Searching for Multiple Targets Using Probabilistic Quadtrees

Stefano Carpin, Derek Burch, and Timothy H. Chung

***Abstract*— We consider the problem of searching for an unknown number of static targets inside an assigned area. The search problem is tackled using Probabilisitic Quadtrees (PQ), a data structure we recently introduced. Probabilistic quadtrees allow for a variable resolution representation and naturally induce a search problem where the searcher needs to choose not only where to sense, but also the sensing resolution. Through a Bayesian approach accommodating faulty sensors returning both false positives and missed detections, a posterior distribution about the location of the targets is propagated during the search effort. In this paper we extend our previous findings by considering the problem of searching for an unknown number of targets. Moreover, we substitute our formerly used heuristic with an approach based on information gain and expected costs. Finally, we provide some convergence results showing that in the worst case our model provides the same results as uniform grids, thus guaranteeing that the representation we propose gracefully degrades towards a known model. Extensive simulation results substantiate the properties of the method we propose, and we also show that our variable resolution method outperforms traditional methods based on uniform resolution grids.**

I. Introduction

Autonomous robots are playing an increasingly important role in search related tasks. Robotic searchers can be sent to inaccessible or dangerous areas to locate items of interest. Moreover, as these platforms become more and more affordable and robust, the use of teams of aerial vehicles that cooperatively and autonomously search an assigned area is also becoming a viable alternative. Applications include urban search and rescue, surveillance of sensitive areas, and intruder detection, just to name a few.

In this context, the advent of easy to deploy and control vertical take off and landing (VTOL) platforms offers novel possibilities in the area of robotic search. The problem of autonomous control of robotic helicopters has been studied in depth because these platform offer numerous control challenges [16]. On the contrary, quadrotor aerial platforms like the AirRobot depicted in Fig. 1 are much simpler to control and their increasing presence is spurring new energy into the field of robotic search using aerial vehicles. When compared with fixed wing vehicles, VTOL platforms offer the advantage of being able to hover above an area of interest. Moreover, when equipped with downward pointing cameras for target detection, a change in elevation translates to a change in size of the sensor footprint, as well as a variation in the sensor accuracy. This fact was already noticed and investigated in [17].

S. Carpin and D. Burch are with the School of Engineering, University of California, Merced, CA, USA. T.H. Chung is with the Naval Postgraduate School, Monterey, CA, USA.

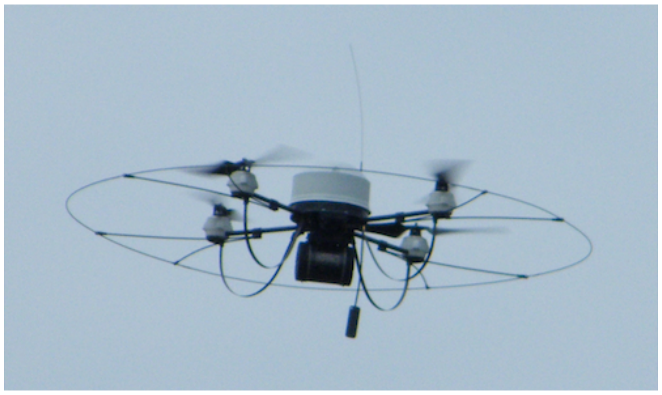
Fig. 1. The AirRobot is an easy to control VTOL vehicle that can hover over areas of interest. When the robot changes its elevation while keeping its position the area covered by the downward pointing camera varies, as well as the accuracy of the target detection algorithm.

These considerations recently motivated us to develop a variable resolution data structure to study tasks where the searcher is equipped with a sensor with variable footprint and accuracy [4]. Using a Bayesian approach, throughout the mission the searcher propagates a posterior about the probability that an object of interest is located in a given area. Our framework accommodates sensors with false positives and missed detection, and dynamically updates the search space in order to allocate more search effort where promising locations emerge. The search algorithm determines not only where to search next, but also the sensing accuracy. The intuition is that the algorithm handles the tradeoff between sensing a wide area with low resolution versus observing a small region with higher precision.

Building upon our recent work, this paper offers three novel contributions:

- While in [4] we considered the case of searching for a single target, here we consider the more general case where an unknown number of targets may be present in the area being searched. This possibility implies significant changes in the posterior being propagated and in the complexity of the Bayesian update.
- In the planning stage we introduce a new information gain based function in order to decide where to sense next. Information gain largely outperforms our previously chosen heuristic based on probability only, and also naturally leads to a simple statement of the search stopping criterion.
- We provide theoretical results showing that in the limit a hierarchical representation converges to a uniform one, thus guaranteeing that under the assumption of infor-

mative sensors all targets will be eventually discovered.

The rest of the paper is organized as follows. Section II shortly discusses related literature. The search problem studied in this paper is formalized in Section III. The search strategy based on entropic information gain is introduced in Section IV, whereas Section V discusses the stopping criterion and how decisions are formulated as the search mission evolves. Theoretical properties are discussed in section VI. Simulation results are illustrated in Section VII, and conclusions and future work are presented in Section VIII.

II. Related work

Search theory dates back to World War II, and the works by Koopman [6] and Stone [15] offer a classic treatise of this area from an operational research perspective. From a robotic point of view the line of work most similar to our approach was performed by Furukawa and colleagues in a series of papers appeared in the last years [1], [2], [10], [19]. Therein the authors cast the search problem as a Baysian framework accounting for faulty sensors. The authors, however, rely on a uniform representation of the environment, and when dealing with multiple targets they assume their number is known. Moreover, their implementation focuses on UAVs flying at a constant elevation, and therefore sensor accuracy does not vary during the mission. In [10] the authors present one of the few examples of search architectures based on a spatial representation that is reconfigured during the search. However their model does not feature any hierarchical layering, but rather consists of planar shapes whose boundary varies as the search evolves.

The problem of searching for an unknown number of targets is considered less frequently, but is obviously more important from a practical point of view. In a demining task Bryant and Carthel [3] consider the problem of estimating the overall unknown number of targets based on sensor performance and number of detections. They show that the number of missed targets follows a negative binomial distribution. This result, however, is based on the assumption that the sensor may miss targets, but does not provide for false positives. In the controls community Hussein *et al.* [18] also consider the problem of searching for an unknown number of targets, and they also use information entropy in order to formulate search decisions like where to search next and when to stop. Their formulation, however, relies on uniform grid representations and does not consider sensors with variable performance.

For sake of completeness, we shall also mention that Kraetzschmar *et al.* also used a data structured called *Probabilistic Quadtrees* [8]. However, since their goal is solving a different problem (i.e., mapping), the probabilistic updates are completely different, and henceforth the similarity between the two works is just in the name and in the common use of a well known variable resolution representation.

III. Problem definition

We here revisit and modify the data structure we introduced in [4] in order to account for the possible presence of multiple targets. The data structure presented in [4] will be called Type1 PQ in the following and it can be used only to search for a single intruder. The data structure we present in this paper is instead dubbed Type2 PQ because it can handle multiple targets. For sake of simplicity, let us assume the search domain $\mathcal{A}$ is a square with an edge L units long. At most one intruder (also called *target* in the following) may be located in a unit size square and therefore the number of intruders in the area is an integer between 0 and L^2. The search area may then be notionally divided into L^2 cells c_i of area 1, but this subdivision is purely conceptual, since our method is explicitly designed to avoid dealing with this uniform representation. The presence of a target inside cell c is modeled by a Bernoulli random variable X_c. We assume the various X_c are independent and identically distributed (iid). A prior distribution about the variables X_c may or may not be available. A single searcher is tasked with the goal of sensing the environment and to eventually output a *search decision* that may be either negative (no intruder detected) or positive (one or more intruders detected). In the latter case for every intruder located the searcher shall also identify the cell where it is located.

A. Search space

We associate with the search domain $\mathcal{A}$ a *probabilistic quadtree* (PQ) $\mathcal{T}$. Quadtrees are a variable resolution, spatial representation heavily used in computational geometry and robotics [5]. In essence a quadtree is a rooted tree where every internal node has four children. Every node is associated with a planar square[1] region, and the root of $\mathcal{T}$ is associated with the whole region $\mathcal{A}$. The recursive invariant in a quadtree is that the region of an internal node is split into four equally sized squares, and each of them is then associated with one of the children. Given a node $n \in \mathcal{T}$, we indicate with $R(n) \subset \mathbb{R}^2$ the square region associated with n. Throughout this paper we will consider quadtrees whose leaves may be at different depths, thus fully exploiting their variable resolution nature.

A probabilistic quadtree is a quadtree where every node n in $\mathcal{T}$ is associated with a binary indicator random variable X_n indicating the event *at least one intruder present in area R(n)*. The reader should note that X_n coincides with X_c only when the region $R(n)$ associated with node n coincides with cell c. However, variables X_n are defined also for internal nodes whose associated area is larger than 1 and may then host more than one target. For every node we define

$$p_n = \Pr[X_n = 1].$$

Conversely, we introduce

$$q_n = 1 - p_n = 1 - \Pr[X_n = 1] = \Pr[X_n = 0], \qquad (1)$$

[1]In general it is rectangular, but for sake of simplicity we here assume it is square.

i.e., q_n is the probability that no targets are present in node n. Because of the independence assumption, if n_i and n_j are two nodes whose associated regions do not intersect, then

$$Pr[X_i = 1, X_j = 1] = \Pr[X_i = 1] \Pr[X_j = 1] = p_i p_j.$$

If n is an internal node and $n_1, \dots, n_4$ are its children, because of the independence assumption the following constraint holds

$$q_n = q_1 q_2 q_3 q_4 \tag{2}$$

because the parent node is intruder free if and only if all of its four children are intruder free. This relationship is necessarily recursive, i.e., it is applied also to n's children. The probabilistic formulation is completed noting that the model also allows study of situations where no intruder is located in the search area. Therefore an *outside node* $n_\emptyset$ is introduced, and its indicator variable $I_{n_\emptyset}$ is true when no intruder is located inside $\mathcal{A}$. If r is the root node in $\mathcal{T}$, it follows that

$$p_{n_\emptyset} = 1 - p_r.$$

We conclude this subsection assuming a maximum depth $\mathcal{D}$ for $\mathcal{T}$ is given. Under this assumption, nodes at depth $\mathcal{D}$ are associated with regions of area 1. That is to say that in a full probabilistic tree with all leaves at depth $\mathcal{D}$ its leaves provide the same partition of $\mathcal{A}$ induced by the uniform grid. If the searcher reports *Intruder located in cell* c_i, then cell c_i must be a region associated with a node at maximum depth. In other words, the decision *target detected* must be associated with a cell at the highest resolution. In the following, if n is a node then $d(n)$ is its depth, and we assume the root is at depth 0. Finally, let $\mathcal{L}(\mathcal{T})$ be the set of leaves of $\mathcal{T}$, and $\mathcal{N}(\mathcal{T})$ be its set of nodes. It is important to recall that leaves may or may not be at depth $\mathcal{D}$.

B. Sensor model

The hierarchical search space is advantageous when it can be coupled with a sensor offering variable resolution. In fact this research is motivated by UAVs equipped with downward pointing cameras. As the UAV varies its altitude, the camera captures regions of different size. Moreover its accuracy varies with altitude, i.e., identifying an object of interest while flying at high elevation is harder than when flying at close range [13]. Figure 2 shows some images illustrating this standpoint.

The output of each sensor reading is assumed to be a Bernoulli detection variable, assuming value 1 if and only if at least one intruder is detected. In order to complete its task, the searcher may sense at different locations. Given an area $\mathcal{A}$ and its associated probabilistic quadtree $\mathcal{T}$, we assume the searcher cannot move and sense at arbitrary locations, but is rather constrained to sense when it is placed at the center of the region associated with one of the nodes n in $\mathcal{T}$. In this case we say that the searcher senses at node n. According to this model, if the searcher queries its sensor when it is located at node n, its sensor scans the associated region $R(n)$. It follows that when the searcher scans the area associated with a node deep in the tree it flies at low elevation

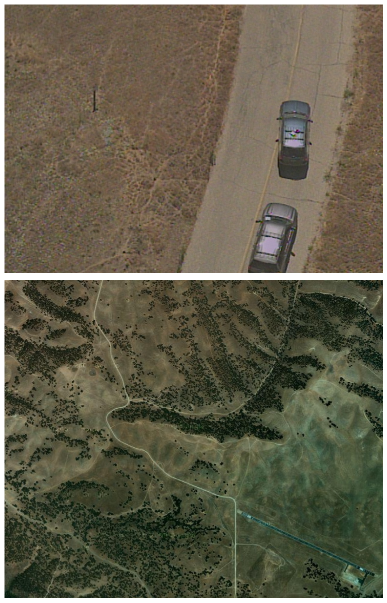
Fig. 2. Aerial images collected by a UAV flying at Camp Roberts, CA. The top image shows an image of two cars taken at close range. The bottom image shows a picture of the same area taken from a much higher altitude. While the scanned area greatly increased, the ability to recognize objects of interest (cars in this case) almost vanished.

(and then small sensing area and high accuracy), whereas if it scans an area associated with a node closer to the root the searcher is located at a higher altitude. The Bernoulli variable Z_n^t is then the output of the sensor when used at time t in node n. The sensor is assumed to be imperfect, i.e., prone to missed detections and false positives. These errors are modeled as follows (note that we omit time because error profiles are assumed stationary with respect to time):

$$\Pr[Z_n = 1 | X_n = 0] = \alpha(d(n))$$

$$\Pr[Z_n = 0 | X_n = 1] = \beta(d(n)).$$

Therefore α measures the false positive rate, whereas β relates to the missed detection error. Dependence on depth $d(n)$ signifies that accuracy depends on the size of the area being sensed, or, equivalently, from the altitude of the UAV. Such detection probabilities can be determined empirically, as contained in search and rescue manuals [13], or from theoretical models for altitude-dependent detections [7]. This dependence is therefore captured by the depth of the node $d(n)$, with the intuition that when sensing at a higher altitude a larger area is scanned, and therefore more errors are possible. This notion is formalized by the following inequalities:

1) $d_1 < d_2 \Rightarrow \alpha(d_1) > \alpha(d_2)$
2) $d_1 < d_2 \Rightarrow \beta(d_1) > \beta(d_2)$

where the reader should recall that nodes closer to the root are associate with larger areas and have lower depth. These inequalities therefore state that error rates of either type increase when sensing larger areas.

C. Bayesian updates

The estimation process is bootstrapped assuming an initial tree $\mathcal{T}$ is available (its construction is described in a later section). It is assumed that for every leaf node n a prior probability $p_n^{(0)} = P[X_n = 1]$ is available. Therefore, recursively applying Eq. 1 and Eq. 2 from the bottom up we can compute the initial values of p_n for every internal node in $\mathcal{T}$. By integrating successive sensor readings we aim to compute the following posterior for each node

$$p_n^{(t)} = \Pr[X_n = 1 | Z_{n_1}^1, Z_{n_2}^2, \dots, Z_{n_t}^t]$$

where the reader should note that the various readings may have occurred also at nodes $n_i \neq n$. The hierarchical structure we are dealing with requires some extra care. If we assume all readings happened on leaves of the tree, and we consider only updating the posterior of the leaves, then $p_n^{(t)}$ can be computed using the standard recursive Bayesian update rule

$$p_n^{(t)} = \frac{\Pr[Z_{n_t}^t | X_n = 1] p_n^{(t-1)}}{\Pr[Z_{n_t}^t]}. \tag{3}$$

If $n_t \neq n$, in Eq. 3 we set $\Pr[Z_{n_t}^t] = 1$ and $\Pr[Z_{n_t}^t | X_n = 1] = 1$, so that the posterior about node n is not altered after sensing a different node[2]. Note that this is consistent with the independence assumption we introduced, but is different from the Type1 PQ structure where all nodes were correlated with each other. If $n_t = n$ we instead use the sensor model previously introduced and update the posterior. After sensing leaf node n_t at time t and having updated its posterior according to Eq. 3, the posterior for all its ancestors can be updated as well using again Eq. 1 and Eq. 2, and this step concludes the update of the probabilistic quadtree. The situation is different when the searcher senses an internal node. This extension is detailed in the following.

Updates when sensing at an internal node. When sensing at an internal node n, its posterior can be updated using Eq. 3, and its ancestors' probabilities can also be updated as outlined above. However, the change in probability shall also be propagated to the descendants of n. The precise formula for this update depends on the sensor being used. In the simulations presented later on we opt for a *uniform* approach, i.e., the change is propagated from n to its four children $n_1, \dots, n_4$, and then recursively to its descendants. Let

$$k = \frac{\log q_n^{(t)}}{\log q_n^{(t-1)}}.$$

Then, the updated q probability of each children n_i is

$$q_{n_i}^{(t)} = (q_{n_i}^{(t-1)})^k. \tag{4}$$

Eq. 4 ensures that the constraint given in Eq. 2 is preserved. In a practical scenario, based on the characteristics of the sensor being used one could implement the update differently as long as the constraint expressed by Eq. 2 is preserved. We conclude this section noting that in the model we presented in [4] the integration of every sensor reading had complexity $\mathcal{O}(|\mathcal{N}(\mathcal{T})|)$ because all nodes in the tree had to be updated due to their correlation. On the contrary, for the model proposed herein a sensor reading performed simply at a single node at depth d can be integrated in time $\mathcal{O}(4^{\mathcal{D}-d})$. Considering that for a full tree we have $|T| \in \mathcal{O}(4^{\mathcal{D}})$ the speedup can be up to $\mathcal{O}(4^d)$.

IV. Entropy of a probabilistic quadtree and search based on information gain

In order to characterize the uncertainty of the search being performed, a measure of *entropy* of probabilistic quadtrees is introduced. The entropy of the $\mathcal{T}$ is defined as

$$H(\mathcal{T}) = -\sum_{n \in \mathcal{L}(\mathcal{T})} p_n \log_2 p_n \tag{5}$$

i.e., it is the sum of entropy associated with every leaf. Because of the symmetry in the definition of entropy we can use either p or $q = 1 - p$ when computing the entropy of the individual binary random variables associated with the leaves. The definition is supported by the independence between the indicator variables associated with the leaves. Indicator variables associated with internal nodes are deterministic functions of the variables associated with the leaves. Therefore Eq. 5 indeed is the entropy of the ensemble of variables variables $X_1, \dots, X_{|\mathcal{T}|}$, as defined in [12]. In fact, all uncertainty in the tree is concentrated on the leaves, because once the values of those random variables is known, then there is no uncertainty in internal nodes.

Decisions about where[3] to sense during the mission are not taken offline, but the searcher rather decides its sensing locations online, based on the posterior being iteratively updated. In [4] we proposed selection of the next sensing location as the node $n \in \mathcal{T}$ maximizing the following function:

$$J \triangleq \frac{p(n) \cdot 4^{d(n)}}{\text{cost}(n', n)}$$

where n' is the current robot location and cost(n', n) is the cost of moving from n' to n. The rationale for this heuristic is to favor locations with high probability and at greater depth while penalizing nodes too far away. This heuristic behaves reasonably well in practice and has the advantage of requiring only linear time for its evaluation. However, in this paper we embrace a more principled approach based on (expected) information gain, as already considered in the area of robotic mapping [14]. Given a PQ $\mathcal{T}$, the information gain associated with node n is

$$I(n) = H(\mathcal{T}) - E_{Z_n}[H(\mathcal{T}|n)]$$

where $E_{Z_n}[H(\mathcal{T}|n)]$ is the expected entropy of $\mathcal{T}$ with respect to the sensor reading Z_n obtained when sensing at

[2]In practice one would not update those probabilities at all. However we make this clear in order to get a model that can be consistently applied to all nodes.

[3]The reader should recall that since the searcher is constrained to sense at positions associated with nodes in the probabilistic quadtree then the sensing location also defines the sensing resolution because the position is univocally related to the depth d in the tree.

node n. In order to account also for travelled distance when choosing where to sense next, we use the following weighted function proposed in [14]

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

where $D(n^*, n)$ is the Euclidean distance between node n and the current searcher position n^*. Throughout the experiments we use a value of $\gamma = 0.5$, i.e., distance and information gain are equally weighted. Hence when the searcher needs to decide where to sense next it will pick the node n as follows:

$$n = \arg\max_n I'(n).$$

The reader shall however note that with this approach the computation of n takes quadratic time in $\mathcal{N}(\mathcal{T})$, whereas using J takes linear time.

V. Stopping the search and formulating a search decision

In a Type2 PQ the number of intruders present in the area may be any number between 0 and L^2. This number may or may not be known beforehand. Moreover, the time devoted to the search mission may or may not be fixed *a priori*. The combination of these factors leads to numerous different situations and to different decision criteria. If the number of targets is known, the searcher may terminate the search effort as soon as it reaches sufficient confidence about the location of the targets. This may happen even before the whole search domain is inspected. Conversely, if the number of targets is unknown the presence of a target in a given region cannot be ruled out before the area is sensed. We focus on the latter situation because it is more general. Moreover, the searcher has limited autonomy, i.e., there is an upper bound T on the duration of the search mission, but we assume there is an incentive in terminating the search as soon as sufficient information is available to formulate a search decision. These assumptions are coherent with indications provided by search and rescue teams. We use the stopping criterion proposed in [18]. Let the quantity U be defined as follows:

$$U(\mathcal{T}) = \frac{\sum_{n_i \in \mathcal{L}(\mathcal{T})} H(n_i)}{|\mathcal{L}(\mathcal{T})| H_{max}}$$

where $H(n_i) = -p_i \log_2 p_i$ and H_{max} is the largest entropy for any node in the tree. The search effort terminates when the allotted time T expires or when each leaf has an entropy below $\varepsilon \cdot U(\mathcal{T})$. Evidently, by decreasing the value of ε one forces the searcher to collect more information (i.e., decrease entropy) before stopping the search.

Finally, one has to consider that decisions taken at the end of the search mission suffer from two types of errors as it is for sensing, i.e., there may be missed detections and false positives. These two errors are not necessarily equally severe, and therefore a cost-based approach is considered [11][4]. Let D_n be the binary decision made about node n at the end of the search effort. The decision is binary with $D_n = 0$ indicating the decision *No target*. When decisions are based on a single sensor reading the expected cost C_n associated with node n is

$$\begin{aligned} C_n = &\Pr[D_n = 1|X_i = 0]\Pr[X_i = 0]C_{10} + \\ &\Pr[D_n = 0|X_i = 1]\Pr[X_i = 1]C_{01} + \\ &\Pr[D_n = 0|X_i = 0]\Pr[X_i = 0]C_{00} + \\ &\Pr[D_n = 1|X_i = 1]\Pr[X_i = 1]C_{11}. \end{aligned}$$

C_{ij} is the cost incurred when decision $D_n = i$ is made and $X_n = j$. In general one may consider situations where a cost is incurred also when the correct decision is taken (hence C_{ii} terms are not null). In the simplest case D_n is formulated based on a single sensor reading Z, and given a sensor model $\Pr[Z|X_n]$ the problem is then how to map sensor readings into decisions so that C_n is minimized. In this situation it is known [11] that one should choose $D_n = 0$ if

$$\frac{\Pr[Z|X_n = 0]}{\Pr[Z|X_n = 1]} > \frac{(C_{11} - C_{01})\Pr[X_n = 1]}{(C_{00} - C_{10})\Pr[X_n = 0]}$$

and $D_n = 1$ otherwise. In the situation we consider in this paper, multiple sensor readings are combined together in order to propagate a posterior over time. Let T be the duration of the search mission, and let $Z^1, \dots, Z^m$ be the sequence of sensor readings covering node n. This is in general a subsequence of the whole set of sensor readings the searcher collected during the mission; therefore $m \leq T$. Decision $D_i^{(m)}$ is therefore the decision concerning node n after integrating all m sensor inputs. The previous decision rule can then be generalized as follows: decide $D_i^{(m)} = 0$ if

$$\frac{\Pr[Z^m|X_n = 0]}{\Pr[Z^m|X_n = 1]} > \frac{(C_{11} - C_{01})\Pr[X_n = 1|Z^1 \dots Z^{m-1}]}{(C_{00} - C_{10})\Pr[X_n = 0|Z^1 \dots Z^{m-1}]}. \quad (6)$$

and 1 otherwise. From an implementation standpoint one may be tempted to formulate a decision D_n for node n at the end of the search process. However it is easily seen that then one must recall for each node when it was sensed. It is therefore more practical to update the decision variables D_n after every sensing step. In this way the algorithm behaves according to an *anytime* paradigm.

A. Search algorithm

At this point we have all the elements to offer an algorithmic sketch of the search strategy we presented, outlined in Algorithm 1. Inside a loop governed by the stopping condition formerly described, the searcher selects the next cell to sense (line 4), and after collecting a sensor reading Z_n^t it updates the posterior based on the rules enforcing the Type2 PQ constraint (lines 6 to 11).

Line 12 shows how the tree $\mathcal{T}$ is iteratively refined during the search, i.e., nodes are added to the tree when the sensor returns a detection at a leaf node not at the maximum depth. In this case the probability of the newly created nodes is initialized by enforcing the constraint in Eq. 2. Next, the

[4]Sometimes cost is called risk.

Algorithm 1 Searching for multiple intruders with Type 2 PQ

```
1: $\mathcal{T} \leftarrow$InitializeTree(Prior)
2: searchDone $\leftarrow$ **false**
3: **while not** searchDone **do**
4:   $n \leftarrow \arg\max_n I'(n),\ n \in \mathcal{N}(\mathcal{T})$
5:   Move to node $n$ and get sensor reading $Z_n^t$
6:   Update $p_n$ using Eq. 3
7:   Update decision for node $n$ using Eq. 6
8:   **for all** $m$ s.t. $m$ is an ancestor or descendant of $n$ **do**
9:     Update $p_m$ using Eq. 2 (ancestor) or 4 (descendant)
10:    Update decision for node $m$ using Eq. 6
11:  **end for**
12:  **if** $Z_n^t$ = 1 **and** $n \in \mathcal{L}(\mathcal{T})$ **and** $d(n) < \mathcal{D}$ **then**
13:    Create 4 children for node $n$ and initialize their probability with $q = \sqrt[4]{q_n}$
14:  **end if**
15:  Compute $U(\mathcal{T})$
16:  searchDone $\leftarrow$ **true**
17:  **for all** $n \in \mathcal{L}(\mathcal{T})$ **do**
18:    **if** $-p_n \log p_n > \varepsilon U(\mathcal{T})$ **then**
19:      searchDone $\leftarrow$ **false**
20:    **end if**
21:  **end for**
22: **end while**
23: **return** all leaves at depth $\mathcal{D}$ with $D_n = 1$
```

termination condition is evaluated (lines 15 to 21) and if it is not satisfied the loop restarts. Finally, we note that in line 23 the searcher returns as confirmed target locations only the leaves at maximum depth for which the decision variable D_n has been set to 1. This is coherent with the requirement of returning target locations only with maximum precision.

VI. Theoretical findings

In this section we sketch some *convergence* results for Type 2 PQ based on the search strategy we formulated. The take home message is that in the worst case scenario a Type2 PQ performs as a uniform grid. Hence, the performance of this well studied search strategy provides an upper bound for the performance of our new method. However, as it will be evidenced in the next section, in practice Type 2 PQ never approach this limit situation, thus providing a large increase in performance.

We recall that $\mathcal{N}(\mathcal{T})$ is the set of nodes in $\mathcal{T}$ and $\mathcal{L}(\mathcal{T})$ is the set of leaf nodes. Let $\mathcal{F}(\mathcal{T})$ be the set of nodes in the full tree of depth $\mathcal{D}$. In other words, $\mathcal{F}(\mathcal{T})$ is the full probabilistic quadtree after it has been fully expanded, i.e., all its leaves are at depth $\mathcal{D}$. Because the topology of the quadtree changes during the search, sets $\mathcal{N}(\mathcal{T})$ and $\mathcal{L}(\mathcal{T})$ should be indexed by time, like $\mathcal{N}_t(\mathcal{T})$ to indicate the set of nodes at time t. Finally, for each time step t, $\mathcal{N}_t(\mathcal{T}) \subseteq \mathcal{F}(\mathcal{T})$.

Assumptions: $\forall d: \quad 0 < \alpha(d) < 1$, and $0 < \beta(d) < 1$ i.e., the sensor has non-null error rates in the form of either false positives or missed detections. Moreover, we assume $\alpha(d), \beta(d) \neq 0.5$.

The motivation of these two assumptions are both practical and theoretical. If we had an error-free sensor some of the convergence results do not hold anymore (for example Lemma 2), although the main properties can still be stated in a different way. However, we are interested in inference using faulty sensors, and if one had a perfect sensor a different strategy would be used. For example, one could permanently prune part of the search space as soon as a no-detection is returned. The second hypothesis, $\alpha(d), \beta(d) \neq 0.5$, implies that the sensor is *informative*, i.e., it does not return a reading that is unrelated to the sensed region[5].

Lemma 1: If the searcher does not stop searching, then the number of times a leaf node is sensed is unbounded.

Proof. (sketch) A leaf node can not be permanently ignored from a certain time because otherwise its entropy will not decrease while the others will. Therefore, eventually, it will be the node maximizing the function I' and will then be selected. □

Lemma 2: When time diverges the probabilistic quadtree becomes fully filled, i.e.,

$$\lim_{t\to\infty} \mathcal{N}_t(\mathcal{T}) = \mathcal{F}(\mathcal{T}).$$

Proof. The proof is by contradiction. Let us assume that for each t, $\mathcal{N}_t(\mathcal{T}) \neq \mathcal{F}(\mathcal{T})$. Since the tree only expands and never contracts, this means that there is at least one leaf node n that is not at the maximum depth $\mathcal{D}$. Let k ($k = 0$ or 1) be the true value of the indicator variable X_n associated with $R(n)$. Because of the the hypotheses made about the sensor, $\Pr[Z_n = 0 | X_n = k] < 1$. After node n is sensed m times, the probability that node is not expanded (and then remains a leaf) is $\Pr[Z_n = 0 | X_n = k]^m$. Clearly, when m diverges this probability converges to 0. Because of Lemma 1 node n is visited infinitely often, and then the result follows. □

It is now straightforward to prove the following result.

Lemma 3: When time diverges the entropy of tree $\mathcal{T}_t$ converges to 0:

$$\lim_{t\to\infty} H(\mathcal{T}_t) = 0$$

Proof. (sketch). Based on our definition, the entropy of a tree $\mathcal{T}$ is the sum of the entropy of its leaves. Hence it suffices to show that the entropy of every indicator variable associated with the leaves of $\mathcal{T}$ converges to 0 when the number of steps grows. From Lemma 2 we know that for $t \to \infty$ the set of leaves of $\mathcal{T}$ approaches the set of leaves of $\mathcal{F}(\mathcal{T})$, i.e., the tree becomes full. Moreover, Lemma 1 states that every leaf will be visited infinitely often. Hence, because of the hypotheses we made about the sensor, in particular $\alpha(d), \beta(d) \neq 0.5$, it is straightforward to see that the for each leaf node n at the deepest level, its associated posterior p_n will eventually converge to either 0 or 1 (see also [9]). Therefore the entropy of every indicator variable associated with the leaves converges to 0 and then the claim follows. □

[5]A sensor with an error rate $\alpha(d), \beta(d) > 0.5$, is still useful by considering 1-α or $1 - \beta$; see also [9].

VII. Simulation results

In this section we present simulation results aiming to outline the performance of the approach we propose[6]. Throughout this section we assume that the searcher is not provided with any prior information about the number or location of targets inside the search area. Therefore the probabilistic quadtree is initialized using a uniform prior. To be specific, the tree is initialized to be a full quadtree of depth 5, i.e., a quadtree whose leaves are all at depth 5. The search area $\mathcal{A}$ is assumed to be a square whose edge L is 256 units long. Results presented refer to 100 different benchmark missions. Every benchmark mission was generated assuming each of the cells at maximum resolution hosts an intruder with a certain probability p_i. The value of p_i was chosen so that the expected[7] number of targets is 5.

In all missions we assumed that C_{01}, i.e., the cost for a missed detection, is 1000, whereas C_{10}, i.e., the cost for a false positive is 10. Moreover we set the upper bound on the search time to $T = 25000$.

We present two classes of experiments. The first batch aims to display the properties of the approach we propose. The second shows how a variable resolution representation largely outperforms search strategies based on uniform representations. A companion video also shows a full mission.

A. Performance of variable resolution search

Based on extensive tests we performed, the most critical parameter is ε, i.e., the value governing when to terminate the search. This value necessarily influences the accuracy of results in terms of expected cost, but also the time to terminate a search mission. Table I summarizes how relevant quantities vary for different values of ε. The table displays Expected Time to Termination (ETTT), the number of instances in which the searcher had to terminate its search because its allotted time expired (OT, overtime), the average cost (C), the cumulative number of false positives (FP), and the cumulative number of missed detections (MD).

ε	ETTT	OT	C	FP	MD
0.1	19522.47	16	624	24	6
0.08	20189.54	19	424	24	4
0.05	21344.27	34	324	24	3

TABLE I

Searcher performance for different values of ε. Data are collected out of 100 benchmark trials. Note that the last two columns display cumulative values, i.e., the sum of all false positives and missed detection in 100 runs.

The reader should note that in no mission is there more than one missed detection, whereas the number of false positives has been always below three per mission. It can be observed that, as expected, a decrease in ε implies an improvement in terms of performance (i.e., the number of missed detections decreases) and an increase in time needed to terminate a mission, with a significant growth in the number of missions that terminate because the allotted time limit has been reached (OT column in Table I). Without changing other parameters, further decreases in ε will not lead to any improvement in terms of missed detections because the three missions where a missed detection occurred terminated because the allotted time limit was reached. This outcome should not be regarded as a failure of the algorithm, but rather as a consequence of imposing a relatively short upper bound time T in order to complete the assignment.

The companion video shows an example mission illustrating how the posterior p_i varies during the effort (left panel) and how the hierarchical representation is iteratively refined during the search (right panel). The video also evidences how the searcher changes its elevation during the search in order to adjust the sensing resolution. Fig. 3 displays the trend of $U(\mathcal{T})$ (blue line) for the search mission illustrated in the video. The green line shows instead the number of nodes with entropy larger than $\varepsilon \cdot U(\mathcal{T})$. According to the stopping criterion we implemented, the mission terminates when this number reaches 0. The reader shall note that even though the mission terminates after 17369 steps, $U(\mathcal{T})$ was evaluated only 641 times, i.e., only after the posterior is updated because of a sensor reading.

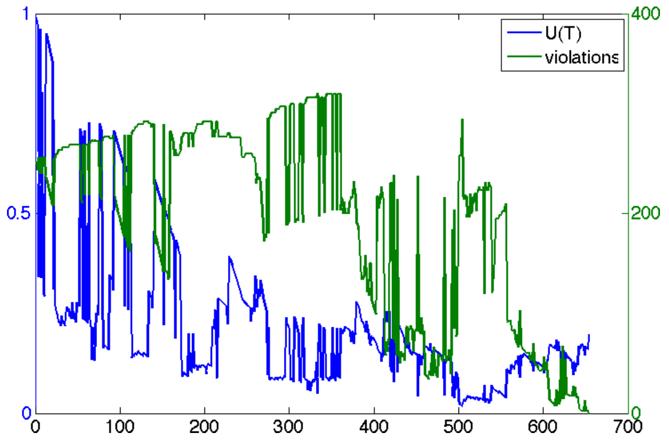

Fig. 3. Trend of $U(\mathcal{T})$ and number of nodes with entropy larger than $\varepsilon \cdot U(\mathcal{T})$ for the mission displayed in the companion video.

Finally, Fig. 4 shows a histogram of the number of nodes in the quadtrees at the end of the search effort for the different values of ε. This histogram outlines the effectiveness of the proposed technique in terms of limiting the search space. The uniform grid guaranteeing the same accuracy would require $256^2 = 65536$ cells. In order to put the histogram into the right perspective, in each trial the PQ is initialized as a tree with 341 nodes.

B. Comparison with uniform representations

We finally compare our PQ based search with a search method based on uniform representations. Using a uniform grid at maximum resolution (i.e., where every leaf has unary

[6]Matlab code reproducing the results presented in this section is available for download at `http://robotics.ucmerced.edu`.

[7]Random variables indicating the presence of a target inside a cell at maxim depth are iid. The overall number of targets follows then a binomial distribution of parameters $\mathcal{B} = (L^2, p_i)$ whose expectation is $L^2 p_i$.

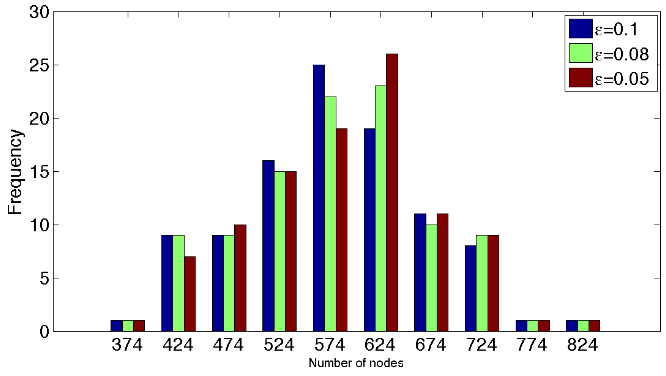

Fig. 4. Distribution of the number of nodes in the PQ at the end of 100 benchmark problems for different ε values.

area) is not feasible because in the search space (65536 cells, as mentioned above) is too large to be considered within the time bound of $T = 25000$ (it takes one time step for the robot to move from one cell to an adjacent one). We therefore run a uniform search with a uniform grid initialized as the uniform tree of depth 6, and leave all other parameters unchanged. The reader should note that in this case the searcher uses a grid where every cell has area 64 (8 $\times 8$); therefore the precision of an indication *target present* is nominally 64 times less accurate than the one provided by the PQ. However, because of the larger search space to explore, the searcher fails to formulate any search decision within the allotted time limit T, and also fails to identify any intruder. Indeed, given the size of the search area $\mathcal{A}$ and the variable resolution of the sensor, the chosen set of parameters is unbearable for the searcher using a fixed resolution grid, whereas it can be handled by a searcher using PQ. These findings are consistent with what we found in [4] where uniform grids were also found to underperform when dealing with Type1 PQ.

VIII. Conclusions and future work

With this paper we argue that the most basic theoretical properties of probabilistic quadtrees have been investigated. Extensive simulation results presented in this manuscript and in [4] show that this representation offers a competitive alternative to uniform spatial representation. From our perspective the next major effort will therefore be devoted to the implementation and field validation of this framework on the AirRobot platform shown in Fig. 1. Efforts are underway to define and implement a target detection algorithm using the robot borne camera and to characterize its performance in terms of false positives and missed detections. Additional investigations into the sensitivity of search and decision parameters, e.g., different costs of decision errors, are also forthcoming in future studies. Finally, we plan to extend this framework in order to coordinate multiple searchers operating in the same area.

Acknolwdgments

The first author is partially supported by the Naval Postgraduate School, Monterey under grant #N00244-11-1-0023. The third author is partially supported by the Office of Naval Research and the NPS-USSOCOM Field Experimentation Cooperative.

References

[1] F. Bougault, T. Furukawa, and H. Durrant-Whyte. Optimal search for a lost target in a bayesian world. In S. Yuta et al., editor, *Field and service robotics*, volume 24 of *STAR*, pages 209–222. Springer, 2006.

[2] F. Bourgault, T. Furukawa, and H. Durrant-Whyte. Coordinated decentralized search for a lost target in a Bayesian world. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 48–53, 2003.

[3] K. Bryant and C. Carthel. A Bayesian approach to predicting an unknown number of targets based on sensor performance. In *Proceedingso of the International Conference on Information Fusion*, 2006.

[4] 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.

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

[6] Bernard O Koopman. Search and Its Optimization. *The American Mathematical Monthly*, 86(7):527–540, 1979.

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

[8] G. Kraetzschmar, G. Pages Gassull, and K. Uhl. Probabilistic quadtrees for variable-resolution mapping of large environments. In *Proceedings of the 5th IFAC/EURON Conference*, 2004.

[9] M. Kress, R. Szechtman, and J.S. Jones. Ecient employment of non-reactive sensors. *Military Operations Research*, 13:45–57, 2008.

[10] B. Lavis, T. Furukawa, and H. Durrant-Whyte. Dynamic space reconfiguration for Bayesian search and tracking with moving targets. *Autonomous Robots*, 24:387–389, 2008.

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

[12] D.J.C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

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

[14] C. Stachniss. *Robotic Mapping and Exploration*, volume 55 of *STAR*. Springer, 2009.

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

[16] J Tang, A. Singh, N. Goehausen, and Pieter Abbeel. Parametrized maneuver learning for autonomous helicopter flight. In *Proceedings IEEE International Conference on Robotics and Automation*, 2010.

[17] 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.

[18] Y. Wang, I.I. Hussein, D.R. Brown III, and R.S. Erwin. Cost-aware Bayesian sequential decision-making for domain search and object classification. In *Proceedings of the IEEE International Conference on Decision and Control*, pages 7196 – 7201, 2010.

[19] E. Wong, F. Bourgault, and T. Furukawa. Multi-vehicle bayesian search for multiple lost targets. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3169–3174, 2005.