Landmark-Based Robot Motion Planning

Anthony Lazanas and Jean-Claude Latombe
lazanas@cs.stanford.edu latombe@cs.stanford.edu
Robotics Laboratory
Department of Computer Science, Stanford University
Stanford, CA 94305

Abstract

This paper describes a reduced version of the general motion planning problem in the presence of uncertainty and a complete polynomial algorithm solving it. This algorithm computes a guaranteed plan by backchaining non-directional preimages of the goal until one fully contains the set of possible initial positions of the robot. It assumes that landmarks are scattered across the workspace, that robot control and sensing are perfect within the fields of influence of these landmarks, and that control is imperfect and sensing null outside these fields. The satisfaction of these assumptions may require the robot and/or its workspace to be specifically engineered. This leads us to view robot/workspace engineering as a means to make planning problems tractable. The algorithm was implemented in an operational planner and run on many examples. Non-implemented extensions of the planner are also discussed.

Introduction

Robots must deal with errors in control and sensing. Since the general motion planning problem in the presence of uncertainty seems intrinsically hard [2], a promising line of research is to identify a restricted, but still useful subclass of problems that can be solved in polynomial time.

We consider here a class of planning problems in the context of the navigation of a mobile robot. We assume that *landmarks* are scattered across the robot's two-dimensional workspace. Each landmark is a feature that the robot can sense and identify if it is located in some appropriate subset of the workspace [10, 5]. This subset is the "field of influence" of the landmark. We assume that robot control and sensing are perfect in the fields of influence of the landmarks, and that control is imperfect and sensing is null outside any such field. Given an initial region where the robot is known to be, and a goal region where we would like the robot to go, the problem is to plan motion commands whose execution guarantees that the robot will move into the goal and stop there. The motion commands should also prevent the robot from colliding with stationary obstacles placed in the workspace.

We describe an implemented planning method that *backchains non-directional preimages* (weakest preconditions) of the goal, until one preimage encloses the initial region [11, 6]. Each non-directional preimage is computed as a set of directional preimages for critical directions of motion [4, 1]. At every iteration, the intersection of the non-directional preimage with the fields of influence of the landmarks define the intermediate goal from which to backchain. The algorithm takes polynomial time in the total number of landmarks and obstacles. It is complete with respect to the problems it attacks, that is, it produces a guaranteed plan, whenever one such plan exists, and returns failure, otherwise. Interestingly, once a motion plan has been generated, the assumption that control and sensing are perfect in landmark areas can be relaxed to some extent without losing plan guaranteedness.

Previous methods to compute motion plans under uncertainty include the non-implemented algorithms of [4, 3], which take exponential time in the size of the input problem, and the implemented algorithm described of [8], which is both exponential and incomplete. The polynomiality and completeness of our algorithm derive from the combination the two notions of a landmark and a non-directional preimage. Since landmarks require the workspace to include appropriate features and the robot to be equipped with sensors able to detect them, this combination is an illustration of the interplay between engineering and algorithmic complexity in robotics. As software becomes more critical in modern robots, the importance of this interplay will increase. Robots will have to be designed with the tractability of their programming in mind.

Problem Statement

The robot is a point moving in a plane, the *workspace*, containing stationary forbidden circular regions, the *obstacle disks*. The robot can move in either one of two control modes, the *perfect* and the *imperfect* modes.

The perfect control mode can only be used in some stationary circular regions, the *landmark disks*, model-

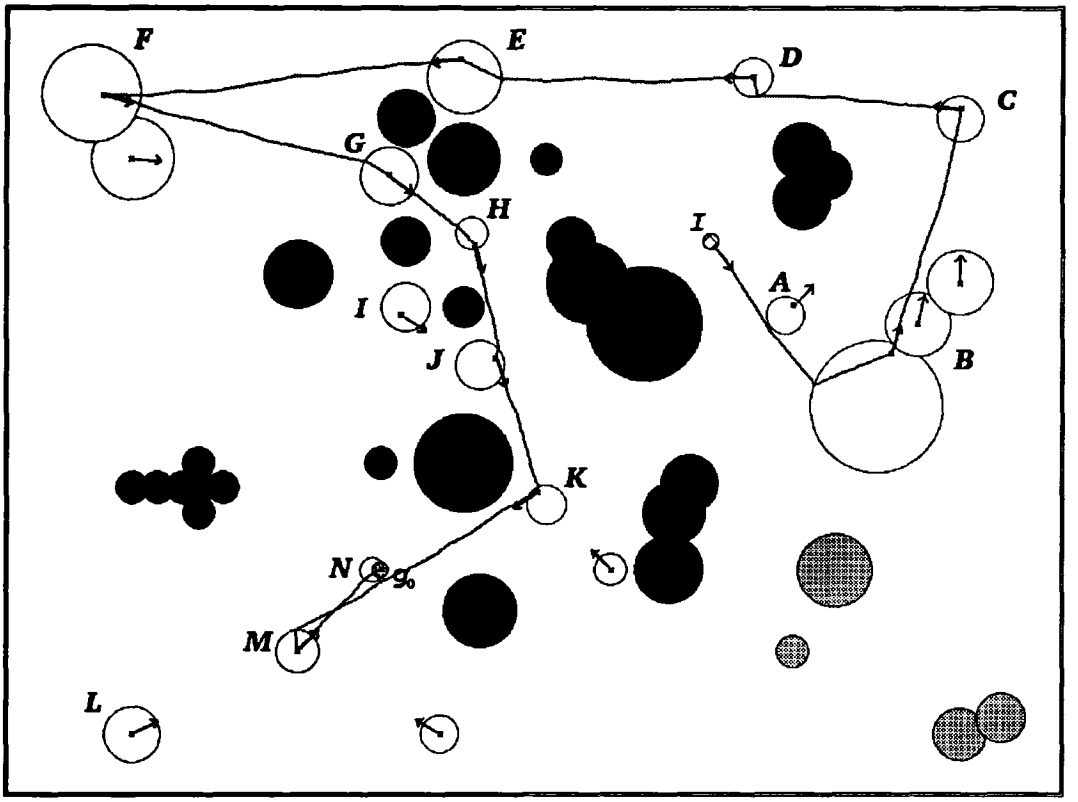

Figure 1: Example of a planning problem

ing the fields of influence of the landmarks. These disks have null intersection with the obstacle disks. When the robot is in a landmark disk, it knows its position exactly and it has perfect control over its motions. Some disks may intersect each other, creating larger areas, called *landmark areas*, through which the robot can move in the perfect control mode. A motion command in the perfect control mode is called a *P-command*.

A motion command in the imperfect control mode, called an *I-command*, is described by a pair $(d, \mathcal{L})$, where $d \in S^1$ is a direction in the plane (the *commanded direction*) and $\mathcal{L}$ is a set of landmark disks (the *termination set*). This command can be executed from anywhere in the plane outside the obstacle disks. The robot follows a path whose tangent at any point makes an angle with the direction d that is no greater than some prespecified angle θ (the *directional uncertainty*). The robot stops as soon as it enters a landmark disk in $\mathcal{L}$.

The robot has no sense of time, which means that the modulus of its velocity is irrelevant to the planning problem.

At planning time, the initial position of the robot is known to be anywhere in a specified *initial region* $\mathcal{I}$ that consists of one or several disks. Each initial-region disk may be disjoint from the landmark areas, or it may overlap some of them, or it may be entirely contained in one of them. The robot must move into a given *goal region* $\mathcal{G}_0$, which is any subset of the workspace whose intersection with the landmark disks is easily computable. The problem is to generate a *guaranteed motion plan*, i.e., an algorithm made up of I- and P-commands whose execution guarantees that the robot will be in $\mathcal{G}_0$ when the execution of the plan terminates. The robot is not allowed to collide with any of the obstacle disks.

This problem is a simplification of a real mobile-robot navigation problem, but it captures its most important aspects [9].

In the sequel we assume that there are ℓ disks (landmarks and obstacles) in total.

Example

Fig. 1 illustrates the previous statement with an example run using the implemented planner. The workspace contains 23 landmark disks (shown white or grey) forming 19 landmark areas, and 25 obstacle disks (shown black). The directional uncertainty θ is set to 0.09 radian. The initial and goal regions are two small disks designated by $\mathcal{I}$ and $\mathcal{G}_0$, respectively. The white landmark disks are those with which the planner has associated I-commands. The arrow attached to a white disk is the commanded direction of motion of an I-command planned to attain another set of disks. There is at least one arrow per white landmark area not intersecting the

goal.

Execution begins with performing the I-command attached to $\mathcal{I}$. When the robot reaches a landmark disk in the termination set of this command, it is guaranteed that a P-command is attached to this disk to attain a point in the current landmark area that is either a goal point (if the goal region intersects this landmark area) or such that an I-command is associated with it (the arrows shown in the figure are drawn from such points). In the first case, plan execution terminates when the goal point is attained. In the second case, the I-command is executed, and so on.

The figure also shows the path produced by a sample execution of the plan. This path first takes the robot from the initial region to the landmark area designated by B. From there, it successively attains and traverses the landmark areas marked C, D, E, F, G, H, J, K, M, and N. The P-command associated with N takes the robot to $\mathcal{G}_0$ where it stops. Due to control errors, other executions of the same plan could produce other paths traversing different landmark disks.

Directional Preimage

Consider a goal region $\mathcal{G}$. We define the *kernel* of $\mathcal{G}$ as the largest set of landmark disks such that, if the robot is in one of them, it can attain the goal by executing a single P-command. Thus, the kernel of $\mathcal{G}$, denoted by $K(\mathcal{G})$, is the set of all the landmark areas having a non-zero intersection with $\mathcal{G}$. The disks in $K(\mathcal{G})$ are called the *kernel disks*. The other landmark disks are called the *non-kernel disks*.

The *directional preimage* of $\mathcal{G}$, for any given commanded direction of motion d, is the region $P(\mathcal{G}, d)$ defined as the largest subset of the workspace such that, if the robot executes the I-command $(d, K(\mathcal{G}))$ from any position in $P(\mathcal{G}, d)$, then it is guaranteed to reach $K(\mathcal{G})$ and thus to stop in $K(\mathcal{G})$. From the entry point in the kernel, the robot can attain $\mathcal{G}$ by executing a P-command.

Fig. 2 shows an example of a directional preimage: the black disks are obstacles; all other disks are kernel landmark disks; the preimage is striped. $P(\mathcal{G}, d)$ consists of one or several connected subsets bounded by circular segments (*arcs*) and straight segments (*edges*). Each arc is a subset of the boundary of a kernel disk or an obstacle. Let the *left ray* (resp. *right ray*) of a kernel disk L be the half-line tangent to L erected from the tangency point in the direction pointed by $\pi+d+\theta$ (resp. $\pi+d-\theta$), with L on its right-hand side (resp. left-hand side). The *right ray* (resp. *left ray*) of an obstacle disk is defined in the same way, but with orientation $\pi+d+\theta$ (resp. $\pi+d-\theta$), with the obstacle on its right-hand side (resp. left-hand side). Each edge of $P(\mathcal{G}, d)$ is contained in the right or left ray of some kernel disk or obstacle disk. One extremity of the edge is the tangency point of the ray; the other extremity is the first intersection point of the ray with another kernel disk, an obstacle disk, or another erected ray. If two edges share the same endpoint, they form a *spike*.

Lemma 1 *The boundary of a directional preimage has size $O(\ell)$. It is computed in $O(\ell \log \ell)$ time and $O(\ell)$ space.*

Non-Directional Preimage

Consider the directional preimage $P(\mathcal{G}, d)$ when d varies continuously over S^1. The topology of the preimage and/or the topology of its intersection with the initial-region disks and the non-kernel landmark disks change at a finite number of critical directions corresponding to events caused by the motion of an edge or a spike of the preimage relative to the kernel landmark disks, the obstacle disks, the initial-region disks, and the non-kernel landmark disks. We call the critical directions where the topology of the directional preimage does change the *D-critical* directions and those where the topology of the intersection of the directional preimage with the initial-region disks (resp. the non-kernel landmark disks) changes, other than the D-critical directions, the *I-critical* (resp. *L-critical*) directions. Fig. 3 illustrates the D-critical directions caused by kernel landmark disks. See [9] for a description of the other critical directions. We assume that the various disks are in general position, i.e., no two events occur simultaneously.

Lemma 2 *There are $O(\ell^3)$ D-critical directions, $O(\ell^2)$ I-critical directions, and $O(\ell^3)$ L-critical directions.*[1]

Let $(d_{c_1}, \ldots, d_{c_p})$ be the cyclic list of all critical directions in counterclockwise order and $I_1, \ldots, I_p$ be the regular intervals between them, with $I_i = (d_{c_i}, d_{c_{i+1 (\mathrm{mod}\ p)}})$. For any interval I_i, let d_{nc_i} be any direction in I_i. In order to characterize all the directional preimages of $\mathcal{G}$ and their intersection with $\mathcal{I}$ and the non-kernel disks, it suffices to compute $P(\mathcal{G}, d)$ for all $d \in \{d_{nc_1}, d_{c_1}, d_{nc_2}, \ldots, d_{c_p}\}$. The set $NP(\mathcal{G})$ of all these directional preimages is called the *non-directional preimage* of $\mathcal{G}$ [4].

To compute the non-directional preimage we first construct the directional preimage of $\mathcal{G}$ for an arbitrary direction $d_\alpha \in S^1$. The rest of the computation is an alternation of event scheduling and event processing phases.

Lemma 3 *The computation of the non-directional preimage takes $O(\ell^3 \log \ell)$ time and $O(\ell^3)$ space.*

Planning Method

If the initial region $\mathcal{I}$ is not contained in the goal region $\mathcal{G}_0$, the planner first computes the kernel $K(\mathcal{G}_0)$. If $K(\mathcal{G}_0) = \emptyset$, the planner returns failure. Otherwise, it associates a P-command to reach a goal point with every landmark disk in this kernel. If $\mathcal{I} \subset K(\mathcal{G}_0)$ the planner returns success.

[1] We conjecture, however, that there are only $O(\ell^2)$ critical directions in total.

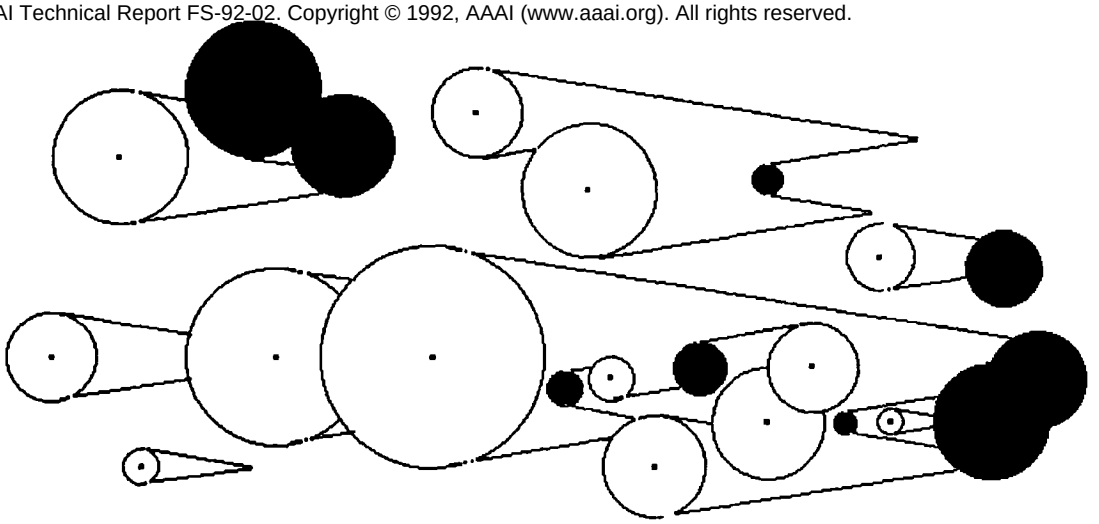

Figure 2: A directional preimage

Assume that $\mathcal{I} \not\subset K(\mathcal{G}_0)$. The planner then computes the non-directional preimage $NP(\mathcal{G}_0)$. If $NP(\mathcal{G}_0)$ contains a directional preimage $P(\mathcal{G}_0, d)$ that includes $\mathcal{I}$, then the planner attaches the I-command $(d, K(\mathcal{G}_0))$ to $\mathcal{I}$ and returns success. Otherwise, for every landmark area $LA \not\subset K(\mathcal{G}_0)$ that has a non-zero intersection with a directional preimage $P(\mathcal{G}_0, d)$ in $NP(\mathcal{G}_0)$, an "exit point" is arbitrarily selected in $LA \cap P(\mathcal{G}_0, d)$ and the I-command $(d, K(\mathcal{G}_0))$ is attached to this point. (If the same area LA intersects several directional preimages, only one intersection is used to produce the I-command.)

The union of the directional preimages in $NP(\mathcal{G}_0)$ is now recursively considered as a goal $\mathcal{G}_1$. Its kernel $K(\mathcal{G}_1)$ is constructed. By construction, $K(\mathcal{G}_1) \supseteq K(\mathcal{G}_0)$. If $K(\mathcal{G}_1) = K(\mathcal{G}_0)$, the planner terminates with failure since it cannot compute a larger non-directional preimage than $NP(\mathcal{G}_0)$. Otherwise, every landmark area in $K(\mathcal{G}_1) \backslash K(\mathcal{G}_0)$ contains one disk L with an exit point and an I-command attached to it. With every other disk in the landmark area, the planner associates a P-command to reach the exit point in L. If $\mathcal{I} \subset K(\mathcal{G}_1)$ the planner returns success, else it computes the non-directional preimage of $\mathcal{G}_1$, and so on.

During this backchaining process, the set of landmark areas in the kernels of the successive goals increases monotonically. At every iteration, either there is a new landmark area in the kernel, and the planner proceeds further, or there is no new area, and the planner terminates with failure. The planner terminates with success whenever it has constructed a kernel $K(\mathcal{G}_n)$ containing $\mathcal{I}$ or a non-directional preimage $NP(\mathcal{G}_n)$ that includes a directional preimage containing $\mathcal{I}$. Let s be the number of landmark areas. The number of iterations is bounded by s. Thus, $n \leq s$.

Theorem 1 *The planning algorithm is complete and generates optimal guaranteed plans.*[2] *It takes* $O(s\ell^3 + \ell^3 \log \ell)$ *time and* $O(\ell^3)$ *space.*

The computed plan is represented as an unordered set of motion commands distributed over the initial-region disks and landmark disks (see Fig. 1). This kind of representation is reminiscent of the concept of a "reaction plan" introduced in classical AI planning research [12]. If the input problem admits no guaranteed plans, the planner nevertheless constructs such a plan, but with no motion command attached to the initial-region disks. Then the robot can try to enter one of the landmark areas where a command is available, by performing an initial random motion with reflection on the obstacles' boundary. If an unexpected event occurs at execution time, the robot may attempt to reconnect to the plan in the same way.

Experimental Results

The above planning method is implemented in C on a DECstation 5000, along with a robot simulator. We experimented with it on many examples. The example shown in Fig. 1 is one of them. It required the computation of 12 successive non-directional preimages and took about 3.5 minutes of computation time. See [9] for more experimental examples.

Discussion and Extensions

These are non-implemented extensions of the planner:

- *Uncertainty in landmark areas:* Perhaps the less realistic assumption in the problem statement of Section 2 is that control and sensing are perfect in landmark areas. However, this assumption can be partially eliminated. Indeed, a landmark area is included in a kernel

[2] Here a plan is said to be *optimal* if the maximal number of executed motion commands is minimal over all possible guaranteed plans.

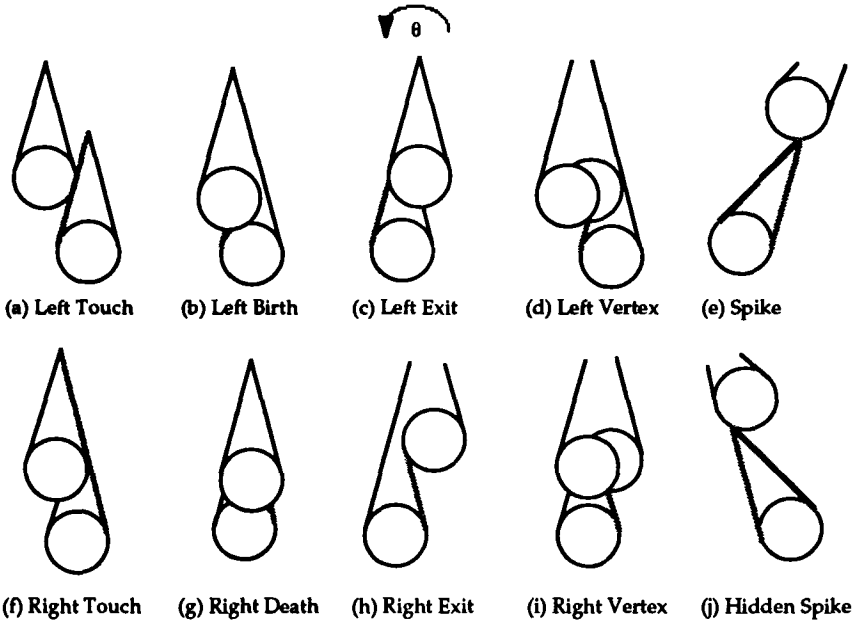

Figure 3: Some D-critical directions

if it has a non-zero intersection with the goal or a previously computed directional preimage. At execution time, if the robot enters this area, it must only be guaranteed that it can move into this non-zero intersection before it executes the next step in the plan. This does not require perfect sensing or control.

- *Landmark and obstacle geometry:* In our current implemention, the landmark and obstacle areas must be unions of circular disks. But the algorithm can be easily adapted to deal with areas described as generalized polygons bounded by straight and circular edges. It can also accept landmark and obstacles areas that touch each other. The degree of any curve traced by a spike remains no greater than 4. If the workspace contains s landmark areas bounded by $O(\ell)$ edges and arcs and the obstacle areas are bounded by $O(\ell)$ edges and arcs, the time complexity of the planner is $O(s\ell^3 + \ell^3 \log \ell)$. Representing landmark and obstacle areas as generalized polygons is a very realistic model for most applications.

- *Compliant motions:* The planning algorithm can be extended to allow compliant motion commands making the robot slide in obstacle boundaries [11]. This extension would enlarge the set of problems admitting guaranteed plans. In general, it would also allow the planner to produce shorter motion plans.

- *Varying directional uncertainty:* Let $P_\theta(\mathcal{G}, d)$ denote the directional preimage of $\mathcal{G}$ for the direction d computed with the directional uncertainty θ. Let us say that a value θ_c of the directional uncertainty is *critical* if there exists $d \in S^1$ such that the intersections of $P_{\theta_c+\epsilon}(\mathcal{G}_i, d)$ and $P_{\theta_c-\epsilon}(\mathcal{G}_i, d)$, for an arbitrarily small ϵ, with the initial-region and the non-kernel disks are topologically different. The algebraic equations defining θ_c can be established by tracking the variations of the non-directional preimage contour when θ varies. The values of θ_c can be exploited to adapt the directional uncertainty used by the planner to the failures and successes met while executing previous plans.

Conclusion

We described a complete polynomial planning algorithm for mobile-robot navigation in the presence of uncertainty. The algorithm solves a class of problems where landmarks are scattered across the workspace. With the possible exception of [7], previous algorithms to plan motion strategies under uncertainty were either exponential in the size of the input problem, or incomplete, or both. Our work shows that it is possible to identify a restricted, but still realistic, subclass of planning problems that can be solved in polynomial time. This subclass is obtained through assumptions whose satisfaction may require prior engineering of the robot and/or its workspace. In our case, this implies the creation of adequate landmarks, by taking advantage of the natural features of the workspace, or introducing artificial beacons, or using specific sensors. This work gives a formal justification to robot/workspace engineering: make planning problems tractable. Engineering the robot and the workspace has its own cost and we would like to minimize it. Our future research will address this issue.

Acknowledgments: This research has been partially funded by DARPA contract DAAA21-89-C0002 and ONR contract N00014-92-J-1809.

References

[1] Briggs, A.J., "An efficient Algorithm for One-Step Planar Compliant Motion Planning with Uncertainty," *Proc. of the 5th Annual ACM Symp. on Computational Geometry*, Saarbruchen, Germany, 1989, pp. 187-196.

[2] Canny, J.F. and Reif, J., "New Lower Bound Techniques for Robot Motion Planning Problems," *27th IEEE Symposium on Foundations of Computer Science*, Los Angeles, CA, 1987, pp. 49-60.

[3] Canny, J.F., "On Computability of Fine Motion Plans," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Scottsdale, AZ, 1989, pp. 177-182.

[4] Donald, B.R., "The Complexity of Planar Compliant Motion Planning Under Uncertainty," *Algorithmica*, 5, 1990, pp. 353-382.

[5] Donald, B.R. and Jennings, J., "Sensor Interpretation and Task-Directed Planning Using Perceptual Equivalence Classes," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Sacramento, CA, 1991, pp. 190-197.

[6] Erdmann, M., *On Motion Planning with Uncertainty*, Tech. Rep. 810, AI Lab., MIT, Cambridge, MA, 1984.

[7] Friedman, J., *Computational Aspects of Compliant Motion Planning*, Ph.D. Dissertation, Report No. STAN-CS-91-1368, Dept. of Computer Science, Stanford University, Stanford, CA, 1991.

[8] Latombe, J.C., Lazanas, A., and Shekhar, S., "Robot Motion Planning with Uncertainty in Control and Sensing," *Artificial Intelligence J.*, 52(1), 1991, pp. 1-47.

[9] Lazanas, A., and Latombe, J.C., *Landmark-Based Robot Navigation*, Tech. Rep. STAN-CS-92-1428, Dept. of Computer Science, Stanford, CA, 1992.

[10] Levitt, T.S., Lawton, D.T., Chelberg, D.M. and Nelson, P.C., "Qualitative Navigation," *Image Understanding Workshop*, Los Angeles, CA, 1987, pp. 447-465.

[11] Lozano-Pérez, T., Mason, M.T. and Taylor, R.H., "Automatic Synthesis of Fine-Motion Strategies for Robots," *Int. J. of Robotics Research*, 3(1), 1984, pp. 3-24.

[12] Schoppers, M.J., *Representation and Automatic Synthesis of Reaction Plans*, Ph.D. Dissertation, Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 1989.