over the area. As demonstrated in (a, c), flank environments contain threats that allow robots to move freely, while divided environments (b, d) requires the robots to cross threatened area for travel in the environment. Yellow cells represent the starting positions of the robots - five robots in (a, b) and four in (c, d)

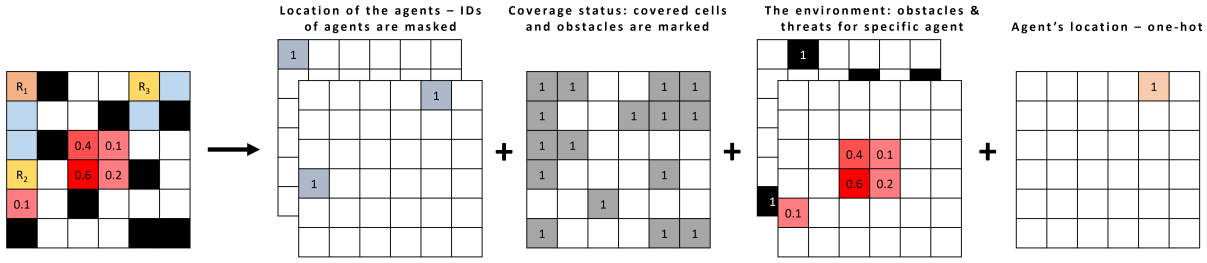

Fig. 3. The encoding of observation as fed to the neural network. The observation consists of the following layers: a one-hot vector for each cell that encodes the type of robot in that cell (if there are no robots, the vector is zeroed); the coverage status of each cell (marked with light-blue in the figure) in the environment, such that obstacles are marked as covered; the risk and obstacles distribution as the observing robot experience them; and a one-hot representing the location of the observing robot.

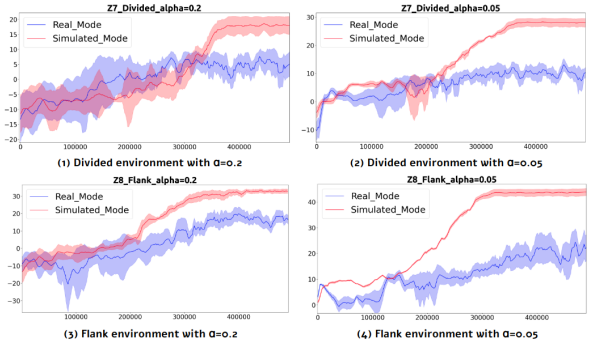

Fig. 4. Value of the optimization criterion in single-agent environments, as obtained by the Q-learning with the real mode (blue) and with simulated mode (red). Note that obstacles do not count as part of the reward, since they are not covered.

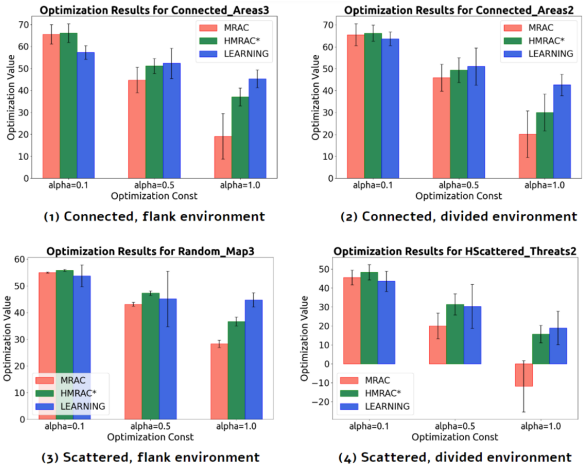

Fig. 5. Value of the optimization criterion in different environments, for (from left to right) $\alpha = 0.1, 0.5, 1.0$.

enough. In some cases, the learning takes a very long time to converge, especially in bigger environments, and frequently converges into a sub-optimal solution. Hence, for lower values of α it is recommended to use HMRAC*, while for higher values of α (e.g., $\alpha \approx 1$) utilizing learning achieves better results.

VI. Conclusions

In this paper, we described the heterogeneous multi-agent adversarial coverage and presented a heuristic algorithm and a learning model for finding optimal paths for the robot that maximizes an optimization criterion that considers both coverage time and coverage success. Nevertheless, the adversarial coverage problem can be extended in other directions, such as time-dependent threats or partial-observable environments, or heterogeneous robots that differ not only in their immunity to threats but also in their travel cost. We believe that future work on these topics may benefit from the ideas presented in this paper.

Acknowledgment

This research was funded in part by ISF grant number 1563/22.

REFERENCES

[1] R. Almadhoun, T. Taha, L. Seneviratne, and Y. Zweiri, "A survey on multi-robot coverage path planning for model reconstruction and mapping," *SN Applied Sciences*, vol. 1, pp. 1–24, 2019.

[2] L. Merino, F. Caballero, J. R. Martínez-de Dios, I. Maza, and A. Ollero, "An unmanned aircraft system for automatic forest fire monitoring and measurement," *Journal of Intelligent & Robotic Systems*, vol. 65, pp. 533–548, 2012.

[3] W. DeBusk, "Unmanned aerial vehicle systems for disaster relief: Tornado alley," in *AIAA Infotech@ Aerospace 2010*, 2010, p. 3506.

[4] J. Walker, "Search and rescue robots–current applications on land, sea, and air," *EMERJ-The AI Research and Advisory Company*, 2019.

[5] R. Yehoshua, N. Agmon, and G. A. Kaminka, "Safest path adversarial coverage," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3027–3032.

[6] ——, "Robotic adversarial coverage of known environments," *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1419–1444, 2016.

[7] R. Yehoshua and N. Agmon, "Multi-robot adversarial coverage," in *Proceedings of the Twenty-second European Conference on Artificial Intelligence*, 2016, pp. 1493–1501.

[8] S. Jorgensen, R. H. Chen, M. B. Milam, and M. Pavone, "The team surviving orienteers problem: routing teams of robots in uncertain environments with survival constraints," *Autonomous Robots*, vol. 42, pp. 927–952, 2018.

[9] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[10] R. Yehoshua, N. Agmon, and G. A. Kaminka, "Frontier-based rtdp: A new approach to solving the robotic adversarial coverage problem." in *AAMAS*, 2015, pp. 861–869.

[11] L. Zhou and P. Tokekar, "Multi-robot coordination and planning in uncertain and adversarial environments," *Current Robotics Reports*, vol. 2, pp. 147–157, 2021.

[12] M. Ishat-E-Rabban and P. Tokekar, "Failure-resilient coverage maximization with multiple robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3894–3901, 2021.

[13] B. Schlotfeldt, V. Tzoumas, D. Thakur, and G. J. Pappas, "Resilient active information gathering with mobile robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4309–4316.

[14] G. Shi, L. Zhou, and P. Tokekar, "Robust multiple-path orienteering problem: Securing against adversarial attacks," *IEEE Transactions on Robotics*, 2023.

[15] A. K. Lakshmanan, R. E. Mohan, B. Ramalingam, A. V. Le, P. Veerajagadeshwar, K. Tiwari, and M. Ilyas, "Complete coverage path planning using reinforcement learning for tetromino based cleaning and maintenance robot," *Automation in Construction*, vol. 112, p. 103078, 2020.

[16] O. Boufous, "Deep reinforcement learning for complete coverage path planning in unknown environments," 2020.

[17] J. Xiao, G. Wang, Y. Zhang, and L. Cheng, "A distributed multi-agent dynamic area coverage algorithm based on reinforcement learning," *IEEE Access*, vol. 8, pp. 33 511–33 521, 2020.

[18] J. Hu, H. Coffin, J. Whitman, M. Travers, and H. Choset, "Large-scale heterogeneous multi-robot coverage via domain decomposition and generative allocation," in *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2022, pp. 52–67.

[19] J. Rückin, L. Jin, and M. Popović, "Adaptive informative path planning using deep reinforcement learning for uav-based active sensing," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 4473–4479.

[20] W. Böhmer, V. Kurin, and S. Whiteson, "Deep coordination graphs," in *International Conference on Machine Learning*. PMLR, 2020, pp. 980–991.