Online Patrolling Using Hierarchical Spatial Representations

Nicola Basilico and Stefano Carpin

Abstract— **Unmanned Aerial Vehicles (UAVs) can be an effective technology for security applications involving patrolling and search missions. Defining online patrolling strategies for UAVs presents challenges related both to classical patrolling, as periodic monitoring of the environment, and to search, as accurate localization and identification of the mission-related activities. In this paper, we deal with this problem considering probabilistic intrusions and a variable resolution sensing model that naturally applies to the domain of UAVs. We present three online single–robot patrolling strategies exploiting a variable resolution paradigm to represent the environment that has recently shown promising results for search problems. The approach uses a hierarchical representation based on probabilistic quadtrees that allows UAVs to tradeoff sensing accuracy with sensing area. The model is extended by adding stochastic arrivals of intruders in space and time. Obtained results validate this approach for online patrolling against approaches based on uniform grids.**

I. Introduction

In this paper, we show how our recently introduced probabilistic quadtrees (PQ) for search can be conveniently used to solve the more general and harder patrolling problem. As such, this contribution places itself at the intersection between problems known in literature as *patrolling* and *search*. Patrolling is the task of monitoring an environment to protect it from malicious activities denoted as intrusions or attacks. The use of Unmanned Aerial Vehicles (UAVs) for patrolling missions is already ongoing, but thus far these vehicles are usually remotely controlled by a human operator. However, there is a great interest in deploying fully autonomous UAVs, or to provide UAV operators with decision support systems that could provide online hints about good *patrolling strategies*. The design of effective patrolling strategies is a challenging task. A patrolling strategy can be generally defined as the mechanism used to decide how the UAV should move during the mission. Often times this problem is modeled as periodic information gathering via exploration and solved in an offline fashion. Proposed approaches range from frequency–constrained exploration to more complex scenarios where game–theoretical adversarial settings are studied (see Section II for some pointers to related work).

Search is the problem of localizing one or more stationary items in a given environment (the case of moving targets is usually denoted as the pursuit problem, and will not be considered herein). Also in this field, different techniques to design strategies for UAVs have been proposed. Differently from patrolling, these approaches typically tackle the problem online, considering realistic sensing models in probabilistic frameworks, often using a Bayesian approach. Recent works [4], [5] showed that variable–resolution environment representations well adapt to UAV–based search, improving both search effectiveness and computational time. In essence, these methods explicitly consider the tradeoff between sensing a large area with a reduced accuracy, versus sensing a smaller area with a higher accuracy. This kind of tradeoff is particularly relevant for vertical take-off and landing (VTOL) UAVs that can hover above areas of interest while varying their elevation.

In this work, we study strategies for *online patrolling*, a problem sharing features with both patrolling and search. In the scenario we consider, a single patroller operates in an environment characterized by areas with different importance and attack patterns. When an intruder enters the environment, the patroller incurs in a penalty that linearly grows over time, and is proportional to the importance of the area where the intruder broke into. In order to stop paying the penalty, the intruder has to find out where the intrusion occurred and then remove the intruder. The overall objective is to minimize the penalty accrued over long periods of time. A precise definition of these terms will be given in Section III. The contribution of this paper are three. First we formulate the online patrolling problem and we note its differences with the recently formulated persistent patrol problem. Next, we show how PQs lend themselves to the definition of online patrolling strategies incorporating both information available a-priori with evidence gathered during the patrolling effort. Finally, we experimentally contrast these strategies among themselves, and we show how they dramatically overcome strategies based on uniform representations.

II. Related Work

Research into algorithms to design effective patrolling and search strategies is abundant and considers numerous variants. We here discuss only selected contributions and outline differences motivating this study.

Proposed techniques for patrolling strategies can be roughly defined in two classes. Algorithms in the first class focus on providing exploration patterns aiming to optimize some objective function. Widely adopted objective functions include efficient coverage, frequency or idleness optimization, as well as approaches oriented to information decay minimization. One example of work in this class is [7], where strategies based on different definitions of idleness are studied. Approaches in the second class consider patrolling

The authors are with the School of Engineering, University of California, Merced. email: {nbasilico,scarpin}@ucmerced.edu. This work is partially supported by the Office of Naval Research under grant #N00244-11-1-002. The views and conclusions in this article are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ONR.

problems as adversarial scenarios. Patroller and intruder are modeled as rational agents competing in a game. The patrolling strategy is determined then by computing the game's equilibrium under some solution concept. Recent works in this class are [1], [10]. These papers share the assumption of a given intrusion model (implicit or explicit) that is fully known in advance. Therefore, once a strategy is computed, its realization does not depend on additional information collected during the mission. As such, these can be seen as *open-loop* methods, i.e., they do not take advantage of information collected during the patrolling effort. For this reason, they do not align with the spirit of this work which aims to provide closed-loop patrolling strategies. Moreover, realistic sensing models and actions typically fall outside their scope.

Under very idealistic assumptions, our problem shares some aspects with dynamic vehicle routing problems in robotic networks. Bullo et al. [3] give an insightful study of this problem, providing an extensive analysis about the properties of different strategies. However, the strong assumption that targets entering the patrolled area are immediately spotted and the fact that no sensing process is assumed make this problem fundamentally different than the one addressed here. Perhaps, the closest work has been recently proposed by Frazzoli et al. [8], where the persistent patrol problem (PPP) is studied. The authors consider a situation where intruders are not immediately detected when they enter the patrolled area, but only when after they fall inside the sensing range of one of the searchers. However, sensors errors are neglected, i.e., an intruder is revealed as soon as it falls inside the sensing range (and only if that happens, i.e., there are no false positives).

Our loss function is minimized by minimizing waiting time, i.e., the time interval between the moment an intruder enters the patrolled area and the moment it is detected and removed. This problem is related to the *dynamic traveling repair problem* (DTPR) introduced by Bertsimas and Van Ryzin [2]. In DTRP one aims to minimize the waiting time between a stochastic arrival of an element in the environment and the moment it is visited (or serviced). Even the deterministic, simpler version when all demands are known upfront is known to be NP-complete [11]. Note that the problem we are solving is not simpler, because arrivals are not immediately revealed, but have to be discovered by the searcher. Moreover, simply visiting a location where an intruder (demand) is located does not necessarily eliminate it, because of the inherent noise in the sensing process.

The problem of localizing targets in an environment collecting noisy perceptions has been extensively studied in the robotic literature. Recent works [4], [5] investigated, with promising results, the use of variable–resolution environment representations. In this paper we extend this model for the problem of online patrolling.

III. Online Patrolling

We start by describing the online patrolling problem on a uniform grid, and we defer its extension to a hierarchical setting to a later section. This problem differs from the PPP defined in [8] inasmuch as it accounts for sensing errors, either in the form of false positives or missed detections. In the following, we will use the terms agent, patroller, searcher as synonyms. Similarly, the words attacker and intruder will be used interchangeably. The patrolling setting considered in this work is defined over a square area $\mathcal{A}$ composed by L^2 equally sized square cells. Time is discrete and develops in steps. The area $\mathcal{A}$ is subject to malicious activities, or *attacks*, that can start at any time in any cell and last until they are detected by the patroller. The function $l : \mathcal{A} \to \mathbb{R}^+$ assigns a *loss* to every cell. For a cell $c \in \mathcal{A}$, $l(c)$ represents the penalty incurred by the patroller in a single time step due to the presence of an intruder in c. Let $a : \mathcal{A} \times T \to \{0, 1\}$ be the function describing whether cell c is attacked at time t or not, i.e. $a(c, t) = 1$ if and only if there is an intruder in c at time t. Under these hypotheses, the overall loss incurred by the patroller over the period $[0, T]$ is given by

$$\rho = \sum_{t=0}^{T} \sum_{c \in \mathcal{A}} a(c, t) l(c). \tag{1}$$

The goal of the patroller is to minimize ρ, i.e., to minimize the cumulative loss due to attackers entering the patrolled area. To complete the formalization of this problem, we need to formulate a model for the attackers, i.e., how they enter $\mathcal{A}$, and also an action space for the patroller, so that it can detect and remove attackers.

We assume attacks are probabilistically distributed in time and in space. More precisely, an attack is defined by a pair (t, c), where t is the arrival time and $c \in \mathcal{A}$ is the attacked cell. Arrivals are independent and their interarrival time is determined by a Poisson process with known rate λ (note that this is a general, largely adopted model; see e.g. [3]). Once an attack arrives in the environment, the attacked cell is determined by randomly selecting a cell from $\mathcal{A}$ with a mass distribution proportional to the loss function l. In other words, the attacker is more likely to pick a cell associated with a higher loss.

In order to minimize the loss ρ defined in Eq. 1, the patroller needs to localize and remove attackers. Therefore it performs two operations, namely *sensing* to determine whether an attacker is in a given cell, and *clear* to remove an intruder from a given cell. The patroller is equipped with a binary sensor returning $Z = 1$ if at least an attacker is perceived inside the sensed cell and $Z = 0$ otherwise. The sensor is assumed to be faulty, i.e., to return both false positives and missed detections. Error rates are stationary and known. In the following we will indicate with α the false positive rate and with β the missed detection rate.

Embracing a Bayesian view, we assume that before the patrolling effort starts the patroller is provided[1] with a prior about the probability that attackers are located inside each cell at time $t = 0$. Then, querying the sensor during the mission, a posterior about this probability can be propagated

[1]If this knowledge is not available, then one would just start with a uniform uninformative prior.

over time by using standard Bayes updates (see e.g., [4]). More specifically, let $\phi^{(t)}(c)$ with $c \in \mathcal{A}$ be the probability of having an attack in cell c at time t.

The distribution of attacks in time requires the patroller to account for arrivals that may take place during the mission and that may occur also in already visited areas. This dynamic phenomenon can be accounted for by a *detrimental* effect that increases ϕ over time. Formally, for every cell c let be p_c be the probability of having an attack in c within one time unit. Assume the last sensing in c occurred at time t and let $\phi^{(t)}(c)$ be the posterior updated after integrating such reading. Then, assuming no sensing is made in c for the next Δt time steps, we have:

$$\phi^{(t+\Delta t)}(c) = 1 - (1 - \phi^{(t)}(c))(1 - p_c)^{\Delta t}. \tag{2}$$

From the assumptions formerly made about spatial and temporal distributions of the attacks, it is clear that p_c is a function of $l(c)$ and λ.

To remove an intruder from the environment, the patroller performs a *clear* action, e.g. it dispatches a human to make contact with the intruder. The searcher's decision to initiate a clearing action in cell c at time t is modeled with a binary variable D_c^t where $D_c^t = 1$ means clear c at time t and $D_c^t = 0$ otherwise. The value of this variable is determined taking into account the last sensing measurement Z performed at time t in cell c (recalling that $Z = 1$ if an intruder is sensed). Given that $a(c,t) = 1$ if an attack is present in cell c at time k and $a(c,t) = 0$ otherwise, and called C_{ij} the cost of taking decision $D_c^k = i$ when $a(c,t) = j$, then $D_c^t = 1$ if (see [9]):

$$\frac{P[Z|a(c,t)=0]}{P[Z|a(c,t)=1]} \leq \frac{(C_{11} - C_{01})\phi^{(t)}(c)}{(C_{00} - C_{10})(1 - \phi^{(t)}(c))}. \tag{3}$$

If an attack is present in cell c at time t and the patroller performs a clearing action (i.e., $D_c^t = 1$), such attack is removed and $a(c,t')$ is set to 0 for $t' > t$, although this value might be set to 1 again if a new arrival occurs at c. The introduction of costs C_{ij} prevents the definition of trivial strategies where the searcher decides to clear each cell it visits because such operation comes at no cost.

IV. Online Patrolling on a Uniform Grid

Representing the environment with a uniform grid is the most basic and classic approach to address online patrolling. In this section, we introduce a simple myopic patrolling strategy that will serve as comparative reference for strategies operating on the more efficient hierarchical representations. Moreover, this strategy also unveils some interesting aspects of the online patrolling problem.

We define a cell evaluation function $\mathcal{L}(c)$, quantifying the *goodness* of cell c, given the current time t and the current location q of the patroller. In this context goodness is measured as the ratio between saved and incurred expected loss if the next sensing action will take place in c. More formally, we denote with $L_t(c, \Delta t)$ the expected penalty accumulated by the searcher if not sensing cell c in the time interval $(t, t + \Delta t]$:

$$L_t(c, \Delta t) = l(c) \sum_{z=1}^{\Delta t} \phi^{(t+z)}(c).$$

Note that $\phi^{(t+z)}(c)$ is obtained by applying the detrimental factor as reported in Eq. 2. Indicating with $T(q,c)$ the time[2] needed to move between two cells q and c, then the evaluation function is defined as:

$$\mathcal{L}(c) = \frac{P[D_c^{k+T(q,c)} = 1] l(c)}{\sum_{w \in \mathcal{A}} L_t(w, T(q,c))}.$$

The numerator computes the expected saved loss, i.e., the loss of cell c weighted by the probability of clearing the attack (notice that this probability accounts also the probability that an attack is present). This quantity is then divided by the expected amount of penalty accrued while traveling to c. Throughout the mission, then, the next sensing location $g \in \mathcal{A}$ is selected as $g = \arg\max_{c \in \mathcal{A}} \mathcal{L}(c)$.

V. Probabilistic Quadtrees for Search

Carpin and Chung recently introduced so-called probabilistic quadtrees (PQ) for search in [4], [5]. We here shortly summarize this data structure and we refer the reader to the aforementioned papers for a detailed description, and to Fig. 1 for a graphical depiction of the idea. PQs are introduced to model scenarios where one can tradeoff sensing accuracy with sensing area, i.e., one has to decide between scanning a large area with lower accuracy, or a smaller area with larger accuracy. This observation was already made in [13] and naturally applies to VTOL vehicles that can hover over interesting areas while ascending or descending and then varying their sensing accuracy. Quadtrees are standard geometric data structures used in computational geometry [6]. Given an uniform grid, it is possible to associate with it a quadtree such that its leaves at the deepest level coincide with the cells of the uniform grid. A probabilistic quadtree is a quadtree where every node n is not only associated with a square area, but is also paired with a Bernoulli random variable X_n indicating whether there is one or more intruders inside the associated area. Consistently with the notation formerly introduced, in a PQ we set $\phi(n) = \Pr[X_n = 1]$. For the problem studied in this paper there can be an arbitrary number of intruders inside $\mathcal{A}$, and therefore we rely on the Type2 PQ we presented in [4]. In a Type2 PQ probabilities in the tree are recursively related as follows. Let n be an internal node in the tree and let $n_1, \ldots n_4$ be its four children. Then, the following constraint holds

$$(1 - \phi(n)) = \Pi_{i=1}^{4}(1 - \phi(n_i))$$

because there is no intruder in the parent node n if and only if there is no intruder in any of its children. The constraint is enforced throughout the tree. PQs are useful when they are associated with sensors with variable resolution [12].

[2]Without loss of generality, we assume the patroller moves with unary velocity, therefore T coincides with the distance between the two cells.

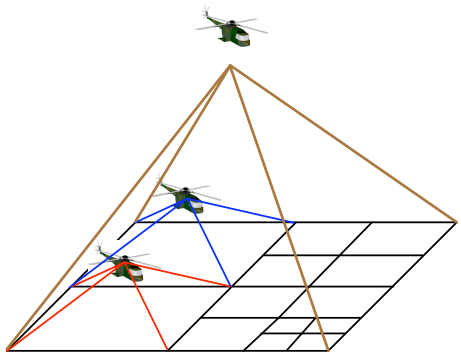


Fig. 1. The figure depicts the intuition behind PQs. UAVs flying at higher elevations sense larger areas but are more likely to suffer from erroneous sensor readings. On the contrary, UAVs at lower elevations are more precise but gather information about a smaller fraction of the area to be patrolled.

We embrace a sensor model where the patroller is equipped with a binary sensor returning 1 if at least an intruder is perceived inside the sensed area and 0 otherwise. The sensor is assumed to be faulty, i.e., to return both false positives and missed detections. Error rates are supposed to be elevation dependent and we assume that they vary with elevation as $\alpha(d) \geq \alpha(d+1)$, where d indicates the depth in the tree and the root is assumed to be at level 1. In other words, as the UAV decreases its elevation (i.e., it increases d), both its sensing area and error rates decrease. Identical assumptions are made for β.

As more and more sensor readings are collected by the patroller, a posterior about the various $\phi(n)$s can be efficiently maintained. The reader is referred to the aforementioned papers for details about how the elementary Bayes update formula can be extended to deal with the tree. We conclude this short recap noticing that one of the strengths of PQs is obviously found in their variable resolution, i.e., one initializes the data structure with a coarse resolution, and refines it only where, through repeated sensing, an increased probability of an ongoing attack occurs. As it will be evidenced in Sec. VII, the size of the tree grows slowly, thus enforcing a convenient limit on the size of the search space, and then accelerating the planning process.

VI. Strategies for Patrolling on Quadtrees

In this section we present 3 patrolling strategies based on PQs. All these strategies are myopic, i.e., they just plan on the next sensing location without committing to a number of decisions in the future. Receding horizon strategies may give better results, but turn out to be computationally more involved, and are therefore not considered here.

Given that for the patrolling problem there will be in general more than one attacker simultaneously present in the area, one may be tempted to reuse the same entropy-guided search strategy we presented in [4]. It can however be quickly assessed that such idea works well for search but not for the online patrolling. In fact, to minimize the loss function we hypothesized, the agent shall devote most of its sensing efforts to areas where through repeated sensing a high ϕ emerged. However, this does not happen when considering entropy only. This is easily understood observing that entropy is a symmetric function in $[0, 1]$ and a cell with probability ε and a cell with probability $1 - \varepsilon$ have the same entropy. However, for the problem at hand one shall obviously focus on the latter and ignore the former.

Before describing the three strategies, we shortly describe a simple extension of the hierarchical structure that is useful for the problem at hand. Consider a leaf node n at the maximum depth. By definition the leaf is associated with a binary indicator variable X_n indicating whether there is an intruder in the leaf or not. Because the expectation of an indicator variable is equal to the probability of the variable itself, it follows that when initially a full PQ is created, the expected number of intruders in every node (leaf or internal) can be computed. In fact, the expected number of intruders in the leaves is given by the probabilities, whereas the number of intruders in the areas associated with the internal nodes is obtained by recursively adding the values of its direct descendants. Once this information is computed during the initialization, it can be maintained at run time while the posterior is updated. Therefore from now onwards we assume that for every node we have an estimate of the expected number of attackers located inside the associated area. This value will be indicated as $\mu(n)$.

A. Density based patrolling

Assume the searcher is located at node n^*. Then for every node n the following quantity $d(n) = \mu(n)/A(n)$ is computed. The ratio measures *Density*, i.e., it estimates the expected number of intruders per surface unit. To account also for travel costs, normalized densities and normalized travel times are linearly combined:

$$J_1(n) = \left[\gamma \frac{d(n)}{\max_{n' \in \mathcal{T}} d(n')} - (1-\gamma) \frac{T(n^*, n)}{\max_{n' \in \mathcal{T}} T(n^*, n')}\right]$$

where $T(n, n^*)$ is the travel time and $\mathcal{T}$ is the set of nodes in the PQ. The agent then will move and sense to the node maximizing J_1, i.e., $n = \arg\max_n J_1(n)$.

Note that the same approach can be used in the uniform case, provided that one substitutes $d(n)$ with $\phi(n)$, given that the two coincides in the uniform case.

B. Weighted density based patrolling

The second strategy we consider is similar to the previous one, but biases the effort towards nodes associated with a higher loss. In other words, if two nodes have the same density d, the patroller shall consider first the one with a higher loss in order to minimize the penalty it pays. The product $d(n) \cdot l(n)$ biases the search too much. Instead, we partition the search area in k *classes* and associate and importance factor related to the loss to every class. Then, after $d(n)$ is computed as for the previous strategy, these values are rescaled according to the importance factor (hence the name *Weighted Density*). The function J_2 is then defined as J_1 above, with the difference that each density $d(n)$ is

multiplied by $w(n)$. In the following, we opt for $k = 2$ classes. The first class has $w(n) = 1$, whereas the second has $w(n) = 0.8$. In order to assign nodes to classes, we sort all nodes by decreasing values of l, and we assign the first 25% to the first class, and the remaining to the second class.

C. Hybrid patrolling

Finally, we propose a hybrid strategy mixing the classic *lawn mower*[3] sweep pattern with the weighted density strategy we just presented. In other words, the searcher alternates phases where it moves and senses the environment using the sweep strategy, with phases where it uses the computed posterior and the loss function to drive its sensing efforts. The rationale behind this strategy is to implement a strategy where the searcher first performs a quick gross-grain reconnaissance of the environment to identify regions that are likely to have been attacked, and then spends time to remove the intruders before going back to a new sweep round. Evidently, by considering different criteria for switching between these two behaviors, a continuum of strategies exist. A systematic investigation of these criteria is beyond the scope of this paper and left for future investigation. In this paper, we consider a single strategy where the searcher equally splits its time between sweeping and searching using weighted density. While sweeping, the searcher restricts itself to a regular grid whose resolution is coarser than the maximum resolution achievable with the PQ. In this way a complete sweep of the environment can be completed relatively quick. Another reason to consider the *Hybrid Search* is to assess the value of a strategy that partially exploits the hierarchical structure and partially commits to a uniform representation.

D. Absence of undetected intrusion

In this subsection, we show how the *Density* strategy guarantees the absence of undetected intrusions. This property implies that each intrusion is always found in a finite time or, alternatively, that no intrusion can stay in the environment undefinitely without being discovered. Notice that this property does not guarantee to clear the environment, i.e., having no active intrusion after some time.

We present the results for the uniform grid because it is simpler, and the generalization to the hierarchical structure just follows by considering a careful extension. However, in order to do so one has to slightly alter the function J_1 formerly defined in order to properly break ties that can emerge in pathological (and unlikely) cases. For this purpose, assuming that two cells sharing an edge have distance 1, it is useful to introduce the set $T_1(m) = \{n \in \mathcal{A} \mid T(m,n) = 1\}$. Note that $|T_1(m)| \leq 4$ when considering Euclidean distances between cells. The modified strategy is given by the following definition.

[3]The lawn mower pattern is a classic search strategy where the searcher divides the search domain in a regular grid and then starts from the top left corner, moves all the way down, then moves one column to the right, moves all the way up, and so on. When the grid is over the searcher moves back to the beginning and restarts.

Consider the situation where the patroller is currently in cell m and it determines that according to J_1 the next cell to be visited is $n \in T_1(m)$. Then, it discards n and rather visits the cell in $T_1(m)$ that it has not visited since the longest time (this could indeed be n, but it can also be different). A policy implementing this paradigm is said to be *round robin*.

Theorem 1: Consider the round robin version of the density based search used on regular grids. If $p_c \neq 0$ for every cell in $\mathcal{A}$ and $\alpha, \beta \neq 0.5$, then each intruder will be eventually found.

Proof Sketch: Assuming that sensors are informative[4], i.e., $\alpha, \beta \neq 0.5$, if an intruder is present in cell c and the number of times c is visited is unbounded, then it will eventually be found and removed based on the decision criterion given in Eq. 3. Hence, the proof is equivalent to showing that every cell is visited an unbounded number of times. Without loss of generality, assume there is exactly one cell c that is not visited an unbounded number of times, and let m be a cell such that $T(m,c) = 1$. Therefore, m is visited an unbounded number of times and let us consider the decision taken by the agent after it visits m. If $\forall a \notin T_1(m)\ J_1(c) > J_1(a)$, then the agent will pick c rather then any a. This condition is equivalent to the following:

$$\gamma\frac{\phi(c)}{\phi_{max}} - (1-\gamma)\frac{T(c,m)}{T_{max}} > \gamma\frac{\phi(a)}{\phi_{max}} - (1-\gamma)\frac{T(a,m)}{T_{max}}.$$

Through algebraic manipulation it is possible to show that the inequality is true when

$$\phi(c) > 1 - \frac{1-\gamma}{\gamma L\sqrt{2}}(\sqrt{2}-1)$$

where we used the hypothesis $T(c,m) = 1$, we assumed the worst case $\phi(a) = 1$, and we indicated with $L\sqrt{2}$ the length of the diagonal of $\mathcal{A}$. Because of they hypothesis $p_c > 0$ and of Eq. 2, $\lim_{t\to\infty}\phi(c) = 1$ and therefore the previous condition is eventually verified so when in m the agent will prefer c versus $a \notin T_1(m)$. Finally, note that the *round robin* hypothesis guarantees that c will be eventually selected even if there exits cells in $T_1(m)$ giving a higher value for J_1. □

VII. Experimental Results

A battery of 100 benchmarks has been designed to assess the performance of the hierarchical patrolling strategies we presented in Section VI[5]. Each benchmark consists of a sequence of attacks in a square grid with 256×256 =65536 cells. The arrivals of attacks in the area to be patrolled is distributed in time and space. Attacks are generated according to a Poisson process with exponential interarrival times characterized by $\lambda = 1/95$. Attacks are distributed in space according to the bidimensional probability distribution displayed in Fig. 2. Note that this probability distribution has the same shape of the loss function, i.e., attacks are more

[4]A sensor with error rate larger than 0.5 is still informative, i.e. one may consider the complement of the value it returns. The only case to avoid is $\alpha, \beta = 0.5$ because in such case the reading is unrelated to the content of the cell.

[5]Matlab® code and datasets used in this section are available on `http://robotics.ucmerced.edu`.

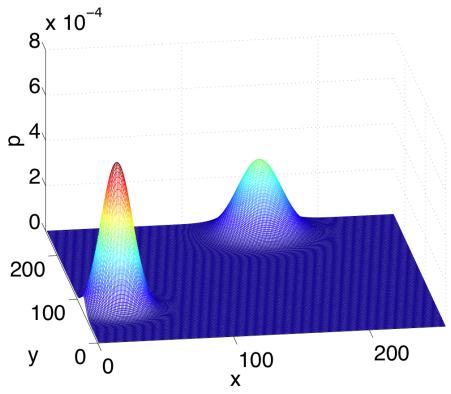


Fig. 2. Probability density for the attacks locations in the patrolled area.

likely to occur in regions associated with a higher loss. Every benchmark consists of up to 75 attacks, but this information is not known to the patroller. Finally, every search strategy is evaluated over a time horizon with 60000 time steps. Note that a single sensing operation takes 1 time step, and moving between two points p_1 and p_2 takes time $\lceil ||p_2 - p_1||_2 \rceil$.

Preliminarily, we have solved every benchmark with an *optimal patroller*. The optimal patroller is an idealized patroller that is immediately notified about the location of an attack as soon as it arrives in the environment and is not affected by any error. At each arrival time, the optimal patroller updates the list of active attacks to clear and computes the optimal plan[6] aiming to remove all intruders in the area while minimizing the overall loss. Evidently, the optimal patroller operates under very favorable conditions, i.e., it is not required to discover intruders by using noisy sensors. However, such abstraction is useful to assess the performance of the more realistic agents we consider. We will perform comparisons using the competitive factor, namely the ratio between total penalties accumulated by the optimal patroller and a patroller using some strategy, respectively.

A. Uniform Grid

In this subsection we discuss experiments with the uniform environment representation. By showing these results we aim at giving insights about the challenges of online patrolling with a uniform grid and providing a comparative baseline for the hierarchical case.

The strategy defined in Sec. IV (denoted as *Greedy*) is compared with other three strategies called *Sweep*, *Random* and *Density*, respectively. *Sweep* and *Random* are strategies that plan just considering the environment topology, i.e., they do not maintain any posterior. The first one simply executes a lawn mower pattern while the second one randomly selects the next sensing location randomly picking from the adjacent ones with a uniform distribution. To compensate for their poor planning principles, these two strategies are endowed with error–free sensors. The *Density* strategy is the one presented in Sec. VI-A adapted for the uniform grid case. To keep a correspondence with the hierarchical scenario, we consider the uniform grid as a constraint over altitude and resolution (given that the maximum resolution resides in a 256×256 grid). More precisely, a patroller executing a mission over a grid of edge L is equivalent to a patroller using a hierarchical representation constrained to depth $1 + log_4(L^2)$ (recall that the root is assumed be at depth 1). For space reasons, we limit to present results obtained on a 32×32 grid.

Fig. 3 depicts the trend of the average penalty accumulated during the mission. *Random* and *Sweep* seem to show a good trend in the first part of the mission. This is mainly due to the fact both start in an area with high loss and are equipped with accurate sensors. Afterwards they tend to diverge and conclude, as expected, obtaining the two lowest performances.

An interesting comparison emerges between *Greedy* and *Density*. Surprisingly, the last one obtains the best performance at the end of the mission. However, by looking at how the penalty evolves in time, it can be noted that in the first part of the mission *Greedy* prevails. The reason behind this is that, as time passes, the attacks at some point stop, thus deviating from the assumption of Poisson arrivals. Then *Greedy* strategy achieves the best performance while arrivals adhere to the hypotheses, and deteriorates later on.

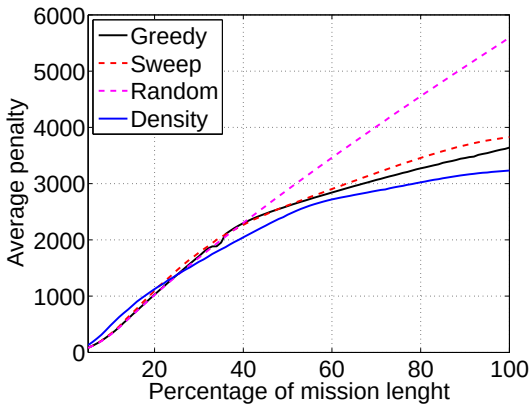


Fig. 3. Average penalty during missions on the uniform grid.

Tab. I reports competitive factors w.r.t the optimal patroller. As it can be seen performance are very low. When moving on a uniform grid of edge L, the distance between two adjacent cells is $2^{(log_4(256^2) - log_4(L^2))}$. This increases the penalty accrued when moving between cells.

B. Hierarchical Grid

Next we illustrate the performance of the three hierarchical patrolling strategies and we contrast their performance. Fig. 4 shows the average number of nodes in the PQ while the mission evolves. This dataset shows the average for the *Weighted Density* strategy but similar trends are observed for the others. The important aspect to observe is that even

[6]This problem is in general hard to solve. However, the idealized patroller we consider is endowed with the ability to solve it instantaneously. Practically, this is obtained by stopping the clock computing the loss while the patroller solves the problem. Note also that, because of the spacing in time of the arrivals, the optimal patroller repeatedly solves modestly sized instances of the DTRP, therefore this problem is solvable.

Uniform Grid	
Density	850
Greedy	959
Sweep	1007
Random	1429

Hierarchical Grid	
Density	43
Weighted Density	36
Hybrid	96

TABLE I

AVERAGE RATIOS BETWEEN THE LOSS ACCRUED BY THE VARIOUS STRATEGIES AND THE OPTIMAL PATROLLER.

though the patrolling effort continues for an extended period of time, the size of the tree remains bounded. To put the numbers into perspective, one should consider the fully expanded PQ would have 65536 leaves.

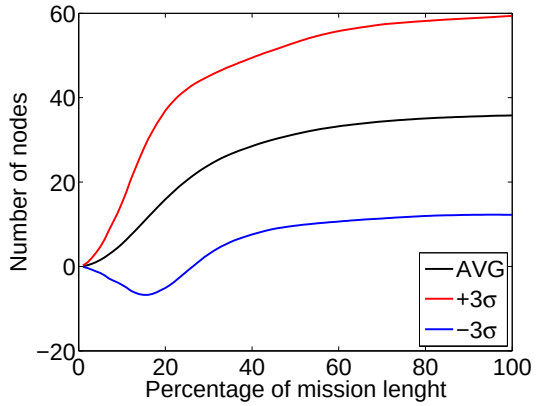
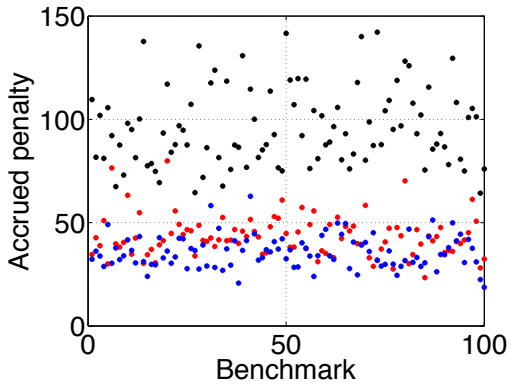


Fig. 4. Average number of nodes in the PQ as the mission unfolds.

Fig. 5 shows the loss accrued by the three patrolling strategies while solving the 100 benchmarks. It can be observed that *Weighted Loss* is the best, followed by *Density*. The *Hybrid* method instead performs significantly worse. This is justified by the fact that, while the patroller sweeps, it does not eliminate intruders, and the penalty accumulated in that stage offsets the advantage it may gain by quickly identifying areas where intruders entered. Finally, in Tab. I we show the competitive factors for the hierarchical strategies. It can be seen that when compared with the uniform strategies hierarchical methods achieve large advantages. A companion video shows an example patrolling mission, with the left panel illustrating the propagated posterior and the right panel offering a top view indicating the position of the patroller and the locations of intruders.

VIII. CONCLUSIONS

In this paper we have considered the problem of online patrolling using our recently introduced probabilistic quadtree representation. The online patrolling problem is mostly similar to the recently introduced persistent patrol problem, but it considers the additional complications arising from faulty sensors. For the hierarchical case we have considered three different myopic strategies. While these may seem heuristic, one should recall that we are dealing with a problem that is easily reduced to the DTRP problem and therefore does not lend itself to efficient optimal solutions. Our simulation results show that the hierarchical representation largely outperforms uniform ones, thus corroborating



s

Fig. 5. Accrued loss by the hierarchical patrolling strategies when solving the 100 benchmarks (Red: *Density*; Blue: *Weighted Density*; Black: *Hybrid*).

a result we already outlined for the simpler problem of search. Moreover, the use of a hybrid strategy mixing ideas coming from uniform representations with the hierarchical one showed to be inferior to the purely hierarchical approach, thus confirming the goodness of the latter. Currently, we are working to implement the presented algorithms to control an AirRobot equipped with a nadir camera engaged in a patrolling mission in a rural area.

REFERENCES

[1] N. Basilico, N. Gatti, and F. Villa. Asynchronous multi-robot patrolling against intrusion in arbitrary topologies. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 1224–1229, 2010.

[2] D.J. Bertsimas and G. Van Ryzin. A stochastic and dynamic vehicle routing problem in the Euclidean plane. *Operations Research*, 39(4):601–615, 1991.

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

[4] S. Carpin, D.A. Burch, and T.H. Chung. Searching for multiple targets using probabilistic quadtrees. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4536–4543, 2011.

[5] T.H. Chung and S. Carpin. Multiscale search using probabilistic quadtrees. In *Proceeding of the IEEE International Conference on Robotics and Automation*, pages 2546–2543, 2011.

[6] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry*. Springer, 2000.

[7] Y. Elmaliach, N. Agmon, and G. Kaminka. Multi-robot area patrol under frequency constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 385–390, 2007.

[8] V. Anh Huyun, J.J. Enright, and E. Frazzoli. Persistent patrol with limited-range onboard sensors. In *Proceedings of the IEEE International Conference on Decision and Control*, pages 7661–7668, 2010.

[9] L.C. Ludeman. *Random Processes : Filtering, Estimation, and Detection*. Wiley-IEEE Press, 2003.

[10] P. Paruchuri, J. Pearce, M. Tambe, F. Ordonez, and S. Kraus. An efficient heuristic approach for security against multiple adversaries. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS)*, pages 311–318, Honolulu, USA, May 14-18 2007.

[11] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM*, 23:555–565, 1976.

[12] National Search and Rescue Committee. United States National Search and Rescue Supplement, 2000.

[13] S. Waharte, A. Symington, and N. Trigoni. Probabilistic search with agile UAVs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2840–2845, 2010.