Variable Resolution Search with Quadrotors: Theory and Practice

Stefano Carpin
University of California, Merced
5200 N Lake Rd, Merced, CA 95343, USA

Nicola Basilico
Polytechnic University of Milan
via Ponzio 33/a, Milan, 20133, Italy

Derek Burch
University of California, Merced
5200 N Lake Rd, Merced, CA 95343, USA

Timothy H. Chung
Naval Postgraduate School
777 Dyer Road, Monterey, CA, 93943, USA

Mathias Kölsch
Naval Postgraduate School
700 Dyer Road, Monterey, CA, 93943, USA

Abstract

This paper presents a variable resolution framework for autonomously searching stationary targets in a bounded area. Theoretical formulations are also described for using a *probabilistic quadtree* data structure, which incorporates imperfect Bayesian (false positive and false negative) detections and informs the searcher's route based on optimizing information gain. Live-fly field experimentation results using a quadrotor unmanned aerial vehicle validate the proposed methodologies and demonstrate an integrated system with autonomous control and embedded object detection for probabilistic search in realistic operational settings. Lessons learned from these field trials include characterization of altitude-dependent detection performance and we also present a benchmark data set of outdoor aerial imagery for search and detection applications.

1 Introduction

The prominence of and the increasing reliance on unmanned systems for information gathering tasks in military and civilian contexts have been met with equally fervent advances in the theoretical and applied robotics communities. Especially invaluable in intelligence, surveillance, and reconnaissance (ISR) missions, a growing family of unmanned aerial vehicles (UAVs) has significantly and positively impacted mission areas such as homeland security for border protection; maritime domain awareness such as in counter piracy operations; force protection of personnel, e.g., in supply convoys; and urban search and rescue. The advent of smaller, tactical UAVs, such as the quadrotor platform, offers enhanced, if not novel, capabilities for collecting mission-relevant information in these and other contexts. For example, quadrotors offer motion control authority in any direction, can loiter in place, and can move vertically, thus varying the field of view while remaining stationed above the same point. These properties make them ideal for missions where the goal is the tactical search for an object or person of interest, such as in the detection of a stolen vehicle or the search and rescue of a downed aircraft or pilot.

Yet with these platforms and a constantly expanding set of sensing modalities also comes a deluge of sensor data, exceeding the abilities of operators and analysts alike to process, exploit, and disseminate the information in the form of actionable decisions relevant to the mission. Though offline processing of the data is possible albeit still intensive, many tasks require real-time decisions to be made based on the observed scene. In particular, in missions such as physical search for one or more targets, the ability to adaptively update the search route with new information from observations is key to optimizing the probability of detection or time until the target(s) are found. The intuition that adaptive search performs better than offline methods has been investigated in Hubenco et al. (2011), where it is shown that when the sensor is informative an adaptive search strategy outperforms offline approaches ignoring data acquired while the mission develops. The approach we propose falls under the category of online, adaptive search methods.

It is in these information-rich contexts where guidance for the operators in the form of decision support tools (also known as tactical decision aids) can make significant improvements in mission performance. In particular, the operator or commander can utilize a given decision support tool which automates much of the decision processes. Recognizing the inextricable interactions necessary between humans and their robotic counterparts when conducting complex missions, a growing trend in these manned-unmanned teams is to use decision support tools to facilitate cooperation and a common informational picture. Integral elements to such an interface in the context of target search includes: (a) optimized search route recommendations stemming from algorithmic or analytic methods, and also (b) automated detection and perception of targets using machine vision techniques.

This paper proposes a theoretical framework to address both of these elements and also demonstrates its effectiveness and applicability through field experiments with a quadrotor UAV in a realistic operational environment. The main contributions of the presented work include a novel variable resolution data structure and theoretical approach for conducting probabilistic search of a target. Further, the field experiments in an outdoor, large-scale setting described herein and the experimental validation of the proposed approach using a quadrotor UAV provide valuable data and operational insights into realistic applications of search.

The remainder of this manuscript is organized as follows. Related work is discussed in Section 2. We formulate a model for variable resolution search in Section 3 that leverages the quadrotor's strengths as an ISR platform and introduces the *probabilistic quadtree* data structure. Section 4 describes the variable resolution search process which includes computing optimized search paths and updating the searcher's belief of target presence or absence. These analytic methods are experimentally validated in outdoor field conditions, from which collection of a unique image dataset and characterization of the automated detection methods are summarized in Section 5. Section 6 concludes the paper with discussions and avenues for future work.

2 Related work

The problem of search for objects stems from Koopman (1979) and the theory of search, where applied probability models coupled with mathematical optimization led not only to the effective defense against U-boats in World War II but also to the birth of operations research as a discipline. Since this time, many researchers have explored the mathematical foundations of search theory. For a survey of the literature, see Benkoski et al. (1991) for works through 1991 and Chung et al. (2011) for more recent results. These classical works largely investigate the offline optimization of search allocation, where optimized plans are determined (often laboriously) and executed without incorporating newly gathered information or potentially changing objectives. The primary measures of performance used in these studies include metrics such as maximal probability of detection of the target(s) or the minimal time until detection occurs for one or more targets. A recent work that combines both these two metrics within a search and rescue application domain has been presented in (Lin and Goodrich, 2009). Constant altitude UAVs with fixed camera footprint size are considered. Given a prior probability distribution, the problem is to generate paths that maximize detection probability and minimize needed time. Such a problem is modeled with combinatorial optimization and different algorithms to tackle it are proposed.

However, with the advent of improved computational resources, *adaptive* search plans (Stone, 1989) are possible. These methods, such as optimal control techniques for search trajectories (Foraker, 2011) or computational constructs like POMDPs (Eagle, 1984), incorporate new observations to inform their next actions, which benefits from the most updated information to guide the search process. Works aligned with the search models presented in this paper include those by Furukawa, Hedrick and respective colleagues. In particular, Bourgault et al. (2006) and Lavis et al. (2008) cast the search problem as a Bayesian framework that accounts for imperfect sensors, that is, sensors providing false positive and negative detections. Likewise, Tisdale et al. (2009) exploit Bayesian filtering over a grid where sensing and localization errors are integrated to provide a control policy. Optimal search paths are planned in a receding horizon setting, with the aim of guiding a team of searching UAVs. Authors discuss paths evaluation in terms of entropy-based information gain and expected detection probability. Sometimes, assumptions about the probabilistic framework can improve the search strategy. Some work in this direction has been recently presented in (Bonnie et al., 2012). This work studies the problem of probabilistic search in a continuous domain where a robot is endowed with a binary faulty sensor. In building a probabilistic framework, authors adopt a sensor model whose accuracy is affected by the distance between the robot and the object to be found. Under the hypothesis that prior and sensor model belong to the exponential family of distributions, some properties about the posterior are shown, i.e., that it belongs to a finitely parameterizable family of functions. Exploiting this result, a probabilistic search strategy is defined via gradient ascent methods. Recently, some interest has been devoted to the study of this problem under unreliable prior probability distributions. A work following this research line is (Sisso et al., 2010) where robustness to initial uncertainty is explicit considered in optimizing search strategies. The development of adaptive plans stemming from the search literature extends also to similar domains such as patrolling. Relevant work in this domain has been carried on by Frazzoli and colleagues (Huynh et al., 2010). The authors consider the problem of detecting a number of dynamic targets, searching for effective strategies and analyzing their performance. A common line in these works, much like the majority of other related works, is the reliance on a uniform representation of the environment in which the search strategy is constrained to operate.

Another common aspect of such works, and of search literature in general, is the assumption of a model for the object to be searched for, also called the *target*. Broadly speaking, a target model can be described along two main dimensions: stationary/moving target and single/multiple targets. Considering this, the aforementioned sample of works can be divided as follows. Bourgault et al. (2006), Lin and Goodrich (2009), Tisdale et al. (2009), and Bonnie et al. (2012) deal with a single stationary target while Huynh et al. (2010) and Sisso et al. (2010) consider multiple stationary ones. Eagle (1984) and Bourgault et al. (2006) address the case of a single moving target while techniques suitable for multiple moving ones are proposed by Lavis et al. (2008) and Foraker (2011). Maintaining probabilistic knowledge about multiple targets or integrating a target's movement model in the decision process can challenge a search strategy's definition and obtained performance, see for example (Carpin et al., 2011). To align with the performed field experiments and to primarily focus on probabilistic quadtrees, this work builds on (Chung and Carpin, 2011) and considers a single and stationary target.

Though still nascent, efforts to conduct live-fly experiments with unmanned aerial vehicles continues to motivate this line of research in search and surveillance problems (see Kendoul (2012) for a recent survey on unmanned rotorcraft systems). Besides ISR applications, search and rescue and disaster mitigation continue to be areas where the use of UAVs promises to bring great benefits (Goodrich et al., 2008; Pratt et al., 2009; Schmid et al., 2012). The use of search models in a decision support context to guide the search has also been demonstrated by Jones et al. (2011), where the authors deployed an integrated system employing fixed-wing UAVs to conduct search for target vehicles in the field. This work combines various elements necessary for real-world implementation, including guidance, navigation and control and networked flight assets with search planning and automated image processing. Similarly, the search problem formulated as a mixed integer program is demonstrated in field experiments in Chung et al. (2009), allowing for allocation of multiple heterogeneous aerial sensors looking for walking individuals to conduct a search and identification mission. However, in both of the previous works, the sensor platforms are also limited to fixed sensor footprints and an arbitrary uniform discretization of the search area.

A key challenge to robust implementation, however, remains the computational tractability of the above approaches, especially with the demand of adaptive and iterative calculations necessary to update the (probabilistic) representations of the current information state. Specifically, the representations in the aforementioned works rely on uniform discretization and/or static partitioning of the search environment. However, this representation at fixed resolution restricts the problem solutions to be either computationally infeasible (for cases of moderate resolutions) or operationally irrelevant (for solvable but low resolutions). Such is the nature of search problems, where the search regions are large but the subject of search is relatively small (National Search And Rescue Committee, 2000).

In this manner, multi-scale methods, i.e., those that can vary the resolution of the search effort and/or environment representation during the course of the search process, are both requisite and promising for addressing search scenarios of practical interest. Broadly speaking, multi-scale representations are largely adopted in different research fields such as Computer Graphics (see (Pajarola and Gobbetti, 2007) for a particular application example) and Computational Geometry (see (de Berg et al., 2000) for an extensive treatment of the subject). Considering our domain, recent works by Schwager et al. (2011) explore discretizing the environment using Voronoi partitions, adapting the sizes based on a balancing of information content within these partitions. However, this formulation does not lend itself to computational advantages offered by a multi-resolution data structure. In other works, Waharte et al. (2010) examine how varying altitudes of an aerial sensor platform induces observations of partial areas which may overlap multiple grid cells; however, these insights are not explicitly used to construct a variable resolution representation of the search environment nor to inform the search process.

3 A model for variable resolution search

In this section we present our variable resolution model. The model is developed to solve search problems where the UAV looks for one or more stationary items of interest possibly located inside a given area. Our assumptions, corroborated by our experimental setup described in Section 5, are the following.

1. The UAV searcher is equipped with a downward pointing camera and the area captured by the camera therefore varies as the UAV alters its elevation. This variation in the sensed area, already evidenced in Waharte et al. (2010), implies that objects of interest appear in images at different sizes, depending on the searcher elevation.
2. The searcher uses a target detection algorithm to identify objects of interest inside acquired imagery. The target detection algorithm is prone to false positive and missed detections and its performance depends on the elevation. We assume error rates are known before the mission starts, i.e., they can be estimated off-line from previously acquired imagery.

These two assumptions are discussed in more detail in the following.

3.1 Probabilistic quadtrees (PQs)

The first hypothesis calls for a variable resolution spatial representation accounting for the trade off between field of view and elevation. We opt for a representation based on quadtrees, a popular data structure heavily used in computational geometry (de Berg et al., 2000). Figure 1 illustrates the idea. When the searcher hovers at a higher elevation its camera sees a larger area, while at lower elevations the area is smaller. With a fixed resolution camera this means a lower accuracy at higher elevations and vice versa.

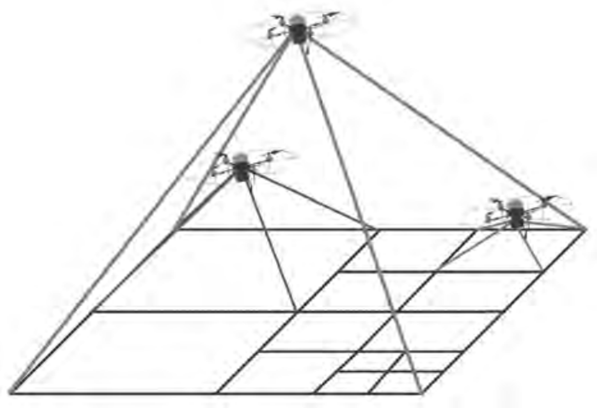


Figure 1: When flying at different elevations, the nadir-pointing camera captures areas of different size. The probabilistic quadtree structure is used to model this tradeoff.

According to its classic definition, a quadtree $\mathcal{T}$ is a rooted tree in which every node n is associated with a square[1] area $A(n)$. Every internal node n has four children, and the area associated with its children is obtained by splitting $A(n)$ in four equally sized squares. In our search application we dictate that the node at the root of the tree is associated with the entire search area. Note that there is a well defined relationship between the size $A(n)$, the depth of node n in the tree, and the elevation of the UAV when taking a picture covering $A(n)$. Specific examples will be given while discussing the experimental setup in Section 5. To every node n we associate a binary random variable X_n indicating whether one or more targets are located inside $A(n)$ or not, and we define $p_n = \Pr[X_n = 1]$, i.e., p_n is the probability that a target is inside $A(n)$. Because of the association between nodes and random variables, we dubbed this data structure *probabilistic quadtree* (PQ). Note that $\mathcal{A}$ needs not be a full tree, i.e., its leaves may have different depth. Therefore the tree may have deeper branches (and thus higher resolution) in some areas and shallower ones in others. Probabilistic quadtrees introduce a novelty with respect to the customary approach embraced by numerous techniques in literature where search develops over uniform representations (see, for example, Lavis et al. (2008) and Lin and Goodrich (2009)). A full probabilistic quadtree is associated with a stack of regular grids (see also Figure 4), i.e., one for each level. Once a maximum depth d is defined for a PQ, the grid associated with the maximum depth induces the maximum resolution achievable, i.e., the size of $A(n)$ when n is a node at the maximum depth. We close this section introducing two additional symbols used in the following. For a tree $\mathcal{T}$ we use $\mathcal{L}(\mathcal{T})$ to indicate the set of leaves of $\mathcal{T}$, and k to indicate the number of nodes in $\mathcal{T}$.

In our model the searcher uses its target detection algorithm only when it is located at a position associated with a node in the PQ. When traveling between nodes in the tree the target detection

[1]We here assume it is a square for simplicity, but the approach equally works when the area is a rectangle.

algorithm is not used. This hypothesis is corroborated by our field experience. With a lightweight quadrotor images taken while moving from point to point are very blurry and greatly compromise the performance of the target detection algorithm. Therefore images are only captured when the robot is stationary after having reached a planned target point.

3.2 Target detection algorithm

The second hypothesis defines the behavior of the target detection algorithm. Target detection operates as a binary classifier that, given an image, returns 0 if no targets are located inside the image and 1 otherwise. Because of unavoidable errors, its functioning is described probabilistically as follows. Let Z_n be the value returned by the target detector after analyzing an image covering $A(n)$. Z_n is modeled as a binary random variable and we define

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

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

where $d(n)$ is the depth of n in quadtree $\mathcal{T}$. In the remaining of this paper we hypothesize these probabilities are known to the searcher. The formulas formalize our assumption that error rates for false positive α and missed detection β are elevation dependent (recall that there is a one-to-one correspondence between elevation and node depth $d(n)$ and that X_n indicates whether there is a target in $A(n)$ or not). Although one could anticipate error rates are monotonic, our experience with off-the-shelf target detection algorithm shows this is not always the case (see Section 5.3) and the trend depends on the specific detector being used.

4 Aerial search with probabilistic quadtrees

We use probabilistic quadtrees to solve two types of search problems. In the first one, coined Type1 PQ and introduced in (Chung and Carpin, 2011), at most one target is located inside the search area. In the second one (Type2 PQ), an arbitrary number of targets may be present. We here discuss Type1 PQ only, because it is related to the field experiments presented later on. We refer the reader to (Carpin et al., 2011) for a discussion about the case with multiple targets. For the sake of completeness, in the following we include a comprehensive discussion about Type1 PQ.

In our search problem a target is found when it is localized inside a square with a given size s. Once a square search area $\mathcal{S}$ and a desired size s are specified, the maximum PQ depth $\mathcal{D}$ is therefore determined.[2] During the search effort new nodes will be added to the PQ and its depth will increase with the objective of generating leaves at depth $\mathcal{D}$ covering the area where the target is located. This tree expansion is a prerequisite for being able to eventually locate a target inside a cell of side s.

[2]To be precise, one should further assume that the ratio between the side of $\mathcal{S}$ and s is a power of 2. When this is not the case the ratio can be achieved by arbitrarily enlarging $\mathcal{S}$. Without loss of generality, in the following we will ignore this technicality.

A Type1 PQ problem is defined as follows. Let $\mathcal{T}$ be a PQ and let r be its root node with $A(r) = \mathcal{S}$, i.e., the root of the tree is associated with the entire search area. Assuming there is at most one target inside $\mathcal{S}$, plan a sequence of sensing operations so that the searcher eventually terminates with a search decision. A search decision can either be *No target present* or *Target found in node n* where n is a node in $\mathcal{T}$ at maximum depth $\mathcal{D}$.

Before discussing the individual steps to solve the search problem, we remark that if n is an internal node in a Type1 PQ $\mathcal{T}$ and n_1, n_2, n_3, n_4 are its four children, then the following relationship holds because there is at most one target:

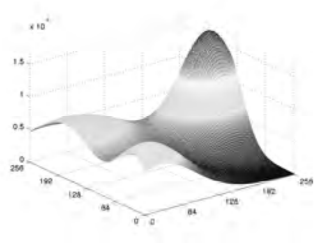
$$p_n = p_1 + p_2 + p_3 + p_4. \tag{1}$$

Moreover, notice that knowing there is at most one target implies all variables X_n associated with leaves are correlated, as already observed in (Chung, 2010).

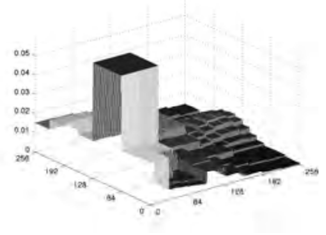
4.1 Initializing a probabilistic quadtree

Probabilistic quadtrees can be initialized by taking advantage of informative priors about the location of targets, or based on uniform uninformative priors when such information is not available. We assume priors represented on a uniform grid discretization of the environment are provided to the algorithm, as this is the format commonly used by human operators performing search operations in the field (see e.g., National SAR Committee (2000)). The goal of the initialization procedure is therefore to compress a prior given over a uniform, fine-grain grid into a coarser, faster-to-search PQ structure. To this end, we start building a full, maximum depth PQ whose leaves are associated with the same grid cells provided by the uniform grid. Then, the tree is iteratively pruned to reduce its size to a target value so that processing can be faster. Pruning works as follows. For every internal node n we compute the difference between the highest and lowest probability stored in its four children. We call this value *disparity* and store it with the node. Next, we sort internal nodes in ascending order by their disparity value and repeatedly remove their children from the tree starting from those with lowest disparity. Every time a set of children is removed the size of the tree decreases. However, because of Eq. 1 every parent stores the sum of the probability of its children, and therefore prior information stored in nodes being removed is not lost, but rather aggregated on the parent. We stop this process when the tree size falls below a target size. The rationale behind this idea is that nodes with low disparity have a set of children with similar probability and these can be adequately represented by their parent. Following this method, the final PQ will have a higher resolution where the prior significantly varies, and a lower resolution in areas where the prior is flat, thus truly exploiting its variable resolution nature. Figure 2 illustrates this idea with some examples. One should finally notice that different pruning strategies could be implemented to reduce the size of the initially full tree, but the rest of the search strategy would not be affected, though results are, of course, in general dependent on the specific initialization performed.

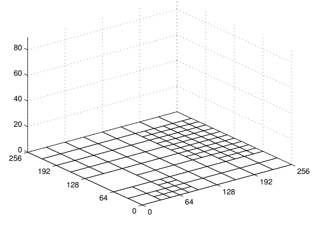
Initialization is finalized by creating a dummy node $n_\emptyset$ representing the event that no target is located inside the search area. Its probability $p_{n_\emptyset}$ is always set to $1 - \sum_{n \in \mathcal{L}(\mathcal{T})} p_n$, i.e., its value normalizes the overall probability to 1 (i.e., a target is present or is not present.) Note that we consider $n_\emptyset$ to be a leaf, so it belongs to the previously defined set $\mathcal{L}(\mathcal{T})$. While $n_\emptyset$ is introduced



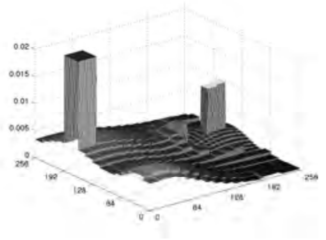
(a) Prior



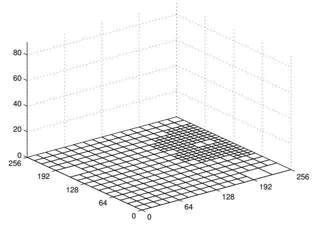
(b) PQ prior with 150 nodes



(c) Quadtree with 150 nodes



(d) PQ prior with 500 nodes



(e) Quadtree with 500 nodes

Figure 2: Subfigure a) shows a prior represented over a uniform grid with 65536 cells. Subfigures b) and d) show the PQ representing the same prior with 150 and 500 nodes, respectively. Subfigures c) and e) show the variable resolution spatial representation induced by the leaves of the quadtrees. For the PQs, note that high peaks are associated with large areas because they account for the total probability of the area. Note also that the spatial subdivision is finer in areas with high variability in the prior and coarser where there is less variability.

primarily for normalization purposes, its explicit inclusion in $\mathcal{L}(\mathcal{T})$ will simplify the notation used in some derivations presented in the Appendix.

4.2 Updating a probabilistic quadtree

During the search mission the searcher repeatedly acquires images of area $A(n)$ for different nodes n and then runs the target detector obtaining a detection result Z_n. Based on this value, all probabilities associated with nodes in $\mathcal{T}$ are updated and the tree is possibly expanded. These two steps are described in the following.

Probability updates. First, p_n is updated using Bayes formula and the aforementioned sensor model. This update will violate the constraint given by Eq. 1 and therefore changes need to be propagated to both ancestors and descendants of n, if any. Probability values for all nodes' ancestors of n are updated by recursively applying Eq. 1 from the parent of n towards the root. If n is an internal node, then its probability change needs to be recursively propagated to its descendants as well. This update could be done in different ways and we opt for a proportional propagation. More precisely, let $\delta p = p_n^t - p_n^{t-1}$ be the change of probability in p_n due to the Bayes update between time $t-1$

and t. Let $p_{n_i}^{t-1}$ and $p_{n_i}^t$ be the probability associated with its i-th child of n at time $t-1$ and t, i.e., before and after the update. These values are then related by the following equation

$$p_{n_i}^t = p_{n_i}^{t-1} + \left(\frac{p_{n_i}^{t-1}}{p_n^{t-1}} \right) \delta p. \tag{2}$$

i.e., variation δ is proportionally redistributed over all children of n. This update is then recursively propagated to all descendants of n. The reader should notice that any downwards propagating strategy preserving the constraint given by Eq. 1 is legitimate. The update in Eq. 2 favors simplicity and fits well the hypothesis that the target detection algorithm just indicates whether a target is inside the image, but does not provide information about its position (see Section 3.2). Under these assumptions, it appears meaningful to spread the update on all descendants based on their prior, rather than favoring one of them.

Finally, since at most one target is present in the search area, all random variables X_i associated with leafs in $\mathcal{T}$ are correlated and a change in the probability value associated with one leaf implies a change in all other leaves because these value must add to 1. Evidently, the process just described generates a change of probability in one or more leaf descendants of n, and therefore the probability values for all leaves need to be updated. The following closed-form formulas were derived in (Chung, 2010). We start by defining Φ and Ψ, two functions of the detection variable obtained scanning node n at time t:

$$\Phi(Z_n^t) = (1 - Z_n^t)(1 - \alpha(d(n))) + Z_n^t \alpha(d(n))$$
$$\Psi(Z_n^t) = (1 - Z_n^t)\beta(d(n)) + Z_n^t(1 - \beta(d(n)))$$

These two functions introduce the role of false positive and false negative error conditional probabilities and are related to the normalization factor for the following Bayesian update for the posterior of cell m given that sensing happened in cell n

$$p_m^t = \frac{\Theta_m(Z_n^t) p_m^{t-1}}{\Phi(Z_n^t)(1 - p_n^t) + \Psi(Z_n^t) p_n^t}. \tag{3}$$

where the term

$$\Theta_m(Z_n^t) = \begin{cases} \Psi(Z_n^t) & \text{if } m = n \\ \Phi(Z_m^t) & \text{if } m \neq n \end{cases}$$

considers the case where sensing happened in m or in n. Recognizing that the posterior depends not only on the detection variable itself but also on the sensed location, the above compact form for the Bayesian update succinctly captures the various cases found in discrete search.

After all leaves have been updated, Eq. 1 is repeatedly computed from the leaves to the root to restore the constraint at the internal nodes. We conclude this discussion noting that the complexity of the update is $\mathcal{O}(k)$, i.e., linear in the number of nodes in $\mathcal{T}$. It is important to remember that this linear complexity is not a major drawback for a PQ, because the data structure has far fewer nodes when compared to a grid with uniform resolution.

Tree expansion. Tree $\mathcal{T}$ will be expanded when n is a leaf node at depth $d < \mathcal{D}$ and $Z_n = 1$. This approach to refinement is motivated by the requirement that a target can be localized only within a leaf of maximum depth $\mathcal{D}$, i.e., in a cell with the finest resolution. Therefore, refinement of the variable resolution data structure occurs only when a detection is returned at a resolution insufficient to positively localize a target. Expansion consists in making n an internal node by adding four children. The probability of the four new nodes is initialized by equally splitting p_n in four, thus immediately enforcing the constraint given in Eq. 1.

4.3 Planning where to sense next

The role of the planner is to decide where the UAV searcher should sense next. We opt for a weighted information-gain approach, similar to (Stachniss, 2009). When deciding where to sense next, the searcher chooses the cell maximizing the expected information gain, i.e., the cell maximizing the expected decrease of entropy of the tree. In a Type 1 PQ the entropy is easily determined to be the sum of the entropies associated with the leaves, where the entropy of a leaf node n is $-p_n \log_2 p_n$. In order to avoid oscillating behaviors between far away cells, information gain is combined with traveled distance. When deciding where to go next, the searcher computes the following function for every node n in $\mathcal{T}$:

$$I'(n) = \left[\gamma \frac{I(n)}{\max_{n' \in \mathcal{T}} I(n')} - (1-\gamma)\frac{D(n^*, n)}{\max_{n' \in \mathcal{T}} D(n^*, n')}\right]. \tag{4}$$

In the former equation, $I(n)$ is the expected information gain obtained when sensing node n, and $D(n, n^*)$ is the distance between candidate node n and node n^* where the UAV is currently located. Parameter γ sets the relative weight of the two components, and in Section 5 we will evaluate how it impacts the performance of the algorithm. One could also notice that γ balances between exploration and exploitation. A large value of γ produces a global search behavior with the searcher more inclined to travel longer distances to inspect promising areas, whereas smaller γ values induce a more local behavior. A different way to describe this search strategy is saying that it is spatially global but temporally greedy, because it embraces a single step plan. It should also be noted that one could consider a multi-step strategy, but this comes at a significantly higher computational cost, and our preliminary experiments did not outline a meaningful performance increase. After $I'(n)$ has been computed for all nodes, the UAV then travels to the node n_{next} defined as

$$n_{next} = \arg\max_{n \in \mathcal{T}} I'(n). \tag{5}$$

Note that n_{next} encodes both a position on the grid and an elevation because it is en element in the PQ structure. A straightforward computation for n_{next} leads to an algorithm with complexity $\mathcal{O}(k^2)$. The reason is that to compute n_{next} one has to be compute $I'(n)$ for each of the k nodes in $\mathcal{T}$, and from Eq. 4 one can compute $I'(n)$ in $O(k)$ because of the need to compute $\max_{n' \in \mathcal{T}}$. However, exploiting the structural properties we introduced when defining a PQ and the fact that all the random variables X_n associated with leaves are correlated, it is possible to determine n_{next} in $\mathcal{O}(k)$. A detailed derivation of these results is offered in the Appendix.

4.4 Terminating the search

The search terminates when either $p_{n_\emptyset}$ exceeds a certain threshold p_{nt} (*n*o *t*arget), or when the probability of a node n at maximum depth $\mathcal{D}$ crosses a possibly different threshold p_{tf} (*t*arget *f*ound). Evidently, in the former case the searcher output for the search decision will be *No target present* whereas in the latter the decision will be *Target found in node n*.

5 Field Experiments

The proposed search algorithm has been experimentally validated on a data set collected at the McMillan air field at Camp Roberts, California on November 5-6, 2011[3].

5.1 Aerial Platform

Aerial imagery was collected using an AirRobot commercial platform (see Figure 3). The AirRobot is a quadrotor with brushless and gearless electric motors with a diameter of 1m. The platform can fly up to 300m with a maximum climbing speed of 2m/s and maximum horizontal speed of 10m/s. Its battery ensures a flight time of up to 30 minutes. The AirRobot has a payload of 200g and is equipped with a gimbaled color camera with a resolution of 640×480 pixels. Being a closed proprietary platform, it is not easy to exchange its camera for a better one; therefore we used the default one even though, as evidenced later, it is often far from optimal. Additional sensors include GPS, an elevation sensor, and a gyroscope. The platform has minimal onboard computational capabilities and receives commands through a digital communication channel ensuring a deployment radius of up to 1500m with direct line of sight. The AirRobot can be operated either manually with an operator control unit, or via a proprietary API. Irrespective of the control mode, data from the robot (imagery and telemetry) can be streamed to a PC connected to the communication station for online processing or storage.

5.2 Data set

Autonomous operation of UAVs at Camp Roberts is limited due to safety concerns; therefore it has not been possible to run the planner while controlling the AirRobot in real time. Our data collection effort has therefore been aimed to collect images and telemetry data to be able to synthesize various offline missions at a later time. During data collection the AirRobot was manually flown though waypoints of interest while time-stamped images and telemetry data (GPS, elevation, roll/pitch/yaw) were recorded at a frequency of 30 Hz. Due to other air traffic and deconfliction rules, during the first day the platform was restricted to fly below 400ft (122m), whereas sustained winds on the second day imposed a ceiling at 200ft (61m). These constraints determined the size of the probabilistic quadtrees considered in the two experiments. Given that in both cases leaf nodes were located at 25ft (7.62m), it followed that during the first experiment the quadtree had depth

[3]Code implementing the experiments described in this section is available for download on `http://robotics.ucmerced.edu` Datasets are also available upon request.

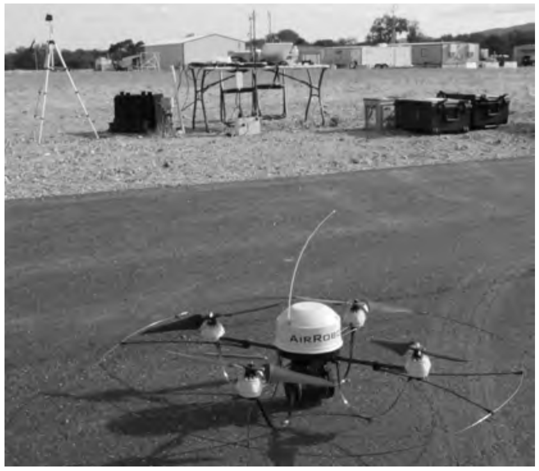


Figure 3: The AirRobot aerial platform used for field experiments. In the background the control station and the laptop logging information collected during the mission can be seen.

5, whereas in the second one it had depth 4. Figure 4 displays the grids composing the depth 5 PQ generated on the first day, as well as a top view of the area.

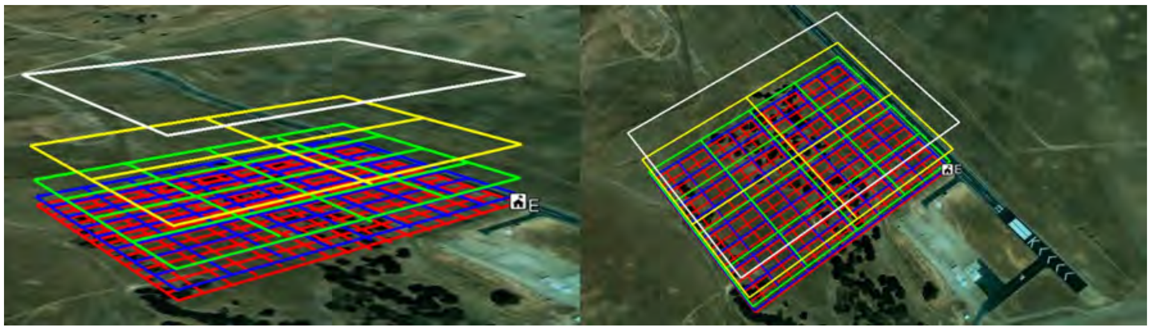


Figure 4: The left figure shows the five levels associated with a PQ of depth 5. The top level (white) is at 400 feet, the second (yellow) is at 200, the third (green) is at 100, the fourth (blue) is at 50 and the fifth (red) is at 25 feet. The grids are overlaid to the area at Camp Roberts where data collection took place. The white rectangle is 252×336 m^2, whereas red cells measure 15.75×21 m^2. The right figure features a top view of the area where data collection took place.

During the two days different objects were positioned in the dirt next to the McMillan air strip and served as targets being sought. In the first day the target was a car, whereas on the second day a mannequin and boxes of different sizes and color were placed in the same area. Figure 5 displays pictures of the car captured by the onboard camera while the AirRobot flew at the different elevations associated with the depth 5 PQ used in the first day. Figure 6 shows instead various pictures of the targets used during the second day.

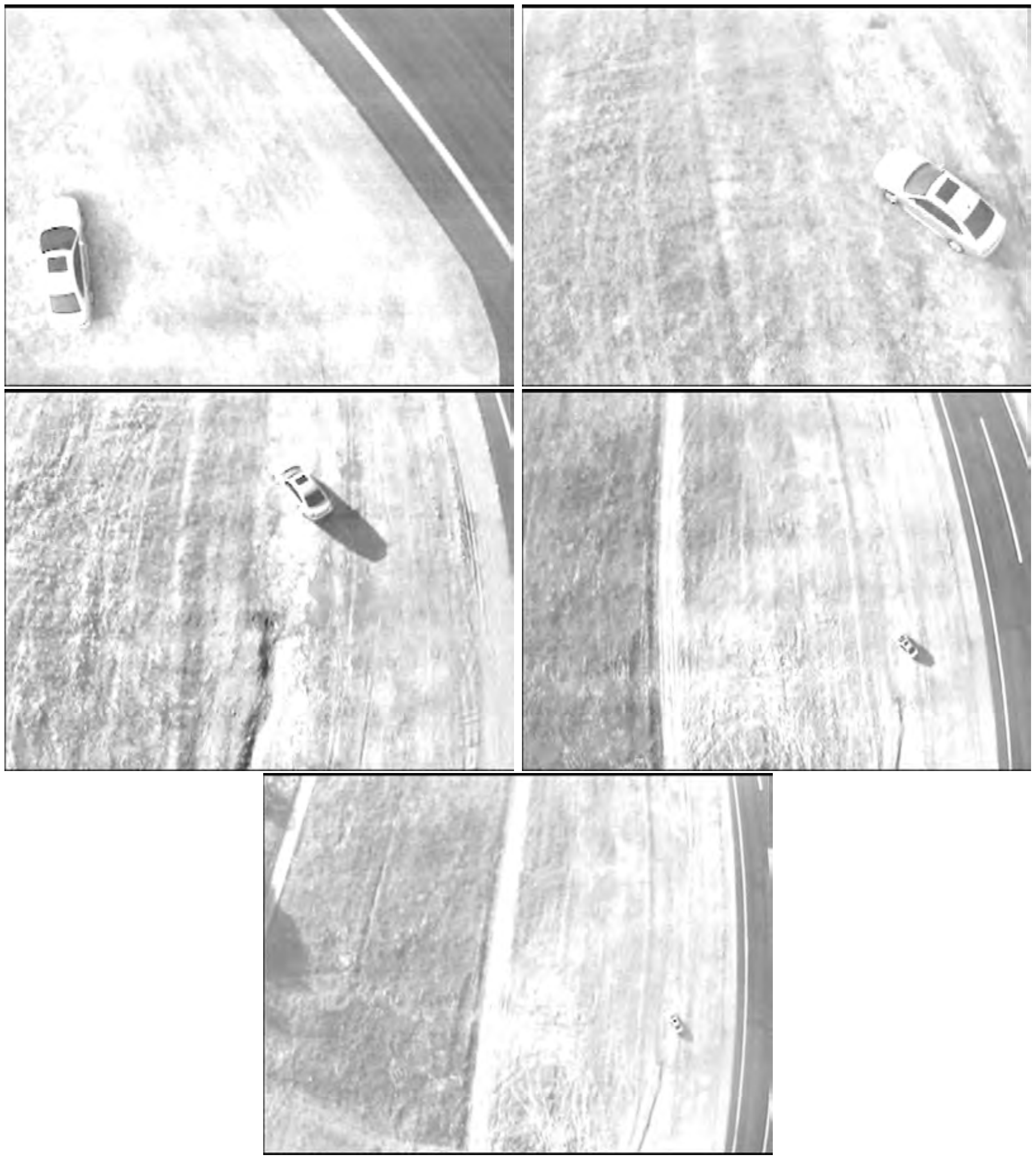


Figure 5: The figure shows five pictures of the car taken by the onboard AirRobot camera. Pictures were taken at the elevations associated with the depth 5 PQ used during the first day, i.e. 25ft, 50ft, 100ft, 200ft, and 400ft.

The reader shall notice that because of environmental conditions (bright sun, sustained wind) and limitations of the used camera images turned out to be overexposed and often blurred, thus posing additional challenges for the target detection algorithm described in the next subsection.

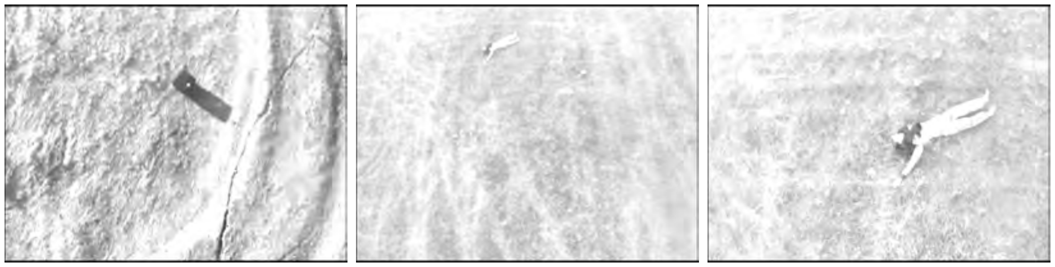


Figure 6: Top left: image of a box taken from the AirRobot flying at 25 ft, i.e., at the deepest level in the PQ. The center and right image show the mannequin from an elevation of 50 ft and 25ft.

Experiments presented in Section 5.4 rely on the first day's dataset. Images were filtered to retain only those taken at elevations consistent with the depth 5 PQ shown in Figure 4, and were later manually labeled to separate images showing the target car from those not showing it. This process produced 829 images showing the car from different elevations and 16264 not showing it. Data sets are available to the community upon request.

5.3 Target Detection

Target recognition was implemented using the off-the-shelf object detector provided with OpenCV (Bradski and Kaehler, 2008). The method is based on the Viola-Jones method and uses Haar features while being trained using both positive and negative examples. Training was performed using a different set of aerial images taken at Camp Robert by a different UAV. Training images feature a variety of backgrounds and image quality was in general much higher than what we experienced during our field experiments (Zaborowski, 2011). The object detector is capable of returning multiple detections in the same image and it also localizes the detected object within the image. However, to integrate it with our framework we treat its output as binary, i.e., we simply look at whether one or more targets were detected or not. Images captured by the AirRobot, in fact, suffered from motion blur and overexposure and returned in general a large number of false positives (in general, multiple per image). To mitigate this problem a set of depth-dependent thresholds were used to disambiguate positive from negative detections. Figure 7 shows how error rates α and β vary with the elevation. The reader will notice that the performance of the sensor is far from optimal, and is particularly bad at high elevations. However, as it will be evidenced in this section, the performance of the searcher is not too negatively influenced by the poor sensor, thus showing that this method can be used also when the sensor is not carefully tuned.

5.4 Search mission under realistic conditions

We have tested the proposed framework synthesizing numerous missions from the first day data set formerly described. We recall that the PQ has depth 5 and therefore its deepest level is associated with a 16×16 grid. During data collection the car was always parked at the same spot, and therefore to test the algorithm under a variety of conditions, missions are synthesized as follows. At the beginning of every mission the car is randomly assigned to one of the 256 cells at the deepest

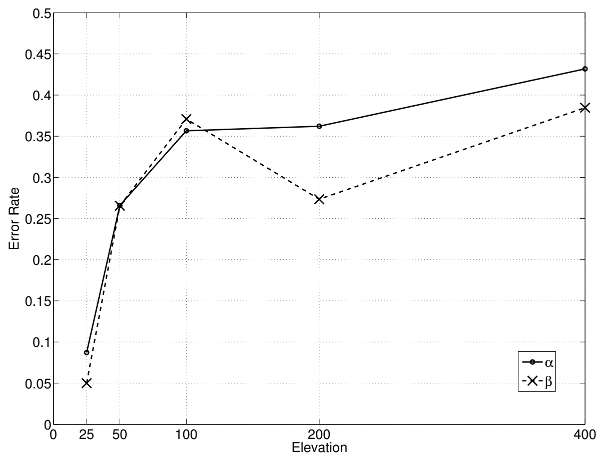


Figure 7: Error trends for false positives (α) and missed detections (β) as function of the elevation.

level. Based on this information we pre-compute, for every elevation, from which grid cells the car will be visible or not. Next, when the planner executes the mission, images are passed to the image detector after being randomly chosen from the appropriate set, i.e., the set of positives or negatives associated with the current AirRobot elevation. Random image selection exposes the classifier to positive or negative images taken from the appropriate elevation but varying in pitch, roll, blur, and captured area, thus more closely resembling the unavoidable time variance that would occur in a real world mission.

In our set of experiments the prior was chosen to be a symmetric Gaussian distribution centered in the middle of the searching area. To test the algorithm under different conditions the covariance of the Gaussian assumed different values (see Figure 8 and Table 1 for details).

The planner was therefore exposed to both cases where a narrow prior indicating strong confidence was given, as well as cases where a less certain a priori knowledge was presented. In all cases the algorithm started assuming that a target was present in the search area with probability 0.75. The search algorithm terminates when it reaches a confidence of 0.97 that the target is in a given cell, or outside the area. A time budget of 10000 time steps was also included, i.e., the searcher quits the mission if it has not arrived at a search decision within this limit (which mimics the limited endurance of the UAV). To put this number into perspective, in the simulation it is assumed that querying the sensor costs 5 time steps, and moving diagonally through the environment takes 362 time steps. For every prior we run 256 experiments, varying the position of the car through all cells associated with the deepest level of the PQ. The reader should therefore note that because of the given σ^2, the prior is fixed whereas the position of the car varies, we are testing the search algorithm using both correct and incorrect priors. Table 1 summarizes the results. In this experiment, the parameter γ in Eq. 4 is set to 0.6, as we experimentally determined that this value offers the best trade off (see later discussion).

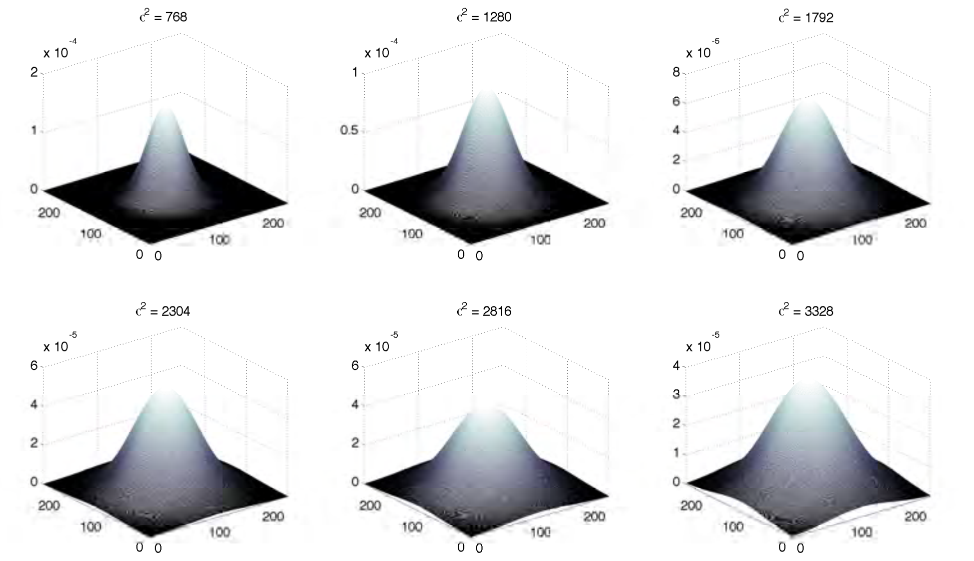


Figure 8: Different priors used in the experiments described in the following.

	$\sigma^2 = 768$	$\sigma^2 = 1280$	$\sigma^2 = 1792$	$\sigma^2 = 2304$	$\sigma^2 = 2816$	$\sigma^2 = 2816$
# detections	254	253	249	252	250	252
# false alarms	2	2	6	3	3	3
# timeouts	0	1	1	1	3	1
Avg. TTD	4159.57	3469.91	2943.67	2641.12	2484.49	2516.60
Std. dev. TTD	2224.87	1648.06	1628.32	1644.04	1517.31	1466.04

Table 1: Performance of the search algorithm with $\gamma = 0.6$ for priors with increasing variance (TTD: Time To Detection in time steps, as defined in Section 5.4).

Table 1 shows a rather consistent performance in terms of number of correct detections, with more than 97% of the search missions correctly terminating with a correct detection. Eq. 4 defines the behavior of the planner and critically depends on the choice of parameter γ. Values of γ larger than 0.5 yield a more myopic behavior with the searcher aggressively moving towards locations with high expected information gain, whereas lower γ values implement a more conservative approach aiming to minimize traveled distance. To assess the sensitivity to γ, we have repeated the former experiment with different γ values. Figure 9 displays how the average time to detection varies with σ^2 for different values of γ. Note that displayed times refer only to missions when the searcher terminates with a correct decision.

It can be seen that for each γ value, the trend of the average time to detection is the same, i.e., it decreases as σ^2 increases. This trend is justified by the fact that averages are taken over all possible locations of the target in the search area. Sharp priors induced by low σ^2 values assume strong

	$\sigma^2 = 768$	$\sigma^2 = 1280$	$\sigma^2 = 1792$	$\sigma^2 = 2304$	$\sigma^2 = 2816$	$\sigma^2 = 2816$
# detections	230	230	229	231	240	234
# false alarms	0	0	0	0	0	0
# timeouts	26	26	27	25	16	22
Avg. TTD	3836.22	3578.09	3445.53	3409.54	3995.27	3506.07
Std. dev. TTD	2546.70	2413.36	2315.37	2191.20	2383.65	2386.06

Table 2: Performance of the algorithm operating on the uniform grid using an on-line lawn mower search strategy (TTD: Time To Detection in time steps, as defined in Section 5.4).

	$\sigma^2 = 768$	$\sigma^2 = 1280$	$\sigma^2 = 1792$	$\sigma^2 = 2304$	$\sigma^2 = 2816$	$\sigma^2 = 2816$
# detections	120	143	162	177	190	207
# false alarms	0	0	0	0	0	0
# timeouts	126	113	94	79	66	49
Avg. TTD	3541.54	3854.91	4136.43	4242.72	4447.07	4634.27
Std. dev. TTD	2755.18	2892.32	2790.06	2768.21	2875.92	2937.76

Table 3: Performance of the algorithm operating on the uniform grid with $\gamma = 0.6$ for priors with increasing variance (TTD: Time To Detection in time steps, as defined in Section 5.4).

a priori confidence that the target is located in the central area, but in most tests this is not the case because the position of the target varies throughout the search area. On average, therefore, smoother priors offer better performance for this specific batch of experiments. Nevertheless, the observed trend is the same for every γ. Our former considerations about the impact of priors are corroborated by Figure 10.

Figure 10 shows how the average time to detection varies with σ^2 when the target is located in the central area. To be precise, we define the central area to be the 3×3 square centrally located in the search domain[4]. Averages are taken with respect to all γ values considered. This figure shows that when the location of the target is coherent with the given prior, sharper *a priori* information indeed gives an advantage. The last figure we include, Figure 11, provides the rationale for the choice of $\gamma = 0.6$.

Considering both Figure 9 and Figure 11, we see that $\gamma = 0.6$ offers the best trade off in terms of speed and accuracy. In fact, while Figure 9 shows that $\gamma = 0.4$ and $\gamma = 0.5$ exhibit better time performance when the target is located, Figure 11 evidences that these two values yield a higher percentage of missions ending with a timeout, i.e., the searcher does not formulate a decision within the allotted time budget. This behavior is consistent with the more conservative search pattern induced by lower γ values, as the searcher tends to be more stationary and therefore explores less. Figure 11(c) shows instead that the percentage of false alarms (i.e., those searches terminating with the searcher reporting the target in the wrong location) is consistently low for each γ values. Therefore, in conclusion we opt for $\gamma = 0.6$ as the preferred value. Of course, in a different operational scenario, this value could be different. The above considerations, however, give some indications about how to pick such a value.

[4]Recall that the planner operates on a 16×16 grid at the deepest level.

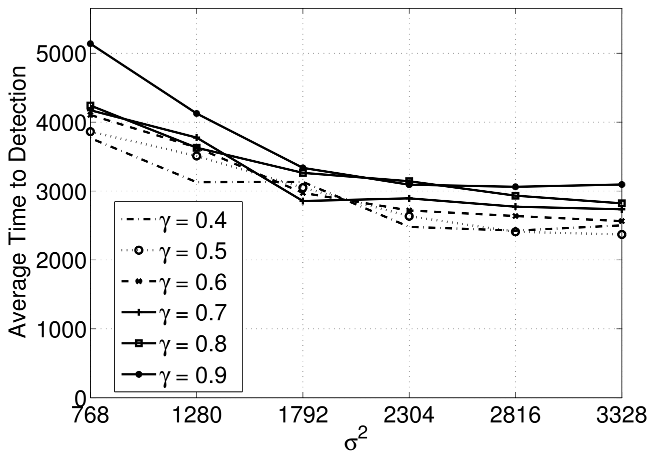


Figure 9: Average time to detection for different σ^2 values.

5.5 Comparison with other search techniques

We conclude this section with two additional sets of experiments aimed at highlighting the strength of the hierarchical method we propose. In particular, we want to show the inherent advantage of the hierarchical model and search algorithm when compared with strategies based on non-hierarchical representations. The comparison is then against two strategies operating on a uniform grid. The resolution of the uniform grid is the same as the resolution of the deepest level in the PQ. To ensure a fair comparison, the searcher is equipped with the same sensor and incurs the same travel costs. Note that in this case, the searcher always uses the sensor at its best resolution, because it operates at the deepest level. All other parameters were also tuned equally.

5.5.1 Lawn mower on a uniform grid

A classic search strategy is the lawn mower pattern, in which the searcher moves along a predetermined pattern on a regular grid. This strategy can be on-line or off-line. In an on-line version, the searcher decides whether it should move to the next grid cell or remain at the current one based on the last sensed value. A simple rule to implement this approach consists of moving forward if the sensor did not detect any target, or remain in the same cell and sense again if a target was detected. Because of the inherent sensing errors, it takes more than one positive detection to push the posterior of a grid cell above the critical threshold p_{tf}, and the on-line strategy is motivated by this observation. In an off-line version, on the contrary, the decision to move to the next grid cell or stay stationary is unrelated to the values returned by the sensor. For example, the searcher could scan each cell a fixed number of times before moving forward. In the experiments we performed, we used the on-line version we previously described. Table 2 summarizes the results we obtained and should be compared to Table 1.

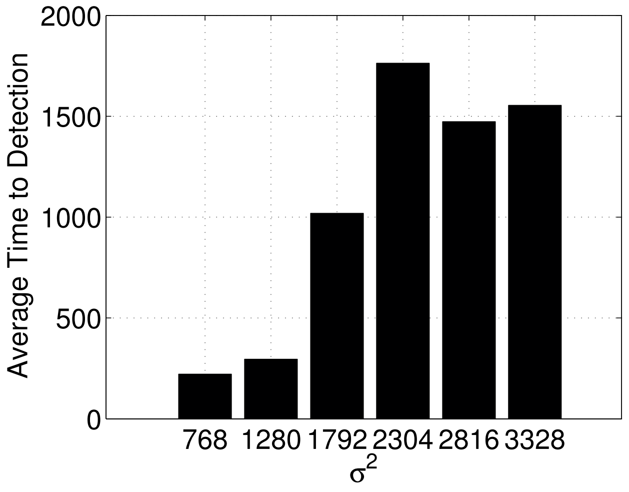


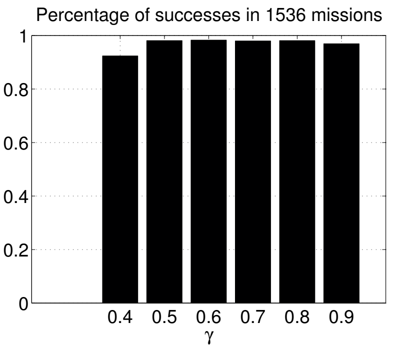
Figure 10: Average time to detection when the target is located in the central area of the search domain.

As expected, the performance of the searcher is basically independent from σ^2 because the sequence of actions taken by the searcher is not influenced by the prior. However, in this case the searcher suffers a much higher number of timeouts.

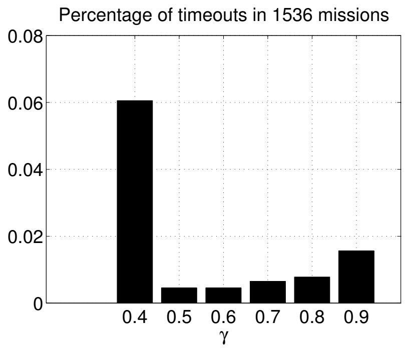
5.5.2 Information gain on a uniform grid

Finally, we implemented a strategy where the searcher decides where to sense next based on information gain (i.e., Eq. 4 and Eq. 5) but its search space is given by the uniform grid. Note that, like the method we presented in this paper, this strategy is spatially global but temporally greedy, thus providing a fair baseline for comparison. In this case we fix $\gamma = 0.6$ because of the conclusions we drew while evaluating the PQ strategy. Different values of γ do not give significantly different results. Table 3 shows the results we obtained when searching on the uniform grid and should be contrasted with Table 1.

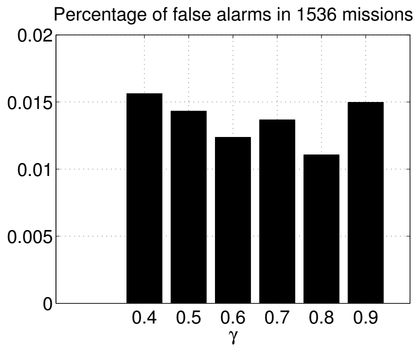
A quick inspection of the tables shows two facts. First, the strategy operating on the uniform grid incurs a much higher number of timeouts when the prior is sharply concentrated in one region (that is, due to smaller values of σ^2). On the contrary, the PQ algorithm is basically insensitive to this parameter and thus much more robust to misleading priors. The second conclusion is that even when a target is correctly located, the uniform algorithm spends significantly more time, as evidenced by comparing the Average Time To Detection row in the tables.



(a) Percentage of successes



(b) Percentage of timeouts



(c) Percentage of false alarms

Figure 11: Performance of the algorithms for different values of γ. Percentages are referred to a total of 1536 missions because for each of the 6 different σ^2 values 256 missions were run. The reader should note different scales are used in the three charts.

6 Discussion and Conclusions

We presented a novel framework for probabilistic search with a quadrotor aerial robot, leveraging its robust capabilities to dynamically improve the search for targets. By introducing the probabilistic quadtree data structure, we provided the analytic foundation which enables variable resolution search in a computationally efficient manner, including algorithms for Bayesian updates to the PQ from imperfect detections and for computing the most informative location and resolution to inspect next. Such automated computations are essential for providing real-time and mathematically grounded recommendations to operators and commanders in decision support contexts. Further, the proposed models were implemented and experimentally validated in realistic operational contexts at Camp Roberts, leveraging access to airspace and real field conditions such as changing terrain, variable weather, and joint flight operations. A major contribution of this work stemmed from experimental characterization of detection models for variable resolution applications. The integrated system presented in this paper, combining the capable aerial robot platform, autonomous search planning, and automated target detection using machine vision, addresses a current and future need for enhanced operations of tactical UAVs such as the quadrotor. Future directions include further experimental validations of both platform and detection methods, with potential deployment of the proposed variable resolution search algorithms using other aerial platforms and/or sensor payloads.

Appendix: the complexity of deciding where to sense next

We show how the node n_{next} defined in Eq. 5 can be computed in time $\mathcal{O}(k)$. Based on its definition, to compute n_{next} it is necessary to determine $I'(n)$ for every node in $\mathcal{T}$. Referring to Eq. 4, it is immediate to see that the second term involving $D(n^*, n)/\max_{n' \in \mathcal{T}} D(n^*, n')$ leads to an aggregate time complexity of $\mathcal{O}(k)$. For a given node n the term $D(n^*, n)$ can, in fact, be computed in constant time because n^* is fixed, whereas the denominator can be precomputed in $\mathcal{O}(k)$.

The first term involving $I(n)$ in Eq. 4 requires instead a more careful analysis. The expected information gain when sensing at node n is defined as

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

where $\mathbb{E}_{Z_n}$ is the expectation with respect to the value Z_n sensed when scanning node n, and $H(\mathcal{T})$ is the entropy[5] of $\mathcal{T}$ before incorporating the sensor reading Z_n. Because the probability values stored in all leaves are correlated, when computing $H(\mathcal{T}|Z_n)$ for a specific sensor reading Z_n one needs to compute not only how p_n will change, but also how the probabilities in all leaves will change. The computational approach described in Section 4.2 can be used for this computation and has time complexity $\mathcal{O}(k)$. Since this computation is needed for all nodes in $\mathcal{T}$, the overall time complexity to compute n_{next} would then be $\mathcal{O}(k^2)$. To improve this bound, we need to exploit the inherent properties of the PQ structure. We start by observing that a sensing operation at node n partitions the set $\mathcal{L}(\mathcal{T})$ in two sets, i.e., the leaves having n as ancestor, and the remaining ones. We indicate the former set as $\mathcal{I}_n(\mathcal{T})$ and the latter as $\mathcal{O}_n(\mathcal{T})$. $\mathcal{I}_n(\mathcal{T})$ is the set of leaves *seen* when sensing at node n (*inside* the field for view, hence the letter $\mathcal{I}$) whereas the $\mathcal{O}_n(\mathcal{T})$ are those *outside* the field of view. Based on this notation, assuming that sensing happens at time $t-1$ we can therefore rewrite the conditional entropy as follows:

$$H(\mathcal{T}|Z_n) = -\sum_{i \in \mathcal{I}_n(\mathcal{T})} p_i^t \log_2 p_i^t - \sum_{i \in \mathcal{O}_n(\mathcal{T})} p_i^t \log_2 p_i^t \qquad (6)$$

where the various p_i^t are the probability values at time t after incorporating the reading Z_n into the posterior at time $t-1$.

By exploiting the quadtree structure and by explicitly considering the distinction between inside and outside leaves, we can differently write the Bayes update rule. Given an observation Z_n at node n we define

$$\eta_n = \frac{\Pr[Z_n = z | X_n = 1]}{\Pr[Z_n = z]}$$

[5]We recall that $H(\mathcal{T})$ is defined as $H(\mathcal{T}) = \sum_{i \in \mathcal{L}(\mathcal{T})} -p_i \log_2 p_i$.

so that $p_n^t = \eta_n p_n^{t-1}$. Combining this definition with Eq. 2, it is easy to see that for every node $i \in \mathcal{I}_n(\mathcal{T})$ the same relationship holds, i.e., $p_i^t = \eta_n p_i^{t-1}$.

A similar reasoning can be applied to nodes in $\mathcal{O}_n(\mathcal{T})$. More precisely, we define

$$\zeta_n = \frac{\Pr[Z_n = z | X_n = 0]}{\Pr[Z_n = z]}$$

and then for every $i \in \mathcal{O}_n(\mathcal{T})$ we can write $p_i^t = \zeta_n p_i^{t-1}$.

Using these symbols, Eq. 6 can then be rewritten as

$$H(\mathcal{T}|Z_n) = -\sum_{i\in\mathcal{I}_n(\mathcal{T})} \eta_n p_i^{t-1} \log_2 \eta_n p_i^{t-1} - \sum_{i\in\mathcal{O}_n(\mathcal{T})} \zeta_n p_i^{t-1} \log_2 \zeta_n p_i^{t-1}.$$

Using the logarithm property $\log(ab) = \log a + \log b$ we have that:

$$\begin{aligned} H(\mathcal{T}|Z_n) = &-\sum_{i\in\mathcal{I}_n(\mathcal{T})} \left[\eta_n p_i^{t-1} \log_2 \eta_n + \eta_n p_i^{t-1} \log_2 p_i^{t-1}\right] \\ &-\sum_{i\in\mathcal{O}_n(\mathcal{T})} \left[\zeta_n p_i^{t-1} \log_2 \zeta_n + \zeta_n p_i^{t-1} \log_2 p_i^{t-1}\right] \end{aligned}$$

and by simple algebraic manipulation we have:

$$\begin{aligned} H(\mathcal{T}|Z_n) = &-\eta_n \log_2 \eta_n \sum_{i\in\mathcal{I}_n(\mathcal{T})} p_i^{t-1} - \eta_n \sum_{i\in\mathcal{I}_n(\mathcal{T})} p_i^{t-1} \log_2 p_i^{t-1} \\ &-\zeta_n \log_2 \zeta_n \sum_{i\in\mathcal{O}_n(\mathcal{T})} p_i^{t-1} - \zeta_n \sum_{i\in\mathcal{O}_n(\mathcal{T})} p_i^{t-1} \log_2 p_i^{t-1}. \end{aligned}$$

Because of the PQ constraint defined in Eq. 1, we have that $\sum_{i\in\mathcal{I}_n(\mathcal{T})} p_i^{t-1} = p_n^{t-1}$ and $\sum_{i\in\mathcal{O}_n(\mathcal{T})} p_i^{t-1} = 1 - p_n^{t-1}$. Furthermore, for leaf nodes in the tree we define $h_i^{t-1} = p_i^{t-1} \log_2 p_i^{t-1}$ and we therefore obtain the following expression for $H(\mathcal{T}|Z_n)$

$$H(\mathcal{T}|Z_n) = -\eta_n \log_2 \eta_n p_n^{t-1} - \eta_n \sum_{i\in\mathcal{I}_n(\mathcal{T})} h_i^{t-1} - \zeta_n \log_2 \zeta_n (1 - p_n^{t-1}) - \zeta_n \sum_{i\in\mathcal{O}_n(\mathcal{T})} h_i^{t-1}.$$

To remove the remaining two sums we have to extend the definition of h for internal nodes too. For an internal node n we define h_n^{t-1} as the sum of the h values at time $t-1$ of all the leaves having

n as ancestor. A simple bottom up algorithm sweeping $\mathcal{T}$ from the leaves to the root can be used to compute h_n^{t-1} for all nodes in the $\mathcal{T}$ in time $\mathcal{O}(k)$. With this definition of h, we get the final expression for $H(\mathcal{T}|Z_n)$ where we use the fact that $\sum_{i\in\mathcal{O}_n(\mathcal{T})} h_i^{t-1} = -H(\mathcal{T}) - h_n^{t-1}$

$$H(\mathcal{T}|Z_n) = -\eta_n \left[\log_2 \eta_n p_n^{t-1} + h_n^{t-1}\right] - \zeta_n \left[\log_2 \zeta_n(1-p_n^{t-1}) + (-H(\mathcal{T}) - h_n^{t-1})\right].$$

In this final expression, all quantities are known upfront (probabilities at time $t-1$), can be computed in constant time (η_n and ζ_n), or can be found in a lookup table (h_n^{t-1}) that is computed once for all nodes in time $\mathcal{O}(k)$. Therefore, n_{next} can be computed in $\mathcal{O}(k)$.

Acknowledgments

Nicola Basilico was with the University of California, Merced while doing this work. Stefano Carpin was partially supported by NPS-ONR under award # N00244-11-1-0023. Nicola Basilico and Stefano Carpin were partially supported by DARPA under grant #HR0011-09-C-0002. Any opinions, findings, and conclusions or recommendations expressed in these materials are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the funding agencies of the U.S. Government. This paper extends preliminary results presented at ICRA 2011 (Chung and Carpin, 2011) and IROS 2011 (Carpin et al., 2011). The authors thank Kevin Jones for having shared aerial images needed to obtain Figure 7.

References

Benkoski, S. J., Monticino, M. G., and Weisinger, J. R. (1991). A Survey of the Search Theory Literature. *Naval Research Logistics*, 38(4):469–494.

Bonnie, D., Candido, S., Bretl, T., and Hutchinson, S. (2012, May). Modelling search with a binary sensor utilizing self-conjugacy of the exponential family. In *Proceedings of the IEEE International Conference on Robotics and Automation*, St.Paul, MN.

Bourgault, F., Furukawa, T., and Durrant-Whyte, H. F. (2006). Optimal Search for a Lost Target in a Bayesian World. *Field and Service Robotics (STAR Springer Tracts in Advanced Robotics)*, 24:209–222.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer Vision with the OpenCV library.* O'Really.

Carpin, S., Burch, D. A., and Chung, T. H. (2011, September). Searching for Multiple Targets Using Probabilistic Quadtrees. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA.

Chung, T., Kress, M., and Royset, J. (2009, May). Probabilistic search optimization and mission assignment for heterogeneous autonomous agents. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan.

Chung, T. H. (2010). On Probabilistic Search Decisions Under Searcher Motion Constraints. In G.S. Chirikjian et al., editor, *Workshop on Algorithmic Foundations of Robotics*, volume 57 of *STAR*, pages 501–516. Springer.

Chung, T. H. and Carpin, S. (2011, May). Multiscale search using probabilistic quadtrees. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Shanghai, China.

Chung, T. H., Hollinger, G. A., and Isler, V. (2011). Search and Pursuit-Evasion in Mobile Robotics: A Survey. *Autonomous Robots*, 31(4):299—-316.

de Berg, M., Schwarzkopf, O., van Kreveld, M., and Overmars, M. (2000). *Computational Geometry*. Springer.

Eagle, J. N. (1984). The Optimal Search for a Moving Target When the Search Path is Constrained. *Operations Research*, 32(5):1107–1115.

Foraker, J. (2011). *Optimal Search for Moving Targets in Continuous Time and Space using Consistent Approximations*. Phd tesis, Naval Postgraduate School, Monterey, CA.

Goodrich, M., Morse, B., Gerhardt, D., Cooper, J., Quinley, M., Adams, J., and Humphrey, C. (2008). Supporting wilderness search and rescue using a camera-equipped mini UAV. *Journal of Field Robotics*, 25(1):89–110.

Hubenco, A., Fonoberov, V., Mattehew, G., and Mezić, I. (2011). Multiscale adatptive search. *IEEE Trasactions on Systems, Man, and Cybernetics – part B: Cybernetics*, 41(4):1076–1087.

Huynh, V. A., Enright, J., and Frazzoli, E. (2010, December). Persistent patrol with limited-range on-board sensors. In *Proceedings of the IEEE Conference on Decision and Control*, Atlanta, GA.

Jones, K. D., Dobrokhodov, V., Kaminer, I., Chung, T. H., Clement, M. R., Kolsch, M., and Zaborowski, R. (2011, August). Cooperative Autonomy for the Masses - Fundamental Steps Toward Enabling Complex Multi-Asset Missions with Simple Point-and-Click Tasking. In *Proceedings of AUVSI Unmanned Systems North America 2011*, Washington, D.C.

Kendoul, F. (2012). Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378.

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

Lavis, B., Furukawa, T., and Durrant Whyte, H. F. (2008). Dynamic Space Reconfiguration for Bayesian Search and Tracking with Moving Targets. *Autonomous Robots*, 24(4):387–399.

Lin, L. and Goodrich, M. (2009, October). UAV intelligent path planning for wilderness search and rescue. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, St.Louis, MO.

National SAR Committee (2000). United States National Search and Rescue Supplement. Technical report, National Search And Rescue Committee, Washington DC.

Pajarola, R. and Gobbetti, E. (2007). Survey of semi-regular multiresolution models for interactive terrain rendering. *The Visual Computer*, 23(8):583–605.

Pratt, K., Murphy, R., Stover, S., and Griffin, C. (2009). CONOPS and autonomy recommendations for VTOL small unmanned aerial system based on hurricane Katrina operations. *Journal of Field Robotics*, 26(8):636–650.

Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., Grixa, I., Ruess, F., Suppa, M., and Burschka, D. (2012). Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue. *Robotics and Automation Magazine*, 19(3):46–56.

Schwager, M., Julian, B. J., Angermann, M., and Rus, D. (2011). Eyes in the Sky: Decentralized Control for the Deployment of Robotic Camera Networks. *Proceedings of the IEEE*, 99(9):1–21.

Sisso, I., Shima, T., and Ben-Haim, Y. (2010). Info-gap approach to multiagent search under severe uncertainty. *IEEE Transaction on Robotics*, 26(6):1032–1041.

Stachniss, C. (2009). *Robotic Mapping and Exploration*, volume 55 of *"STAR Springer tracts in advanced robotics"*. Springer.

Stone, L. D. (1989). *Theory of Optimal Search*. Academic Press, 2nd edition.

Tisdale, J., Kim, Z., and Hedrick, J. (2009). Autonomous UAV path planning and estimation. *Robotics & Automation Magazine, IEEE*, 16(2):35–42.

Waharte, S., Symington, A., and Trigoni, N. (2010, May). Probabilistic search with agile UAVs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, AK.

Zaborowski, R. M. (2011). *Onboard and Parts-based Object Detection from Aerial Imagery*. Master's, Naval Postgraduate School, Monterey, CA .