Deploying Teams of Heterogeneous UAVs in Cooperative Two-Level Surveillance Missions

Nicola Basilico   Stefano Carpin

Abstract— **We consider the problem of providing surveillance to a grid area using multiple heterogeneous UAVs, named sentinels and searchers, with complementary sensing and actuation capabilities. We consider probabilistic attacks and we analyze the expected performance with respect to the team deployment. We then introduce the problem of finding minmax deployments that result in the most desirable worst case performance caused by an attack. We present an algorithm to compute deployments while trading off solution's quality and computational effort and we qualitatively and quantitatively analyze it.**

I. Introduction

Intelligence, surveillance, and reconnaissance (ISR) applications continue to receive growing attention from the multi-robot systems research community. In this domain, autonomous robots (and UAVs in particular) can introduce advantages like increased efficiency, resilience and lower risks for humans. Recent research showed how performing surveillance at multiple levels of granularity can be effective. Alongside works where the environment is modeled using multi-scale representations [2], [8], recent interest has been devoted to two-level paradigms, where *broad-area* surveillance provides wide-scope but imprecise detections, triggering *local-investigation* operations to disambiguate such detections by close inspection [3]. Applications include for example wildfire detection [11] and monitoring of agricultural fields [9], where the large extension of the environment makes an accurate whole coverage not viable and requires to balance surveillance scope and accuracy.

In this paper, we use the two-level paradigm on a hierarchical surveillance framework with heterogeneous UAVs we call *sentinels* and *searchers*. Sentinels are tasked with guarding large areas to detect domain-relevant events we shall call *attacks*. Upon detection, a sentinel dispatches one or more searchers to collect local information and continues its monitoring task. In this approach robots undertake complementary tasks. Sentinels fly at higher elevations, are subject to higher error rates, but can scan larger areas of the environment recognizing the presence of an attack. They are, however, unable to gather additional information about its precise location. Searchers, instead, operate at lower elevations and have lower error rates. They are capable of localizing attacks once they are notified by a sentinel, but they lack the initial detection ability. After an attack occurs, sentinels cooperate to dispatch a searcher and localize it.

N. Basilico is with the Computer Science Department, University of Milan, Milan, Italy.

S. Carpin is with the School of Engineering, University of California, Merced, CA, USA.

Building upon our previous work [3], we provide an analytical description of the expected performance of a system of this kind and we leverage such characterization to formulate the problem of computing the optimal system deployment. We take a minmax stance, seeking for the smallest worst damage an attack could cause to the monitored environment, and we define an algorithm for this purpose trying to balance the needed computational efforts with the solution quality. We then experimentally evaluate our solutions with a qualitative analysis and we assess their performance with respect to different parameterizations.

The rest of the paper is organized as follows. Related work is discussed in Section II and Section III introduces our setting. The deployment problem is addressed in Section IV and experiments are presented in Section V. Finally, Section VI concludes the paper.

II. Related work

Search theory goes back to world war II and has been extensively studied in an offline setting [12], [17]. Continuous developments in the area of autonomous vehicles and the recent explosion in the use of UAVs have added further momentum to this line of research. The reader is referred to [6] and references therein for a detailed assessment of the state of the art. In our previous work, we have considered variable resolution search with a single agent [2], [6], [7], [8]. The elevation-resolution tradeoff was first observed in [18], and similar ideas were recently proposed in [15] where the variable resolution coverage methods proved to introduce a number of advantages with respect to state-of-the-art uniform coverage approaches. Our method exploits basic concepts from queuing theory to characterize the performance of the collective search over an area. Such general approach has been introduced by Bullo et al. [5]. However, their method does not consider sensing errors. Other works considering settings where the use of autonomous agents provides uninterrupted monitoring over a search domain are, for example, [10], [16]. With respect to such contributions, we address a more general scenario, allowing teams of heterogeneous agents. The problem of finding an optimal deployment for our system builds on top of the scenario we introduced in [3] where we studied the quality of a fixed and non-overlapping arrangement of sentinels. In this work, we extend such model to obtain a formulation of the deployment problem and we study a method to solve it. To the best of our knowledge, this formulation is novel. Some similarities with well-known environment partitioning methods can be recognized [14].

However, the addressed scenario and method we study is radically different from the classical partitioning problem.

III. Basic setting

Environment. Let $\mathcal{G}$ be the search domain assumed to be a discrete representation of some area $\mathcal{A} \subseteq \mathbb{R}^2$ to be protected and modeled as a rectangular grid partitioned into equally sized square cells. A cell is indicated as c, the coordinates of its center are $(c_x, c_y) \in \mathcal{A}$, and each cell's edge length is denoted with e. Every cell represents some area within which *attacks* can take place. The presence of an attack in cell c produces a penalty that we model with a *loss* value $l(c)$. Such value represents the penalty per time unit incurred by the system while an attack is present in cell c. At each time, multiple attacks may occur in $\mathcal{G}$, and if more than one attack target the same cell at the same time we treat them a single joint attack for that cell. Once a cell is attacked, the attack persists until it is detected. As explained in the following, this will be performed by a searcher agent which tries to localize attacks. Let $a(c,t)$ be a binary function indicating whether cell c is under attack at time t, i.e., $a(c,t) = 1$ if and only if there is an attack in cell c at time t. If we consider a finite time horizon T, the overall loss accrued by the system over the whole environment is defined as follows:

$$\mathcal{L} = \sum_{c \in \mathcal{G}} l(c) \int_0^T a(c,t)dt. \tag{1}$$

This quantity will be our performance metric for evaluating our surveillance system. The objective is to deploy sentinels and searchers to contain the loss $\mathcal{L}$.

Attacks. We model attacks probabilistically, along a temporal and a spatial dimension. First, we model the inter-arrival time between attacks as a random variable with a known probability density function. Second, when an attack takes place, we assume that the targeted cell c is drawn from a probability mass distribution $p(c)$ over $\mathcal{G}$. As common in literature, we assume independent arrivals and $p(c)$ proportional to $l(c)$. Considering a probabilistic model like this can be effective in situations where the attacker's behavior does not depend on the surveillance strategy. Doing otherwise would entail a number of assumptions on the attacker planning and observation capabilities as well as different resolution techniques that in literature often came under game theoretical frameworks [4], [1]. Our model, instead, only requires that the environment is known to an attacker, making it more inclined to target those more profitable regions with higher l values.

Sentinels and searchers. The surveillance team is composed by M sentinels and N searchers which are tasked with localizing and removing attacks in order to contain the overall loss $\mathcal{L}$. Each sentinel is assigned to monitor a rectangular sub-grid of $\mathcal{G}$ denoted as $\mathcal{G}_i$. The ith sentinel repeatedly scans its area with period Δ. Each scan returns a binary reading, indicating whether an attack is present in $\mathcal{G}_i$ or not. We allow the various $\mathcal{G}_i$s not to form a partition of $\mathcal{G}$. In particular, they may have different size, we allow overlaps between areas monitored by different sentinels, and we admit situations where $\mathcal{G}$ is not fully covered by the union of all the $\mathcal{G}_i$s (for example, if it is acceptable not to cover a subset of $\mathcal{G}$ when it is known that no attack will occur therein). This hypothesis marks a strong difference from our recent work presented in [3], where we analyzed only partitions of $\mathcal{G}$ made of equally sized sub-areas. Notice also that in [3] we just considered the problem of analyzing a given deployment, whereas in this work the goal is to construct deployments optimizing an objective function defined in the sequel. Scans operated by each sentinels are affected by errors. We indicate with α_i the probability that sentinel i receives a false positive, and with β_i the probability of a missed detection. As corroborated by the field experiments presented in [6], we assume a correlation between the size of $\mathcal{G}_i$ and the error rates. In particular, larger areas are associated with higher probabilities of getting false positives and missed detections. Due to the possible sensing errors, the outcome of each scan is a random variable. When considering the sequence of readings performed by a single sentinel, we assume that the random variables are independent and identically distributed. Furthermore, we assume independence between readings by different sentinels. Once a sentinel i gets a detection, a searcher is dispatched over $\mathcal{G}_i$ with the task of localizing the possible attack. To accomplish this goal, the searcher scans individual cells of $\mathcal{G}_i$ employing some search strategy. Its scans incur in the same sensing error/area correlation adopted for the sentinel. However, sensors readings for searchers will be more accurate than sentinels' because they scan single cells on the grid. Moreover, each searcher has a limited traveling budget B. That is, if the searcher locates the attack before traveling B distance units, it removes it and terminates its mission. If instead the searcher does not locate any attack within the distance budget, it terminates its mission and reports to the sentinel that no attack was found. For simplicity, we assume that searchers move at constant speed, thus we do not distinguish between temporal and distance costs. Under such assumption the time to travel from a cell to the adjacent one is e.

In this work, we extend [3] by considering multiple sentinels that can be placed in the environment without restrictions, allowing overlaps and uncovered zones. We then leverage our theoretical analysis in dealing with the sentinel placement problem, i.e., determining the area $\mathcal{G}_i$ to associate to each sentinel to bound the loss $\mathcal{L}$. This is a fundamental and challenging requirement for our basic setting, especially when considering realistic deployments. In our analysis, we hypothesize to know the spatiotemporal distribution characterizing the attacks, the error ratios associated with each agent, and the search strategy employed by each searcher. Such hypotheses simplify our model, but also favor analytical tractability allowing us to derive a set of results to evaluate more realistic scenarios from a baseline comparison perspective.

IV. The sentinel placement problem

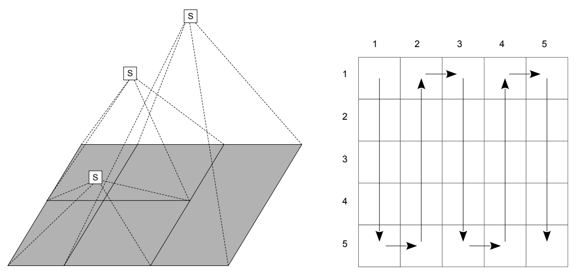
We address the problem of finding a placement for each sentinel such that $\mathcal{L}$ is minimized. Let $s_i = (c_x^i, c_y^i, h^i)$ be the pose of the ith sentinel, meaning that it is positioned at elevation h^i above a cell whose center coordinates are c_x^i and c_y^i (we will omit superscript i when not necessary). Let $\mathcal{S}$ be a set of n allowable poses. Formally, a deployment E is defined as $E = (s_1, s_2, \dots, s_M)$, with $s_i \in \mathcal{S}$. Note that although in the sequel it may seem that E is a sequence, it is in fact a subset of $\mathcal{S}$, so no repeated elements are allowed (therefore there cannot be two sentinels in the same place) and the order does not matter because all sentinels have the same sensing capabilities and can then be swapped. We start by defining a method to evaluate the value of a single sentinel's pose and we extend such method to the evaluation of a whole deployment of M sentinels. Assuming each sentinel is equipped with the same sensor, e.g., a downward pointing camera, there is a one-to-one correspondence between $s_i = (c_x^i, c_y^i, h^i)$ and the assigned area $\mathcal{G}_i$ (see Figure 1(a)). Hence, we will use s_i as a proxy for $\mathcal{G}_i$, and *vice versa*.

To evaluate the performance of a sentinel located at some pose s we need to provide a description of the process generating the attacks. As common in literature, we hypothesize that the inter-arrival time between attacks is distributed according to an exponential random variable with intensity λ. Each attack will target cell c with probability $p(c) = \frac{l(c)}{\sum_{c \in G} l(c)}$. Against this background, we define how to compute an upper bound for $v_s(c)$, that is the expected loss that a single sentinel located at s would expect to receive from cell c. Such quantity is formally defined as:

$$v_s(c) = \mathbb{E}\left[l(c) \int_0^T a(c,t)dt \right]. \qquad (2)$$

Let $\mathcal{G}_i$ be the area associated with s_i. Every cell in $\mathcal{G}_i$ can be identified with a row and a column indexes $1 \le i \le I$ and $1 \le j \le J$, respectively, and we will indicate as $c_{i,j}$ the cells in $\mathcal{G}_i$ (notice that I and J also denote the height and width of sub-grid $\mathcal{G}_i$). Once dispatched, a searcher will cover $\mathcal{G}_i$ using a lawn mower pattern, i.e., it starts sweeping from cell $c_{1,1}$ up to cell $c_{I,1}$, then moves to cell $c_{I,2}$ sweeps up to cell $c_{1,2}$ and so on until cell $c_{I,J}$ is eventually reached (see Fig. 1(b)). At that point it reverses its direction of travel and repeats the same pattern in the opposite direction. We assume that moving from one cell to an adjacent one takes a travel cost of e and that such back-and-forth sweep pattern is repeated until either the whole budget B is consumed or an attack is found. Finally we assume searchers operate independently from one another.

Requiring each searcher to follow this predetermined non-informed sweep strategy is a simplification we introduced to make the theoretical analysis of the model's performance more tractable. Indeed, characterizing the possible probabilistic realizations of an informed (on-line) search strategy as well as the synergies arising when two or more searchers operate in the same sub-grid is not an easy task and would narrow the scope of our findings to the search strategies obeying to such assumptions. Instead, we can safely state that an upper bound for $v_s(c)$ under such sweep strategy is



(a) (b)

Fig. 1. (a) A scheme of sentinels deployment (surveillance area in gray); (b) the sweep pattern of searchers (in this example $I = J = 5$).

an upper bound for other better informed search methods, since predetermined fully covering patterns like this can be thought as baseline methods. In addition, lawn mower patterns are strategies currently used in many domains, e.g., maritime search. Operatively, an upper bound for $v_s(c)$ can be computed as $l(c)W_c$, where W_c is an upper bound for the expected waiting time of an attack in c before it is detected. To compute W_c we need to account for the following terms:

- the time between the arrival of an attack at $c \in \mathcal{G}_i$ and a scan performed by sentinel i, we denote it as $\zeta \in [0, \Delta]$;
- given that an attack is in $\mathcal{G}_i$, the number of scans that a sentinel has to perform before dispatching the first searcher, we denote it as $t_{tr} > 0$;
- given that an attack is present in $\mathcal{G}_i$, the number of scans that a sentinel has to perform before dispatching the ith searcher with $i > 1$, we denote it as $t_{tr}^i > 0$;
- the number of dispatches (searchers) that a sentinel has to perform to obtain the first *successful* dispatch, that is dispatching the searcher that will detect the attack, we denote it as n_s;
- the time employed by the successful searcher to detect the attack, we denote this value as s.

By calling $\lambda_c = \lambda p(c)$ the attack arrival rate limited to cell c, and N the probability that an attack takes place somewhere in $\mathcal{G}_i$ during an interval of Δ time units, we can compute $\mathbb{E}[\zeta]$, $\mathbb{E}[t_{tr}^1]$, and $\mathbb{E}[t_{tr}]$ (see [3] for details):

$$\mathbb{E}[\zeta] = \Delta - \frac{1 - e^{-\lambda_c \Delta} - \lambda_c \Delta e^{-\lambda_c \Delta}}{\lambda_c (1 - e^{-\lambda_c \Delta})}$$

$$\mathbb{E}[t_{tr}] = \frac{1}{1 - (1-N)(1-\beta_i(h)) + N\alpha_i(h)}$$

$$\mathbb{E}[t_{tr}^1] = 1 - \beta_i(h) + \frac{\beta_i(h)}{1 - N\alpha_i(h) - (1-N)(1-\beta_i(h))}.$$

Notice that we made the dependency from the sentinel's altitude h explicit in the error rates $\alpha_i(h)$ and $\beta_i(h)$.

Given the sweep pattern, any cell c is guaranteed to be scanned at least m times, where $m = \left\lfloor \frac{B}{(|\mathcal{A}_s|-1)e} \right\rfloor$. Then, the expected number of dispatches required to have the successful one is given by the mean of a geometric variable as $\mathbb{E}[n_s] = \frac{1}{1-\beta_s^m}$. where we indicated with β_s the false negatives rate for a searcher (we do not highlight the dependency from the altitude since we assume that every searcher operates at the same constant altitude.) Finally, the expected time the successful searcher has to spend to detect the attack in a cell $c_{x,y} \in \mathcal{G}_i$ can be obtained defining the following function:

$$D(k) = \begin{cases} D_1 & \text{if } k \text{ is odd} \\ D_2 & \text{otherwise} \end{cases}$$

where D_1 is the time (distance) required to travel from cell $c_{1,1}$ to cell $c_{x,y}$ whereas D_2 is the time (distance) between cells $c_{x,y}$ and $c_{I,J}$. Thus, we define the time at which the successful searcher performs the k-th visit to cell $c_{x,y}$ as $t(k) = D(k) + (k-1)(IJ-1)e$. Therefore, by applying the definition of expected value we have that

$$\mathbb{E}[s] = \sum_{k=1}^{m} t(k)\beta_s(h_s)^{k-1}(1-\beta_s(h_s)).$$

We can now derive the following upper bound:

$$v_s(c) \leq W_c\big(\mathbb{E}[\zeta] + \mathbb{E}[t^1_{tr}] + (\mathbb{E}[n_s]-1)\mathbb{E}[t_{tr}] + \mathbb{E}[s]\big). \quad (3)$$

We will refer to the right hand side of the previous inequality as $\bar{v}_s(c)$. For every cell $c \in \mathcal{G}_i$ such value is computed as per Eq. 3 whereas when c does not fall in the area covered by the sentinel we set $\bar{v}_s(c) = \infty$ if $l(c) > 0$ and $\bar{v}_s(c) = 0$ otherwise. This choice of values models the fact that when a cell is not covered by the sentinel, arbitrarily large penalties can be incurred (recall that here we consider a single sentinel) while if the cell will not be subject to attacks, its expected penalty will be null independently on whether it is covered or not.

Given such values, we introduce our definition of optimal deployment as the one that minimizes a value function which, for a given deployment E, is defined as follows:

$$v(E) = \max_{c \in \mathcal{G}} \min_{s \in E} \bar{v}_s(c). \quad (4)$$

The rationale behind Eq. 4 is the following. Given a cell c, different sentinels will cover it depending on their poses in E, inducing different values for the expected loss upper bound. Combining such values in some aggregation function would require to consider synergy relations arising between sentinels sharing the same cell c. Instead of trying to model such dynamics we take a bounding approach, considering the minimum value, that is, the best performance achieved by one of the sentinels above c. Since we are not accounting for interferences, such value constitutes an upper bound for c's expected loss when multiple sentinels operate on it. Then we take the maximum to adopt a worst-case stance, namely we define the performance of a deployment E as the worst penalty a single attack can cause. In Table I, we show a straightforward numeric example of the application of such method in a small setting with 4 cells and 4 candidate poses (rows correspond to deployments, columns to cells and the values denote the obtained expected loss). Obviously,

	c_1	c_2	c_3	c_4	$v(\{s_i\})$
s_1	11	15	16	13	16
s_2	10	18	20	12	20
s_3	12	16	12	10	16
s_4	∞	4	7	∞	∞

	c_1	c_2	c_3	c_4	$v(E)$
$E = \{s_1, s_2, s_3\}$	10	15	12	10	15
$E = \{s_1, s_2, s_3, s_4\}$	10	4	7	10	10

TABLE I
SIMPLE EXAMPLE ON THE APPLICATION OF EQ. 4.

searching for the deployment providing a best worst case is not the only rationale that can be followed when computing an effective deployment of sentinels, and several different functions can be proposed as an alternative to Eq. 4. However, such a function does not require any parametrization, simplifying the task of analyzing its solutions.

Algorithm 1 FindDeployment(ρ)

```
1: $E = \{\}$
2: for all $k \in \{1 \dots \lceil \frac{M}{\rho} \rceil\}$ do
3:     $\mathcal{Q} \leftarrow$ set of deployments with $\min\{\rho, M - \rho(k-1)\}$ sentinels
4:     $Q^* = \arg\max_{Q \in \mathcal{Q}} \Delta_v(Q \mid E)$
5:     $E = E \cup Q^*$
6: end for
```

In Algorithm 1, we provide a method to compute deployments for teams of sentinels which makes use of the evaluation method we defined. The algorithm decomposes the problem of finding an optimal deployment for M sentinels into a number of subproblems where, at each of them, an optimal deployment Q^* on a number $\rho \in [1, M]$ of sentinels is computed. The optimality of a deployment Q at the ith iteration is defined by the maximization of the marginal gain obtained in v when including Q in the union of the previously computed deployments (E). Such quantity is defined as $\Delta_v(Q \mid E) = v(E) - V(E \cup Q)$. Clearly, such decomposition of the problem is not without loss of optimality, but it enables a trade-off between the required computation time and the obtained solution's quality, which is controllable by means of parameter ρ. When $\rho = 1$ the algorithm is purely greedy, whereas when $\rho = M$ the algorithm performs a brute force search over set of all possible deployments. It is useful to keep these two extremes in mind when considering the computational complexity and optimality. Notice that, contrarily to what intuition would suggest, function v is not submodular. Considering again the example of Table I, it easy to see that adding pose s_4 to $\{s_1, s_2, s_3\}$ yields a strictly larger marginal gain than adding the same pose to $\{s_1, s_2\}$. As a consequence, no simple approximation bound can be derived for the greedy version of the algorithm (see [13] for a discussion on this subject).

A. Computational complexity

We recall that M is the number of sentinels and $n = |\mathcal{S}|$ is the set of possible sentinel locations. Let $g = |\mathcal{G}|$ be the number of cells in the grid $\mathcal{G}$. For $\rho = 1$ the algorithm greedily grows E by adding sentinels one after one. At each of the M iterations the algorithm computes $\mathcal{O}(ng)$ values for $\bar{v}_s(c)$ (see Eqs. 2, 3 and 4) to determine where to place the next sentinel and so the overall complexity is $\mathcal{O}(Mng)$. However, a more efficient computation for the greedy method is possible. In time $\mathcal{O}(ng)$ one can precompute the value $\bar{v}_s(c)$ for each combination of cell in $\mathcal{G}$ and pose in $\mathcal{S}$ (see Eqs. 3 and 4). These values can be though as arranged in a table with n rows and g columns. Then, for each of the n we can determine the maximum in complexity $\mathcal{O}(ng)$. After this step, for each row (i.e., sentinel pose), we have determined the cell in $\mathcal{G}$ giving the worst performance (highest loss). At this point the algorithm needs to pick the M rows with the worst case largest value, and this can be efficiently done by sorting the maximum values and extracting M rows with the largest values. Adding all complexities together, the greedy approach can then be implemented with time complexity $\mathcal{O}(ng + n \log n)$.

At the other end of the spectrum, when $\rho = M$ we have the optimal brute force algorithm evaluating each of the $\binom{n}{M}$ possible deployments. Assuming again that one precomputes all values for $\bar{v}_s(c)$ in time $\mathcal{O}(ng)$ and stores the maximum of the row, its complexity is then $\mathcal{O}(n!)$ and is evidently intractable for cases of practical interest.

The intermediate solution allocating blocks of ρ sentinels at a time, has complexity $\mathcal{O}(\frac{M}{\rho}\binom{n}{\rho})$. This result follows because we solve by brute force $\frac{M}{\rho}$ instances of the problem with ρ sentinels and each subproblem implies the evaluation of $\binom{n}{\rho}$ candidate solutions. For small values of ρ the overall complexity can be approximated as $\mathcal{O}(Mn^{\rho})$. Note that in this case we assumed that the precomputation time $\mathcal{O}(ng)$ is dominated and then ignored.

V. Experimental evaluation

We considered a variable number of sentinels with the task of protecting a 16×16 grid. Since we assume that UAVs are the reference implementation for our system, we faced the problem of defining a realistic sensing model. To this aim, we adopted a model inspired by the one we studied in our previous work [6] where the obtained data were supported by field experiments with a real UAV. More precisely, we considered five different altitudes reserving the lowest one to searchers. In the same way, false positive and missed detection rates trends are derived from the data we gathered in such experimental campaign. For each altitude, we considered 16^2 candidate poses, each one of them assumed to be above the center of a cell of $\mathcal{G}$. Notice that the resolution of $\mathcal{S}$ is a sensitive parameter for the problem we are trying to solve. Obviously, fine grained discretizations would led to better solutions whereas having coarse spaces can introduce potentially arbitrary suboptimality. On the other hand having a sparse space of locations can be advantageous in terms of computation time. Determining an optimal density for $\mathcal{S}$ is an interesting problem which can be tackled exploiting different properties of our setting. For example, it is easy to show that there exists an altitude $\bar{h}$ such that poses above it

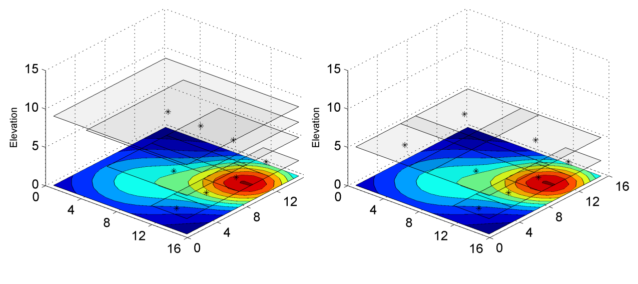
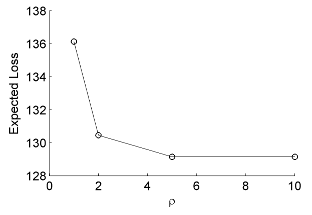


Fig. 2. Example of two deployments found with Algorithm 1 for a team of $M = 8$ sentinels with $\rho = 1$ (a) and $\rho = M$ (b).

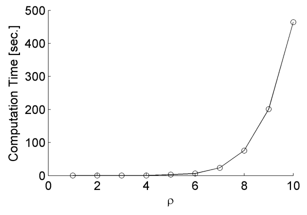
can be safely discarded and that a principle of locality holds, that is close locations tend to provide the same performance suggesting that good solutions should show some kind of uniform spatial distribution. We leave the study of such problem as future work and we concentrate instead on what results could be obtained with our method given $\mathcal{S}$ defined as described above.

An example of deployment found with our method is depicted in Fig 2. In this experimental phase, we used a team of 8 sentinels and a bimodal loss distribution obtained by adding two Gaussian functions. In the figures, the asterisks represent sentinels while gray rectangles depict sub-areas $\mathcal{G}_i$s. We can qualitatively compare the deployment found by setting $\rho = 1$ in Algorithm 1 (Fig. 2(a)) against the optimal deployment found through exhaustive search (Fig. 2(b)). The first feature that can be observed is that both deployments tend to concentrate more resources above the areas with higher loss, indeed both allocate a number of sentinels at low altitude in order to contain the expect loss from the cells which are likely to cause more damage if under attack. A difference can be noticed when considering how the two cover the environment as a whole. The greedy method, being forced by construction to select a pose at a time, first tries to guarantee complete coverage by selecting a high altitude pose, then it tries to refine the solution in the same iterative way. The optimal method, instead, was able to find a better allocation to guarantee complete coverage in which no sentinel has to dispatch searchers on the whole grid. Allocating sentinels at lower altitudes, the optimal deployments provides potential benefits also in terms of energy consumption. This is a trend we observed rather frequently when comparing greedy (and, in general, those found with $\rho < M$) with optimal solutions.

Fig. 3 shows the trade-off between solution's quality and the required computational effort and gives some insights on how to tune ρ. We considered a team of 10 sentinels with random loss distribution where each $l(c)$ is drawn from with uniform probability from $[1, 100]$ (each point is the average over 10 different runs). Fig. 3(a) shows how the solution's quality converges to the optimum as we adopt increasing values of ρ while Fig. 3(b) shows the corresponding computation times exposing the exponential growth rate as Algorithm 1 performs exhaustive searches. The trade-off



(a)



(b)

Fig. 3. Values and computation times obtained for different values of ρ with $M = 10$ sentinels and random loss distribution.

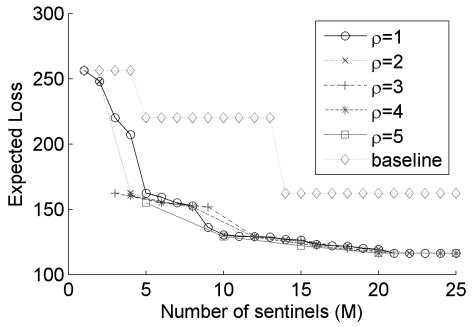


Fig. 4. Varying the number of sentinels for different values of ρ.

described by such curves suggests that we could obtain fairly good solution within a reasonable time (for the settings we considered, the solution improvements are small for $\rho > 5$).

Finally, Fig. 4 depicts how the expected loss improves when adding more sentinels in a setting with a random loss distribution. The graph gives an insight on the trend of marginal gains achieved by the different iterations of Algorithm 1 and it provides an assessment on how profitable adding a sentinel to an already deployed team can be. As it can be seen, the curves for different values of ρ converge towards the same optimal value entailing the existence of a maximum number of sentinels beyond which no significant improvement can be obtained. This can be an important aspect when choosing the number of sentinels to deploy to protect a given environment. The figure also shows results with a non-informed baseline deployment strategy that works as follows. It deploys sentinels one by one filling altitude levels from the the highest to the lowest. A level is considered filled as soon as each cell of the environment can be perceived by at least one sentinel in that level (locations are chosen trying to keep the amount of cells in $\mathcal{G}_i$ the same for each sentinel i). The comparison with such method shows how, in general, non-informed strategies achieve poor performance when compared to informed ones. The steps that can be observed in the curve correspond to situations where the added sentinel starts seeing the cell causing the maximum expected loss. Since we use a random loss distribution, such maximum value is provided by more cells, randomly scattered on the grid. A drop in the penalty is then observed each time a level is filled.

VI. Conclusions and future work

We studied the problem of finding deployments for a surveillance team composed by heterogeneous robots that we called sentinels and searchers. Sentinels persistently monitor large portions of the environment and dispatch searchers once the presence of an attack is detected. Searchers have to localize the attack after having been dispatched. We defined the problem of finding the optimal deployment as the one that results in the minimum penalty against the most damaging attack and we provided a simple algorithm to search for it.

In the future, we will include heuristic techniques to reduce the size of the set of candidate locations trying to bound the loss in terms of solution quality. Along the same lines, comparisons with other evaluation approaches, other than the minmax, are possible, including the use of functions which could achieve some degree of approximation when optimized with simple methods (e.g., submodular functions).

References

[1] N. Agmon, V. Sadov, G. Kaminka, and S. Kraus. The impact of adversarial knowledge on adversarial planning in perimeter patrol. In *AAMAS*, pages 55–62, 2008.

[2] N. Basilico and S. Carpin. Online patrolling using hierarchical spatial representations. In *ICRA*, pages 2163–2169, 2012.

[3] N. Basilico, T.H. Chung, and S. Carpin. Distributed online patrolling with multi-agent teams of sentinels and searchers. In *DARS*, 2014.

[4] N. Basilico, N. Gatti, and F. Amigoni. Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder. *ARTIF INTELL*, 184–185:78–123, 2012.

[5] F. Bullo, E. Frazzoli, M. Pavone, K. Savla, and S.L. Smith. Dynamic vehicle routing for robotic systems. *Proceedings of the IEEE*, 99(9):1482–1504, 2011.

[6] S. Carpin, D. Burch, N. Basilico, T.H. Chung, and M. Kölsch. Variable resolution search with quadrotors: theory and practice. *J FIELD ROBOT*, 30(5):685–701, 2013.

[7] S. Carpin, D.A. Burch, and T.H. Chung. Searching for multiple targets using probabilistic quadtrees. In *IROS*, pages 4536–4543, 2011.

[8] T.H. Chung and S. Carpin. Multiscale search using probabilistic quadtrees. In *ICRA*, pages 2546–2543, 2011.

[9] A.-J. Garcia-Sanchez, F. Garcia-Sanchez, and J. Garcia-Haro. Wireless sensor network deployment for integrating video-surveillance and data-monitoring in precision agriculture over distributed crops. *Computers and Electronics in Agriculture*, 75(2):288 – 303, 2011.

[10] V. A. Huynh, J. J. Enright, and E. Frazzoli. Persistent patrol with limited-range on-board sensors. In *CDC*, pages 7661–7668. IEEE, 2010.

[11] B.C. Ko, J.O. Park, and J.-Y. Nam. Spatiotemporal bag-of-features for early wildfire smoke detection. *Image and Vision Computing*, 31(10):786 – 795, 2013.

[12] B.O. Koopman. The Theory of Search, Part {II}. Target Detection. *OPER RES*, 4(5):503–531, 1956.

[13] A. Krause and C. Guestrin. Submodularity and its applications in optimized information gathering. *ACM TIST*, 2(4), 2011.

[14] M. Pavone, A Arsie, E. Frazzoli, and F. Bullo. Distributed algorithms for environment partitioning in mobile robotic networks. *IEEE T AUTOMAT CONTR*, 56(8):1834–1848, 2011.

[15] S.A. Sadat, J. Wawerla, and R.T. Vaughan. Recursive non-uniform coverage of unknown terrains for uavs. In *IROS*, 2014.

[16] M. Schwager, B.J. Julian, M. Angermann, and D. Rus. Eyes in the sky: Decentralized control for the deployment of robotic camera networks. *Proceedings of the IEEE*, 99(9):1541–1561, 2011.

[17] L.D. Stone. *Theory of optimal search.* MAS of INFORMS, 2nd edition, 2007.

[18] S. Waharte, A. Symington, and N. Trigoni. Probabilistic search with agile UAVs. In *ICRA*, pages 2840–2845, 2010.