Robot Motion Planning in a Changing, Partially Predictable Environment

Steven M. LaValle
lavalle@cs.uiuc.edu

Rajeev Sharma
rajeev@cs.uiuc.edu

The Beckman Institute
University of Illinois, Urbana, IL 61801

Abstract

In this paper we present a framework for analyzing and determining robot motion plans for situations in which the robot is affected by an environment that probabilistically changes over time. In general, motion planning under uncertainty has recently received substantial interest, and in particular a changing-environment has been recognized as an important aspect of motion planing under uncertainty. We model the environment as a finite-state Markov process, and the robot executes a motion strategy that is conditioned on its current position and the state of the environment. Optimality of a robot strategy is evaluated in terms of a performance functional that depends on the environment, robot actions, and a precise encoding of relevant preferences. By using a simple, yet powerful computation technique that is based on dynamic programming, we can numerically compute optimal robot strategies for a wide class of problems, surpassing previous results in this context that were obtained analytically. Several computed motion planning examples are presented.

1 Introduction

The modeling and analysis of uncertainty is crucial to the design of an autonomous system that navigates in a realistic environment. Four important aspects of uncertainty in robot motion planning are:

1. Partially predictable robot actions (e.g., [1, 3, 4])
2. Imperfect sensing (e.g., [3, 5])
3. Incomplete environment information (e.g., [2, 12])
4. Changing, partially predictable environment (e.g., [10, 11])

This first aspect indicates situations in which some form of motion command is given to a robot; however, the actual direction or path chosen by the robot can only be known to lie within some set, or is sampled from a known probability distribution. With imperfect sensing, the information that the robot has regarding its current configuration may be incomplete or inaccurate, which complicates the selection of a robot plan that adequately accomplishes some goal. Incomplete environment information represents a difficult aspect of uncertainty in which the robot does not have complete knowledge of structure of the workspace, but instead the structure is known to belong to some set of alternatives (or in some cases is completely unknown).

The final aspect of uncertainty is the primary focus of this paper. We will present motion planning problems with this form of uncertainty in isolation, to analyze and demonstrate the effects of a changing environment on a robot strategy. We assume that the changing portion of the environment can at least be predicted with probability distributions. We then present a method of treating this problem in a stochastic framework, which uses some general notions from stochastic optimal control theory [6]. Using this framework, we can compute numerical solutions that are optimal with respect to some criteria that precisely characterize the problem of interest. One could also combine a changing, partially predictable environment with other forms of robot uncertainty. In an approach similar to the one presented here, motion planning with partially predictable actions and sensing uncertainty has been recently addressed [9], facilitating such a combination.

A model of uncertainty due to a changing environment was introduced in [10, 11]. The key idea was to abstract the dynamic component of the environment in terms of discrete events or alarms and use a probabilistic framework to analyze the expected performance of different motion planning strategies. However, the development of analytical, optimal solutions under this framework becomes quite difficult with a more complicated environment than those presented in [10] and [11]. This motivates the approach taken in this paper, which is able to model more complicated changing environments and uses dynamic programming to achieve numerical solutions.

One further benefit is the ability to experiment with a variety of cases by varying the definition of a precise performance functional and transition probabilities (these terms are defined in Section 2). These variations allow us to study the situations in which the optimal behavior of the robot changes considerably, offering a greater understanding of the interaction of a robot with a changing, partially predictable environment.

Section 2 presents the general framework and definitions. In Section 3 the framework is applied to a corridor crossing problem, which was introduced in [10]. In Section 4, we obtain optimal solutions for a servicing problem, which was introduced in [11]. Section 5 provides a concluding discussion.

2 Our General Framework

In this section we define the general concepts and terminology that form the basis for our framework. Since the interaction of the robot with a changing environment

is of fundamental importance, time must be explicitly considered in both motion planning and execution. We consider a discretized representation of time as *stages*, with an index $k \in \{1, 2, \ldots, K\}$. Stage k refers to time $(k-1)\Delta t$. We generally take Δt sufficiently small to approximate continuous paths. This appropriately reflects a situation in which a real robot is limited to some sampling rate for acquiring sensor information and executing motion commands. A finite K is only used to prevent us from developing a special treatment of infinite stages. In practice, an appropriate value of K is automatically determined during the execution of the dynamic programming (see Section 2.1).

In robot motion planning, the position of $\mathcal{A}$ in a workspace is usually represented by a point in an n-dimensional *configuration space*, $\mathcal{C}$, in which n is the number of degrees of freedom of $\mathcal{A}$ [7]. In this paper we are generally interested in planning paths in which the robot configurations are guaranteed to lie in $\mathcal{C}_{free}$, which is the subset of the configuration space in which the robot is not in contact with obstacles. In particular, for the examples presented here, we take $\mathcal{C}_{free} \subseteq \Re^2$. We could alternatively execute compliant motions and use the subset $\mathcal{C}_{valid}$ [3, 8].

For geometric motion planning problems without uncertainty, the space of possible situations that can occur is sufficiently characterized by $\mathcal{C}$. In our context, the environment can additionally interfere with the motion plans of $\mathcal{A}$. We consequently define a finite set, E, of *environment states*. At a given stage, k, the environment is in some state $e_k \in E$, which is known to the robot. To uniquely identify all of the possible situations that can occur in our changing environment problem, we define a *state space* as $X \subseteq \mathcal{C} \times E$. The state of $\mathcal{A}$ at stage k is denoted by x_k, which represents both a configuration of $\mathcal{A}$ in the geometric sense, and an environment state, e_k. The environment states in E form a partition of the state space, X. Each time the environment state changes, the robot is forced into a different portion of X (see Figure 1). By using the state space representation, we could

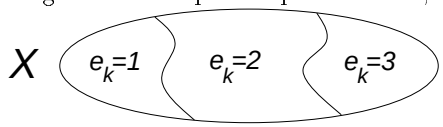

Figure 1. The environment process can also be considered as a partition of X.

for instance model a situation in which $\mathcal{C}_{free}$ can change over time. A different environment state might represent the fact that a corridor has been blocked, and the corresponding subset of the state space represents the resulting $\mathcal{C}_{free}$. It will be seen in the coming sections that a other situations can be modeled by this form of state space.

We additionally consider the environment as a finite-state Markov chain, which we call the *environment process*. At the initial stage, $k = 1$, the environment state, $e_1 \in E$ is given. For a given environment state e_k, the next environment state, e_{k+1} is specified as a probability distribution over E. This probability distribution is defined by a vector P_i such that $P_i[j] = P(e_{k+1} = j | e_k = i)$. Figure 2 depicts an example environment process for which $E = \{1, 2, 3\}$. The environment state transition probabilities can be specified as

$$\begin{array}{lccc} P_1 = & [0.8 & 0.1 & 0.1] \\ P_2 = & [0.1 & 0.6 & 0.3] \\ P_3 = & [0.2 & 0.5 & 0.3]. \end{array} \qquad (1)$$

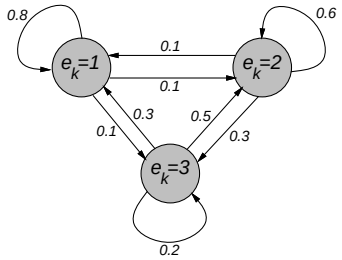

Figure 2. The environment process can be considered as a finite-state Markov chain with state transition probabilities.

For this example, the environment process is independent of the configuration of $\mathcal{A}$. In general, however, we allow $\mathcal{A}$ to have influence over the environment by defining the environment transition probabilities as $P(e_{k+1}|x_k, u_k)$. This is a function of both the state, x_k, and the robot action u_k, which will be defined next. We could also incorporate environment transition probabilities that change over time; however, we preclude this type of model in this presentation.

An *action* (or command), which is denoted by u_k, can be issued to $\mathcal{A}$ at each stage k. We let U denote the *action space* for $\mathcal{A}$, requiring that $u_k \in U$. To describe the effect of a robot action with respect to state, we define a *state transition distribution* as $P(x_{k+1}|x_k, u_k)$. In this paper we assume that if e_{k+1} is additionally given, then x_{k+1} is completely determined. Note, however, that $\mathcal{A}$ only has information about future environment state through an environment state transition probability distribution.

The following state transition distribution is sufficiently general to characterize the examples presented in this paper. We have $\mathcal{C} \subset \Re^2$. We define the action space as $U = [0, 2\pi) \cup \{\emptyset\}$. If $u_k \in [0, 2\pi)$, then $\mathcal{A}$ attempts to move a distance $||v||\Delta t$ toward a direction in $\mathcal{C}$, in which $||v||$ denotes some fixed speed for $\mathcal{A}$. If $u_k = \emptyset$, then the robot remains motionless.

Consider the case in which $x_k \in \mathcal{C}_{free}$ is at a distance of at least $||v||\Delta t$ from the obstacles. If $\mathcal{A}$ chooses action $u_k \neq \emptyset$ from state x_k, then[1]

$$x_{k+1} = \begin{bmatrix} x_{k,1} + ||v||\Delta t\ cos(u_k) \\ x_{k,2} + ||v||\Delta t\ sin(u_k) \\ e_{k+1} \end{bmatrix}, \qquad (2)$$

in which the environment state e_{k+1} is known to be sampled from $P(e_{k+1}|x_k, u_k)$. We can thus consider a finite-valued random variable X_{k+1} with corresponding distribution $P(x_{k+1}|x_k, u_k)$, which can be inferred from the given model. If $u_k = \emptyset$, then $x_{k,1} = x_{k+1,1}$ and $x_{k,2} = x_{k+1,2}$; however, e_{k+1} is not necessarily equal to e_k. We currently prohibit the robot from considering motion directions that collide with obstacles; however, one could also consider compliant motion [3, 8].

We now define the notion of a robot plan or strategy in our stochastic framework. At first it might seem appropriate to define some action u_k for each stage; however, we want plans that are prepared for the various contingencies presented by the changing environment. Therefore, we define a *strategy at stage* k of $\mathcal{A}$ as a function

[1] We use the notation $x_{k,i}$ to refer to the i^{th} element of the vector x_k.

$\gamma_k : X \rightarrow U$. For each state, x_k, a strategy yields an action $u_k = \gamma_k(x_k)$. The set of mappings $\{\gamma_1, \gamma_2, \ldots, \gamma_K\}$ is denoted by γ and termed a *strategy* of $\mathcal{A}$. This is equivalent to a control law or policy in control theory [6]. For the examples that we present in this paper, the decision functions γ_k will be the same for all k (i.e., each robot action depends only on the current state, and not the particular stage).

Some subset G of the state space X is defined as the *goal region*. The robot terminates if $x_k \in G$. By this we mean that if $x_k \in G$, then for all k' such that $k \leq k' \leq K$, we choose $u'_k = \emptyset$. This is similar to the notion of a *termination condition* in [3, 7], but in our work there is no uncertainty associated with the present state (or configuration) of $\mathcal{A}$.

We use the notation $w(\gamma, x_1, \mathbf{e})$ to refer to the path taken through the state space by the implementation of γ, an initial state, x_1, and a given environment state sequence $\mathbf{e} = \{e_1, e_2, \ldots, e_K\}$. We refer to $w(\gamma, x_1, \mathbf{e})$ as a *sample path* for γ (given x_1 and $\mathbf{e}$). We also define $W(\gamma, x_1)$, which is a random process that takes on values of sample paths once $\mathbf{e}$ is known. Note that the probability distribution of $\mathbf{e}$ can be directly determined from γ, x_1, and the state transition distribution; therefore, the probability distribution over the sample paths (which defines $W(\gamma, x_1)$) is known.

We can encode the objectives that are to be achieved by a nonnegative real-valued functional $L(x_1, ..., x_{K+1}, u_1, ..., u_K)$, called the *loss functional* (or performance functional) of $\mathcal{A}$. A robot strategy that produces less loss will be considered preferable. We will be interested in considering loss that is accumulated at each stage; hence, it is assumed that the loss functional can be expressed as

$$L(x_1, ...x_{K+1}, u_1, ..., u_K) = \sum_{k=1}^{K} l_k(x_k, u_k) + l_{K+1}(x_{K+1}). \tag{3}$$

We state that $l_k(x_k, \emptyset) = 0$, implying that there is no additional loss for choosing $\emptyset$.

The ultimate goal of the planner is to determine an optimal strategy $\gamma^* = \{\gamma_1^*, \gamma_2^*, \ldots, \gamma_K^*\}$ that causes L to be minimized in an expected sense.

2.1 Computing optimal strategies

One of the primary advantages of our framework is that a straightforward numerical computation procedure can be used to determine optimal strategies. We employ the dynamic programming principle on the state space to recursively determine γ^*. Although the computational cost of dynamic programming increases exponentially in the dimension of the state space, for a given dimension the algorithm is very efficient. We expect the computational approach to be reasonable for problems as large as a three dimensional configuration space with several environment states. This dimensionality includes many interesting motion planning problems (see [7]); however, for more difficult problems some sort of suboptimal solution technique might be required.

The expected loss obtained by starting from stage k, and implementing the portion of the optimal strategy, $\{\gamma_k^*, \ldots, \gamma_K^*\}$, can be represented as

$$\bar{L}_k^*(x_k) = E\left\{\sum_{i=k}^{K} l_i(x_i, \gamma_i^*(x_i)) + l_{K+1}(x_{K+1})\right\}. \tag{4}$$

The expectation is taken over the possible environment sequences, $\mathbf{e}$.

The dynamic programming principle [6] states that $\bar{L}_k^*(x_k)$ can be obtained from $\bar{L}_{k+1}^*(x_{k+1})$ by the following recurrence:

$\bar{L}_k^*(x_k) =$

$$\min_{u_k}\left\{l_k(x_k, u_k) + \sum_{x_{k+1}} \bar{L}_k^*(x_{k+1}) P(x_{k+1}|x_k, u_k) dx_k\right\}. \tag{5}$$

Note that the sum in (5) is taken over a finite number of states, which can be reached using (2).

At stage $K+1$, we can use the last term of (3) to obtain $\bar{L}_{K+1}^*(x_{K+1}) = l_{K+1}(x_{K+1})$. The loss functional $\bar{L}_K^*$ can be determined from $\bar{L}_{K+1}^*$ through (5). Using the $u_K \in U$ that minimizes (5) at x_K, we define $\gamma_K^*(x_K) = u_K$. If $x_K \in G$, then $\gamma_k(x_k) = \emptyset$. We then apply (5) again, using $\bar{L}_K^*$ to obtain $\bar{L}_{K-1}^*$ and γ_{K-1}^*. Eventually, the loss values stabilize, and we terminate when $|\bar{L}_k^*(x) - \bar{L}_{k+1}^*(x)|$ becomes small for all $x \in X$. After the algorithm terminates the resulting stage is designated as $k = 1$, and an explicit prior choice of K is not necessary. Finally, we take $\gamma^* = \{\gamma_1^*, \ldots, \gamma_K^*\}$.

In our implementation of the dynamic programming algorithm, we quantize the state space into an array, typically of size $40 \times 40 \times |E|$, in which $|E|$ represents the number of environment states. Furthermore, the action set is quantized (typically into 32 or 64 values, excluding $\emptyset$) for the evaluation of (5). The resulting loss function $\bar{L}_1^*(x_1)$ shares similarities with the concept of a global navigation function in motion planning [7], and the corresponding *wavefront expansion method* from that context can be viewed as a special form of dynamic programming.

3 A Corridor Crossing Problem

In this section we will consider a corridor crossing problem that generalizes the problem treated analytically in [10]. This serves as the first illustration of the general framework presented in Section 2. It clearly demonstrates how dynamic programming is used to obtain the numerically optimal solution. Although the optimal solution was obtained analytically in [10], the model presented here can be changed easily to reflect a modified behavior of the robot, which makes the approach more flexible.

Suppose that the goal of a robot is to move from a point A to a point B across a corridor that has other moving objects. The motion of the objects in the environment is not known *a priori*. There are no fixed obstacles, so under the absence of other moving objects the robot would proceed in a straight line from A to B. However, while the robot is within the corridor an approaching object will cause it to dart to the safety of the opposite side of the corridor. In order to model the situation it was suggested in [10] that this dynamic behavior of the robot be modeled in terms of "alarms" which follow a probabilistic distribution. In particular, a Poisson distribution was used to reflect the independence of the alarms. Here we generalize the model in terms of the framework presented earlier. The environment changes are captured in terms of the two states—alarm and no-alarm states, and the model is developed as follows.

We define a subset $S \subset X$ as a shelter region. When the robot is in this portion of the state space, the alarm condition can be ignored. For this example, we consider the other side of the corridor to be a shelter region, while the interior of the corridor is unsafe.

We consider a two-state environment process. The environment states are $E = \{0, 1\}$, in which $e_k = 0$ indicates that the alarm is off at stage k, and $e_k = 1$ indicates that the alarm is on at stage k. The transition probabilities are generally specified as $P_0 = [p_0 \quad 1 - p_0]$ and $P_1 = [0 \quad 1]$. By the definition of P_1, once the alarm is on, it remains on for the duration of the execution.

The following represents a general loss functional that applies to the models in Sections 3 and 4:
$L(x_1, ...x_{K+1}, u_1, ..., u_K) =$

$$\begin{cases} \sum_{k=1}^{K} l_k(x_k, u_k) & \text{if } x_{K+1} \in G \\ \\ c_f & \text{otherwise} \end{cases} . \qquad (6)$$

If $u_k \neq \emptyset$, then the term l_k is defined as

$$l_k(x_k, u_k) = \begin{cases} c_u & \text{if } x_{k,3} = 0 \text{ or } x_k \in S \\ c_u + c_a & \text{otherwise} \end{cases} . \qquad (7)$$

Otherwise $l_k = 0$.

A cost of c_f is incurred if the goal is not completed. If this value is taken to be as large as possible with respect to the other numbers, then the optimal strategy γ^* will always solve the goal. The variable c_f is included, however, in case one wants to consider not solving the goal and accepting some penalty. At each stage, a fixed cost c_u is added. Since the distance that the robot can move at each stage is fixed, the cost c_u simply becomes a measure of the distance traveled along the path by the robot. The cost c_a represents the importance of the alarm condition. If c_a is small, then the robot can choose to ignore it with little penalty. If c_a is large, then the robot is forced to respond. We note that this adds to the flexibility of our approach compared to the model considered in [10] where c_a was assumed to be infinitely high and the robot was forced to respond to an alarm.

In this section we present and compare the numerically optimal strategies for two different cases. We consider Case 1, in which $c_a = 2$, and Case 2, in which $c_a = 10$, while the following conditions remain fixed. The configuration space for this problem ranges from 0 to 100 along each axis. The robot has the ability to move a fixed distance $\|v\|\Delta t = 2$ at each stage. We take $c_u = 1$ and $c_f = 1000$. We take $p_0 = 0.98$. The starting configuration is $(80, 95)$, and the initial environment state is $e_1 = 0$ (representing $x_1 = (80, 95, 0)$). The goal, $G \subset X$, is a set of two points, $(10, 10, 0)$ and $(10, 10, 1)$.

By using the dynamic programming algorithm described in Section 2.1, we obtained an optimal strategy, γ^*, for each of Case 1 and Case 2. Figures 3.a and 3.b depict the level sets of $\bar{L}_1^*(x_1)$ for Case 1. Figure 3.a corresponds to the subset of X in which $e = 0$. Figure 3.b corresponds to the remaining portion of X, in which $e = 1$. The difference between the contour lines in Figures 3.a and 3.b provides a clear illustration of the effect that the alarm has on the planning problem. The values for $\bar{L}_1^*(x_1)$ increases monotonically in both figures as the distance from the goal increases; however, there is a much sharper increase when $e = 1$. Furthermore, the shelter has the effect of spreading out the contours since a region is provided in which the alarm penalty has no effect.

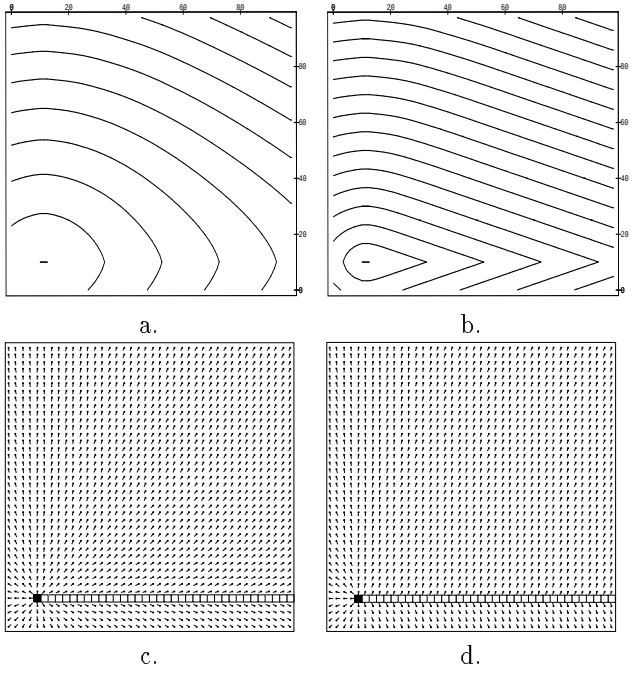

Figure 3. Level sets of $\bar{L}_1^*(x_1)$ and robot strategies for a corridor crossing problem. See Section 3 for details.

Figures 3.c and 3.d depict the computed optimal strategy, γ^* for Case 1. Figure 3.c corresponds to the subset of X in which $e = 0$, and Figure 3.d corresponds to $e = 1$. At each quantized location in $x_k \in X$, a direction is indicated that specifies the action $u_k = \gamma_k^*(x_k)$ that the robot will take when in that location. In these figures, and in figures from Section 4, white enclosed regions indicate S, and black enclosed regions indicate G. One can observe from these figures that when e becomes 1, the robot action changes from a direction that points close to the goal, to pointing almost vertically downward.

Figure 4.a shows a sample path, $w(\gamma^*, x_1, \mathbf{e}^0)$, for Case 1, in which $\mathbf{e}^0 = \{0, 0, \ldots, 0\}$. This corresponds to the situation in which the robot is prepared for alarms, but none occur. Figure 4.b represents Case 2.

A path $w(\gamma^*, x_1, \mathbf{e}^0)$ is equivalent to the notion of a *static path*, which was presented in [10]. In that work, these paths were shown analytically to be exponential in form, which appears to be true for our numerically optimal strategy. Further, the exponential curve is sharper in Figure 4.b due to the increased cost c_a, which was observed for an increased Poisson frequency in [10].

Figures 4.c and 4.d show 30 sample paths drawn from the random process $W(\gamma^*, x_1)$, for Case 1 and Case 2, respectively. This corresponds to the behavior that would be observed if the models are accurate, and the robot repeatly executes the optimal strategy γ^*.

4 A Servicing Problem

In this section we consider a servicing problem, which is a generalization of the problem considered analytically in [11].

The point robot moves in a plane that is populated with a certain *service area* (or shelters [11]). The changing environment is again captured in terms of *alarms*

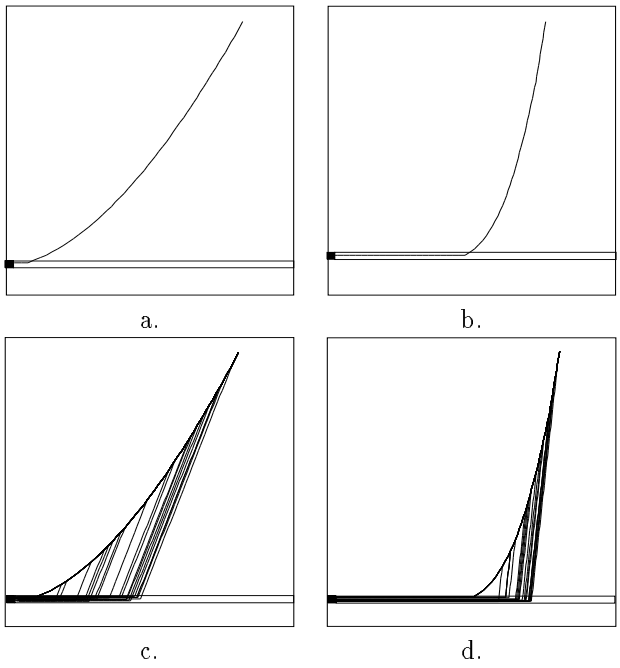

Figure 4. Sample paths of $W(\gamma^*, x_1)$ for a corridor crossing problem. See Section 3 for details.

that are detected on-line and follow a Poisson distribution. When the environment is in the alarm state the robot should move to the nearest part of the service area. After this area is reached, the robot should continue its motion toward a goal position. The problem is to find a strategy that minimizes the expected cost, when there are penalties for both path length and ignoring an alarm. A simple variation of this problem was considered in [11], in which the service area was restricted to be a set of points in the plane and analytical solutions were presented for the limiting cases when the alarms have a very high or a very low frequency. We model a more general version of that problem and present the optimal solution which is computed numerically.

Let $S \subseteq X$ represent the service area. Note that S is not necessarily connected. When an alarm is on, the robot can respond by going to a point in S and turning it off. The environment states are $E = \{0, 1\}$, in which $e_k = 0$ indicates that the alarm off at stage k, and $e_k = 1$ indicates that the alarm is on at stage k. The transition probabilities depend on the state of the robot. We define $P_0 = [p_0 \quad 1 - p_0]$. If $e_k = 1$ and $x_k \notin S$, then we have $P_1 = [0 \quad 1]$. However, if $e_k = 1$ and $x_k \in S$, then $P_1 = [1 \quad 0]$.

In this section we present and compare the numerically optimal strategies for three different cases. We consider Case 1, in which $c_a = 2$, and Case 2, in which $c_a = 20$. The following conditions remain fixed. The configuration space for this problem ranges from 0 to 100 along each axis. The robot has the ability to move a fixed distance $\|v\|\Delta t = 2$ at each stage. We take $c_u = 1$ and $c_f = 1000$. We take $p_0 = 0.90$. The initial state is $x_1 = (5, 95, 0)$. The goal, G, is a set of two points, $(95, 5, 0)$ and $(95, 5, 1)$.

Figure 5.a shows the sample path $w(\gamma^*, x_1, \mathbf{e}^0)$ for Case 1, and Figure 5.b shows $w(\gamma^*, x_1, \mathbf{e}^0)$ for Case 2. Figures 5.c and 5.d show 30 sample paths from $W(\gamma^*, x_1)$ for Case 1 and Case 2, respectively. For Case 1, the robot path is pulled toward the service area; however,

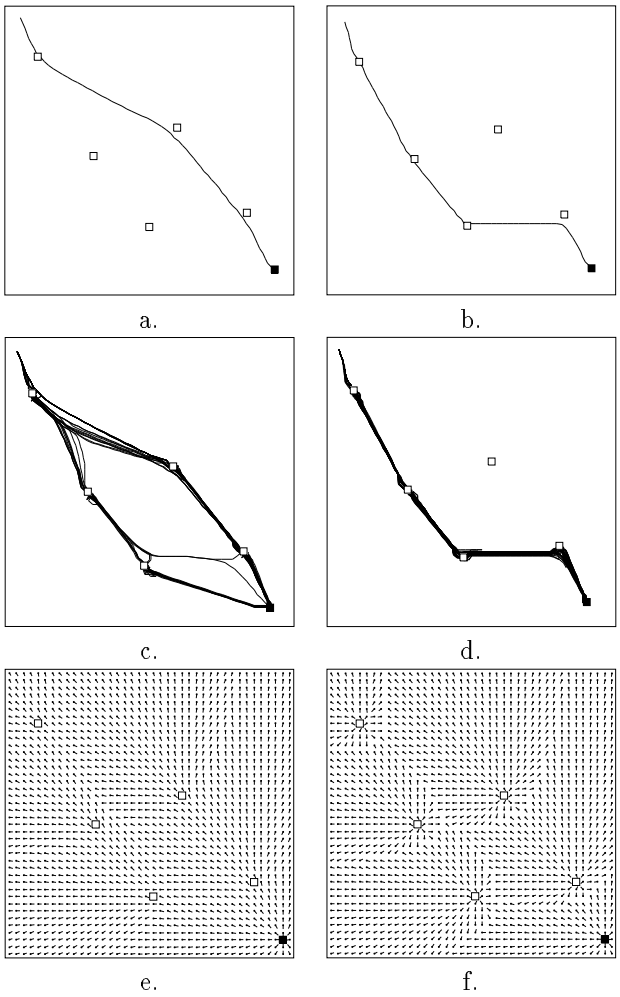

Figure 5. Sample paths and a strategy for a servicing problem. See Section 4 for details.

in Case 2 the penalty is so strong that the robot plans to move in nearly straight line segments from shelter to shelter. This coincides with the Delaunay path, which was shown to be the analytic solution for this problem when the alarm frequency is high [11]. We were able to compute a numerically optimal γ^* for Case 1 and Case 2, for which the analytical solutions are not known.

Figure 6 depicts the level sets of $\bar{L}_1^*(x_1)$ for both cases. Figures 6.a and 6.b depict the level sets for Case 1, Figures 6.c and 6.d depict the level sets for Case 2. It is important to note the circular curves that tend to surround the components of the service area. The service area causes wells to appear in $\bar{L}_1^*(x_1)$, which increase in size as c_a increases.

In Figure 7 we show results for a servicing problem in which there are also obstacles in the workspace. For this problem, no analytical results are yet known [11]. We consider Case 1, in which $c_a = 3$, and Case 2 in which $c_a = 8$, while the following conditions remain fixed. The configuration space again ranges from 0 to 100 along each axis. The robot has the ability to move a fixed distance $\|v\|\Delta t = 2$ at each stage. We take $c_u = 1$ and $c_f = 10000$. We take $p_0 = 0.90$. The initial state is $x_1 = (5, 95, 0)$. The goal, G, is a set of two points, $(95, 40, 0)$ and $(95, 40, 1)$. Figure 7.a shows the sample path $w(\gamma^*, x_1, \mathbf{e}^0)$ for Case 1, and Figure 7.b shows $w(\gamma^*, x_1, \mathbf{e}^0)$ for Case 2. For Case 2, the penalty c_a is

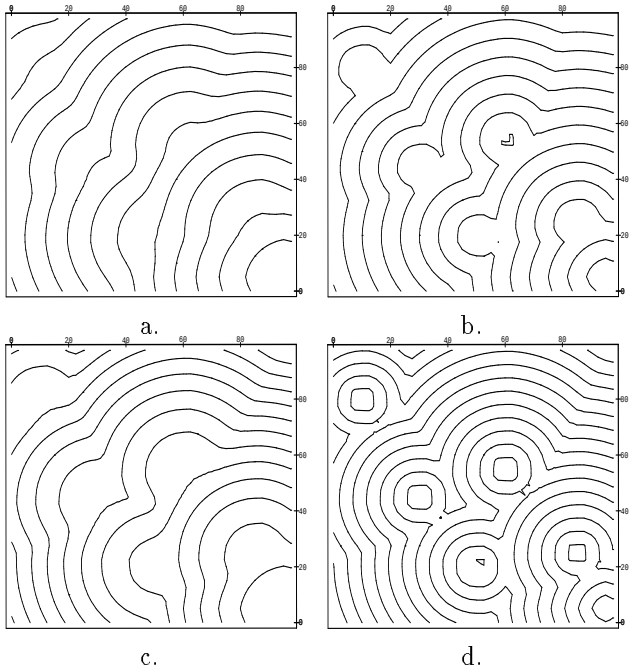

Figure 6. Level sets of $\bar{L}_1^*(x_1)$ for a servicing problem. See Section 4 for details.

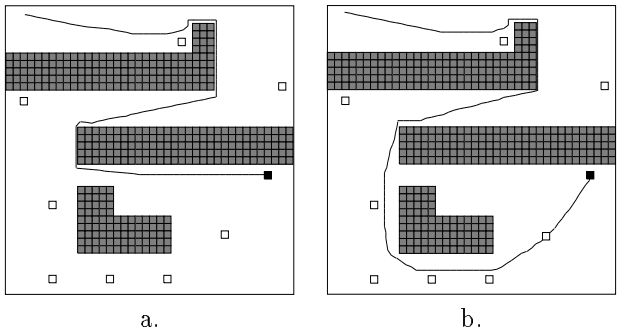

Figure 7. Sample paths for the Servicing problem with obstacles. See Section 4 for details.

high enough to cause the robot to choose paths that have a greater length, as long as the service area is nearby to keep the overall cost low.

5 Discussion

We have presented a general framework for analyzing and determining robot motion plans under a partially predictable, changing environment. The environment is modeled as a finite-state Markov process in which the set of environment states is chosen to represent a particular changing situation. Under this probabilistic framework we define the concept of optimality in the expected sense and then present a method for computing the iterative solutions using a dynamic programming approach. Although this only provides a numerically optimal solution, the computation scheme is powerful enough to handle a wide class of motion planning problems that go far beyond what is known in literature about the analytical solutions for the uncertain and changing environment.

We consider the specific examples explored in this paper as initial experiments with a general framework for dealing with a partially predictable, changing environment. Several generalizations and applications of the models can be considered.

We could consider a set of M service areas, $\{S_1, \ldots, S_M\}$, each performing some separate function. Each service area has its own alarm that indicates that a service needs to be performed by bringing the robot to a state in S_i. Note that each S_i could have multiple components, and the service areas could in general overlap. The environment states would represent various subsets of alarms that are allowed to be on at a given stage.

As mentioned in Section 2, we can also model a scenario in which $\mathcal{C}_{free}$ changes, depending on the environment state. This would lead to an optimal strategy in which the robot takes into account the possibility that the topology of C_{free} might suddenly change at some future time.

Although our discussion was based on *robot* motion planning, the same stochastic framework can be applied to other situations with partially predictable, changing environments. One example would be for designing routing strategies in a computer network via a set of nodes some of which could fail randomly. Another example analogous to that of the problem analyzed in Section 4 would be what we would term as a "Traveling Serviceman Problem": to design an optimal strategy for a technician through a service area with a stochastic modeling of the demands from different stations.

References

[1] K. Basye, T. Dean, J. Kirman, and M. Lejter. A decision-theoretic approach to planning, perception, and control. *IEEE Expert*, 7(4):58–65, August 1992.

[2] B. R. Donald. *Error Detection and Recovery for Robot Motion Planning with Uncertainty*. PhD thesis, MIT, Cambridge, MA, 1987.

[3] M. A. Erdmann. On motion planning with uncertainty. Master's thesis, MIT, Cambridge, MA, August 1984.

[4] K. Y. Goldberg. *Stochastic Plans for Robotic Manipulation*. PhD thesis, Carnegie-Mellon, Pittsburgh, PA, August 1990.

[5] G. D. Hager. *Task-Directed Sensor Fusion and Planning*. Kluwer Academic Publishers, Boston, MA, 1990.

[6] P. R. Kumar and P. Varaiya. *Stochastic Systems*. Prentice Hall, Englewood Cliffs, NJ, 1986.

[7] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[8] J.-C. Latombe, A. Lazanas, and S. Shekhar. Robot motion planning with uncertainty in control and sensing. *Art. Intell.*, 52:1–47, 1991.

[9] S. M. LaValle and S. A. Hutchinson. An objective-based stochastic framework for manipulation planning. In *IEEE/RSJ/GI International Conference on Robots and Systems* (to appear), September 1994.

[10] R. Sharma. Locally Efficient Path Planning in an Uncertain, Dynamic Environment using a Probabilistic Model. *IEEE Transactions on Robotics and Automation*, 8(1):105–110, February 1992.

[11] R. Sharma, D. M. Mount, and Y. Aloimonos. Probabilistic Analysis of Some Navigation Strategies in a Dynamic Environment. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(5):1465–1474, September 1993.

[12] A. Stentz. Optimal and efficient path planning for partially-known environments. In *IEEE Int. Conf. on Robotics and Automation*, pages 3310–3317, 1994.