Hierarchical Decision-theoretic Robotic Surveillance

Nikos Massios and Frans Voorbraak
Institute for Logic, Language and Computation (ILLC),
University of Amsterdam, Plantage Muidergracht 24,
1018 TV Amsterdam, The Netherlands
{massios,fransv}@wins.uva.nl

Abstract

In this paper, we discuss a decision-theoretic strategy for surveillance as a first step towards automating the planning of the movement of an autonomous surveillance robot. We extend a previous proposal by including some heuristics based on an abstract representation of the environment. We show, using a concrete example, how these heuristics allow computationally feasible, finite look-ahead versions of the decision-theoretic strategy to escape local minima, and to better approximate globally optimal solutions.

1 Introduction

In several projects, robots are employed to perform surveillance tasks. See, for example, [2, 9]. However, in most of these projects, the robots are typically used as a kind of flexible sensor platform controlled by some external human operator who makes the high-level decisions on where to go, and how to use the on-board sensors.

We are interested in automating this high-level decision making process to allow an autonomous mobile robot plan itself how to perform the surveillance task. Little work has been done in this area, with the the notable exception of [5, 6]. We believe this autonomous robotic surveillance application is interesting in its own right, but it also provides a test-case for formalisms aimed at dealing both with (low-level) sensor, localisation, and navigation uncertainty and with uncertainty at a more abstract planning level.

By surveillance we understand keeping a close watch over something or someone, with the purpose of detecting the occurrence of some relevant events. Which events are considered relevant depends on the overall reason for the surveillance. A concrete example of what we have in mind is a surveillance robot in an office-like environment, which is equipped with sensors to detect hazards like fires, floodings, and intruders. (This is similar to the system described in [2].)

Although the issue of automatic recognition of such hazards is extremely important, we concentrate our research on the problem of getting the available sensors in such a position that the relevant events are most likely detected. We cannot simply require the detection of *all* relevant events, since in many realistic situations only a part of the environment can be in the sensor range.

Our main concern is to develop strategies or algorithms that make use of the available resources (sensors, manoeuvring capabilities, et cetera) such that the probability, or better, the expected cost, of relevant events remaining undetected is minimised. The fact that detecting relevant events does not guarantee them to be recognised as relevant, can (and should) be modeled by introducing some uncertainty in the classification or recognition of events. Similarly, there is uncertainty introduced by not completely reliable robot navigation and self-localisation.

Using a decision-theoretic criterion, like minimising expected cost, has a number of advantages. For example, one can take into account the fact that some relevant events can have more serious consequences than others. (A fire near a storage room of highly inflammable toxic material is more serious than a flooding of a rarely used basement with no electric outlets.) Also, at least from a theoretical point of view, it seems obvious how to further refine this criterion to incorporate other types of cost, such as the danger of damage to the robot, or the cost of false alarms (in case the recognition of relevant events is not completely reliable). However, several serious problems remain.

One problem is that the expected cost of the whole surveillance plan or decision policy should be minimised. In general, such a policy cannot be reduced to a sequence of simple decision problems, but involves looking (far) ahead at (many) possible outcomes of (many) future actions. Therefore, in realistic applications one will probably need heuristics and approximation methods to arrive at feasible solutions.

In this paper, we discuss an hierarchical version of the minimum expected cost strategy proposed in [8, 10]. Descriptions of the environment at different abstraction levels are used simultaneously, where decisions made at more abstract levels guide the decision making at less abstract levels. Similar applications of abstract representations to obtain computational savings for computationally expensive methods have been proposed before. See, for example, [3] for such a proposal in the context of Markov Decision Problems.

The rest of this paper is organised as follows. In section 2 we describe a simple formal model of an environment in which the surveillance task has to be performed. In section 3 we discuss the minimum expected cost strategy proposed in [8, 10], and briefly mention some other possible strategies. In section 4 we discuss applying the minimum expected cost strategy on a more realistic environment. We introduce a hierarchical minimum expected cost strategy to deal with barrier problems. In section 5 we show how information obtained by unreliable sensors can be used in our expected cost strategies. In sections 6, 7 we summarise our plans for future work and our conclusions.

2 A Simple Environment Model

In this section we introduce a very simple, formal model of the environment for the surveillance task.

Definition 2.1 *An environment E is a tuple $\langle X, A_0, A, C, P_0, P\rangle$, where X is a set $\{X_i : 1 \leq i \leq n\}$ of mutually disjoint spatial areas, or locations, $A_0 \in X$ is the start location of the robot, $A \subseteq X \times X$ represents the relation of immediate accessibility (for the robot) between locations, C is a function assigning to each location $X_i \in X$ the cost $C(X_i)$ associated with not detecting a relevant event occurring at X_i, P_0 is a function assigning to each location $X_i \in X$ the probability $P_0(X_i = 1)$ that at time 0 an event occurs at X_i, and P is a set of transition probabilities. That is, for every $X_i \in X$ and time $t \geq 0$, P contains $P_t(X_i \to 1)$ denoting the (prior) probability that at time t a relevant event starts at X_i, and $P_t(X_i \to 0)$ denoting the (prior) probability that the relevant event at X_i stops at time t.*

For the rest of this paper, we assume that for every $X_i \in X$, $P_0(X_i = 1) = 0$, and $\langle A_0, X_i\rangle \in A$. As a consequence, we can simplify the description of an environment $\langle X, A_0, A, C, P_0, P\rangle$ to $\langle X, A, C, P\rangle$. Further, we assume that $P_t(X_i \to 1)$ and $P_t(X_i \to 0)$ do not depend on the time t, and we drop the subscript t. We also assume that the environment is *connected*, in the sense that for every $X_i, X_j \in X$ there is a path $X_i = Y_1, \ldots, Y_m = X_j$ such that $\langle Y_k, Y_{k+1}\rangle \in A$.

We write r_t to denote the sensor range of the robot at time t, and we assume that $r_0 = A_0$ and for every $t \geq 1$, $r_t \in \{X_i : \langle r_{t-1}, X_i\rangle \in A\}$. If $r_t = X_i$, then we say that X_i is visited at time t. The decision strategies should decide which immediately accessible location to visit next. For the moment, we do not take recognition uncertainty into account and assume a relevant event to be detected whenever it is in the sensor range of the robot.

The above definition provides a very abstract model of the decision problem. For realistic applications, we have to take into account that a robot can have several sensors, each with its own sensor range, that the sensor range is not necessarily an element of X, that the actions of the robot may include changing its location, its orientation, and possibly manipulating aspects of the environment, such as opening a door, and that the exact state and dynamics of the environment, the exact position of the robot in the environment, and the recognition of relevant events are uncertain, et cetera.

In spite of the many simplifying assumptions, the model is sufficiently general to capture the abstract environment used in [6] to experimentally compare different surveillance strategies. In that experimental set-up, all locations are assumed to be immediately accessible from each other, i.e., $A = X \times X$. The dynamics at each location is modeled by a Markov process, and $P(X_i \to 1)$ and $P(X_i \to 0)$ are viewed as transition probabilities between possible states of a location.

In our model, the state of a location also depends on the actions of the agent, as is appropriate for many typical surveillance applications. For example, although the start of a fire can be assumed to have an extraneous cause (nature or some agent other than the surveillance agent), it is typically the case that the detection of a fire influences the probability of the fire being extinguished. The detection of a relevant event should trigger some appropriate response of the surveillance agent, such as raising an alarm, or taking more direct countermeasures against the event. Since the particular actions triggered by a detection are themselves, strictly speaking, not part of the surveillance behaviour of the agent, we will leave them out of our considerations.

We will assume that relevant events do not stop spontaneously, but that they stop immediately when detected by the surveillance agent. Formally, $P(X_i \to 0) = 0$, and $P_{t+1}(X_i = 1) = P_{t+1}(X_i \to 1)$, if $r_t = X_i$. It is possible to introduce some time delay for the countermeasures to take effect, but this raises the problem of deciding how important it is to monitor areas where relevant events are known or have been observed to occur. It is also possible to allow $P(X_i \to 0) > 0$ and to model the effect of the actions triggered by observing a relevant event as an increase in $P(X_i \to 0)$. Our assumption can be viewed as an extreme instance of this possibility.

Given our assumptions, it is possible to express $P_t(X_i = 1)$ in terms of $P(X_i \to 1)$ and the amount of time that has passed since the last visit to X_i.

Proposition 2.1 *Let* $E = \langle X, A_0, A, C, P_0, P \rangle$ *be an environment where* $P(X_i \to 0) = 0$*, and* $X_i = r_t$ *implies that* $P_{t+1}(X_i = 1) = P_{t+1}(X_i \to 1)$*. Then* $P_t(X_i = 1) = 1 - (1 - P(X_i \to 1))^{t-t'}$*, where* t' *is the largest time point* $\leq t$ *such that* $r_{t'} = X_i$*.*

Our model currently contains a number of restrictions, possibly the most severe of which is the lack of sensing and robot localisation uncertainty. One might claim that POMDPs [1] can deal with these types of uncertainty and so POMDPs are better suited for problems where robot navigation is involved. However we have introduced these restrictions in our model because we want to keep a clear view of the problem of abstracting environment representations. It is possible to add sensor uncertainty in our model (see section 5). Furthermore, the problem of abstracting environment representations is also important for POMDPs and in fact there has been some work on abstractions for MDPs in [3, 7].

3 Minimum Expected Cost Strategy

In [5, 6] a surveillance strategy is proposed based on a notion of confidence, which can be viewed as a second-order uncertainty measure. In a part of [6], the notion of confidence is claimed to be related to possibility. Confidence works as follows: whenever sensory information about the state of a location becomes available, the probability of an event occurring at that location at that time is updated, and one is assumed to be very confident about this assessment of the state of the location. Confidence then drops gradually over time during a period in which no fresh sensory information concerning this particular location is obtained.

The policy proposed in [5, 6] is to maximise confidence, i.e., to choose the action that changes the sensor range to the (neighbouring) location which has the lowest degree of confidence attached to it. One problem with this proposal is that the notion of confidence has to be clarified. Also, it is not immediately clear how to incorporate the notion of cost, and the possibility to express the relative importance of events, in the proposal.

If the transition probabilities $P(X_i \to 1)$ are the same for each X_i, then the maximum confidence strategy can be shown to be equivalent to the following.

minimax interval Minimise the maximum time interval between visits of locations by choosing the action that changes the sensor range to the neighbouring location which has not been visited for the longest time.

We propose to use this minimax interval policy as a kind of reference strategy. Since this strategy does not use information about the uncertainties, it can be used to clarify how much other strategies which do use uncertainty information gain in efficiency.

The maximising confidence strategy does use uncertainty information, but the costs are not taken into consideration. An obvious (decision-theoretic) surveillance strategy, which considers both uncertainty and cost, is based on the following decision policy.

minimum expected cost Choose the action that minimises the expected cost.

This decision policy can be interpreted both globally and locally. Under the global interpretation, the action that has to be chosen corresponds to the behaviour of the surveillance agent from the start to the end of the surveillance task. There is not an inherent end to a surveillance task, but in practice each particular task has a limited duration (say, until the next morning when the employees return to the office building, or until the batteries of the robot have to be recharged).

The (global) expected cost EC_T until time T can be computed by the following formula.

$$EC_T = \sum_{1 \leq t \leq T, X_i \neq r_t} P_t(X_i = 1)C(X_i).$$

Notice that a choice to visit X_i at t not only removes the term $P_t(X_i = 1)C(X_i)$ from the above sum, but it also has some indirect benefits, due to the fact that it reduces $P_{t'}(X_i = 1)$ for $t' > t$.

The behaviour of the surveillance agent from the start to the end of the surveillance task can also be viewed as consisting of a sequence of simpler actions. One can apply the above decision policy locally to choose at each time between the possible simple actions by comparing the consequences of these simple actions, or perhaps by comparing the (expected) consequences of small sequences of simple actions. Let us say that an n-step policy compares the (expected) consequences of sequences of n (simple) actions. In that case, the expected cost of a sequence will not be evaluated over all the environment locations (as the global expected cost is evaluated) but only for the environment locations in the action sequence. Of course, the policy is more easily implemented for small n, whereas, in general, it better approximates the global policy for large n.

The following proposition shows that we can expect a difference between the mentioned policies only in the case of varying probabilities (or costs).

Proposition 3.1 *Let $E = \langle X, A_0, A, C, P_0, P\rangle$. Assume that for all $X_i, X_j \in X$, $P(X_i \to 1) = P(X_j \to 1)$ and $C(X_i) = C(X_j)$. Then the maximum confidence policy and the 1-step minimum expected cost policy both reduce to the minimax interval policy.*

In [8, 10] we compare minimum expected cost with maximum confidence and with some other strategies proposed by Fabiani [6] in greater detail. However in this article we want to concentrate on hierarchical expected cost strategies. Before introducing hierarchical expected cost we want to give some remarks and examples that we hope will clarify our notion of a minimum expected cost strategy.

Example 3.1 *Consider an environment $E = \langle X, A_0, A, C, P_0, P\rangle$, where X is a set $\{X_1, X_2\}$ consisting of two rooms, $A = X \times X$, $C(X_1) = C(X_2) = 1$, $P(X_1 \to 1) = 0.5$ and $P(X_2 \to 1) = 0.8$. The minimax interval strategy goes back and forth between both rooms, just as the maximum confidence policy does.*

The strategy based on 1-step minimising expected cost is slightly more complicated. At time 1, room X_2 is chosen because $P(X_2 = 1) = P(X_2 \to 1) = 0.8 > 0.5 = P(X_1 \to 1) = P(X_1 = 1)$. At time 2, $P(X_2 = 1)$ is again 0.8, since room X_2 was visited at the immediately preceding time step. However, $P(X_1 = 1)$ has only increased to 0.75, which is not enough to get chosen. Only at time 3, $P(X_1 = 1) = 0.875$ has increased above the 0.8 probability that a relevant event occurs in room 2. We thus obtain a sequence where room X_1 is only chosen every third time step. See table 1, where for the first six time steps the expected costs (of not visiting a room) are displayed. The 1-step minimum expected cost policy chooses the room with the (maximal) expected cost printed in boldface.

In this example, the 1-step minimum expected cost policy results in a behaviour which seems intuitively appealing, since it clearly reflects the fact that $P(X_1 \to 1)$ is substantially lower than $P(X_2 \to 1)$, whereas minimax interval, just as maximum confidence, treats both rooms the same. However, it can be shown that in this case the 1-step minimum expected cost policy does not minimise the global expected cost, and already the 2-step minimum expected cost policy results in the same back and forth behaviour as prescribed by the minimax interval policy.

It can be shown that if the cost of not detecting a relevant event is constant over the different areas X_i, then, in the long run, the 1-step minimum expected cost policy will result in an exhaustive exploration of the environment. More precisely, one can show the following.

time	X_1	X_2
1	0.5	**0.8**
2	0.75	**0.8**
3	**0.875**	0.8
4	0.5	**0.96**
5	0.75	**0.8**
6	**0.875**	0.8

Table 1: The expected costs of example 3.1.

time	X_1	X_2
1	0.5	**2.4**
2	0.75	**2.4**
3	0.875	**2.4**
4	0.9375	**2.4**

Table 2: The expected costs of example 3.2.

Proposition 3.2 *Let $E = \langle X, A_0, A, C, P_0, P \rangle$. Assume that for all $X_i, X_j \in X$, $C(X_i) = C(X_j)$. Then, in the long run, every $X_i \in X$ is visited when applying the 1-step minimum expected cost policy, and there is a finite upper bound N_i on the length of the time interval between visits of X_i.*

If relevant events will not stop spontaneously before they are detected, exhaustive exploration implies that all relevant events will eventually be detected. In [6], exhaustive exploration is mentioned as a desirable property of a surveillance strategy, and it is mentioned that the maximum confidence strategy achieves exhaustive exploration.

The following simple modification of example 3.1 shows that, in general, proposition 3.2 is no longer valid (and the 1-step minimum expected cost policy is no longer guaranteed to result in an exhaustive exploration of the environment) if the costs are allowed to vary.

Example 3.2 *Consider the situation of example 3.1, but now assume $C(X_1) = 1$ and $C(X_2) = 3$. Then the expected cost of not visiting room 1 has an upper bound of 1, whereas the expected of not visiting room 2 is 2.4 even when it has been visited the previous moment. Therefore, by the 1-step minimum expected cost policy, room 2 will always be chosen. See table 2, where for the first four time steps the expected costs (of not visiting a room) are displayed. The 1-step minimum expected cost policy chooses the room with the (maximal) expected cost printed in boldface.*

This effect of ignoring room 1 can be avoided by allowing the cost of not detecting an event to grow as a function of the time passed since the event has started. However, if the mentioned (expected) costs are correct, then this ignoring of room 1 may be defensible. Intuitively, one should not require a surveillance agent to explore irrelevant or unimportant areas of the environment. Thus, achieving exhaustive exploration is not always a desirable property of a surveillance strategy. This is substantiated by the following.

Proposition 3.3 *In the environment of example 3.2 the 1-step minimum expected cost policy minimises the global expected cost.*

Perhaps a more important problem than possibly preventing an exhaustive exploration of the environment is that the varying cost can form an obstacle to obtaining optimal behaviour using the local (n-step) minimum expected cost policy.

Example 3.3 *Consider an environment* $E = \langle X, A_0, A, C, P_0, P \rangle$*, where* X *is a set* $\{X_0, X_1, X_2\}$ *consisting of three rooms,* $A = \{\langle X_0, X_1 \rangle, \langle X_1, X_0 \rangle, \langle X_1, X_2 \rangle, \langle X_2, X_1 \rangle\}$*,* $C(X_0) = 10$*,* $C(X_1) = 1$*,* $C(X_2) = 3$*,* $P(X_0 \to 1) = 0.1$*,* $P(X_1 \to 1) = 0.5$ *and* $P(X_2 \to 1) = 0.8$*.*

As in example 3.2, the 1-step minimum expected cost policy will always choose room 2. But now the (possibly justifiable) ignoring of room 1 will make it impossible to visit room 0, and in the long run the expected cost of not visiting room 0 will be very high.

Obviously, such problems can be solved theoretically by looking more than one step ahead. But looking ahead many steps is computationally expensive. See [8], where some experiments on the required computation time are summarised. Below, we describe an example of an environment for surveillance which is more realistic than our previous examples, and we propose some heuristics based on an abstract representation of this environment to escape local minima, and to better approximate globally optimal solutions with limited look-ahead.

4 A More Realistic Environment

Our more realistic example is robot surveillance for detecting fires in a simulated real office building of our university (fig. 1:a). The simulation assumes that an *a priori* map of the building is available. This map is an accurate labeled occupancy grid that is composed out of walls, doors, and free space. The relevant events in our example are fires and the locations X_i are rooms and segments of the corridor.

4.1 Environment graph construction

Before applying the decision strategies on the office environment each room is identified and marked. The corridor (room 2) is segmented into many smaller rooms. The places where to segment the corridor are determined by the position

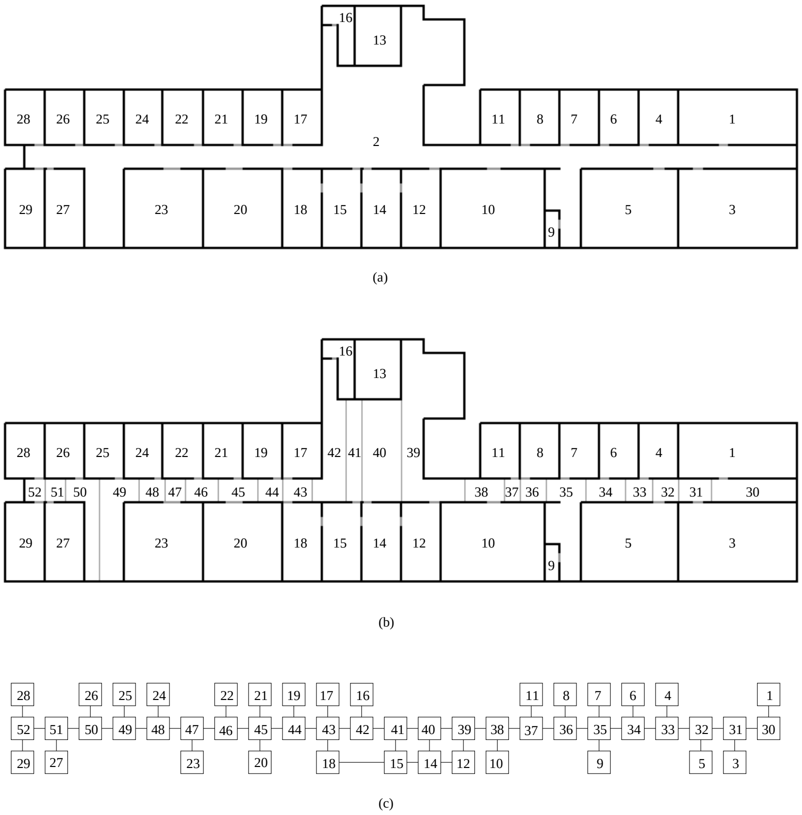


Figure 1: (a) Initial map (b) Segmented map (c) Environment graph.

of the doors connecting other rooms to the corridor. By segmenting the corridor a mapping of all decision points to individual locations is effectively achieved (fig. 1:b). Finally the marked room environment representation is converted into a graph where rooms correspond to graph nodes and doors to graph links (fig. 1:c). This representation is called the *environment graph*.

The n-step minimum expected cost strategy works using the environment graph. Together with the environment graph a wide range of routines is provided that allows the mapping between environment graph nodes and individual room locations and vice versa. Furthermore virtual sensors are supplied that can detect if a fire is present within a certain graph node. Initial fire locations, probabilities and costs are specified at the level of environment graph nodes.

The time needed to take a single decision using an n-step strategy is exponential with $O(b^n)$ time complexity. Where b is the branch factor of the decision tree being explored and n is number of steps to look ahead. We have determined experimentally that the largest n that can be used by our implementation, for real-time behaviour, is around 5 for this type of environment [8].

4.2 The cost barrier problem

It is obvious that under certain circumstances n-step strategies can get into trouble in our environment.

Example 4.1 *Consider the case where* $C(X_i) = 0$ *for* $i = \{6, 7, 8, 9, 10, 11, 34, 35, 36, 37, 38\}$, $C(X_j) = 10$ *for* $j = \{1, 3, 4, 5, 30, 31, 32, 33\}$, $C(X_k) = 1$ *for all* $k \neq i \wedge k \neq j$, $P(X_l) = 0.001$ *for all* l *and* $A_0 = 52$. *In this situation an expected cost barrier is formed by the* X_i*s that a* n*-step strategy with* $n <= 5$ *cannot cross. All the barrier locations* X_i *have an expected cost of 0. A robot starting at 52 can get up to room 39 but then its immediate neighbours will all have expected cost 0 and a visit there will not be justified. The rooms* X_j *will also not be visited because they can only be considered once the robot is in one of the* X_i*s. Therefore the rooms* X_j *will never will be visited although after a sufficiently large period they will almost certainly be on fire and the cost of fire there is relatively high. This is an example of an expected cost barrier.*

We have plotted the number of times each room is visited by the 5-step minimum expected cost strategy (fig. 4:a). Since this strategy cannot overcome the expected cost barrier, the groups of nodes a, b, c and d in the plot are not visited. So neither the nodes beyond the barrier nor the nodes in the barrier are visited.

It is possible to give an argument saying that a 6-step strategy can overcome this barrier but then a larger barrier can always be constructed. Furthermore the decision to use 6 rather than 5 steps has to be based on knowledge about the presence of a barrier. Such knowledge must either be given in advance or it has to be derived based on an analysis of the environment. It is not obvious how such an analysis might be performed. Finally as we have previously mentioned, there is a real-time performance-imposed upper limit on the value of n that can be used.

The proposal is to overcome the barrier problems by means of an hierarchical minimum expected cost strategy. In order to implement an hierarchical minimum expected cost strategy two subtasks are performed.

- The environment graph is abstracted. The aim is to generate an hierarchical model of the environment.
- A reasonable strategy is found. This strategy has to apply minimum expected cost on the hierarchical model of the environment.

The solution we are about to propose is based on heuristics. We cannot guarantee that it works in all cases and we have not estimated how suboptimal it is. To obtain a deeper understanding of the problem we need more experimentation.

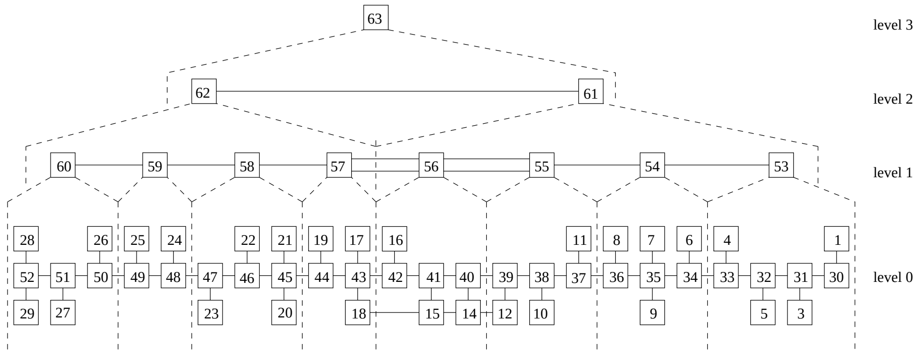


Figure 2: Abstraction tree.

4.3 Abstraction tree construction

The first subtask to perform is abstraction tree construction. To construct the abstraction tree, the environment graph must be repeatedly clustered and abstracted. The clustering process at this moment is manual but the following criteria are taken into account:

1. A path within the cluster should exist between any two nodes in a cluster.
2. No path between any two nodes within a cluster should be greater than the size of look ahead n (we took $n = 5$).
3. The resulting tree should be balanced. This means that at each level all clusters should have approximately the same number of children.

The clustering is performed recursively until the environment graph cannot be further abstracted. The abstract nodes at each level are also interconnected with links that map to doors (fig. 2 (levels 1&2)).

The expected cost of an abstract node is calculated recursively as:

$$\begin{array}{ll} EC_t(X_i) = P_t(X_i = 1)C(X_i) & \text{if } level(X_i) = 0. \\ EC_t(X_i) = \sum_{j=ch(X_i)} EC_t(X_j) & \text{if } level(X_i) > 0. \end{array}$$

where $ch(X_i)$ is the set of children of node X_i. So the expected cost of an abstract node is the sum of the expected costs of all its level 0 descendents.

4.4 Hierarchical minimum expected cost strategy

The second subtask is to apply a strategy on the abstraction tree. We propose the following algorithm:

```
set abstraction level to MAX (where MAX is
    the abstraction level of the most
    abstract level with more than 1 node);

set 'override' to false;
%if 'override' is true, then the abstract
          level overrides MEC at level 0%

while abstraction level > 0 do
 select best next node using 5-step MEC
          at current abstraction level;
 if selected node is present node
   then
     reduce abstraction level by 1;
   else
     set 'override' to true;
     choose best shortest path from present
       (level 0) location to some (level 0)
       location in selected cluster/node;
     set abstraction level to 0;
 end if then else
end while

if 'override' is false
  then
   select best path of length 5 at level
        0 using 5-step committed MEC;
end if then
```

Committed minimum expected cost means that once a path is selected by the strategy the robot sticks to it until its end. Ordinary 5-step strategies reevaluate after each step is taken. So once a path is selected by an ordinary strategy, the first location in the path is visited and then a new 5-step reevaluation takes place. It can be shown that in our environment committed minimum expected cost behaves better that the ordinary version.

To make the hierarchical strategy clearer we give a couple of concrete examples. In the first example *override* is eventually set to true (fig. 3:a).

Example 4.2 *The robot starts at room 4. This is mapped to node 4 at level 0 of the abstraction tree. The strategy has first to decide which node at level 2 should be visited. Suppose its outcome is to stay within the abstract node 61. Then another decision has to be made at level 1. If the decision is to visit node 54 then override is set to true. The robot now has to go to node 54. Consequently the robot moves from 4 to 33, 34 because this is the shortest part from 4 to some descendant of node 54.*

In the second example *override* is never set to true (fig. 3:b).

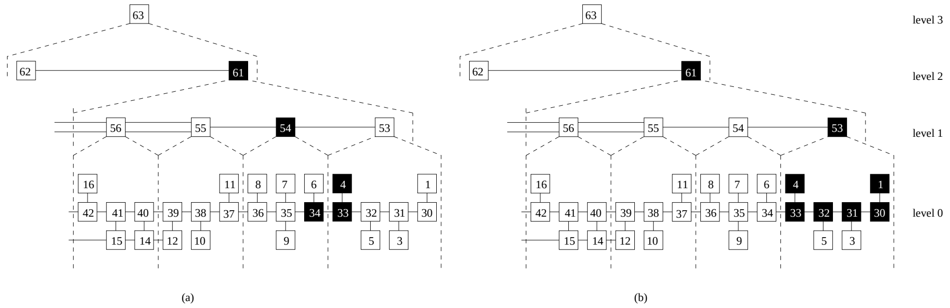


Figure 3: Example 4.2 (a) and 4.3 (b).

Example 4.3 *The robot again begins at room 4 and the same line of reasoning is followed until abstraction level 1. Then level 1 decides to visit node 53. The main loop should be exited and the robot has to use a committed minimum expected cost strategy starting from node 4.*

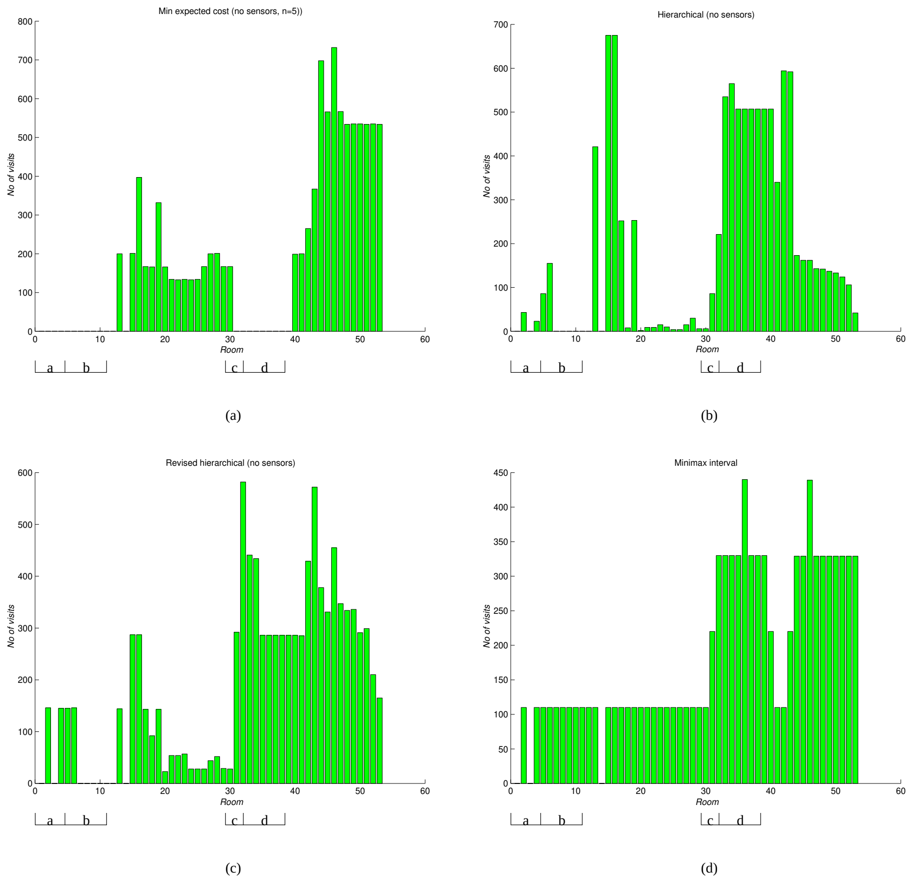
The proposed hierarchical strategy was applied on the environment of example 4.1. The number of times each location was visited was again plotted (fig. 4:b). The location groupings a, c are visited indicating that the robot went beyond the expected cost barrier. This is a positive result. Furthermore the barrier locations on the corridor (group d) are visited when the robot crosses to the other side of the barrier. The barrier locations that are not on the corridor (group b) are correctly not visited since no benefit can be gained by doing so.

One problem we encountered is that the robot frequently swaps between abstract nodes 61 and 62. This in combination with a tendency to want to visit cluster 53, results in more visits to the rooms along the corridor of the barrier (around nodes 33-37). To fix this problem we revised hierarchical MEC to prefer paths that start by staying to current node c.

$$\begin{array}{ll} EC_t([f|r]) = \kappa EC_t([f|r]), \kappa > 1 & \text{if } c = f. \\ EC_t([f|r]) = EC_t([f|r]) & \text{if } c \neq f. \end{array}$$

where f is the first node in the path, r is the rest of the path and κ is a delay constant. The purpose of κ is to delay moving from the current node. The bigger the value of κ the larger the delay. After performing some experiments we have set κ to 1.3. We then replotted the number of visits per room for the revised hierarchical strategy (fig. 4:c). Although now there are some irregular peaks in the plot, the previous problem of swapping between nodes 61 and 62 is solved. Furthermore the nodes in groups a and c are visited more often and this also constitutes an improvement.

The hierarchical strategy succeeds in crossing the barrier. More importantly however, the hierarchical strategy also reduces the total cost. At table 3 you can see the total cost after 10000 iterations for MEC, hierarchical MEC and the revised hierarchical MEC. Even the simple hierarchical MEC strategy reduces



Legend:

a = 1,3,4,5 = rooms that are beyond the barrier.

b = 6,7,8,9,10,11 = rooms that form part of the barrier.

c = 30,31,32,33 = corridor segments that are beyond the barrier.

d = 34,35,36,37,38 = corridor segments that form part of the barrier

Note:

room 0 is disconnected and so it is not visited.

room 2 is the corridor that has been segmented and no longer exists.

room 13 is disconnected and so it is not visited.

Figure 4: Example 4.1 (a) MEC (b) hierarchical MEC (c) revised hierarchical MEC (d) minimax interval.

	minimax interval	MEC	hierarchical MEC	
			original	revised
total cost	37923	635408	103895	34600

Table 3: Total costs after 10000 iterations (without sensors).

the total cost in comparison with MEC, this is an indication that we have improved over the original MEC strategy.

The improvement in total cost arising from making not moving more attractive is quite dramatic. The delay κ is necessary because the expected cost assigned to the abstract nodes is always an overestimation of the actual cost. The expected cost estimate assumes that once the strategy visits an abstract node it stays there long enough for all sub-nodes to be visited. However, it is often the case that the strategy stays in a node only long enough for its expected cost to decrease and for the other nodes to attract the attention of the strategy. When that happens the actual cost is only a portion of the estimated expected cost. We believe that by making not moving more attractive the revised hierarchical strategy stays longer in a node and so our abstracted cost is a better estimate of the real cost.

A possibly more interesting observation that can be made from table 3 is that minimax interval (that does not use the probabilities and costs) performs better than MEC and hierarchical MEC. Only the revised hierarchical MEC outperforms minimax interval and the difference, even then, is not great. Minimax interval moves in the environment in a way that can be almost described as "methodical". Some environment locations are visited more frequently than others only due to the connectivity of the environment (fig. 4:d)

The fact that MEC performs so badly should not be surprising since MEC never passes to the other side of the barrier. Hierarchical MEC appears to be also problematic despite of being an improvement over MEC. It is clear that our hierarchical MEC is not optimal and that there should be a lot of room for improvement. MEC strategies take advantage of the knowledge about the probabilities and costs. This knowledge used effectively should be a greater benefit to MEC strategies. A great deal of work has to still be done to improve the use of this knowledge in our strategies.

5 Unreliable Sensors

We have mentioned in section 2 that it is possible to add sensing uncertainty to our strategies. In this section, we will show how unreliable static fire sensors can be incorporated in our model.

We assume that there is a sensor S_i in every $X_i \in X$. These sensors can return two values, $S_i = 1$ when there is a fire at X_i and $S_i = 0$ when there is no fire at X_i. We further assume that all sensors are unreliable with the same false positive $P(S_i = 1|X_i = 0)$ and false negative $P(S_i = 0|X_i = 1)$ rates. Finally once a sensor reports an alarm, it keeps reporting an alarm until the robot visits its location. At any given time a number of sensors might be reporting a fire and these alarms are transmitted to the robot.

Assuming that no alarms are reported between the time t' of the last visit at X_i and the current time t then $P_t(X_i = 1)$ can be computed using the equation described in proposition 2.1. However, if an alarm is reported at time t'' such that $t' \leq t'' \leq t$, then $P_t(X_i = 1)$ has to be computed in a way that takes into account the alarm.

Assuming that negative sensor evidence does not affect our belief that a fire is present, we propose to start by computing the probability prior to the event (at time t'') of the sensor raising an alarm.

$$P_{prior}(X_i = 1) = 1 - (1 - P(X_i \to 1))^{t''-t'}$$

where t' is the largest time point $\leq t$ such that $r_{t'} = X_i$ and t'' is the time the alarm was raised.

Subsequently the posterior probability after the alarm (at t'') has to be found using Bayes rule:

$$P_{posterior}(X_i = 1) = \frac{P(S_i = 1|X_i = 1) * P_{prior}(X_i = 1)}{P(S_i = 1)}$$

where:

$$\begin{aligned} P(S_i = 1) = \quad & P(S_i = 1|X_i = 1) * P_{prior}(X_i = 1) + \\ & P(S_i = 1|X_i = 0) * P_{prior}(X_i = 0) \end{aligned}$$

Finally the increase in probability from the time of the alarm t'' until the current time t has to be computed using:

$$P_t(X_i = 1) = 1 - (1 - P(X_i \to 1))^{t-t''} * P_{posterior}(X_i = 0)$$

where t'' is the time the alarm at X_i was raised.

The abovementioned probability computation is slightly more involved than that in proposition 2.1. However the computation time for one location is still $O(1)$. This implies that the computation complexity of the surveillance strategies is not affected by the introduction of sensor uncertainty. More importantly, by incorporating the sensory information into our method for computing the probability of fires, we have also incorporated the sensory information to all our MEC strategies. To evaluate the behaviour of the hierarchical MEC strategies while using the sensory information, we run a new set of experiments in a variation of the environment of example 4.1.

Example 5.1 *Consider the environment of example 4.1 but now with a static fire sensor in each room X_i and take $P(S_i = 1|X_i = 0) = P(S_i = 0|X_i = 1) = 0.001$.*

We run our simulation for 10000 steps as before. The total cost for the two hierarchical strategies can be found in table 4. By comparing tables 3, 4 one can see that the hierarchical strategies do substantially better using the sensors than without. Furthermore, now revised hierarchical MEC performs significantly better than minimax interval which is perhaps not surprising given that minimax interval does not use the sensor information.

In figure 5 you can find a graph of the average time to extinguish a fire per room. For exposition purposes, the same negative average time to extinguish (-5) is set for rooms that are never visited. The graph shows that when sensors

	hierarchical MEC	
	original	revised
total cost	57122	22600

Table 4: Total costs after 10000 iterations (with sensors).

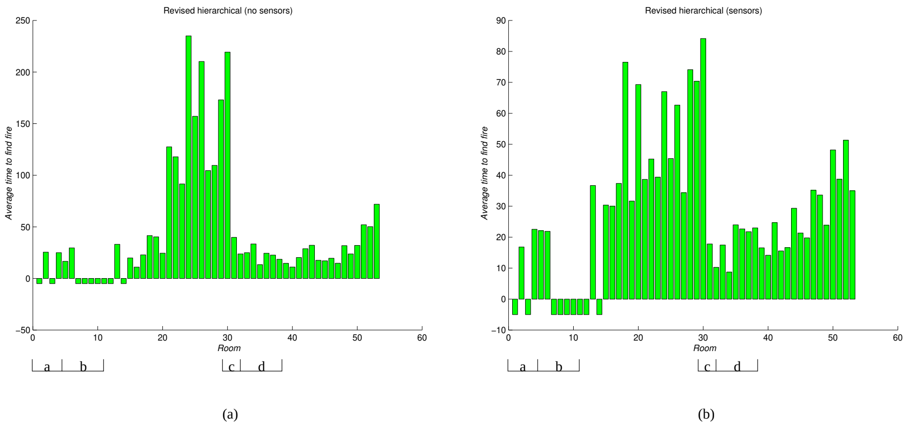


Figure 5: Average time to extinguish fires for each room using revised MEC (a) without sensors (b) with sensors.

are used the average time to extinguish a fire is smaller than when sensors are not used. This decrease in the time fires are active has as net effect the decrease in cost that can be seen in table 4. It can be stated that when using the sensors hierarchical MEC pursues the alarms. For a low enough prior probability of fires and low enough false positive rates the robot should just follow the alarms and almost ignore the time to visit a location.

We have mentioned that hierarchical MEC can deal with the barrier problem. Furthermore, having an environment abstraction allows us to deal with alarms far away from our current location. MEC would not be able to deal with alarms beyond its look-ahead. However in hierarchical MEC the higher decision levels can get information about the alarms despite currently being far from them. Therefore hierarchical MEC deals more efficiently with alarms in addition to dealing with the barrier problem.

Other types of sensing uncertainty can similarly be incorporated into our model. In these simulations we essentially have 2 sensors; the robot that is completely reliable but myopic and the static sensors that are unreliable but are omnipresent. We could change our example by having two types of sensors both mounted on the robot and both having limited (albeit different) sensor ranges. Furthermore in a similar manner localisation uncertainty can be added to our framework.

6 Further Work

Given that our approach is cursory there are several areas that can be improved. Presently we are investigating methods to generate the abstraction tree automatically given an office-like environment. We also plan to investigate whether it is better to have a dynamic abstraction tree instead of a static abstraction tree. By dynamic abstraction we mean an abstraction that is computed every time a decision has to be made and that takes into account the current robot location.

We want to improve our method of estimating the expected cost in the more abstract levels. Our present estimation is *ad hoc* and we have not calculated how suboptimal it is. It would be interesting to consider sound abstraction like that of Doan and Haddawy [4]. There the approximation of the expected cost of abstract nodes is specified using upper and lower bounds on the expected cost so it is more accurate.

As mentioned in the introduction, the use of abstract representations to obtain approximate solutions at a lower computational cost has been proposed before. The relation of the present work with similar proposals in the context of (Partially Observable) Markov Decision Problems, such as [3], has to be investigated.

7 Conclusions

We have previously showed [8, 10] that a minimum expected cost strategy is a promising method for planning in robotic surveillance. In practice, minimum expected cost typically cannot be evaluated globally. Only n-step minimum expected cost strategies are feasible. In order to overcome problems imposed by local minima and barriers we have implemented a hierarchical minimum expected cost strategy.

We believe that abstraction is necessary for large environments. Abstraction is something humans always do when reasoning about possible outcomes of actions. Our hierarchical strategy is *ad hoc* but it should provide the basis on which a more sound strategy can be implemented. Although our method is *ad hoc* we have demonstrated the potential improvement on the unabstracted strategies.

To sum up, we discussed a decision-theoretic strategy for surveillance as a first step towards automating the planning of the movement of an autonomous surveillance robot. We extended a previous proposal by including some heuristics based on an abstract representation of the environment. We showed, using a concrete example, how these heuristics allow computationally feasible, finite look-ahead versions of the decision-theoretic strategy to escape local minima, and to better approximate globally optimal solutions. Finally, we demonstrated the ability to incorporate in our decision theoretic strategy, uncertainty arising from sensors.

Acknowledgments

Thanks to Leo Dorst and Frans Groen for many helpful discussions and comments. The investigations were carried out as part of the PIONIER-project Reasoning with Uncertainty, subsidized by the Netherlands Organization of Scientific Research (NWO), under grant pgs-22-262.

References

[1] A.R. Cassandra, L. P. Kaelbling, and J.A. Kurien. Acting under uncertainty: discrete bayesian models for mobile-robot navigation. In *IROS'96*, 1996.

[2] Cybermotion. Cyberguard. (description of a commercial robotic surveillance system). www.cybermotion.com available since, 1990.

[3] R. Daerden and C. Boutilier. Abstraction and approximate decision-theoretic planning. *Artificial Intelligence*, 89(1):219–283, 1997.

[4] A. Doan and P. Haddawy. Sound abstraction of probabilistic actions in the constraint mass assignment framework. In *UAI'96*, pages 228–235, 1996.

[5] P. Fabiani. A new approach in temporal representation of belief for autonomous observation and surveillance systems. In *ECAI'94*, pages 391–395, 1994.

[6] P. Fabiani. *Représentation dynamique de l' incertain et stratégie de prise d' information pour un système autonome en enviroment évolutif.* PhD thesis, L' école nationale supérieure de l' aéronautique et de l' espace, Toulouse, November 1996.

[7] M. Hauskrecht, N. Meuleau, L. P. Kaelbling, T. Dean, and Craig Boutilier. Hierarchical solution of markov decision processes using macro-actions. In *UAI'98*, pages 220–229, 1998.

[8] N. Massios and F. Voorbraak. Planning strategies for decision-theoretic robotic surveillance. In *NAIC'98*, pages 117–126, November 1998.

[9] D.W. Murphy and J.P. Bott. On the lookout: The air mobile ground security and surveillance system (AMGSSS) has arrived. *Unmanned Systems*, 13(4), 1995.

[10] F. Voorbraak and N. Massios. Decision-theoretic planning for autonomous robotic surveillance. In *ECAI'98 Workshop Decision Theory meets Artificial Intelligence. Qualitative and Quantitative Approaches*, pages 23–32, University of Brighton, August 1998.