Planning and Executing Robot Assembly Strategies in the Presence of Uncertainty

Bruce R. Donald*

TR 89-1060
November 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This paper describes research done in the Computer Science Robotics Laboratory at Cornell University. Support for our robotics research is provided in part by the National Science Foundation under grant no. IRI-8802390 and by a Presidential Young Investigator award, and in part by the Mathematical Sciences Institute.

Planning and Executing Robot Assembly Strategies in the Presence of Uncertainty

A Lecture Delivered to the American Mathematical Society

$\mathcal{AMS}$ Short Course on

Mathematical Questions in Robotics

96th Annual Meeting of the $\mathcal{AMS}$

Louisville, Kentucky

January 16-17, 1990

Bruce R. Donald

Cornell University

Planning and Executing Robot Assembly Strategies in the Presence of Uncertainty

Bruce R. Donald[1]

Abstract

Robot control systems are subject to significant uncertainty and error. Typical robots are also equipped with sensors—force sensors, kinesthetic position sensors, tactile sensors, vision, and so forth. However, these sensors are also subject to significant uncertainty. Finally, the geometrical models of the robot and the environment (parts, obstacles, etc.) cannot be exact—they are accurate only to manufacturing tolerances, or to the accuracy of the sensors used to acquire the models. Uncertainty is an absolutely fundamental problem in robotics, and plans produced under the assumption of no uncertainty are meaningless. What is needed is a principled theory of planning in the presence of uncertainty. Such a theory must not only be computational, but must also take uncertainty into account *a priori*. In motion planning with uncertainty, we exploit compliant motion—sliding on surfaces—in order to effect a "structural" reduction in uncertainty. Such compliant motion plans can be synthesized from a computational analysis of the geometry of the holonomic constraints.

We will present a precise framework for motion planning with uncertainty. In particular, given geometric bounds on the uncertainty in sensing and control, we develop algorithms for generating and verifying compliant motion strategies that are guaranteed to succeed as long as the sensing and control uncertainties lie within the specified bounds. The first results in this theory begin with Lozano-Pérez, Mason, and Taylor [LMT], with subsequent contributions by Mason [Ma2], Erdmann [E], Donald [D], and others. This research has led to a theoretical computational framework for motion planning with uncertainty, which we explore in this focused survey paper.

[1]This paper describes research done in the Computer Science Robotics Laboratory at Cornell University. Support for our robotics research is provided in part by the National Science Foundation under grant No. IRI-8802390 and by a Presidential Young Investigator award, and in part by the Mathematical Sciences Institute.

Contents

1 A Geometric Approach to Robot Planning **2**
1.1 Introduction 2
1.2 Uncertainty and Compliant Motion 4
1.3 Example: Synthesizing Guaranteed Plans 6
1.4 Dynamic Model 7
1.5 Definitions 7
1.6 Model Error 8
1.7 Backprojections 10
1.8 Error Detection and Recovery 11
1.9 Experimental Results 13

2 Mathematical Model **14**
2.1 Research Issues 14
2.1.1 Guaranteed Strategies Under Bounded Control And Sensing Uncertainty . 14
2.1.2 Uncertain Geometry 15
2.1.3 Tolerating Failure: Error Detection and Recovery 15
2.2 Preimages 16
2.2.1 Preliminaries 16
2.2.2 Sensing Model 17
2.2.3 Termination Predicate and Forward Projections 17
2.2.4 Application: Computing Forward Projections 18
2.2.5 The Role of the Forward Projection in Preimages 22
2.2.6 Definition of Preimages 23
2.2.7 Preimage Approximation Theory 25
2.2.8 Erdmann's Structure Equation 27
2.3 Multi-Step Strategies 28
2.4 Representing Model Error 33
2.4.1 A Simple Example: The Variable-Width Peg-In-Hole 33
2.4.2 Pushing 37
2.4.3 Guaranteed Plans 38
2.5 Error Detection and Recovery 39
2.5.1 The Preimage Structure of EDR Regions 42
2.5.2 Application: Algorithms for EDR Strategies 43
2.6 Advanced Topics 44

3 References **45**

1 A Geometric Approach to Robot Planning

1.1 Introduction

For the past seven years, I have been interested in building robot planning systems that can function at the task-level. A task-level specification of a robot plan might have the form, *Put together this disk rotor assembly.* The planner is given geometric models of the parts, and geometric or analytic models of the robot dynamics. Beyond this, the specification, or input to the planner does not mention the specific kinematic and dynamic constraints that the robot must obey; these are determined by the planner using geometrical computation. Typically, there are additional constraints that the planner must also obey, for example, *Construct a robot plan that is robust in the face of uncertainty and error. Use compliant motion where appropriate to reduce uncertainty.* The goal of a task-level planner is to take a task-level specification and to produce a runnable robot program—one which is fully specified in terms of force-control, kinematics, and dynamics—that can accomplish the task.

Major advances in task-level planning can enable robotics to achieve its full potential in the assembly domain. Today, even existing robots cannot be exploited to their full capacity. For example, assembly tasks require compliant motion; however, compliance requires force-control, and such force-control motion strategies are quite difficult for humans to specify. Furthermore, robot assembly programs are very sensitive to the details of geometry. Finally, reprogramming a general purpose robot for a new assembly task can take time on the order of man-months. For these reasons, we have been working on the automatic synthesis of motion strategies for robots. Much work is required to reduce such current task-level technologies to practice.

Research in task-level planning is often characterized as *theoretical robotics.* There are several reasons for this; the first is that much of the work has been concerned with constructing a *theory* of planning. In other words, the computational problem "task-level planning" is not well-specified. Much of our work lies in specifying the computation precisely. Second, given some sort of decomposition of task-level planning into "planning problems", one is immediately driven to ask, *What are the algorithms for these problems? Can plans, in general, be computed? How efficiently can planning algorithms run?* Historically, the nature of these questions has led researchers to apply tools from theoretical computer science, computational geometry, and algebra.

Recently, a great deal of attention has been focused on a particular robotics problem, called the find-path, or generalized movers' problem. In this problem, we ask the purely kinematic question, can a robot system be moved from one configuration to another, without colliding with obstacles? (See, See figs. 1, 2). This is a nicely-defined mathematical problem, and, after much research, at this point its computational complexity is precisely known.

In fact, the neatness of this problem is deceptive, so much so that this formal problem has even been called "the" motion planning problem. From a task-level viewpoint, there

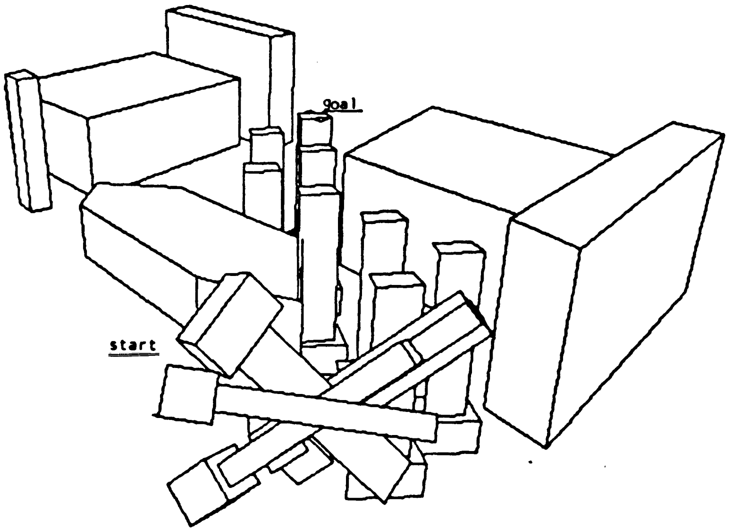

Figure 1: An example of a solution path for the classical find-path, or Movers' problem. This illustration is a "time-lapse" picture of a path found for a hammer-shaped object through a maze of polyhedral obstacles. From [Donald, 87a].

is much hidden in the statement "Can the robot system be moved...?" Specifically, the find-path problem assumes that the robot has a perfect control system that can exactly execute the plan, and that the geometric and analytic models of the robot and obstacles are exact.

In reality, of course, robot control systems are subject to significant uncertainty and error. Typical robots are also equipped with sensors—force sensors, kinesthetic position sensors, tactile sensors, vision, and so forth. However, these sensors are also subject to significant uncertainty. Finally, the geometrical models of the robot and the environment (parts, obstacles, etc.) cannot be exact—they are accurate only to manufacturing tolerances, or to the accuracy of the sensors used to acquire the models. Uncertainty is not a mere engineering detail; in particular, it is characteristically impossible to "patch" these perfect plans in such a way that they will function once uncertainty comes into play. Uncertainty is an absolutely fundamental problem in robotics, and plans produced under the assumption of no uncertainty are meaningless. What is needed is a principled theory of planning in the presence of uncertainty. Such a theory must not only be computational, but must also take uncertainty into account *a priori*. The overlap with exact motion planning algorithms can be stated roughly as follows: exact kinematic planning algorithms provide a computational-geometric theory of holonomic constraints. In motion planning with uncertainty, we exploit compliant motion—sliding on surfaces—in order to effect a "structural" reduction in uncertainty. Such compliant motion plans can be synthesized from a computational analysis of the geometry of the holonomic constraints.

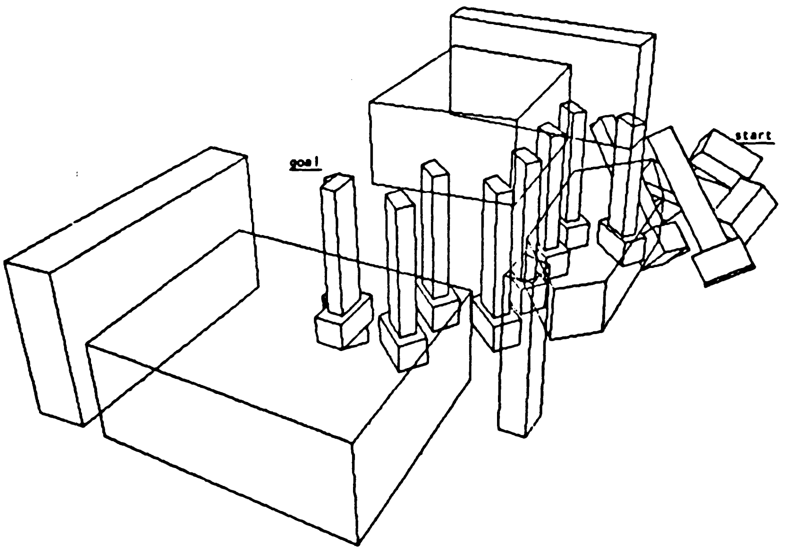

Figure 2: A different view of the solution path for the Movers' problem in fig. 1, with the obstacles "transparent" to allow us to view the rotations better.

1.2 Uncertainty and Compliant Motion

Robots are subject to the following kinds of uncertainty:

1. Inaccuracy and errors in sensing,
2. Inaccuracy and errors in control,
3. Uncertainty about the geometry of the environment.

The last (3) is called "model error", and has received little previous attention. Model error arises because, in general, a robot can have only approximate knowledge of the shape and position of objects in the environment.

We now ask the question:

- How can robots plan and execute tasks (for example, a mechanical assembly using compliant motion) in the presence of these three kinds of uncertainty?

This is perhaps the most fundamental problem in robotics today. We call it the problem of *motion planning with uncertainty.*

In motion planning with uncertainty, the objective is to find a plan which is guaranteed to succeed even when the robot cannot execute it perfectly due to control and sensing uncertainty. With control uncertainty, it is impossible to perform assembly tasks which involve sliding motions using position control alone. To successfully perform assembly tasks, uncertainty must be taken into account, and other types of control must be employed which allow *compliant motion.*

Compliant motion occurs when a robot is commanded to move into an obstacle, but rather than stubbornly obeying its motion command, it complies to the surface of the obstacle. Work on compliant motion[2] attempts to utilize the task geometry to plan motions that reduce the uncertainty in position by maintaining sliding contact with a surface. Plans consisting of such motions can be designed to exploit the geometry of surfaces around the goal to guide the robot. By computing "preimages"[3] of a geometrical goal in configuration space, guaranteed strategies can be synthesized geometrically: We call this a *geometrical theory of planning.* The first results in this theory begin with Lozano-Pérez, Mason, and Taylor (or [LMT]), with subsequent contributions by Mason [Ma2], Erdmann [Erdmann] and Donald [D]. This research has led to a theoretical computational framework for motion planning with uncertainty, which we denote [LMT,E,D]. See [Buc, EM, Bro, CR, Can89a, FHS, LLS] for other allied work.

The [LMT,E,D] framework begins by observing that the use of active compliance enables robots to carry out tasks in the presence of significant sensing and control errors. Compliant motion meets external constraints by specifying how the robot's motion should be modified in response to the forces generated when the constraints are violated. For example, contact with a surface can be guaranteed by maintaining a small force normal to the surface. The remaining degrees of freedom (DOF)—the orthogonal complement of the normal-space—can then be position-controlled. Using this technique, the robot can achieve and retain contact with a surface that may vary significantly in shape and orientation from the programmer's expectations. Generalizations of this principle can be used to accomplish a wide variety of tasks involving constrained motion, e.g., inserting a peg in a hole, or following a weld seam. The specification of particular compliant motions to achieve a task requires knowledge of the geometric constraints imposed by the task. Given a description of the constraints, choices can be made for the compliant motion parameters, e.g., the motion freedoms to be force controlled and those to be position controlled. It is common, however, for position uncertainty to be large enough so that the programmer cannot unambiguously determine which geometric constraints hold at any instant in time. For example, the possible initial configurations for a peg in hole strategy (see fig. 3) may be "topologically" very different, in that different surfaces of the peg and hole are in contact. Under these circumstances, the programmer must employ a combined strategy of force and position control that guarantees reaching the desired final configuration from all the likely initial configurations. We call such a strategy a *motion strategy.*

Motion strategies are quite difficult for humans to specify. Furthermore, robot programs are very sensitive to the details of geometry. For this reason, we have been working

[2]See [Ma] for an introduction and survey.

[3]The *preimage* of a goal [LMT] is the set of configurations from which a particular commanded compliant motion is guaranteed to succeed.

on the automatic synthesis of motion strategies for robots.

Note that compliant motion planning with uncertainty is significantly different from motion planning with perfect sensing and control along completely-known configuration space obstacle boundaries [Kou, HW, BK]. The first difference is physical:

- From a practical point of view, the motion-in-contact plans generated under the assumption of perfect control cannot ever be executed by a physical robot using position control alone.

The second difference is combinatorial:

- The planning of motions in contact with perfect control has the same time-complexity as planning free-space motions; that is, it can be done in time $O(n^r \log n)$ for r degrees of freedom and n faces or surfaces in the environment [C1]; the exponent is worst-case optimal. However, prior to [Donald 87b,88a], there are no upper bounds for planning compliant motions *with* uncertainty. However, for r fixed at 3, the problem is hard for non-deterministic exponential time [CR]. We showed in [Donald 87b,88a] that the planar case, somewhat surprisingly, turns out to be tractable.

The physical difficulty in fact motivates our work. This experimental viewpoint is manifest in [Donald 87b,88b; Jennings, Donald and Campbell]. It is also possible to adopt a more theoretical perspective; in [Donald 88a; Briggs 89] we concentrate on the geometric and combinatorial aspects of the problem.

1.3 Example: Synthesizing Guaranteed Plans

Error Detection and Recovery (EDR) strategies arose as a response to certain inadequacies in the "guaranteed success" planning model. Hence, in order to study EDR, it is first necessary to understand the structure of guaranteed compliant motion strategies, and how they may be synthesized. To this end, in these notes, we wish to give a flavor for the kinds of issues that arise in planning guaranteed motion strategies under uncertainty. We examine a very special case—the planar polygonal case. (That is, the workspace environment ("parts" and "obstacles") are polygonal). First, we will briefly develop a simple dynamic model that is adequate for this situation. Next, we carefully define the computational problem of synthesizing planar guaranteed strategies under uncertainty in sensing and control, and generalize our definition to include uncertainty in the shape of the parts and obstacles. Finally, we hint at the types of computational techniques employed.

For this very simple planning problem, we find that motion plans with a simple structure suffice. This permits us to illustrate by a specific example the situation that we investigated in [Donald 87b, 88a, 88b, 89] for fairly arbitrary plans. Together with the problem of constructing guaranteed compliant motion strategies under uncertainty, we will consider a rather restricted class of strategies, namely those that terminate by sticking on a surface.

1.4 Dynamic Model

Compliant motion is only possible with certain dynamic models. We will employ the generalized damper model [W, Ma]. We assume that the environment is polyhedral, and that it describes the configuration space of the robot, so that the robot is always a point. The planned path consists of r successive motions in directions $v_1, \ldots, v_r$. Each motion terminates when it sticks, due to coulomb friction, on some surface in the environment. Because of control uncertainty, however, the robot cannot move with precisely velocity v_i on the i^{th} motion. Instead, it moves with velocity v_i^{free}, which lies in a cone of velocities $B_{ec}(v_i)$ about v_i. The boundaries of the cone form an angle of ϵ_c with v_i. ϵ_c is called the *control uncertainty*, and $B_{ec}(v_i)$ the *control uncertainty cone about* v_i. It is, in fact, equivalent to regard ϵ_c as specifying that v_i^{free} lies within a ball about v_i in velocity space.

For a compliant motion, the robot moves along an obstacle surface with a sliding velocity v_i^{slide} which is the projection onto the surface of the obstacle of some v_i^{free} in $B_{ec}(v_i)$. Under generalized damper dynamics, the motion of a polyhedral robot without rotations is completely specified by the motion of its reference point in configuration space. See fig 3.

The i^{th} motion terminates by sticking on a surface when the velocity v_i^{free} in $B_{ec}(v_i)$ points into the negative coulomb friction cone on a surface (see fig. 4). Thus sticking on a surface can be non-deterministic. We will assume that motion i can terminate on any reachable surface for which some velocity $v_i^{free} \in B_{ec}(v_i)$ is inside the negative friction cone. Sticking termination is motivated by the fact that a robot with a force-sensing wrist can easily recognize sticking and robustly terminate the motion.

To test whether sticking is possible on some set of (say, goal) edges, we simply perform a geometric cone intersection on each edge. Sticking is possible when the intersection of the cone of velocity uncertainty and the negative friction cone have a non-trivial intersection. Since determining the possibility (or necessity) of sticking reduces to a simple cone intersection, which may be done in constant time per edge, in this preface we will focus on the more difficult issue of computing reachability. Representing friction in our planar polygonal configuration space is easy; see fig. 4. However, the more general question of representing friction in configuration spaces with rotations is subtle; see [Erdmann, BRS].

While robust implementation of generalized damper dynamics is still a research issue, in our robotics laboratory we have recently implemented an experimental force-control system with this dynamic model to test our geometrical planning theories.

1.5 Definitions

We will regard the goal region G as a polyhedral region in configuration space. Since in general we cannot precisely know the initial configuration of the robot, we will also assume that the start region R is some polyhedral region in configuration space.

We now pose three problems (see fig. 6):

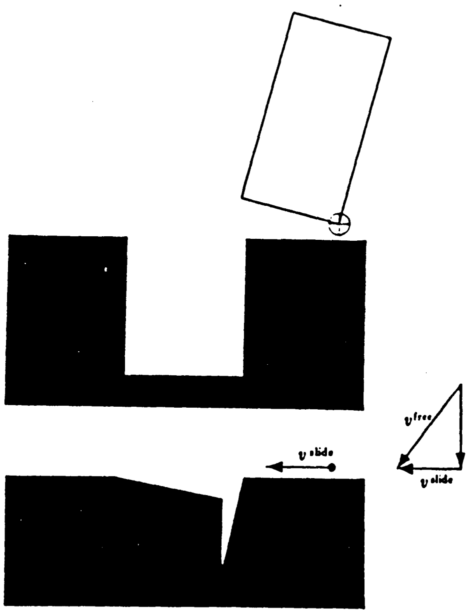

Figure 3: (a) Peg in hole environment. (b) Configuration space, showing the motion of the reference point during compliant motion.

Problem 1: One-Step Compliant Motion Planning with Uncertainty. Given a polyhedral start region R, a polyhedral environment $\mathcal{P}$ of n vertices, control uncertainty ϵ_c, coefficient of friction μ, and a polyhedral goal G, find one commanded motion direction v such that under v, all possible motions from R terminate by sticking in G.

Problem 2: One-Step Compliant Motion Verification. Given $(R, \mathcal{P}, \epsilon_c, \mu, G)$ and v, verify that under v, all possible motions from R terminate by sticking in G.

Problem 3: Compliant Motion Planning with Uncertainty Given $(R, \mathcal{P}, \epsilon_c, \mu, G)$, and an integer r, find a sequence of r motions such that each motion terminates in sticking, and the final motion terminates in the goal. Or, if no such r-step strategy exists, then say so.

1.6 Model Error

We now introduce model error into the picture. How can compliant motion strategies be synthesized in the presence of sensing, control, and geometric model error, such that the strategies are guaranteed to succeed so long as the errors lie within the specified bounds? As an example, consider a peg-in-hole assembly with sensing and control uncertainty, with toleranced parts. We wish to synthesize a compliant motion strategy that is guaranteed to succeed so long as the parts lie within the specified tolerances, and the sensing and control errors lie within the specified bounds.

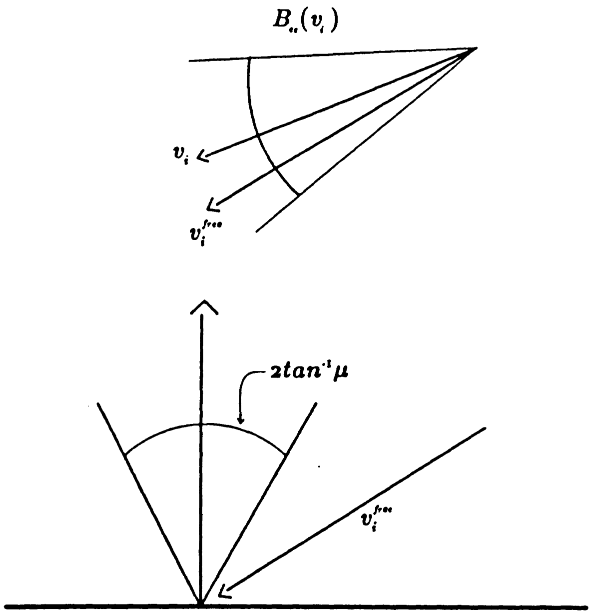

Figure 4: The control uncertainty cone B_{ec} about a nominal commanded velocity v_i. The coulomb friction cone edge makes an angle of $\tan^{-1}\mu$ with the normal. The velocity v_i^{free} will result in sliding on the surface.

We can now state our fourth problem. We parameterize our family of geometries by the parameter space $\alpha_1, \ldots, \alpha_k$, so that $\mathcal{P}(\alpha_1, \ldots, \alpha_k)$ denotes a particular geometry, and each α_i lies within some given interval. Note that this parameterization need not be "continuous". We wish to find a plan which will succeed for all geometries $\mathcal{P}(\alpha_1, \ldots, \alpha_k)$.

Problem 4: Compliant Motion Planning with Uncertainty and Geometric Model Error Given $(R, \mathcal{P}(\alpha_1, \ldots, \alpha_k), \epsilon_c, \mu, G)$, and an integer r, find a sequence of r motions such that, for all possible values of $\alpha_1, \ldots, \alpha_k$, each motion terminates in sticking, and the final motion terminates in the goal. Or, if no such r-step strategy exists, then say so.

We have begun an attack on this problem by introducing additional dimensions to the configuration space; each dimension represents a way in which the parts could parametrically vary. We termed the product space of the motion degrees of freedom and the geometric model variational dimensions "generalized configuration space" and showed how to compute "preimages" [LMT,E] of a geometrical goal in this generalized configuration space. The preimage of a goal is the set of (generalized) configurations from which a particular commanded compliant motion is guaranteed to succeed. Using this technique, we have developed a framework for computing motion strategies that are guaranteed to succeed in the presence of sensing, control, and geometric model uncertainty. The motion strategies comprise sensor-based gross motions, compliant motions, and simple pushing motions.

1.7 Backprojections

We now sketch, for a specific example, a computational approach to the compliant motion planning problem with uncertainty. In [Donald 87b, 88a;b, 89] we have mounted a more systematic attack using similar, albeit considerably generalized methods.

Erdmann [Erdmann] has shown that in the plane, when G is a single edge of the polygonal environment $\mathcal{P}$, then the one-step verification problem (2) can be done in time $O((n + c)\log n)$, where c is the number of intersections encountered by a planar arrangement algorithm.

Erdmann's algorithm makes use of *backprojections*, which he defined as a simplified case of the [LMT] notion of geometrical preimages. The question of goal reachability from a start region can be reduced to deciding the containment of the start region within the backprojection of the goal.

The backprojection $B_\theta(G)$ of a goal G (with respect to a commanded velocity v_θ^*) consists of those configurations guaranteed to enter the goal (under v_θ^*).[4] That is, the backprojection is the set of all positions from which all possible trajectories consistent with the control uncertainty are guaranteed to reach G. See fig. 5. The terms "preimage" and "backprojection" come from viewing motions as "mappings" between subsets of configuration space. Hence the backprojection of a goal is the set of configurations from which a particular commanded compliant motion is guaranteed to succeed. [LMT] envisioned a back-chaining planner that recursively computes preimages of a goal region. Successive subgoals are attained by motion strategies. Each motion terminates when all sensor interpretations indicate that the robot must be within the subgoal.

See fig. 6. Here is the key point about backprojections: Given $(R, \mathcal{P}, \epsilon_c, \mu, G, v_\theta^*)$, the one-step verification problem (2) reduces to testing set containment, i.e., that

$$R \subset B_\theta(G).$$

Erdmann showed that when G is a single edge of a planar environment $\mathcal{P}$, then $B_\theta(G)$ has size $O(n)$ and can be computed as follows (see fig. 7):

(a) Find all vertices in the environment where sticking is *possible* under v_θ^*.

(b) At each of these vertices, erect two rays, parallel to the two edges of the *inverted* velocity cone $-B_{ec}(v_\theta^*)$.

(c) Compute the arrangement from the environment plus these additional $O(n)$ constraints.

(d) Starting at the goal edge, trace out the backprojection region.

An excellent exposition of Erdmann's algorithm can be found in [E]. With John Canny, we implemented a plane-sweep algorithm for backprojections from general polygonal

[4]The star $*$ denotes the ideal, or perfect control velocity. Henceforth, we will typically identify a commanded motion v_θ^* with its angular direction θ.

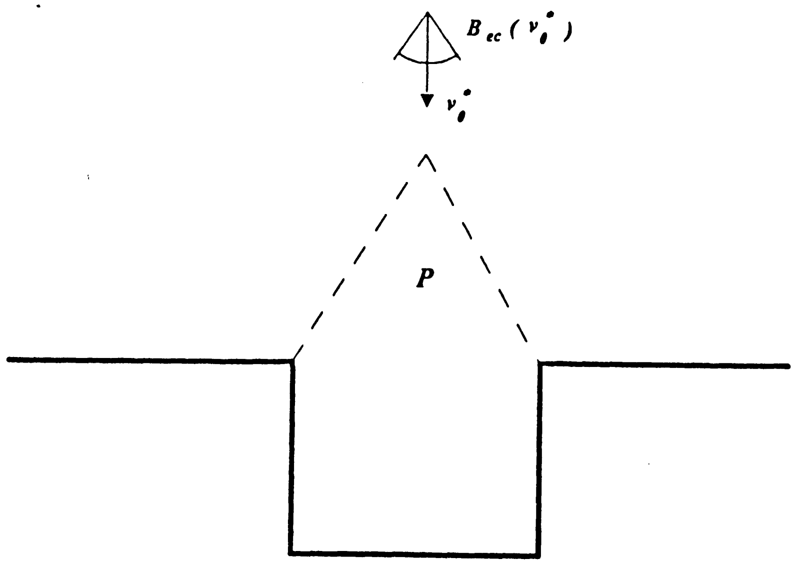

Figure 5: The goal is the region G. Sliding occurs on vertical surfaces, and sticking on horizontal ones. The commanded velocity is v_θ^*, and the control uncertainty is $B_{ec}(v_\theta^*)$. The *backprojection* of G with respect to θ is the region P.

goals. The idea is similar, and interested readers may find details in chapter V. While in general the complexity of these algorithms is $O((n+c)\log n)$, both methods take time $O(n \log n)$ and space $O(n)$ when the goal has $c = O(n)$ intersections with $\mathcal{P}$.

This sketch of the algorithm for verifying a compliant motion strategy gives some flavor for the kind of solution we desire. Consider the remaining problems. The trick to obtaining a computational solution to problem (1) lies in considering all possible backprojections (for all possible θ) simultaneously, and choosing θ that are suitable. In [Donald 87b,88a, Briggs 89] we show that it is possible to do this efficiently, and also provide exact algorithms for problems (3) and (4) as well.

Finally, we must note that there are many variants and facets of the guaranteed compliant motion planning problem; these we address in [Donald 87b, 88a;b, 89, 90]. For these problems, we also gave computational approaches.

1.8 Error Detection and Recovery

A task-level planning system requires a precise theory of Error Diagnosis and Recovery, and a method for generating sensor-based plans with built-in error detection and recovery. We call this the *Error Detection and Recovery (EDR) Theory.* I believe that the chief contribution of our work has been to define and develop a theory of EDR. Perhaps the simplest way to view EDR is as a generalization, or extension, of guaranteed geometrical planning theories. Of course, much work is required to make this notion precise, and

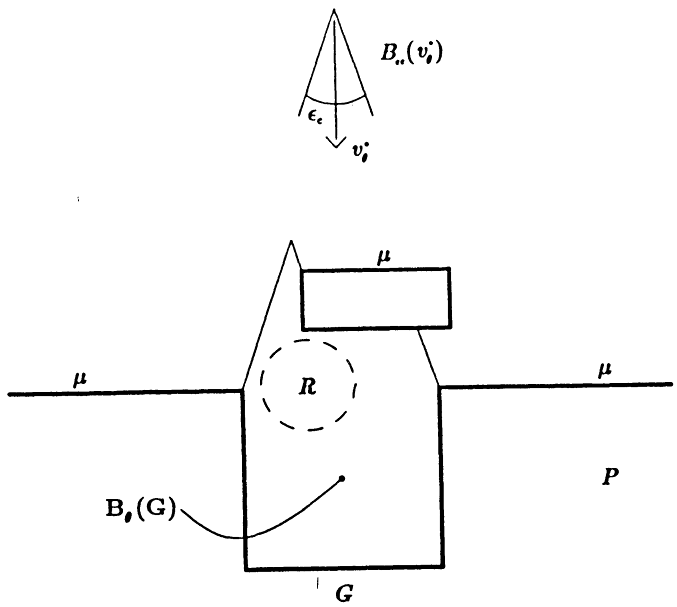

Figure 6: A compliant motion planning problem with goal G, start region R, control uncertainty ϵ_c, obstacles $\mathcal{P}$, and coefficient of friction μ. v_θ^* is a solution which is guaranteed to succeed.

useful; that is the bulk of the work in our monograph [Donald 89], in which in effect, we proposed a "new" theory of planning.

Our approach to EDR observes that there are certain inadequacies with the "guaranteed success" planning model. It is simply not always possible to find plans that are guaranteed to succeed. For example, if tolerancing errors render an assembly infeasible, the plan executor should stop and signal failure. In such cases the insistence on guaranteed success is too restrictive. For this reason we turn to EDR strategies. EDR plans will succeed or fail recognizably: in these more general strategies, there is no possibility that the plan will fail without the executor realizing it. The EDR framework fills a gap when guaranteed plans cannot be found or do not exist: it provides a technology for constructing plans that might work, but fail in a "reasonable" way when they cannot.

The key theoretical issue is: How can we relax the restriction that plans must be guaranteed to succeed, and still retain a theory of planning that is not completely *ad hoc*? We attempt to answer this by giving a constructive definition of EDR strategies. In particular, our approach provides a formal test for verifying whether a given strategy is an EDR strategy. The test is formulated as a decision problem about projection sets in a generalized configuration space which also encodes model error. Roughly speaking, the projection sets represent all possible outcomes of a motion (the *forward projection*), and weakest preconditions for attaining a subgoal (the *preimage*).

Given the formal test for "recognizing" an EDR strategy, we can then test the definition by building a generate-and-test planner. The generator is trivial; the recognizer is an algorithmic embodiment of the formal test. It lies at the heart of this research. A

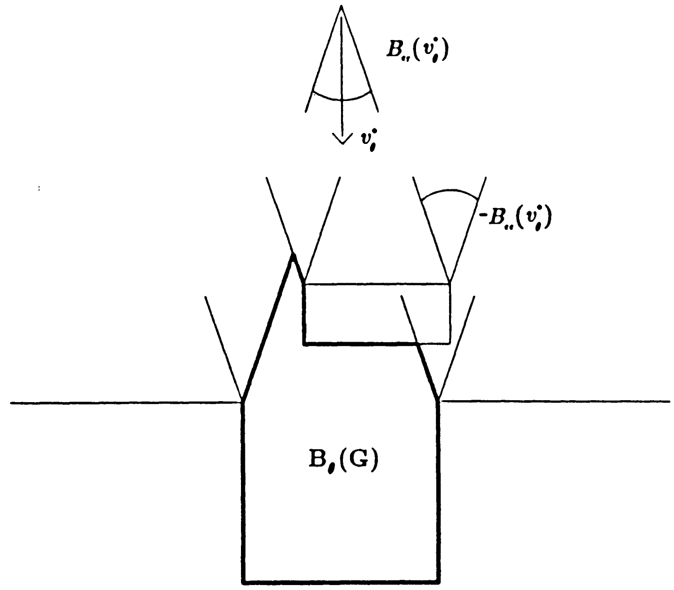

Figure 7: An illustration of Erdmann's algorithm for computing the backprojection $B_\theta(G)$ of a goal G in the plane.

second key component of the planner is a set of techniques for chaining together motions to synthesize multi-step strategies. The planner, called LIMITED, synthesizes robot control programs with built-in sensor-based EDR.

While EDR is largely motivated by the problems of uncertainty and model error, its applicability may be quite broad. EDR has been a persistent but ill-defined theme in both AI and robotics research. We give a constructive, geometric definition for EDR strategies and show how they can be computed. This theory represents what is perhaps the first systematic attack on the problem of error detection and recovery based on geometric and physical reasoning.

1.9 Experimental Results

Since 1987, my group at Cornell has implemented an experimental force-control system on our PUMA robot. In particular, we implemented an approximate generalized damper. Our PUMA 560 robot arm has a VAL-II controller, a Multibus-based 68000 computer system running NRTX (a real-time variant of UNIX) for real-time control, an Astek 6-axis force-sensing wrist, and a Monforte tool-changing gripper with finger positioning and sensing. We closed the servo loop outside of VAL-II to obtain a servo rate we estimate as approximately 20-30 hz. Probably due to the inherent passive compliance in the joints and wrist, the robot can maintain compliant contact between a (gripped) steel bolt and an aluminum surface at speeds of 1-3 inches per second.

This experimental system was sufficiently stable to perform experiments executing the plans generated by LIMITED. In particular, we executed the plans and we performed all the assemblies shown in [Donald 87b, 89], chapter I, figs. 2-13. Again, the interesting thing about these plans is that they were generated by a machine, from task-level descriptions. Thus we were able to build a small slice of a task-level system, right down to the physical robot [Donald 88c, 90; Jennings, Donald and Campbell].

The peg-in-hole plans worked very well, even with tight (1 mm) clearances. The gear-meshing plans also worked, although we found it somewhat tricky to implement the failure mode analysis "oracle" that determines when sticking or breaking contact occurs using force- and position-sensing ([Donald 87b, 88c, 89, 90; Jennings, Donald and Campbell]). For both types of plans, we found that there were unmodelled effects (impact, bouncing) that could have an effect on plan execution, and unmodelled constants (the damping constants, the amount of force to maintain for sliding on a surface, the force termination thresholds) that must be chosen by the runtime executive. A more advanced planner would take these effects and constants into account. A more sophisticated runtime force-control plan executor might adaptively adjust these parameters to a local optimum. In addition, from an experimental viewpoint, much research is needed on the implementation and choice of termination predicates.

2 Mathematical Model

2.1 Research Issues

2.1.1 Guaranteed Strategies Under Bounded Control And Sensing Uncertainty

The gross motion planning problem with no uncertainty has received a great deal of attention recently. In this problem, the state of the robot may be represented as a point in a configuration space. Thus moving from a start to a goal point may be viewed as finding an arc in free space connecting the two points. Since the robot is assumed to have perfect control and sensing, any such arc may be reliably executed once it is found. In particular, given a candidate arc, it may be tested. That is, motion along the arc may be simulated to see whether it is collision free. For example, an algebraic curve may be intersected with semi-algebraic sets defining the configuration space obstacles. In the presence of uncertainty, however, we cannot simply simulate a motion strategy to verify it. Instead, we need some technique for simulating *all* possible orbits, or evolutions of the robot system, under any possible choice of the uncertain parameters. With sensing and control uncertainty, the state of the robot must be viewed as a subset of the configuration space. Motions, then, can be viewed as mappings between these subsets. Of course there are many such subsets! From this perspective, it is clear that a chief contribution of [LMT] has been to identify and give a constructive definition for a privileged class of subsets, called *preimages*, and show that it is necessary and sufficient to search among this class. This framework appears very promising for planning guaranteed motion strategies under sensing and control uncertainty.

2.1.2 Uncertain Geometry

The [LMT] framework assumes no geometric model error. Donald [Don89] reduced the problem of planning guaranteed strategies with sensing, control, *and geometric model* uncertainty[5] to the problem of computing preimages in a (higher dimensional) generalized configuration space $C \times J$, where C is a configuration space representing the motion degrees of freedom for the robot, and J is a parameter space of model error. See sec. 2.3 for details and an example.

2.1.3 Tolerating Failure: Error Detection and Recovery

There are certain inadequacies with the planning model. The insistence that strategies be guaranteed to succeed is too restrictive in practice. To see this, observe that guaranteed strategies do not always exist. For example, in a peg-in-hole problem with model error (fig. 3) there may be no guaranteed strategy for achieving the goal, since the hole may be too small for some model error values. For these values the goal in configuration space does not exist. Because tolerances may cause gross topological changes in configuration space, this problem is particularly prevalent in the presence of model error. More generally, there may be model error values for which the goal may still exist, but it may not be reachable. For example, in a variant of the problem in fig. 6, an obstacle could block the channel to the goal. Then the goal is non-empty, but also not reachable. Finally, and most generally, there may be model error values for which the goal is reachable but not *recognizably* reachable. In this case we still cannot guarantee plans, since a planner cannot know when they have succeeded.

These problems may occur even in the absence of model error. However, without model error a guaranteed plan is often obtainable by back-chaining and adding more steps to the plan. In the presence of model error this technique frequently fails: for example, in the peg-in-hole problem with model error, this technique will not work since no plan of any length can succeed when the hole closes up.

This is why we investigate *error detection and recovery (EDR) strategies*, as introduced in sec. 1.8. Since EDR strategies enjoy the ability to tolerate failure in a way that is robust and mathematically formal, they also provide a theoretical cornerstone for *randomized* or *probabilistic robotic strategies*, as explored by Erdmann [Erd 89] and Goldberg and Mason [GM 89].

[5] We use the terms model error and model uncertainty interchangeably.

2.2 Preimages

2.2.1 Preliminaries

We now elaborate on the definition of preimages subject to the dynamic model introduced in sec. 1.4. After the initial definitions suggested by [LMT,Ma84], this model was proposed by Erdmann [E]. Section 2.2 reviews Erdmann's definitions for the simplest termination predicates. Now, there can be much discussion as to which dynamic model is appropriate for compliant robot control, and what sensor models and termination predicates are more realistic. These are important issues. It is our goal here, to choose a *particular* model, and show how its information content can be formalized, and how, under this model, plans may be analyzed and computed. While in fact I believe that such a model is a reasonable and implementable approximation to actual robot assembly systems, I don't intend to argue that here. It is our purpose here to adopt a simple model, and then pursue the purely mathematical questions relating to reachability, sensor information, recognizability, plan computation, and complexity.

Whereas in section 1 we restricted our consideration to the configuration space $\Re^2$, we now assume our configuration space C is a Lie group, and view control velocities as generators in the Lie algebra. Sec. 1.4 may be formalized by saying that the behavior of our system is specified by a first order differential equation, called the *damper equation*, which linearly relates forces and velocities:

$$\boldsymbol{F} = \boldsymbol{B}(\boldsymbol{v} - \boldsymbol{v}_0) \tag{1}$$

Here, $\boldsymbol{F}$ is the generalized reaction force on the robot, $\boldsymbol{B}$ is the damping matrix of the form $b\boldsymbol{I}$, $\boldsymbol{v}_0$ is the control velocity, and $\boldsymbol{v}$ is the actual velocity the system.[6] One may view the vector $\boldsymbol{F}$ as the net force acting on the robot due to other objects in the robot's environment. The reaction forces due to these objects are specified by coulomb's law. $\boldsymbol{v}_0$ is related to the commanded velocity $\boldsymbol{v}_\theta^*$ as follows: $\boldsymbol{v}_0$ lies in a error ball (or cone) $B_{ec}(\boldsymbol{v}_\theta^*)$ about $\boldsymbol{v}_\theta^*$. Here is how the ball B_{ec} represents error. The possible motion directions that an unobstructed (point) robot[7] may follow when commanded to move in direction $\boldsymbol{v}_\theta^*$ lie within the cone $B_{ec}(\boldsymbol{v}_\theta^*)$. The effect of uncertainty on equation (1) is therefore to substitute for $\boldsymbol{v}_0$ any of the velocities in the error cone about $\boldsymbol{v}_\theta^*$.

As Erdmann [E] points out, to predict the outcome of a commanded motion $\boldsymbol{v}_\theta^*$, we must solve the generalized damper equation (1). Given that the commanded velocity in this equation (1) is subject to uncertainty

$$\boldsymbol{v}_0 \in B_{ec}(\boldsymbol{v}_\theta^*), \tag{2}$$

the solutions are actually *classes* of trajectories. Motion along configuration space surfaces corresponds exactly to motion subject to a holonomic constraint: the damper equation (1) must be solved with contact for the surface in question. See fig. 11.

[6] All forces and velocities are generalized.

[7] Note it suffices to consider point robots in configuration space; see fig. 3.

We will assume that

1. The control velocity $\boldsymbol{v}_0$ can vary arbitrarily within $B_{ec}(\boldsymbol{v}_\theta^*)$.
2. Trajectories must be the result of integrating the first order damper equation (1):

$$\boldsymbol{p}(t_0) = \boldsymbol{p}(0) + \int_0^{t_0} \boldsymbol{v}(t)dt. \tag{3}$$

Definition 2.1 *A trajectory that* satisfies the damper equation with uncertainty relative to a commanded velocity $\boldsymbol{v}_\theta^*$ *is a mapping*

$$\begin{array}{rcl} T : [0, \infty) & \longrightarrow & TC \\ t & \mapsto & (T_p(t), T_v(t)) \end{array} \tag{4}$$

such that at all times t there exists a $\boldsymbol{v}_0(t)$ *satisfying (2) with*

$$\boldsymbol{F}(t) = \boldsymbol{B}(\boldsymbol{v}(t) - \boldsymbol{v}_0(t)), \tag{5}$$

and (3) relates T_p *and* T_v.

Definition 2.2 *Define* $\mathcal{T}(\boldsymbol{v}_\theta^*)$ *to be the set of all trajectories that satisfy the damper equation with uncertainty relative to* $\boldsymbol{v}_\theta^*$.

2.2.2 Sensing Model

Position sensing is specified as follows. For all times t, the sensed position $\boldsymbol{p}^*$ must lie within a ball $B_{ep}(\boldsymbol{p})$ about the actual position $\boldsymbol{p}$.

Now, given the damper equation (1), velocity- and force-sensing are equivalent. Hence we assume that for all times t, the sensed velocity $\boldsymbol{v}^*$ must lie within a ball $B_{ev}(\boldsymbol{v})$ about the actual position $\boldsymbol{v}$.[8]

2.2.3 Termination Predicate and Forward Projections

When the motion is initiated, the termination predicate knows the commanded velocity $\boldsymbol{v}_\theta^*$, the error bounds B_{ec}, B_{ep} and B_{ev}, and the set of goals to be reached, $\{ G_\alpha \}$. In addition, it also knows a bound on the initial conditions; in other words, it knows an initial set R such that the resulting trajectory must originate in R. In this survey, we do not assume that the termination predicate can "remember" past sensor values, although this model has been considered by Mason [Ma2] and Erdmann [E].

We assume that our termination predicate continuously monitors sensing values and time, and that it terminates the motion when the trajectory is guaranteed to have entered a goal. At that point the termination predicate returns which goal $G \in \{ G_\alpha \}$ has been

[8] For technical reasons, we must also assume symmetry, i.e., that $\boldsymbol{p} \in B_{ep}(\boldsymbol{p}^*)$ and $\boldsymbol{p} \in B_{ev}(\boldsymbol{v}^*)$ as well.

achieved. Hence, we view the computational role of the termination predicate as follows: at all times, the termination predicate computes the set of effective interpretations of its sensors, and checks to see whether this set is entirely contained within some goal. When containment is decided, the predicate returns.

In order to model this process mathematically, it is useful to speak of a trajectory as being *consistent* with a sensor value—or the other way around. Hence:

Definition 2.3 *A trajectory T is* sensor consistent at time t with reading $(\boldsymbol{p}^*, \boldsymbol{v}^*)$ *when*

$$(T_p(t), T_v(t)) \in B_{ep}(\boldsymbol{p}^*) \times B_{ev}(\boldsymbol{v}^*). \tag{6}$$

Since the termination predicate knows the controls, and an initial set R in which the motion is guaranteed to originate, it can construct a bound on all possible reachable states. This bound is called the *forward projection.*

Definition 2.4 *The* forward projection at time t of a set of initial conditions R, under commanded velocity $\boldsymbol{v}_\theta^*$ subject to uncertainty (2), *is given by*

$$F_\theta(R, t) = \{\, T(t) \mid T \in \mathcal{T}(\boldsymbol{v}_\theta^*) \text{ and } T_p(0) \in R \,\}. \tag{7}$$

Correspondingly,

Definition 2.5 *The* timeless forward projection *is*

$$F_\theta(R) = \bigcup_{t \geq 0} F_\theta(R, t). \tag{8}$$

2.2.4 Application: Computing Forward Projections

We now consider the computation of forward projections in various simple configuration spaces. Note that forward projections lie in phase space. Now, for a configuration space C, let $\pi : TC \to C$ be the canonical projection map. Buckley [Buc] developed the first algorithms in $C = \Re^3$ for computing the position component of the forward projection $\pi F_\theta(R)$. By the results of Canny and Reif [CR], these sets can have exponential size, and the following decision problem is computationally intractable:

Lemma 2.1 [CR]. *In the configuration space $\Re^3$, given a polyhedral start region R, a polyhedral environment $\mathcal{P}$ of n vertices, control uncertainty ϵ_c, coefficient of friction μ, and a polyhedral goal G, and a commanded motion $\boldsymbol{v}_\theta^*$, deciding containment of a point $\boldsymbol{x}$ in the forward projection $\pi F_\theta(R)$ is $\mathcal{NP}$-hard.*

This immediately implies the following

Corollary 2.2 [CR]. *The One-Step Compliant Motion Verification Problem (problem 2 in sec. 1.5) $(R, \mathcal{P}, \epsilon_c, \mu, G)$ is $\mathcal{NP}$-hard in $\Re^3$.*

However, the planar case turns out to be tractable. By generalizing results of Erdmann [E], Donald and Canny developed forward projection algorithms for that are optimal ($O(n \log n)$) in the plane [Don 89].

We now consider some computational complications of introducing rotations. Note that our discussion generalizes straightforwardly to the configuration space $\Re^3 \times SO(3)$ of rigid body euclidean motions. For the remainder of section 2.2.4, it it convenient to drop the boldface notation for points and tangent vectors to C.

In section 1 we considered the configuration space $\Re^2$ to develop intuition. Suppose we continue to assume the obstacles and peg are polyhedral, but we allow the peg to rotate as well as translate. This lifts the problem into the three dimensional manifold corresponding to the special euclidean group acting on the plane, which we will represent as $C = \Re^2 \times S^1$. This configuration space may be given coordinates (x, y, θ), or, if we desire an "algebraic" configuration space, we may use the coordinates (x, y, u) where $u = \tan \frac{\theta}{2}$. Configuration space obstacles in this space are 3-dimensional semi-algebraic sets (see figs. 8-10, from Brost's work [Brost 89]).[9] These obstacles are generated by one triangular moving "peg" A, and one triangular stationary "obstacle" B. The surfaces bounding these obstacles are ruled, algebraic surfaces; they are simultaneously linear in x and y, and quadratic in u. Each configuration space surface S, or "C-surface" in figs. 8-10 corresponds exactly to a particular holonomic constraint. In particular, each C-surface is "generated" by a particular contact type. That is, each C-surface is generated by the constraint that a vertex of A touch an edge of B, or that an edge of A touch a vertex of B. Note that $C = \Re^2 \times S^1$ is made into a Riemannian manifold by the choice of inner product as follows: If v is a tangent vector at $x \in C$, then $\langle v, v \rangle_x = v \boldsymbol{I}_x v$ where $\boldsymbol{I}_x$ is the moment of inertia tensor at x. This means that $\langle \cdot, \cdot \rangle_x$ is a quadratic form that computes the energy of the system, and hence is a "natural" choice in the sense of Lagrangian dynamics. Each C-surface S hence inherits this inner product. We write the normal to S at x under this inner product as N_x. N_x corresponds to the direction of the pure constraint force for S satisfying the D'Alembert-Lagrange principle.

In free space (the complement of the obstacles), the set of all possible motions simply fans out in a 3D cone. The dynamics of sliding on surfaces, even under generalized damper dynamics, is more complicated. In particular, the dynamics changes from point to point as a s.a. function of orientation. See fig. 11, which shows a typical C-surface S. The dynamics form a differential constraint at each point x on S. The differential constraint is a projection of the 3D velocity uncertainty cone $B_{ec}(v_\theta^*)$ onto the tangent space to S at x. This projection forms a 2D cone; the differential constraint specifies that for a trajectory satisfying the damper equation with uncertainty relative to v_θ^*, and subject to constraint S, that the velocity $T_v(t)$ at $x = T_p(t)$ must lie inside the projected cone. The projection mapping, and hence the cone, varies from point to point. Now, this projection mapping is given by the mechanics of coulomb friction; since friction changes in a non-linear way

[9]These figures are very similar to those in [Brost 89] but were kindly provided to me by the author and are reprinted with permission here.

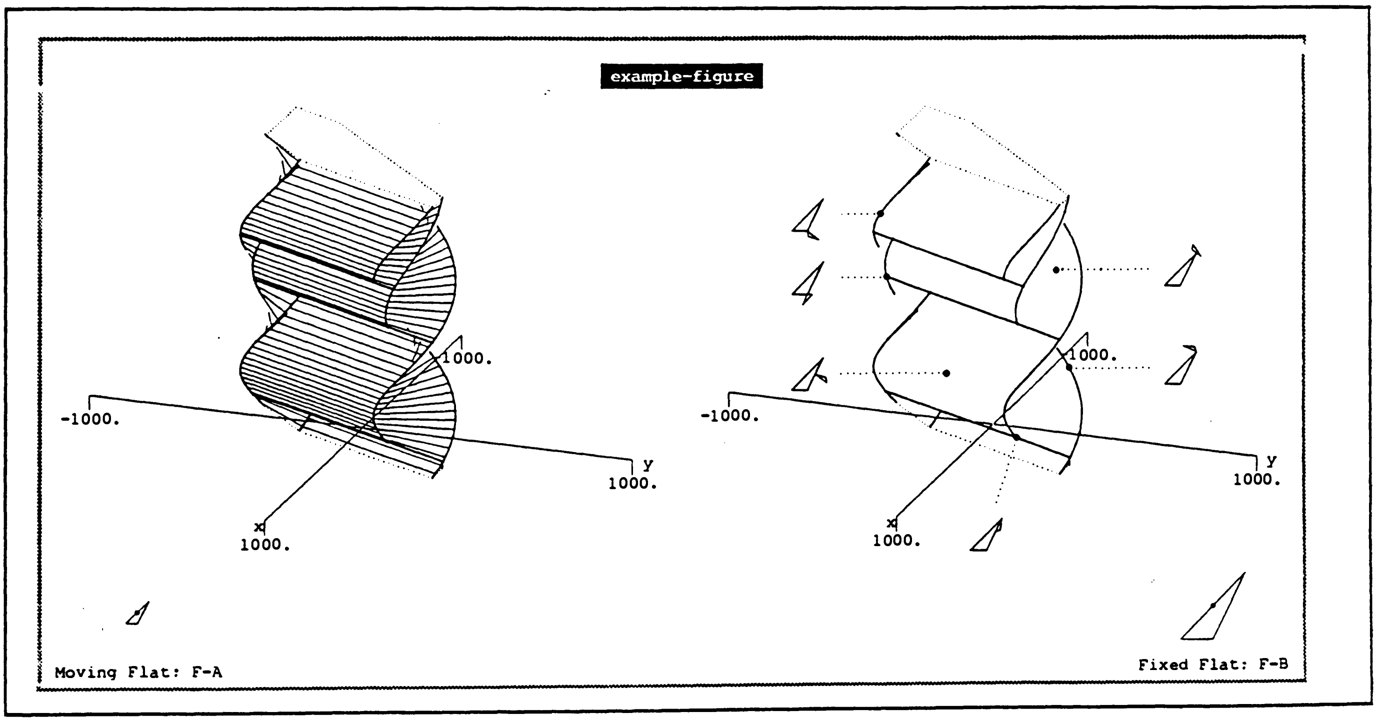

Figure 8: In the configuration space $\Re^2 \times S^1$, shows the configuration space obstacle for a small triangular "peg" A and a large triangular "obstacle" B. This figure shows how each C-surface bounding the obstacle is generated by a contact pair between a vertex (resp. edge) of A and an edge (resp. vertex) of B. Figs. 8-10 reprinted with permission from [Brost 89], and from new figures provided by R. Brost for this article.

over the surface,[10] the projection does also. Erdmann [E] gives the form of the projection map π_x, which is semi-algebraic (s.a.). We note that computing forward projections in $\Re^2 \times S^1$ has both a "dynamical systems" component (finding the outer envelope of all possible evolutions of the dynamical system π_x, subject to a holonomic constraint S), and a "combinatorial" component (there are a finite number of C-surfaces S that we can slide on).

Example: We derive the map π_x for the case of no friction. C is a parallelizable n-manifold, hence $TC \cong C \times \Re^n$. Let $\phi_x : T_eC \to T_xC$ smoothly identify each tangent space with the tangent space at the identity. Now, we may view $x \in S$ as a point in C, and hence identify T_xS with a subspace of T_xC, such that the normal N_x is orthogonal to T_xS. Now, define $p_x : T_xC \to T_xS$ to be the projection of T_xC onto T_xS along N_x. Then it is easily seen that $\pi_x = p_x \circ \phi_x$. With friction, the situation is somewhat more complicated, but the definition of π_x has the same spirit. See Erdmann [E].

[10]That is, the damper equation (1) during contact is non-linear.

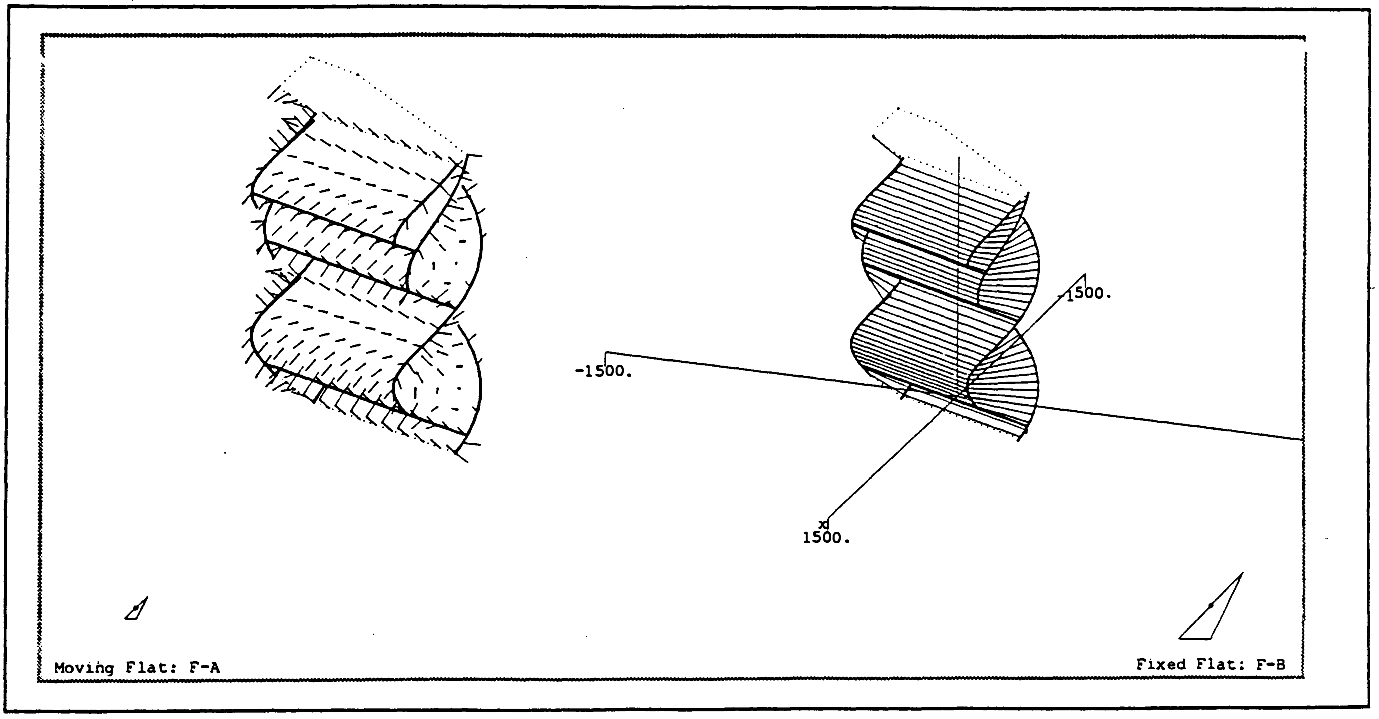

Figure 9: Normals to the C-surfaces. Each C-surface S is ruled and algebraic, and is simultaneously linear in x and y, and quadratic in $u = \tan\frac{\theta}{2}$.

Definition 2.6 *Let S be an algebraic surface in the configuration space C, and $R \subset S$ a set of initial conditions. Let $B_{ec}(v_\theta^*)$ be the control uncertainty cone in the tangent space T_eC to C at the identity. Let $\pi_x : T_eC \rightarrow T_xS$ be a projection map that varies with x and is semi-algebraic. Then the* forward projection $F_\theta(R)|_S$ of R under θ subject to S *is defined as follows:*

$F_\theta(R)|_S$ is the image of all trajectories T with $T_p(0) \in R$, such that at $T(t) = (x, v)$, with controls $v_0(t)$ satisfying (2), one of the following holds:

1. (Maintain contact): $\langle v_0(t), N_x \rangle_x \leq 0$ *and* $v \in \pi_x(B_{ec}(v_\theta^*))$, or
2. (Break Contact): $\langle v_0(t), N_x \rangle_x \geq 0$ *and* $v = v_0(t)$.

The forward projection with time subject to S is defined analogously. We can now state

Problem 5. Suppose S is a ruled, quadric C-surface in the configuration space $\Re^2 \times S^1$, as described above. Find combinatorially precise algorithms for computing $F_\theta(R)|_S$, and for deciding containment of a point in this forward projection.

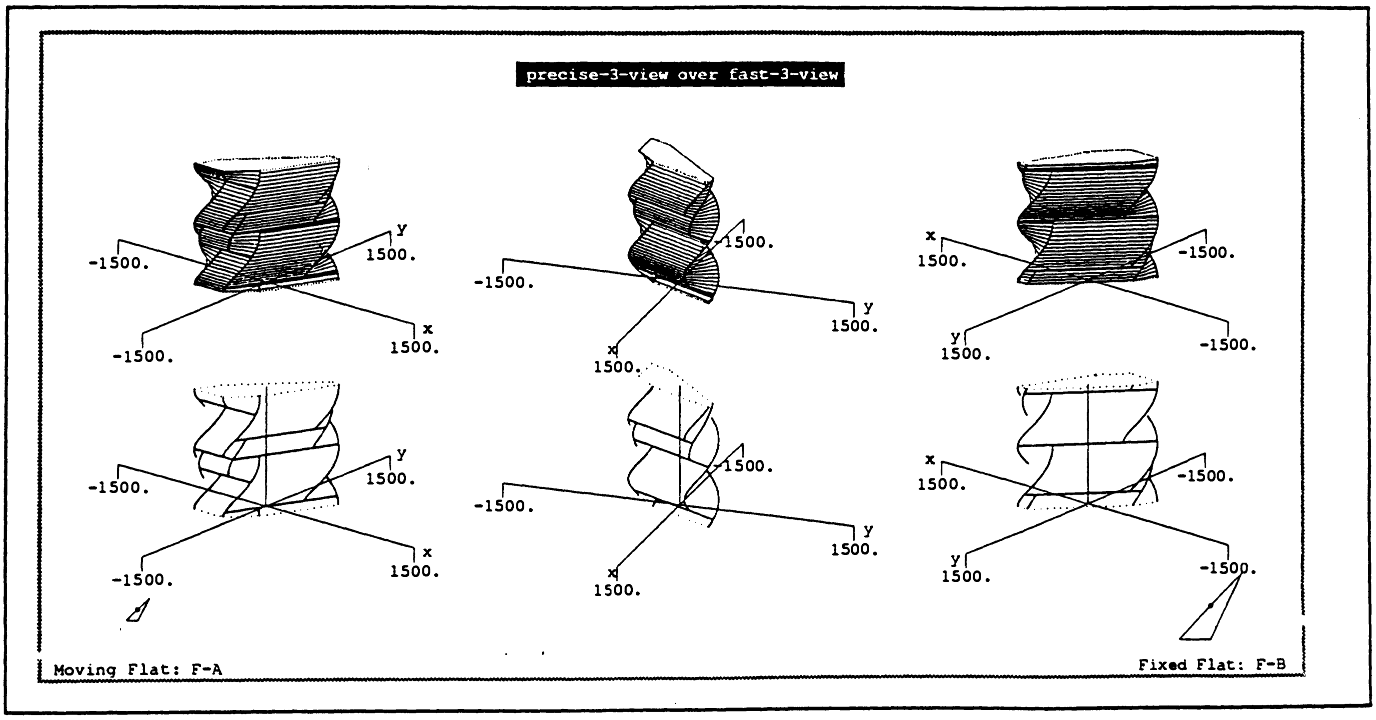

Figure 10: Three different "views" of the configuration space obstacles.

At this time, problem (5) is open, although Erdmann [E] and Brost [Brost 85] have given "numerical" approximation algorithms for backprojections, and Donald [Don 89] has given (similar) "numerical" algorithms for forward projections. In a recent breakthrough, Canny has developed a combinatorially precise algorithm for computing fully-general LMT multi-step strategies in the configuration space $\Re^3$, using the termination predicate in eq. (10). No such algorithms exist yet for $\Re^2 \times S^1$ or $\Re^3 \times SO(3)$, although there are approximation algorithms; some of these apply to restricted subproblems and are precise algorithms [Don88a, Briggs, FHS].

2.2.5 The Role of the Forward Projection in Preimages

Now, knowing where a motion started gives the termination predicate additional constraint on the set of effective sensor interpretations. For example, suppose that the position sensors read $\boldsymbol{p}^*$. *A priori*, this means that the actual position $\boldsymbol{p}$ can lie anywhere in the ball $B_{ep}(\boldsymbol{p}^*)$. However, suppose that the termination predicate has computed the forward projection $\pi F_\theta(R)$; this is an "upper bound" on all reachable positions. If part of the sensor ball $B_{ep}(\boldsymbol{p}^*)$ lies outside the forward projection, then those configurations are clearly excluded as effective sensor interpretations, since they are unreachable. More specifically, we have that

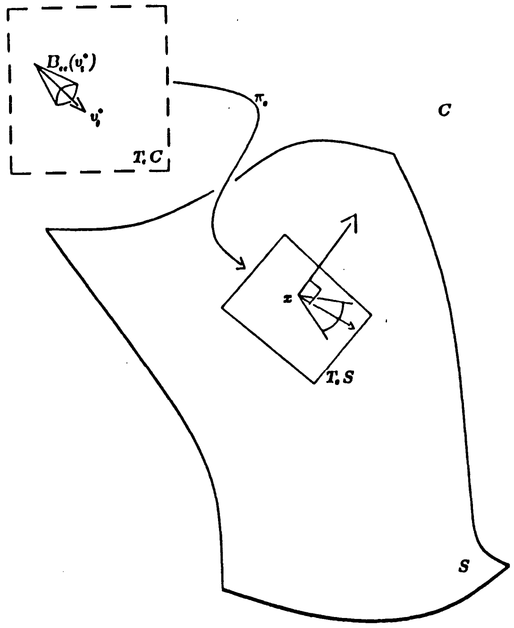

Figure 11: S is an algebraic surface in the lie group C. π_x projects control set B_{ec} from the Lie algebra of C onto the tangent space to S at x. π_x is a semi-algebraic map, (and s.a. parameterized by x). The projected velocity cone is a differential constraint that the velocity of admissible trajectories in TS must obey.

Proposition 2.3 *The effective interpretation of the sensor values $(\boldsymbol{p}^*, \boldsymbol{v}^*)$ at time t is*

$$F_\theta(R,t) \cap B_{ep}(\boldsymbol{p}^*) \times B_{ev}(\boldsymbol{v}^*). \tag{9}$$

2.2.6 Definition of Preimages

We can now define preimages. First, we define the "set of successful controls" $S(R, \{G_\alpha\})$:

Definition 2.7 *For a start region R and a set of goals $\{G_\alpha\}$, Define the set $S(R, \{G_\alpha\})$ as follows. $\boldsymbol{v}_\theta^* \in S(R, \{G_\alpha\})$ if and only if:*

For all trajectories $T \in \mathcal{T}(\boldsymbol{v}_\theta^*)$ consistent with the damper equation under uncertainty, there exists a time $t \geq 0$ such that for all sensor values $(\boldsymbol{p}^*, \boldsymbol{v}^*)$ consistent with $T(t)$, there exists some goal $G \in \{G_\alpha\}$, such that $\pi^{-1}G$ contains the set of effective sensor interpretations (9), that is,

$$F_\theta(R,t) \cap B_{ep}(\boldsymbol{p}^*) \times B_{ev}(\boldsymbol{v}^*) \subset \pi^{-1}G. \tag{10}$$

Then preimages are as follows:

Definition 2.8 *The* Preimage of a set of goals $\{G_\alpha\}$ with respect to a start region R, for a commanded velocity $\boldsymbol{v}_\theta^*$, *is defined as*

$$P_{R,\theta}(\{G_\alpha\}) = \{\boldsymbol{p} \in R \mid \boldsymbol{v}_\theta^* \in S(R, \{G_\alpha\})\}. \tag{11}$$

Lozano-Pérez, Mason, and Taylor [LMT] envisioned a back-chaining planner that would recursively compute preimages of a goal until the start region was contained in a preimage. For example, starting with a goal G_0, one first computes the preimage G_1 of G under θ_1. Then one computes the preimage G_2 of G_1 under θ_2, and so forth, until the start region R is contained in G_n. Then one executes the plan $(\theta_n, \ldots, \theta_1)$, and the termination predicate (10) is guaranteed to recognize entry into each successive subgoal G_i.[11]

One then asks, what characterizes preimage sets, and when are they suitable subgoals for the next level of backchaining? The first result is:

Theorem 2.4 *(Fixed-Point Theorem [LMT]). In order that $P_{R,\theta}(\{G_\alpha\})$ be a suitable subgoal for the next level of backchaining, the equation*

$$R = P_{R,\theta}(\{G_\alpha\}) \tag{12}$$

must hold.

Hence, we may view the problem of guaranteed motion planning with uncertainty as solving the preimage equation (12) for R. The preimage equation is a consequence of the following property of the preimage mapping:

Lemma 2.5 (Erdmann). *The mapping $P_{R,\theta}$ is idempotent, when viewed as a function of R, for a fixed collection of goals.*

Comment: (On distinguishable goal sets). Consider the following preimage equation:

$$P_{R,\theta}(\{G, H\}) = R. \tag{13}$$

We say that the preimage (13) is taken *with respect to* R. (13) means that the preimage of the set of goals $\{G, H\}$, with respect to commanded velocity v_θ^*, is all of R. Note that by def. 2.7, when we have a *set* of goals, the termination predicate must return *which* goal (G or H) has been achieved. This is different from $P_{\theta,R}(G \cup H)$, which means the termination predicate will halt saying "we've terminated in G or H, but I don't know which." The region R appears on both sides of (13) because the preimage depends on knowing where the motion started. Thus solving preimage equations like (13) for R is like finding the fixed point of a recursive equation.

[11]This example does not illustrate branching, or conditional plans.

2.2.7 Preimage Approximation Theory

Preimages exhibit several undesirable qualities that make for computational difficulties. First, in general, maximal preimages are not unique. Second, in some cases, maximal preimages do not even exist. This led Erdmann to pursue an agenda that we may term "preimage approximation theory", in which he attempted to find a preimage-like mathematical object that enjoyed the properties of maximality, uniqueness, and efficient computability. This object he termed the *backprojection*.

Example: Fig. 12 illustrates the difference between backprojections and preimages. Here the radius of position sensing uncertainty is greater than twice the diameter of the hole. Sliding occurs on all surfaces. Furthermore, we assume that the robot has no sense of time (i.e., no clock)—for example, it might be equipped with independent x and y contact sensors that only fire once each.

We formalize this sensor model by assuming that the robot has accurate force-sensing in the y-direction, and infinitely bad force-sensing in the x-direction, and that there is no friction. The start region R can be all of free space. The backprojection $B_\theta(G)$ strictly contains the preimage $P_{R,\theta}(G)$: while all points in the backprojection are guaranteed to reach G, the sensing inaccuracy is so large that the termination predicate cannot tell whether the goal or the left horizontal surface has been reached. Only from the preimage can entry into G be recognized.

We may define the *simple backprojection* $\overline{P_{R,\theta}}$ of a set of goals to be the preimage with perfect position- and velocity- sensing. We observe that every preimage is contained in a simple backprojection. Hence, backprojections only address goal *reachability*, whereas preimages capture reachability *and recognizability*. No notion of termination predicate is needed to formalize the notion of backprojection, although in our case, it is useful pedagogically to think of a termination predicate with "perfect" sensing. Erdmann [E] made precise the notion that "backprojections may be used to approximate preimages." The idea is as follows. First, Erdmann observed that whether or not a point is in a backprojection depends solely on the properties of the point, and not on the properties of neighboring points. That led him to define a "maximal" backprojection $\overline{M_\theta}$ as the union of "simple" backprojections:

$$\overline{M_\theta} = \bigcup_R \overline{P_{R,\theta}}(\{\, G_\alpha \,\}). \tag{14}$$

Note that by idempotency, it suffices to consider only sets R such that $R = \overline{P_{R,\theta}}(\{\, G_\alpha \,\})$. It is clear, therefore, that the union of simple backprojections is also a simple backprojection:

$$\overline{M_\theta} = \overline{P_{\overline{M_\theta},\theta}}(\{\, G_\alpha \,\}). \tag{15}$$

Hence, the absence of termination predicates (or, equivalently, the assumption of perfect termination predicates) makes it possible to define a unique maximal set $\overline{M_\theta}$ such

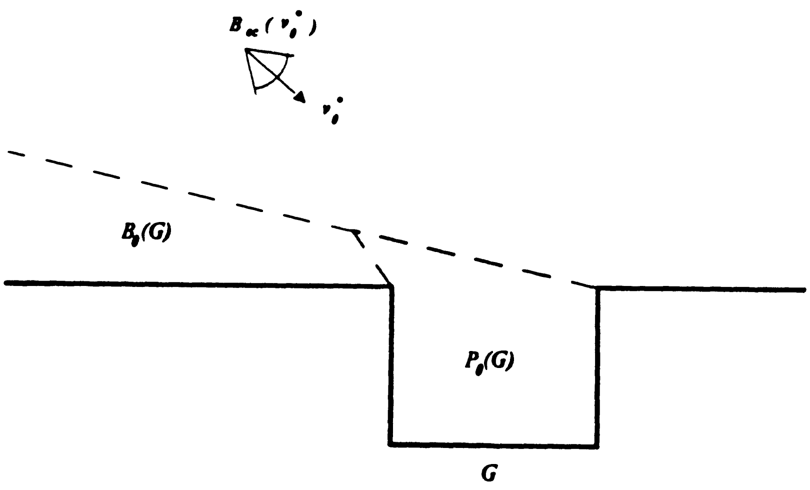

Figure 12: Here, the radius of the position sensing uncertainty ball is twice the width of the hole. Sliding occurs on all surfaces under the control velocities shown. Let the start region R be all of free space. The preimage of the goal under commanded velocity v_θ^* is $P_\theta(G) = P_{R,\theta}(G)$. The backprojection $B_\theta(G)$ strictly contains this preimage: while all points in the backprojection are guaranteed to reach G, the sensing inaccuracy is so large that the termination predicate cannot tell whether the goal or the left horizontal surface has been reached. Only from the preimage can entry into G be recognized.

that under controls v_θ^*, from $\overline{M_\theta}$ we are guaranteed to reach a goal $G \in \{ G_\alpha \}$. Erdmann called the mapping $\theta \times \{ G_\alpha \} \mapsto \overline{M_\theta}$ the *maximal backprojection*, denoted

$$B_\theta(\{ G_\alpha \}) = \overline{M_\theta}. \tag{16}$$

Hence we have

Lemma 2.6 *For all R, θ, and $\{ G_\alpha \}$,*

$$P_{R,\theta}(\{ G_\alpha \}) \subset \overline{P_{R,\theta}}(\{ G_\alpha \}) \subset B_\theta(\{ G_\alpha \}) = B_\theta\Big(\cup_\alpha \{ G_\alpha \}\Big). \tag{17}$$

2.2.8 Erdmann's Structure Equation

Now, Erdmann showed that every preimage $R = P_{R,\theta}(\{ G_\alpha \})$ may be "extended" to an "almost simple" preimage $A(R)$, where $A(R)$ contains R and has the general form

$$A(R) = \pi F_\theta(R) \cap B_\theta(E(R)), \tag{18}$$

where $E(R)$ is a subset of the union of the goals $\cup_\alpha \{ G_\alpha \}$. This says that given R, θ, and $\{ G_\alpha \}$, one may, in principle, "shrink" the goals to compute $E(R)$, and then compute $A(R)$ as in (18).

It remains to define $E(R)$, which Erdmann called the *First Entry Set.*

Definition 2.9 *For a trajectory T, let $\mathcal{S}_T$ be the set of successful termination times for R, θ, and $\{ G_\alpha \}$. That is, $\mathcal{S}_T$ is the set of all times $t \geq 0$ such that for all sensor values $(\boldsymbol{p}^*, \boldsymbol{v}^*)$ consistent with $T(t)$, there exists some goal $G \in \{ G_\alpha \}$, such that $\pi^{-1}G$ contains the set of effective sensor interpretations (eq. (9)).*

Definition 2.10 *Suppose $R = P_{R,\theta}(\{ G_\alpha \})$ is preimage. Define the* First Entry Set $E(R)$ of R *to be the set of all "first entry points" of trajectories that start in R. That is,*

$$E(R) = \{ T_p(t_0) \mid T \in \mathcal{T}(v_\theta^*), T_p(0) \in R, \text{ and } t_0 = \inf(\mathcal{S}_T) \}. \tag{19}$$

Hence, we have that every suitable preimage R is a subset of an "almost simple" preimage" $A(R)$, and $A(R)$ is also a suitable preimage. We also have that $A(R)$, the extension of R, is "maximal" in the sense that A is idempotent. We call eq. (18) the *structure equation* for preimages and backprojections. On a practical note, while computing first entry sets may be in general, as hard as computing preimages, the utility of this definition is that many, easy-to-compute goal-shrinking schemes can quickly be proved correct. That is, a conservative goal-shrinking scheme is correct if it computes a subset G of $E(R)$. Then we may compute a preimage using the structure equation.

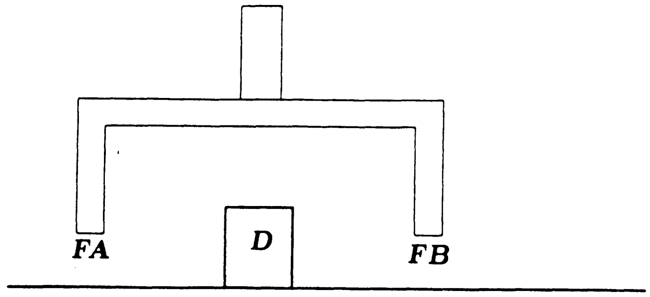

Figure 13: The grasp centering problem. The width of the block D on the table, and the position of the gripper are only known approximately.

2.3 Multi-Step Strategies

Having reduced motion planning under uncertainty to essentially "preimage-theoretic" equations, motion strategies can be synthesized by solving these preimage equations. We now present a worked-out example of a motion plan using preimages. The motion problem is grasp-centering for a robot gripper in the presence of model error. A guaranteed plan is found by solving the preimage equations. The example[12] illustrates the use of the preimage framework to derive a multi-step motion strategy in the presence of model error. The strategy employs time-sensing and force-sensing.

Consider the grasp-centering problem shown in fig. 13. The task is to center the robot gripper over the block D. The gripper can translate but not rotate in the plane. In its start position, the gripper is somewhere over D, such that the bottom of the fingers FA and FB are below the top of D. The width of D is unknown, but must be less than the distance between FA and FB. We assume D is fixed (it cannot be accidentally pushed).

Hence we can regard this as a planning problem with model error. C is taken to be the cartesian plane, and J, the space of model error, is a bounded interval of the positive reals. Our first question is, what does the generalized configuration space $C \times J$ look like? This is easily answered by considering the motion planning problem in fig. 14. The problem is to find a motion strategy for a point robot so that it can achieve a goal exactly

[12]This problem arose in discussions with Tomás Lozano-Pérez, John Canny, and Mike Erdmann. It appears in [Don89].

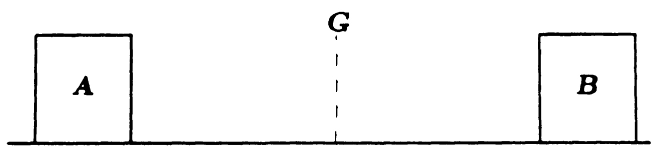

Figure 14: An equivalent problem. A point robot must be navigated halfway between the blocks A and B. The distance between A and B is not known. The robot has force sensing, and a clock. However, it has poor position sensing. We regard C as $\Re^2$ and J, the space of model error, as the bounded interval $(0, d]$ for d positive. The generalized configuration space for this problem is $\Re^2 \times J$, and is shown in fig. 18.

halfway between the blocks A and B. The distance α between A and B is unknown and positive. The point robot is known to start between A and B. Again, the point can translate in the plane. The distance α is the model error parameter. It is easy to see that the problems in figs. 13 and 14 are equivalent. The generalized configuration space for fig. 14 is, in fact, the same as that depicted in fig. 18, as we will see in sec. 2.4.

Next, we assume that the robot has perfect control, perfect velocity sensing, and a perfectly accurate sense of time. However, it has infinite position sensing error. [13]

Now, since the gripper starts over D with the bottom of the fingers below the top of D, and since the robot has perfect control, it suffices to consider the x axis of C. Since the y axis can be ignored, we develop our example in the plane, that is, in the generalized configuration space where C and J are both one-dimensional. This 2D generalized configuration space is shown in fig. 15, which is essentially an x-J cross-section of the full 3D generalized configuration space $\Re^2 \times J$ shown in fig. 18, holding y constant with α constrained to be positive. In fig. 15, L and R are left and right obstacle edge boundaries generated by A and B. The goal is the line in free-space bisecting L and R. The start region T is the triangular region in free-space between L and R. (T is the convex hull of L and R).

[13]This example is easily generalized to non-zero control, time-sensing, and force-sensing error, and finite position-sensing error. This requires giving the goal non-empty interior, however.

Now, since motion across J is not permitted, all motions are parallel to the x axis, that is to say, horizontal in fig. 15. There are only two kinds of motions the planner can command. Let $+$ denote a motion to the right, and $-$ a motion to the left. We assume the robot has perfect control over the magnitude as well as the direction of the commanded velocity.

See fig. 15. Now, if α is a point on the J axis, let E_α be the point on the left obstacle edge L with J coordinate α. We will denote the collection of all such points on L by $\{ E_\alpha \}$. Let S_α denote the maximal line segment within T containing E_α and parallel to G. Formally, if E_α has coordinates (x, α), then S_α is the line segment extending from E_α to (x, d) where d is an upper bound on the distance between A and B. We denote the collection of all lines S_α by $\{ S_\alpha \}$.

At this point we are prepared to derive a motion strategy for centering the grasp, that is, for attaining G from T. The strategy has three steps. The termination conditions for the motions involve time- and force-sensing. Here is the motion strategy in qualitative terms:

Strategy Guarantee-Center

1. *Command a motion to the right. Terminate on the right edge R based on force sensing.*

2. *Command a velocity of known magnitude to the left. Terminate when in contact with the left edge L, using force sensing. Measure the elapsed time of the motion. Compute the distance traversed. This gives exact knowledge of where the motion terminated on L. The effect of this step is to measure the distance α between the blocks.*

3. *Move distance $\frac{\alpha}{2}$ to the right, terminating in G based on time sensing.*

We now derive this strategy by solving the preimage equations for the motion planning problem.

First, note that if the run-time executive knows that the robot is inside a *particular* S_α, then G can be reliably achieved by commanding a motion to the right. Since the robot has perfect control and time sensing, the motion can be terminated after moving distance $\frac{\alpha}{2}$, that is, exactly when the line G is achieved. Using the preimage notation, we write this as

$$P_{+,\{ S_\alpha \}}(G) = \{ S_\alpha \}. \tag{1}$$

Next, we take the collection $\{ S_\alpha \}$ as a set of subgoals, and try to find a motion that can recognizably attain this collection, and, furthermore, can distinguish which S_α the motion achieves. Consider a leftward motion starting from anywhere on the right edge R. The robot does not know where on R the motion starts, however. To recognizably achieve some S_α, such a motion should move leftward, and terminate when force-sensing

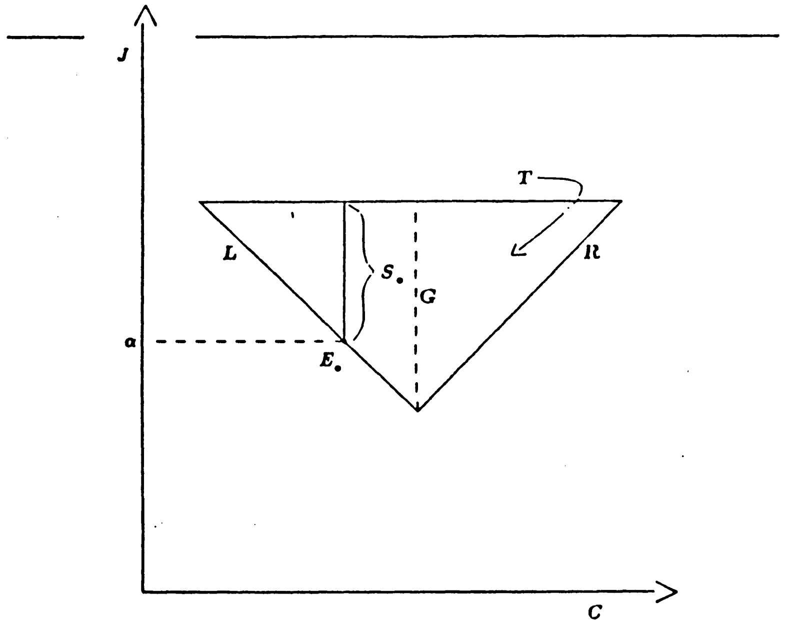

Figure 15: Assuming that the gripper fingers are initially lower than the top of the block D, the y dimension can effectively be ignored. This allows us to examine an x-J cross-section of the 3D generalized configuration space $\Re^2 \times J$ in fig. 18. We treat C as the x axis of motion freedom, yielding a 2D $C \times J$ planning space. L and R are obstacle boundaries in generalized configuration space . The goal is the bisector G between L and R in free-space. The start region T is the triangular region between L and R. E_α is a point on L. S_α is a line in T parallel to G and containing S_α.

indicates that L has been reached. If the termination predicate measures the elapsed time of the motion, and knows the magnitude of the commanded velocity, then it can recognize which point E_α has been reached, and hence which subgoal S_α has been achieved. Writing this down in preimage equations,

$$P_{-,R}(\{ S_\alpha \}) = P_{-,R}(\{ E_\alpha \}) = R. \tag{2}$$

Finally, the right edge R may be achieved from anywhere within the start region T by moving rightward, and terminating when force sensing indicates contact. This is simply

$$P_{+,T}(R) = T. \tag{3}$$

It is instructive to examine the termination conditions for motions (1)–(3). In developing the LMT framework for planning guaranteed strategies, [Erdmann] developed an elegant formalization of the question, "Using sensors and history, when can the termination predicate decide that a motion has recognizably entered a goal G_β?" The answer was as follows. Assume that G_β has been lifted into phase-space. Let R be the start region. The forward projection, $F_\theta(R)$ captures the notion of history: it is all positions and velocities that can be reached given that the motion started in R. At a particular instant t in time, let $B_{ep}(t)$ and $B_{ev}(t)$ be the sets of possible positions and velocities. These are the sensing uncertainty balls about a sensed position and velocity in phase space at time t. Thus sensing provides the information that the actual position and velocity must lie within the set $B_{ep}(t) \times B_{ev}(t)$. The forward projection further constrains the actual position and velocity to lie within $F_\theta(R)$. Thus the termination predicate can terminate the motion as having recognizably reached G_β when

$$F_\theta(R) \cap (B_{ep}(t) \times B_{ev}(t)) \subset G_\beta. \tag{$*$}$$

Now, let $F_\theta(R, t)$ denote the *time-indexed* or *instantaneous forward projection* of R under θ at time t. $F_\theta(R, t)$ denotes the set of positions and velocities that are possibly achievable at elapsed time t, under motion θ, given that the motion started in R. The termination predicate in this case monitors a clock, in addition to position and velocity sensors. In motion (1), only the time-indexed forward projection $F_+(S_\alpha, t)$ is relevant to deciding termination. The motion terminates when $F_+(S_\alpha, t) \subset G$. Motion (3) can be terminated using pure force sensing. It could also be terminated using time, since there exists some t for which $F_+(T, t) = R$. In motion (2), both force sensing and time are required to terminate within a distinguishable E_α. The general form of the termination condition for all three cases is as follows. The termination predicate has the form

$$F_\theta(U, t) \cap (B_{ep}(t) \times B_{ev}(t)) \subset G_\beta$$

for a goal G_β and a start region U. (Assume that all subgoals have been lifted into phase space). In our case, position sensing error is infinite, so $B_{ep}(t)$ is $C \times J$. Let us denote $(C \times J) \times B_{ev}(t)$ by the simpler expression $B_v(t)$. Then the termination conditions for motions (1)–(3) are as follows. For the first motion (3), to terminate, we must have

$$F_+(T,t) \cap B_v(t) \subset R. \tag{4}$$

For the second motion (2) to terminate, we must have

$$F_-(R,t) \cap B_v(t) \subset S_\alpha \tag{5}$$

for some S_α. We think of the termination predicate as "returning" this S_α. Finally, for termination of the last motion (1), we must have

$$F_+(S_\alpha) \cap B_v(t) \subset G, \tag{6}$$

where the S_α in (6) is the same as the one returned by the termination predicate after the second motion as the satisfying assignment for (5).

Finally, note that time is the source of some complexity in this example. This complexity might be removed by employing a distance sensor instead. The output of such a sensor could be modeled as position sensing in J. The sensing action in J would entail measuring the distance between A and B. This relaxes the assumption of no position sensing in the J dimensions, but such modification to the generalized configuration space framework is trivial. With this modification, B_{ep} is simply regarded as a product of a position sensing ball in C and a position sensing set in J.

This concludes the example. We have shown how to derive a multi-step guaranteed motion strategy in the presence of model error. The strategy was derived by solving the preimage equations in generalized configuration space for the motion plan. These preimage equations made the role of time- and force-sensing explicit in deriving conditions for distinguishable termination in a collection of subgoals.

2.4 Representing Model Error

The [LMT] framework assumes no geometric model error. Donald [Don89] reduced the problem of planning guaranteed strategies with sensing, control, *and geometric model* uncertainty to the problem of computing preimages in a (higher dimensional) generalized configuration space, which encodes model error as a kind of nonholonomic constraint. In fact, we used this method implicitly in the previous section 2.3. We new review the reduction by carefully examining some very simple planning problems with model error. These problems in fact motivate the definition of *Error Detection and Recovery (EDR) strategies.* Of course, this does not mean that EDR is limited to situations with model error.

2.4.1 A Simple Example: The Variable-Width Peg-In-Hole

Example 1: Consider fig. 16. There is position sensing uncertainty, so that the start position of the robot is only known to lie within some ball in the plane. The goal is to bring the robot in contact with the right vertical surface of A.

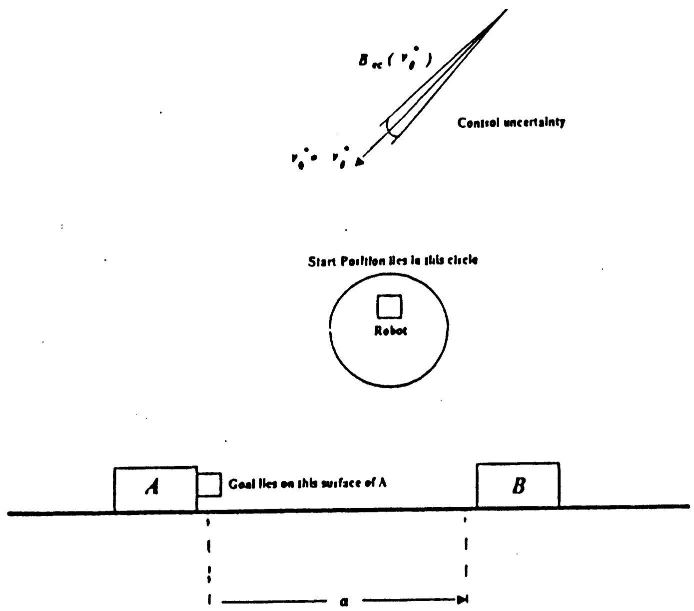

Figure 16: The goal is to bring the robot into contact with the right vertical surface of A. (For example, the "robot" could be a gripper finger). There is position sensing uncertainty, so in the start position the robot is only known to lie within some uncertainty ball. There is also control uncertainty in the commanded velocity to the robot. It is represented as a cone, as shown.

We will simplify the problem so that the computational task is in configuration space. This transformation reduces the planning task for a complicated moving object to navigating a point in configuration space. Consider fig. 17. The configuration point starts out in the region R, which is the position sensing uncertainty ball B_{ep} about some initial sensed position. As usual, to model sliding behavior, we will assume Coulomb friction and generalized damper dynamics, which allows an identification of forces and velocities. Thus the commanded velocity v_0 is related to the effective velocity v by $f = B(v - v_0)$ where f is the effective force on the robot and B is a scalar. Given a nominal commanded velocity v_0^*, the control uncertainty is represented by a cone of velocities (B_{ec} in the figure). The actual commanded velocity v_0 must lie within this cone.

The goal in fig. 17 is to move to the region G. Now, with Coulomb friction, sticking occurs on a surface when the (actual) commanded velocity points into the friction cone. We assume the friction cones are such that sliding occurs (for all possible commanded velocities in B_{ec}) on all surfaces save G, where all velocities stick. We will assume that the planner can monitor position and velocity sensors to determine whether a motion has reached the goal. Velocity sensing is also subject to uncertainty: for an actual velocity v, the sensed velocity lies in some cone B_{ev} of velocities about v.

Now we introduce simple model error. The shape of A and B are known precisely, and the position of A is fixed. However, the position of B, relative to A is not known. B's position is characterized by the distance α. If $\alpha > 0$ the goal is reachable. But if

$\alpha = 0$, then the goal vanishes. No plan can be guaranteed to succeed if $\alpha = 0$ is possible. Suppose we allow α to be negative. In this case the blocks meet and fuse. Eventually, for sufficiently negative α, B will emerge on the other side of A. In this case, the goal "reappears," and may be reachable again.[14] Let us assume that α is bounded, and lies in the interval $[-d_0, d_0]$.

Our task is to find a plan that can attain G in the cases where it is recognizably reachable. Such a plan is called a *guaranteed strategy in the presence of model error.* But the plan cannot be guaranteed for the α where the goal vanishes. In these cases we want the plan to signal failure. Loosely speaking, a motion strategy which achieves the goal when it is recognizably reachable and signals failure when it is not is called an *Error Detection and Recovery (EDR) strategy.* Such strategies are more general than guaranteed strategies, in that they allow plans to fail.

To represent model error, we will choose a parameterization of the possible variation in the environment. The degrees of freedom of this parameterization are considered as additional degrees of freedom in the system. For example, in fig. 17, we have the x and y degrees of freedom of the configuration space. In addition, we have the model error parameter α. A coordinate in this space has the form (x, y, α). The space itself is the cartesian product $\Re^2 \times [-d_0, d_0]$. Each *α-slice* of the space for a particular α is a configuration space with the obstacles A and B instantiated at distance α apart. Fig. 17 is such a slice.

More generally, suppose we have a configuration space C for the degrees of freedom of the moving object. Let J be an arbitrary index set which parameterizes the model error. (Above, J was $[-d_0, d_0]$). Then the *generalized configuration space* with model error is $C \times J$. One way to think of this construction is to imagine a collection of possible "universes", $\{ C_\alpha \}$ for α in J. Each C_α is a configuration space, containing configuration space obstacles. The ambient space for each C_α is some canonical C. $C \times J$ is simply the natural product representing the ambient space of their disjoint union. There is no constraint that J be finite or even countable.

In fig. 18 we show the generalized configuration space for example (1). Note that the goal in generalized configuration space becomes a 2-dimensional surface, and the obstacles are 3-dimensional polyhedra. Note that the goal surface vanishes where A and B meet.

Given a configuration space corresponding to a physical situation, it is well known how to represent motions, forces, velocities, and so forth in it (eg., see [Arnold]). The representations for classical mechanics exploit the geometry of differentiable manifolds. We must develop a similar representation to plan motions, forces, and velocities in generalized configuration space . We develop the following "axioms" for "physics" in $C \times J$.

1. At execution time, the robot finds itself in a particular slice of $C \times J$, (although it

[14]This model is adopted for the purposes of exposition, not for physical plausibility. It is not hard to model the case where the blocks meet but do not fuse.

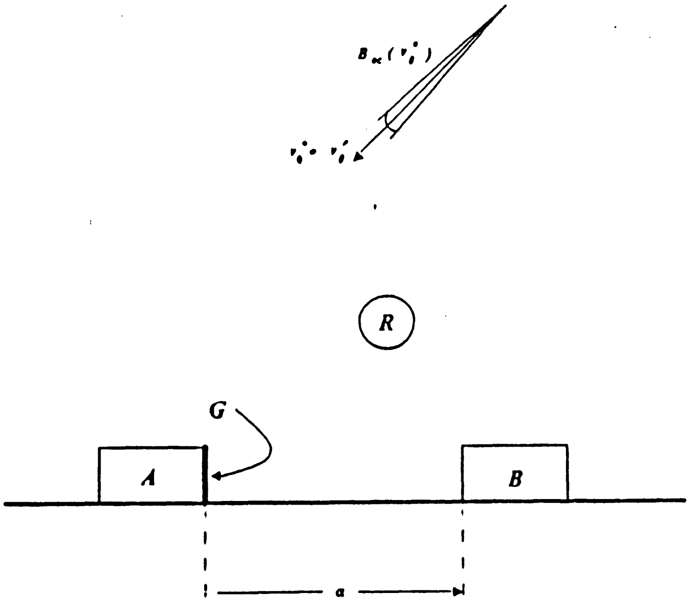

Figure 17: The equivalent problem in configuration space. The blocks A and B, the distance between the blocks α, and the commanded velocity $v_\theta = v_0^*$ with control error cone $B_{ec}(v_0^*)$. The position of A is fixed.

may not know which). Thus we say there is only one "real" universe, α_0 in[15] J. This α_0 is fixed. However, α_0 is not known *a priori.* Thus all motions are confined to a particular (unknown) α_0-slice, such as fig. 17. This is because motions cannot move between universes. In fig. 18, any legal motion in $C \times J$ is everywhere orthogonal to the J-axis and parallel to the x-y plane.

2. Suppose in any α-slice the position sensing uncertainty ball about a given sensed position is some set B_{ep}. The set R in fig. 17 is such a ball. We cannot sense across J: position sensing uncertainty is infinite in the J dimensions.[16] Thus the position sensing uncertainty in $C \times J$ is the cylinder $B_{ep} \times J$. In figs. 17,18, this simply says that x and y are known to some precision, while α is unknown. The initial position in fig. 17 is given by $R \times [-d_0, d_0]$. This cylinder is a 3-dimensional solid, orthogonal to the x-y plane and parallel to the J-axis in fig. 18.

3. Suppose in the configuration space C, the velocity control uncertainty about a given nominal commanded velocity is a cone of velocities B_{ec}. Such a cone is shown in fig. 17. This cone lies in the tangent bundle (or phase-space) TC of C. (Phase space is simply Position-space $\times$ Velocity-space. A point in phase space has the form (x, v), and denotes an instantaneous velocity of v at configuration x).

[15] α_0 is a point in the multi-dimensional space J.

[16] One generalization of the framework would permit and plan for sensing in J. In this case one would employ a bounded sensing uncertainty ball in the J dimensions.

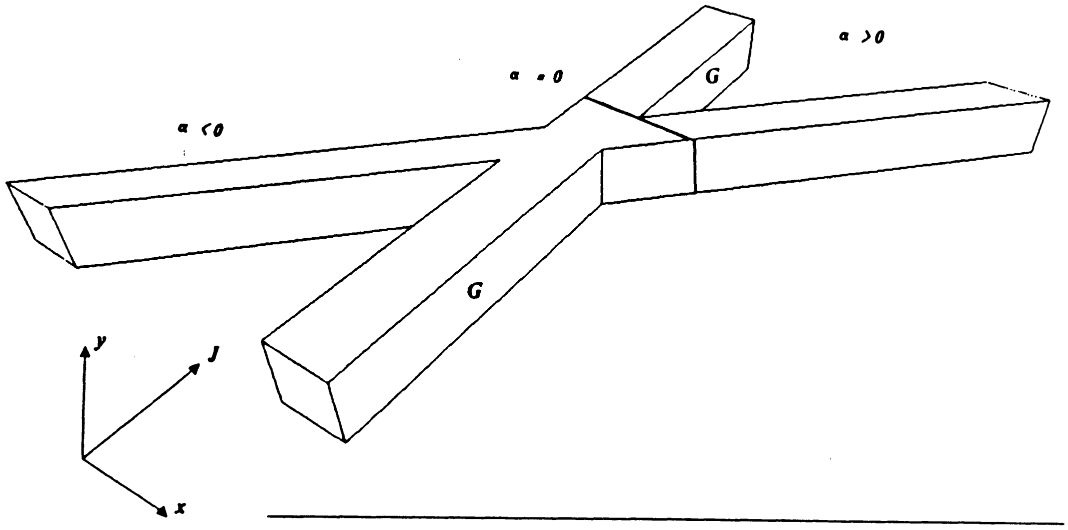

Figure 18: The generalized configuration space obstacles for example (1). The generalized configuration space is three dimensional, having x and y degrees of motion freedom, and an α degree of model error freedom. Legal motions are parallel to the x-y plane, and orthogonal to the J axis.

Phase space represents all possible velocities at all points in C. The phase space for $C \times J$ is obtained by indexing TC by J to obtain $TC \times J$. All velocities in generalized configuration space lie in $TC \times J$. For Ex. (1) $TC \times J$ is $\Re^4 \times [-d_0, d_0]$. The generalized velocity uncertainty cones are two-dimensional, parallel to the x-y plane, and orthogonal to the J axis.

4. Generalized damper dynamics extend straight-forwardly to $C \times J$, so motions satisfy $f = B(v - v_0)$ where f, v, and v_0 lie in $TC \times J$. Thus friction cones from configuration space (see [Erdmann]) naturally embed like generalized velocity cones in $TC \times J$.

These axioms give an intuitive description of the physics of $C \times J$. A formal axiomatization is given in [Don 89]. We have captured the physics of $C \times J$ using a set of *generalized uncertainties*, friction, and control characteristics (1-4). These axioms completely characterize the behavior of motions in $C \times J$.

2.4.2 Pushing

By relaxing axiom (1), above, we can consider a generalization of the model error framework, in which pushing motions are permitted, as well as compliant and gross motions. We relax the assumption that motion between universes is impossible, and permit certain motions across J. Consider example (3). Observe that a displacement in J corresponds to a displacement in the position of the block B. Thus a motion in J should correspond to a motion of B. Suppose the robot can change the position of B by pushing on it, that is, by exerting a force on the surface of B. The key point is that pushing operations may be modeled by observing that commanded forces to the robot may result in changes in the environment. That is, a commanded force to the robot can result in motion in C (sliding) as well as motion in J (pushing the block). Let us develop this notion further.

Our previous discussion assumed that motion across J was impossible. That is, all motion is confined to one α-slice of generalized configuration space. In example (3), this is equivalent to the axiom that B does not move or deform under an applied force. Such an axiom makes sense for applications where B is indeed immovable, for example, if A and B are machined tabs of a connected metal part. However, suppose that B is a block that can slide on the table. Then an applied force on the surface of the block can cause the block to slide. This corresponds to motion in J. In general, the effect of an applied force will be a motion which slides or sticks on the surface of B, and which causes B to slide or stick on the table. This corresponds to a coupled motion in both C and J, that is, a motion *across α-slices of generalized configuration space*. Such a motion is always tangent to a surface in generalized configuration space.

In [Don89], we generalize the description of the physics of $C \times J$ to permit a rigorous account of such motions. This model can then be employed by an automated planner. *Such a planner can construct motion strategies whose primitives are gross motions, compliant motions, and pushing motions.* This model of pushing is used in the gear-meshing example, where a model error parameter—the orientation of B—can be changed due to pushing.

2.4.3 Guaranteed Plans

Recall that a motion strategy is a commanded velocity v_θ^* together with a *termination predicate* which monitors the sensors and decides when the motion has achieved the goal. Given a goal G in configuration space, we can form its *preimage*. The preimage of G is the region in configuration space from which all motions are guaranteed to move into G in such a way that the entry is recognizable. That is, the preimage is the set of all positions from which all possible trajectories consistent with the control uncertainty are guaranteed to reach G recognizably.

Preimages provide a way to construct guaranteed plans for the situation with no model error. Can preimages and backprojections be generalized to situations with model error? The answer is yes. The generalized control and sensing uncertainties in $C \times J$ are given by the physics axioms above. These uncertainties completely determine how motions in generalized configuration space must behave. We form the backprojection of G under these uncertainties. The trick here is to view the motion planning problem with n degrees of motion freedom and k degrees of model error freedom as a planning problem in an $(n + k)$-dimensional generalized configuration space, endowed with the special physics described above. The physics is characterized precisely by axioms defining certain special sensing and control uncertainties in $C \times J$. The definitions and results for pre-images and backprojections (sec. 2.2) in configuration space generalize *mutatis mutandis* to $C \times J$ endowed with this physics; this is proved in [Don89]. Thus our framework reduces the problem of constructing guaranteed motion strategies with model error to computing preimages in a somewhat more complicated, and higher-dimensional configuration space. For details, see [Don89].

2.5 Error Detection and Recovery

If we were exclusively interested in constructing guaranteed motion strategies in the presence of model error, we would be done defining the framework: having reduced the problem to computing preimages in $C \times J$, we could now turn to the important and difficult problems of computing and constructing $C \times J$, and computing preimages in generalized configuration space.

However, guaranteed strategies do not always exist. In example (1), (figs. 16–18) there is no guaranteed strategy for achieving the goal, since the goal may vanish for some values of α. Because tolerances may cause gross topological changes in configuration space, this problem is particularly prevalent in the presence of model error. In a peg-in-hole problem with model error (fig. 3) the goal may also vanish (the hole may close up) for certain regions in J. More generally, there may be values of α for which the goal may still exist, but it may not be reachable. For example, in a variant of the problem in fig. 6, an obstacle could block the channel to the goal. Then G is non-empty, but also not reachable. Finally, and most generally, there may be values of α for which the goal is reachable but not *recognizably* reachable. In this case we still cannot guarantee plans, since a planner cannot know when they have succeeded.

These problems may occur even in the absence of model error. However, without model error a guaranteed plan is often obtainable by back-chaining and adding more steps to the plan. In the presence of model error this technique frequently fails: in example (1), no chain of recursively-computed preimages can ever cover the start region $R \times J$. The failure is due to the peculiar sensing and control characteristics (1-4) in generalized configuration space .

In response, we will develop Error Detection and Recovery (EDR) strategies. These are characterized as follows:

- An EDR strategy should attain the goal when it is recognizably reachable, and signal failure when it is not.
- It should also permit serendipitous achievement of the goal.
- Furthermore, no motion guaranteed to terminate recognizably in the goal should ever be prematurely terminated as a failure.

- Finally, no motion should be terminated as a failure while there is any chance that it might serendipitously achieve the goal due to fortuitous sensing and control events.

These are called the "EDR Axioms", they will be our guiding principles. We now state how such EDR strategies may be constructed; for proofs and more detail, see [Don89].

For technical reasons (see [Don89]), in the EDR theory, we must construct preimages relative not to the initial set R, but rather to the forward projection $F_\theta(R)$. We will henceforth mercifully abbreviate $P_{F_\theta(R),\theta}$ by P_θ.

Suppose that a planning problem is given with two disjoint geometrical goals, G_1 and G_2. We may insist that the run-time executor be able to terminate the strategy and also be able to disambiguate *which* goal has been reached. In this case, we can construct the *preimage of the distinguishable union of* G_1 *and* G_2, which we write as

$$P_\theta(\{\, G_1, G_2 \,\}).$$

If θ is executed starting in this preimage, then the motion can always be recognizably and distinguishably terminated in either G_1 or G_2.

We can characterize EDR strategies geometrically as follows. Suppose the geometric goal is G. The implicit "meaning" of G is: recognizably achieving G is equivalent to "success." We introduce an "additional" goal-like set H, which is disjoint and distinguishable from G, such that when H is recognizably achieved, then failure of the motion may be signalled. That is, we construct an H such that recognizably achieving H is equivalent to "failure." H is called the *EDR region.* Remarkably, H may be selected such that the EDR axioms are satisfied. In [Don89], we derive H as follows. Given a motion θ and a start region R, first we define H using reachability constructs only. Then we test whether H and G are distinguishable using sensors (this is the "formal test" alluded to in the prelude). If so, then by the construction of H, we have

$$R \subset P_\theta(\{\, G, H \,\}).$$

Furthermore, using H, θ is a one-step EDR strategy satisfying the EDR axioms. Here is an idea of what H is like: In fig. 19 the EDR region H is shown (in position space only) for example (1). Consider H as a two-dimensional region in $C \times J$; just a slice of it is shown in fig. 19. Note that in this example, H only exists in the slices in which G vanishes. Here, given the sensing uncertainty bounds of example (1), the termination predicate can distinguish between G and H based on position sensing, velocity sensing, or elapsed time. Clearly, H satisfies the EDR axioms: the motion is guaranteed to terminate recognizably in G iff the motion began in a universe in which G does not vanish. Otherwise, the motion terminates recognizably in H. In the first case, the termination predicate signals success, in the latter, failure.

Here is how we construct H. Recall the *forward projection* $F_\theta(R)$ of a set R under θ is all positions and velocities that are possibly reachable from R under v_θ^* (subject to control uncertainty). Forward projections only address reachability: the termination predicate is

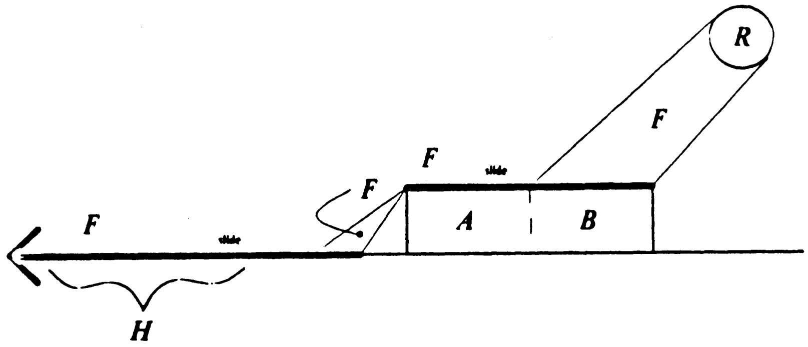

Figure 19: A typical α-slice of the EDR region H, for α small and negative. The goal vanishes in this slice; the dashed line indicates where the goal would be in other slices. Points in H lie within the forward projection (since they are reachable), yet outside the weak-preimage (since the goal is unachievable).

ignored and only the control uncertainty bound and commanded velocity v_θ^* are needed to specify the forward projection.

So far the preimages we have considered are *strong* preimages, in that *all* possible motions are guaranteed to terminate recognizably in the goal. The *weak* preimage [LMT] (with respect to a commanded velocity) is the set of points which could *possibly* enter the goal recognizably, given fortuitous sensing and control events. See fig. 20. We will use the weak preimage to capture the notion of serendipity in the EDR axioms. The idea is that a motion may be terminated in failure as soon as egress from the weak preimage is recognized. The weak preimage is denoted $\hat{P}_\theta(G)$.

We define H_0 to be the set difference of the forward projection minus the weak preimage:

$$H_0 = \pi F_\theta(R) - \hat{P}_\theta(G). \tag{20}$$

Clearly, the motion θ can be terminated as a failure whenever H_0 has been reached, since H_0 is outside the weak preimage; hence the goal cannot be attained under θ from there.

We also define H_s to be all regions where sticking is possible in the weak minus strong preimage:

$$H_s = \{\, x \in \hat{P}_\theta(G) - P_\theta(G) \mid \text{sticking is possible at } x \,\}.$$

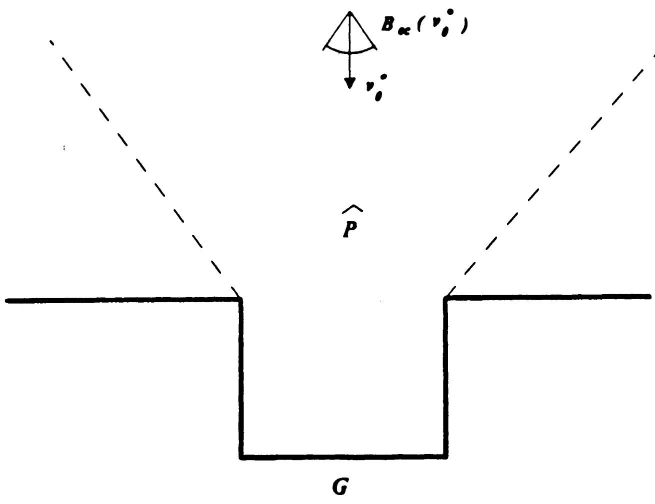

Figure 20: The *weak* preimage of the goal G under v_θ^*. Compare figs. 6 and 12.

See fig. 21. The motion θ should also be terminated as a failure if sticking occurs in H_s. We will decree that the robot has stuck in H_s if its velocity is zero for some duration, or "time-out" period.[17] More precisely, we define H to be a set in phase-space. First, note that H contains all phase-space points (x, v) where x is in H_0. Second, H contains points of the the form $(x, 0)$, where x is in H_s. Thus, viewing phase-space as the tangent bundle to generalized configuration space, H contains the "cylinder" $\pi^{-1}(H_0)$ of velocities over H_0, and the "zero section" $Z(H_s)$ of zero velocities over H_s:

$$H = \pi^{-1}(H_0) \cup Z(H_s).$$

This definition of H almost satisfies the EDR axioms—the only tricky point is that we cannot guarantee that after sticking in H_s for a long time, the robot cannot eventually slide into the goal. This may be handled in principle by introducing a time-out period by which the goal must be reached. That is, our definition of H satisfies the EDR axioms if the goal is specified in phase-space-*time* as the product of $\pi^{-1}(G)$ with a compact time interval.

2.5.1 The Preimage Structure of EDR Regions

Consider an equation such as (13) once more, viewing H as an EDR region:

[17]That is, the termination predicate halts the motion when the velocity is zero for some prespecified duration.

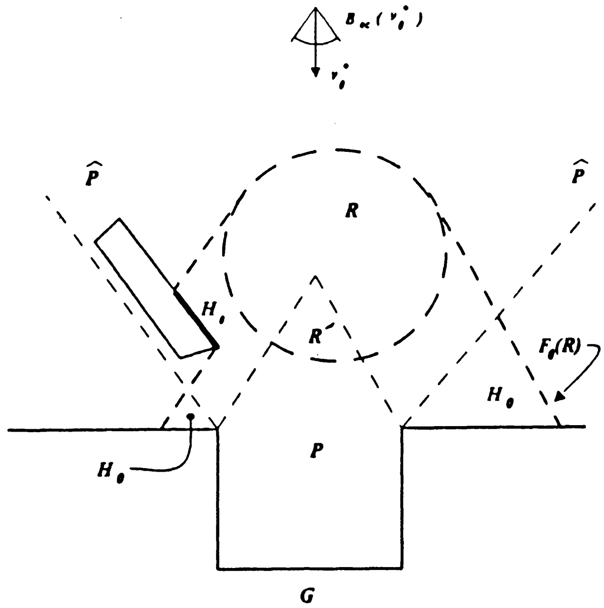

Figure 21: H_0 in eq. (20) is not the entire EDR region. Sticking may occur within the weak preimage in H_s. The EDR region must include H_0 for all possible velocities, and H_s for "sticking velocities."

$$R \subset P_{F_\theta(R),\theta}(\{\, G, H \,\}). \tag{21}$$

We remarked that in general, solving preimage equations like (13) and (21) for R is like finding the fixed point of a recursive equation. However, in the EDR application, we know R, H, and G, so (21) is a constraint which must be true, rather than an equation to solve. Presumably (21) is easier to check than to solve for R.

2.5.2 Application: Algorithms for EDR Strategies

In [Don89], we showed how to compute H in the domain of planar assemblies with model error. We also showed how to compute whether G and H are distinguishable. This is sufficient to generate one-step EDR strategies. These algorithms have been implemented in LIMITED. At a high level, the one-step algorithm is:

Algorithm Planar One-Step EDR

1. *Generate a commanded velocity* v_θ^*.
2. *Compute the EDR region* H *for* v_θ^*,
3. *Determine whether the EDR region* H *and the goal* G *are distinguishable using sensors. If so, then* v_θ^* *yields a one-step EDR strategy which recognizably terminates in* G *or* H *by monitoring position and force sensors.*

4. *Let* $\text{push}_\theta(G)$ *and* $\text{push}_\theta(H)$ *denote the* sticking push-forwards. *They are the set of obstacle edges within* G *and* H, *resp., on which sticking can occur under* v_θ^*. *Determine whether these regions are distinguishable using sensors. If so, then* v_θ^* *yields a one-step EDR strategy which recognizably terminates when sticking is detected.*

Here is how LIMITED decides the question, "Are G and H distinguishable using sensors?"

H and G are distinguishable using position sensing alone if their convolutions (Minkowski sums) by the position sensing error ball B_{ep} do not intersect.

Each obstacle edge of H and G has an associated configuration space friction cone. Two edges are distinguishable using force sensing if the convolutions of their friction cones by the force sensing uncertainty B_{ev} have a trivial intersection.

Similarly, the set of possible sensed reaction forces at an obstacle vertex w of G or H may be found by taking the direct sum of the friction cones of the edges cobounding w, and convolving by B_{ev}. Again, a vertex of H and a vertex (or edge) of G are distinguishable using force sensing if their associated cones of sensed reaction forces have a trivial intersection.

The procedure also works for determining the distinguishability of the push-forwards. Note that the procedure is correct for linear edges, where position- and force-sensing are separable, because the set of possible reaction forces is constant along an edge. For the general case, see [Don89].

2.6 Advanced Topics

Now, we have only addressed the planning of single-step EDR strategies. In order to plan multi-step EDR strategies, the techniques of sec. 2.3 may be employed. Multi-step strategy synthesis is complex, and the theory extends rather far beyond this paper.

We note that, in principle, having reduced both model error and EDR to essentially "preimage-theoretic" equations, multi-step strategies could be synthesized by solving these preimage equations. While this is proved or at least implicit in previous work [LMT,E,Ma2,D], it is far from obvious; furthermore, there are almost no published examples of such strategies. For this reason we presented a worked-out example of a motion plan using preimages in sec. 2.3. We could easily derive EDR as well as guaranteed strategies by solving the preimage equations [Don90].

Preimages are a key underlying tool for the geometric EDR theory, and the LMT framework is in some sense a "universal" method for synthesizing multi-step strategies. However, the technique of solving the preimage equations is not computational. For this reason, we have introduced a construction called the *push-forward* [Don89]. Roughly speaking, the push-forward is that subset of the forward projection where the motion can terminate. Since push-forwards address termination whereas forward projections do not, we may regard them as "dual" to preimages. That is, push-forwards are to forward projections as preimages are to backprojections. Second, the push-forward permits us to develop rather simple algorithms for planning multi-step strategies. These algorithms have been implemented in LIMITED. While the push-forward method for multi-step strategy synthesis is algorithmic, it is less general than the full preimage method (solving the preimage equations). We characterize the loss of power in push-forward algorithms in [Don89].

In [Don89], we presented two EDR plans generated by LIMITED. These were a peg-in-hole insertion strategy with model error, and a gear-meshing plan. Both were two-step plans. The peg-in-hole plan used push-forward techniques. The gear plan used a seemingly unrelated technique called *failure mode analysis.* However, there exists a view of multi-step strategies which essentially unifies all these techniques. This is called the "weak" EDR theory in [Don89]. The motivation behind this theory is that when a motion terminates ambiguously, a subsequent motion may be synthesized which disambiguates the success or failure of the first. Oddly enough, it is not necessary for either motion individually to satisfy the EDR axioms of sec. 2.5. However, when taken together, the two-motion plan can often be considered "equivalent" to a one-step EDR strategy.

The weak EDR theory effectively defines some laws of "composition" that permit two single-step plans to be concatenated into a two-step plan satisfying the EDR axioms. Hence it is often possible to construct multi-step plans that are EDR plans "globally" although not "locally". That is, considered as entire plans, they satisfy the EDR axioms; this is the "global" condition. However, "locally" they are not EDR plans, in that no single step is an EDR strategy. The key to pasting together non-EDR plans to make a global EDR strategy lies in defining certain local "niceness" conditions for how plans must mesh. These are called the *linking conditions.*

As the notation suggests, it is possible to formalize our view of "motions as mappings"—this notion is implicit in the term "preimage." To develop this viewpoint, one considers motions as a certain class of morphisms between distinguishable unions in the powerset of the tangent bundle to generalized configuration space . An EDR theory, then, is a covariant functor associated with a family of quotient maps of the form

$$\pi : TC \times J \to \{ P, \hat{P} - P, \tilde{F} - \hat{P} \}.$$

One focus for future work would be to push such a functorial viewpoint; any category-theoretic formulation of this flavor is still operational in nature and not yet fully abstract.

Acknowledgments: I am very grateful to Randy Brost for providing me with figures 8-10. Also, thanks to Tomás Lozano-Pérez, Matt Mason, Russ Taylor, Mike Erdmann, and John Canny for sharing with me the exciting research I survey in this paper.

3 References

Abraham, R. and Marsden, J. *Foundations of Mechanics,* Benjamin/Cummings, London (1978).

Arnold, V. I. *Mathematical Methods of Classical Mechanics,* Springer-Verlag, New York (1978).

[BK] Bajaj, C. and M. Kim, "Compliant Motion Planning with Geometric Models", *Proc. ACM Symposium on Computational Geometry,* Waterloo, 1987.

Brady, M. et. al. (eds). *Robot Motion: Planning and Control.,* Cambridge, Mass.: MIT Press. (1982).

Briggs, Amy *An Efficient Algorithm for One-Step Compliant Motion Planning with Uncertainty,* 5^{th} ACM Symposium on Computational Geometry, Saarbrucken, Germany. (1989).

Brost, R., "Automatic Grasp Planning in the Presence of Uncertainty", *IEEE International Conference on Robotics and Automation,* San Francisco, April, 1986.

Brost, R. *A State/Action Space Approach to Planning Robot Actions,* Forthcoming Ph.D. Thesis, Computer Science Dept., CMU (to appear).

Brost, R. *Computing Metric and Topological Properties of Confugurarion Space Obstacles,* IEEE Int. Conference on Robotics and Automation, Scottsdale, AZ (1989).

Buckley, S. J. *Planning and Teaching Compliant Motion Strategies,* Ph.D. Thesis. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1987. Also MIT-AI-TR-936 (1987).

Burridge, R., Rajan, V. T., and Schwartz, J. T. *The Peg-In-Hole Problem: Statics and Dynamics of Nearly Rigid Bodies in Frictional Contact,* IEEE ICRA, Raleigh, NC (1983).

[C1] Canny, J.F. *The Complexity of Robot Motion Planning,* Ph.D. Thesis, MIT Department of Electrical Engineering and Computer Science (1987).

[Can89a] John Canny. *On computability of fine motion plans.* , IEEE ICRA, Scottsdale, AZ, (1989)

[CD] Canny, J.F. and Donald, B. R. *Simplified Voronoi Diagrams,* Discrete and Computational Geometry **3** (3) pp. 219-236. (1988).

[CR] Canny, J., and J. Reif, "New Lower Bound Techniques for Robot Motion Planning Problems", *FOCS* (1987).

[D] Donald, B. R. *Robot Motion Planning with Uncertainty in the Geometric Models of the Robot and Environment: A Formal Framework for Error Detection and Recovery,* IEEE International Conference on Robotics and Automation, San Francisco, April (1986a).

Donald, B. R. *A Search Algorithm for Motion Planning with Six Degrees of Freedom,* Artificial Intelligence, 31 (**3**) (1987a).

Donald, B. R. *Error Detection and Recovery for Robot Motion Planning with Uncertainty,* Ph.D. Thesis, Dept. EECS, MIT Artificial Intelligence Laboratory, MIT-AI-TR 982 (1987b).

Donald, B. R. *The Complexity of Planar Compliant Motion Planning with Uncertainty,* Proc. ACM Symposium on Computational Geometry, Urbana, IL June, 1988. *To appear in* Algorithmica. (1988a).

[D] **Donald, B. R.** *A Geometric Approach to Error Detection and Recovery for Robot Motion Planning with Uncertainty,* Artificial Intelligence **37** (1-3), Dec. 1988. pp. 223-271. (1988b).

Donald, B. R. *Planning Multi-Step Error Detection and Recovery Strategies,* Proc. IEEE International Conference on Robotics and Automation, Philadelphia, PA April, 1988. (1988c).

Donald, B. R. *Error Detection and Recovery in Robotics,* Lecture Notes in Computer Science, Vol. 336, Springer-Verlag, New York (1989). (1989).

Donald, B. R. *Planning Multi-Step Error Detection and Recovery Strategies,* International Journal of Robotics Research, **3** (1), March, (In press). (1990).

Draper Laboratories, Fourth Annual Seminar on Robotics and Advanced Assembly Systems, Cambridge, Massachusetts, November, 1983.

Dufay, B., and J. Latombe, "An Approach to Automatic Robot Programming Based on Inductive Learning", in Brady, M., and R. Paul, *Robotics Research: The First International Symposium*, MIT Press, 1984.

Durrant-Whyte, H. *Concerning Uncertain Geometry in Robotics,* Intl. Workshop on Geometric Reasoning, Oxford, England, June (1986).

[E] **Erdmann, M.** *Using Backprojections for Fine Motion Planning with Uncertainty,* IJRR Vol. 5 no. 1 (1986).

Erdmann, M. *On Motion Planning With Uncertainty,* MIT AI Lab, MIT-AI-TR 810 (1984).

Erdmann, M. *On Probabilistic Robot Strategies,* Ph.D. thesis, MIT Department of EECS, MIT A.I. Lab, Cambridge (1989)..

Erdmann, M., and M. Mason, "An Exploration of Sensorless Manipulation", *IEEE International Conference on Robotics and Automation*, San Francisco, April, 1986.

[FHS] **Friedman, J., J. Hershberger and J. Snoeyink** *Compliant Motion in a Simple Polygon,* 5^{th} ACM Symposium on Computational Geometry, Saarbrucken, Germany. (1989).

[GM89] **Goldberg, K., and Mason, M.** *Bayesian Grasping,* Submitted to the IEEE Int'l. COnf. on Robotics and Automation, 1990 (1989).

Hopcroft, J. E., Schwartz, J. T., and Sharir, M. 1984 On the Complexity of Motion Planning for Multiple Independent Objects; *PSPACE*-Hardness of the "Warehouseman's Problem." *International Journal of Robotics Research.* **3**(4):76–88.

Hopcroft J., and Wilfong G., "Motion of Objects in Contact," Int. Jour. Robotics Res. vol 4, no. 4, (1986).

Inoue, H., "Force Feedback in Precise Assembly Tasks", MIT Artificial Intelligence Laboratory, AIM-308, August, 1974.

J. Jennings, B. Donald, and D. Campbell *Towards Experimental Verification of an Automated Compliant Motion Planner based on a Geometric Theory of Error Detection and Recovery,* Proc. IEEE International Conference on Robotics and Automation, Scotsdale, Arizona (1989).

[LLS] **Latombe, J.C., A. Lazanas, and S. Shekhar** *Motion Planning with Uncertainty: Practical Computation of Non-Maximal Preimages,* IEEE/RSJ Intl. Workshop on Intelligent Robots and Systems, Sept. 4-6 (1989).

Laugier, C., "A Program for Automatic Grasping of Objects with a Robot Arm", *Eleventh Symposium of Industrial Robots*, Japan Society of Biomechanisms and Japan Industrial Robot Association, 1981.

Lieberman, L., and M. Wesley, "AUTOPASS: An Automatic Programming System for Computer Controlled Mechanical Assembly", *IBM Journal of Research Development*, Vol. 21, No. 4, 1977, pp. 321-333.

Lozano-Pérez, T., "The Design of a Mechanical Assembly System", S.M. dissertation, MIT Department of Electrical Engineering and Computer Science, also AI-TR-397, MIT Artificial Intelligence Laboratory, 1976.

Lozano–Pérez, T. *Automatic Planning of Manipulator Transfer Movements,* IEEE Trans. on Systems, Man and Cybernetics (SMC–11):681–698 (1981).

[LoP] **Lozano–Pérez, T.** *Spatial Planning: A Configuration Space Approach,* IEEE Trans. on Computers (C–32):108–120 (1983a).

Lozano-Pérez, T., "Robot Programming", *IEEE Proceedings*, 1983b.

Lozano-Pérez, "A Simple Motion Planning Algorithm for General Robot Manipulators," in Proceedings of Fifth National Conference for the American Association of Artificial Intelligence, Philadelphia, 1986, pp. 626–631.

[LMT] **Lozano-Pérez, T., Mason, M. T., and Taylor, R. H.** *Automatic Synthesis of Fine-Motion Strategies for Robots,* Int. J. of Robotics Research, Vol 3, no. 1 (1984).

Lozano-Pérez, T., and Wesley, M. A. *An algorithm for planning collision-free paths among polyhedral obstacles,* Communications of the ACM (22):560-570 (1979).

Lumelsky, V. J. *Continuous Motion Planning in Unknown Environment for a 3D Cartesian Robot Arm.* , *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pp. 1569–1574. (April 7–10, San Francisco, Calif.) (1986).

Mason, M.T. *Compliance and force control for computer controlled manipulators*, IEEE Trans. on Systems, Man and Cybernetics (SMC-11):418-432 (1981).

Mason, M. T. 1986. Mechanics and Planning of Manipulator Pushing Operations. *International Journal of Robotics Research* **5**(3).

[**Ma2**] **Mason, M. T.** *Automatic Planning of Fine Motions: Correctness and Completeness,* 1984 IEEE International Conference on Robotics, Atlanta Ga. (1984).

Natarajan, B. K. 1986 (Oct. 27–29, Toronto, Ontario). An Algorithmic Approach to the Automated Design of Parts Orienters. *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, pp. 132–142.

Ohwovoriole, M., and B. Roth, "A Theory of Parts Mating For Assembly Automation", *Proceedings of the Robot and Man Symposium 81*, Warsaw, Poland, September 1981.

Peshkin, M., "Planning Robotic Manipulation Strategies for Sliding Objects", Ph.D. dissertation, Department of Physics, Carnegie-Mellon University, 1986.

Raibert, M., and J. Craig, "Hybrid Position/Force Control of Manipulators", *Journal of Dynamic Systems, Measurement, and Control*, No. 102, June, 1981, pp. 126-133.

Schwartz J., Hopcroft J., and Sharir M., "Planning, Geometry and Complexity of Robot Motion Planning", Albex Publishing Co., New Jersey, (1987).

Schwartz J. and Yap C. K., "Advances in Robotics," Lawrence Erlbaum associates, Hillside New Jersey, (1986).

Simunovic, S. N., "An Information Approach to Parts Mating", Ph.D. dissertation, Department of Electrical Engineering, Massachusetts Institute of Technology, 1979.

Taylor, R. H. *The Synthesis of Manipulator Control Programs from Task-level Specifications,* Stanford Artificial Intelligence Laboratory, AIM-282, July (1976).

Turk, M., "A Fine-Motion Planning Algorithm", *SPIE Conference on Intelligent Robots and Computer Vision*, Cambridge, Massachusetts, September, 1985.

Udupa S., "Collision Detection and Avoidance in Computer Controlled Manipulators", Proc. 5th Int. Joint. Conf. on Art. Intell., Mass. Inst. Tech. (1977) pp 737-748.

Valade, J., "Automatic Generation of Trajectories for Assembly Tasks", *Sixth European Conference on Artificial Intelligence*, Pisa, Italy, September, 1984.

[**Whi76**] **Whitney, D.**, "Force Feedback Control of Manipulator Fine Motions", *Journal of Dynamic Systems, Measurement, and Control*, June, 1977, pp. 91-97.

[Whi82] **Whitney, D.**, "Quasi-Static Assembly of Compliantly Supported Rigid Parts", *Journal of Dynamic Systems, Measurement, and Control*, Vol. 104, March 1982.

Department of Computer Science, Cornell University, Ithaca, NY 14853.