Stochastic Plans for Robotic Manipulation

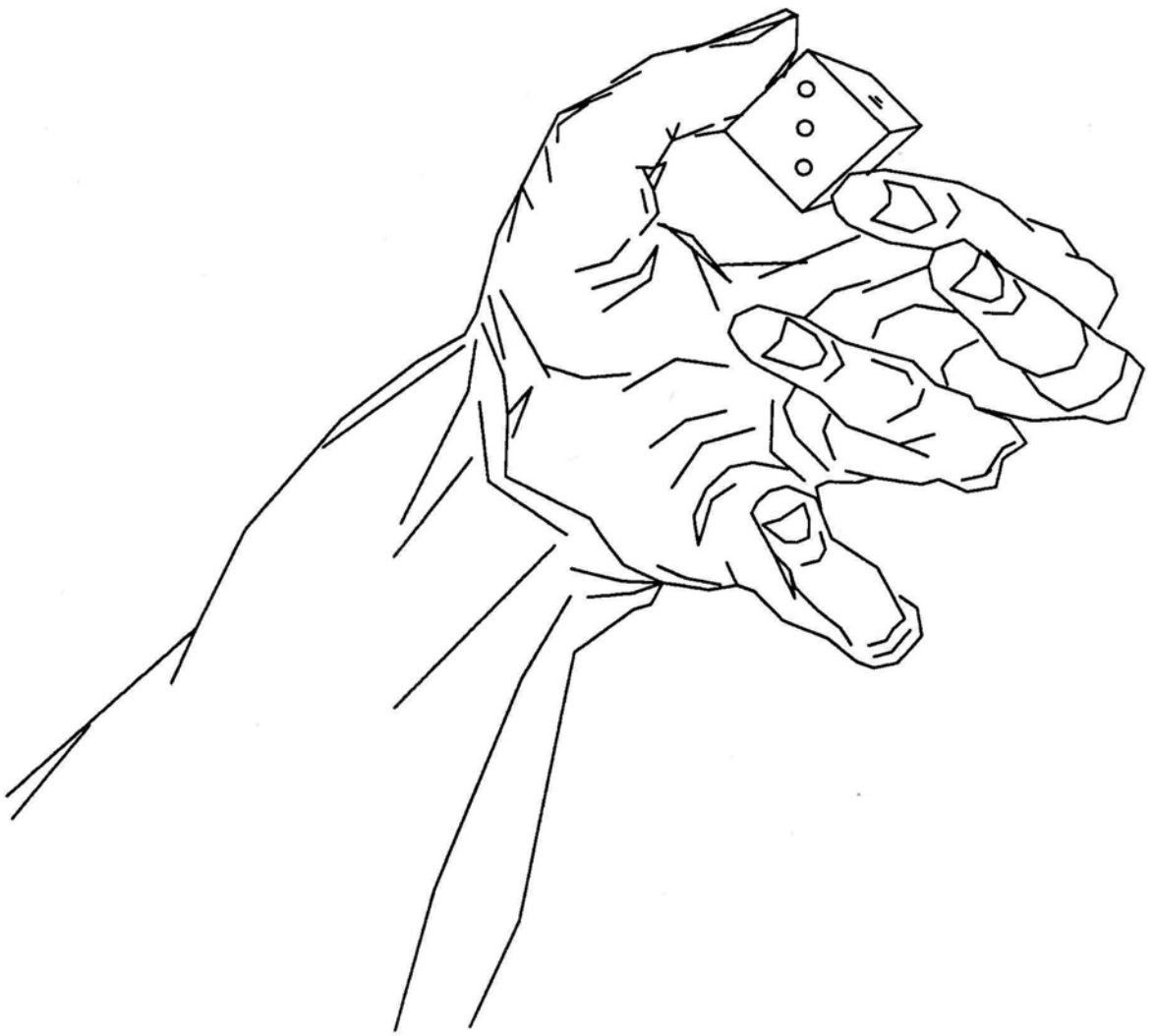
Kenneth Yigael Goldberg

31 August 1990
CMU-CS-90-161

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Supported by NSF grant DMC-8520475 and NASA-Ames grant NCC 2-463. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, or NASA-Ames, or the U.S. Government.



Abstract

Geometric uncertainty is unavoidable when programming robots for physical applications. We propose a stochastic framework for manipulation planning where plans are ranked on the basis of *expected cost*. That is, we express the desirability of states and actions with a cost function and describe uncertainty with probability distributions. We illustrate the approach with a new design for a *programmable parts feeder*: a mechanism that orients two-dimensional parts using a sequence of open-loop mechanical motions. We present a planning algorithm that accepts an n-sided polygonal part as input and, in time $O(n^2)$, generates a stochastically optimal plan for orienting the part.

Dedicated to the memory of my father and his vision of a robotic hoist.

Most importantly, I thank my family, especially my mother, sisters, and grandparents for their consistent support, encouragement, and love through 29 years of learning. Over these years I was lucky to work with some excellent teachers: Theresa Ludwig, Doris Helms, Len Perrett, Peggy Ota, Ruzena Bajcsy, and Philip Rieff, who taught me that you only live once, if then.

I owe a substantial debt to my advisor, Matt Mason, for his guidance and high standards. For his amazing tolerance of my artwork in the lab, his support and confidence in the ability of his students. His good humor fostered a strong sense of solidarity in our lab that made this work a pleasure. I also thank the other members of my thesis committee, Profs. Paul Wright of NYU, Rob Kass of CMU Statistics, and Takeo Kanade for their insightful questions and advice.

For specific contributions to this thesis, I'd like to thank Alan Christiansen, Mike Erdmann, Merrick Furst, Klaus Gross, Kevin Lynch, Barak Pearlmutter, Harry Printz, Russ Taylor, and in particular Randy Brost for his boundless enthusiasm.

I also greatly benefitted from discussions with the following colleagues: Srinivas Akella, Peter Allen, Dina Berkowitz, David Bourne, Johann Borenstein, Ben Brown, John Canny, Lin Chase, Peter Cheeseman, Bruce Donald, Morris DeGroot, Adele Goldberg, Jessica Hodgins, Katsushi Ikeuchi, Guy Jacobsen, Jeff Koechling, Eric Krotkov, Ehud Lenz, Larry Matthies, Gary Miller, Max Mintz, Bud Mishra, Hans Moravec, Mark Nagurka, Balas Natarajan, Shree Nayar, Irv Oppenheim, Mike Peshkin, Marc Raibert, Todd Rockoff, Dean Rubine, Moshe Shoham, Reid Simmons, Herb Simon, Dan Sleator, Rick Szeliski, Roy Taylor, Jeff Trinkle, Richard Wallace, Yu Wang, Larry Wasserman, Jon Webb, and Andy Witkin.

If I had to do it over again I'd come back to Carnegie Mellon's School of Computer Science. Anyone who's spent any time here knows what I mean. Thanks to Nico Habermann for giving me a semester to visit the Technion. And to Mike Acceta, Sylvia Berry, Sharon Burks, Catherine Copetas, Edith Culmer, Jean Harpley, Maggie Muller, and Jim Skees for making everything around here run so smoothly.

I thank my friends in Pittsburgh and those who attended the Logout Bashes. Chateau Wexford: Todd Rockoff and Klaus Gross, and especially Richard Wallace, who shared adventures both inside and outside of Wean Hall. And old friends G.C., D.C., E.C., K.B., C.Z., J.E., J.E., S.F., A.S., J.A., E.G., A.G., and B.C.

I'd also like to thank Ram Nevatia, Ari Requicha, George Bekey, Jerry Mendel, and Ellis Horowitz for inviting me to join the faculty at USC. But that's another story....

Ken Goldberg
Pittsburgh, PA. 1990

Contents

Chapter 1

Introduction

Geometric uncertainty is unavoidable when programming robots for physical applications. Consider programming a robot to pick up parts on an assembly line. Each time a part arrives, its initial position and orientation relative to the robot will be slightly different. Yet we desire a sequence of commands – a plan – that will achieve a desired outcome. Can we find a plan that will work every time? Or at least, most of the time?

To address such questions, a geometric theory of manipulation planning in the presence of uncertainty was outlined by Lozano-Perez, Mason, and Taylor [1984] and extended in a series of papers over the past six years. In this approach, robot actions correspond to mappings on a state (configuration) space. To express uncertainty, both the initial conditions and the desired outcome (the goal) are represented as *subsets* of state space such that the true state of the physical system corresponds to a single element of the current set. A plan that defines a mapping from the initial set into the goal set is *guaranteed* to succeed.

This approach makes a binary distinction between plans that are guaranteed and plans that are not guaranteed. This may be overly conservative in factory applications where plans will be executed many times and occasional failure is tolerable. For cases where a guaranteed plan doesn't exist, we need a way to choose between plans that are not guaranteed. And for cases where more than one guaranteed plan exists, we need a way to choose between guaranteed plans. In short, we need a formalism for *ranking* plans.

If we treat plans as strategies, the theory of games and decisions provides such a formalism. Plans can be ranked by *expected performance* based on a posterior probability distribution and a cost function that relates states to performance. This thesis suggests a stochastic framework for manipulation planning and demonstrates how cost and probability models can be justified either (i) by analyzing system geometry or (ii) through empirical observations. This thesis, together with the independent work of Mike Erdmann [1989], provides a link between well-established methods in statistics and ongoing work in the theory of robot manipulation.

1.1 Background

In the paper, *Sensor-Based Manipulation Planning as a Game With Nature*, Taylor, Mason, and Goldberg [1987], suggested an approach to manipulation planning where sensor data and actions are treated in a common framework: both are viewed as constraints on the set of possible system configurations (states). Erdmann [1989] focussed on probabilistic extensions of this framework, in particular the use of randomization in planning. This thesis focusses on tradeoffs between a plan's cost and its probability of success.

In this section we review the history of robotic manipulation planning in the presence of uncertainty and describe the progression of ideas that led to the current thesis topic.

1.1.1 Robotic Manipulation Planning

We refer to a robot command as an *action*, noting that the exact outcome of an action may be impossible to predict. We refer to a sequence of actions as a *plan*. *Planning* is the process of finding a plan to achieve a desired outcome. An example is path planning, where the objective is to find a sequence of actions that will move the robot from point A to point B without crashing into obstacles along the way [Schwartz *et al.*, 1987]. In manipulation planning, the objective is to find a sequence of actions to move a *part* from state A to state B using mechanical interactions between the robot and the part.

1.1.2 Uncertainty

Anyone who has tried to program a real robot for manipulation is aware of the problem of uncertainty. Given an ideal system of masses and forces, we can in principle predict the resulting motions using Newtonian mechanics. In practice however, we cannot predict the exact position and orientation of the part we want to manipulate. Also, we cannot predict the exact forces that will act on the part.

Friction, for example, is a common source of uncertainty. Consider programming a robot to push a box across a table. When the box is pushed, the reaction torque depends on the box's support distribution, which in turn depends on the microscopic interface between the box and the table [Mason, 1982]. As the box is pushed, the interface changes. There is also a discontinuity between static and dynamic coefficients of friction. Since these quantities are essentially impossible to predict, we cannot predict the exact motion of the box.

1.1.3 Two Approaches to Coping with Uncertainty

When we don't know the state of a system, one approach is to *sense* the state, for example with a visual or tactile sensor. Transforming noisy sensor data into constraints in the state of the system depends on many factors, such as the noise and resolution of the sensor. There is a vast body of literature on methods for processing sensor data.

A more subtle approach to coping with uncertainty is to constrain the state with mechanical compliance, for example by sweeping the robot arm across the table to "scoop up" a block prior to grasping [Inoue, 1974, Mason, 1978]. Another example is the common method for calibrating X-Y plotters. When powered up, the initial position of the pen is unknown. The first step is to drive the pen along the X axis for a few seconds and then along the Y axis for a few seconds. This open-loop strategy drives the pen into a known position (the corner) from which subsequent motions are measured. Open-loop actions are also referred to as *sensor-less actions* [Erdmann and Mason, 1986] or *generalized funnels* [Brost, 1990].

This thesis focusses on the class of *open-loop* plans, where execution proceeds without sensors. An example is shown in Figure 1.1. Since open-loop plans require no sensing and minimal on-line computation, they can be less expensive to implement than sensor strategies.

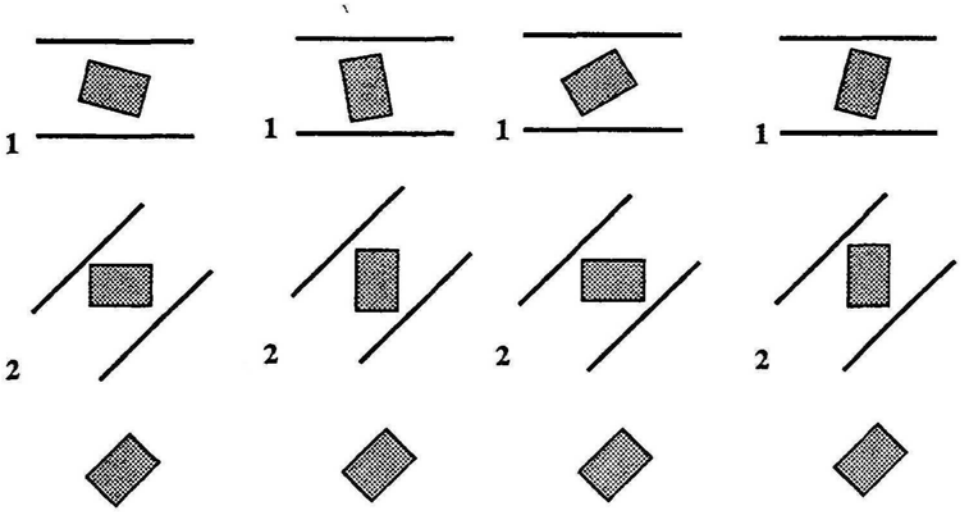


Figure 1.1: Four traces of a two-step plan for orienting a rectangular part using the frictionless parallel-jaw gripper. Each trace runs from top to bottom showing the orientation of both gripper and part after each squeeze action. Although the part's *initial* orientation is different in each trace, its *final* orientation is the same. Note that gripper motion for each case is the same. The plan is open-loop: it does not depend on sensor data.

1.1.4 A Geometric Theory of Manipulation Planning

> We believe that [the geometric theory of manipulation] is the cleanest and most rigorous approach to motion planning with uncertainty proposed so far, and that, despite some theoretical difficulties, it will have a major impact in the future. [Latombe, 1989]

A geometric theory of manipulation planning in the presence of uncertainty was outlined by Lozano-Perez, Mason, and Taylor [1984]. This theory has been explored over the past six years; see Latombe [1989] and Donald [1990] for reviews. The fundamental idea is that manipulation planning can be transformed into a geometric problem in a state (configuration) space, where robot actions correspond to mappings and mechanical properties such as stability, friction, and kinematic constraints are related to geometric conditions.

To express uncertainty, both the initial conditions and the desired outcome (the goal) are represented as *subsets* of state space such that the true state of the physical system corresponds to a single element of the current set. A plan that defines a mapping from the initial set into the goal set is guaranteed to succeed. Such a plan is called a *guaranteed plan* (also called a *fine-motion plan* [Lozano-Perez, Mason, and Taylor , 1984]).

Preimage Backchaining

One way to find a plan is to work backward from the goal set. This method for finding manipulation plans is called the *preimage backchaining* method. Lozano-Perez, Mason, and Taylor [1984] used the peg-in-hole problem as an example, where peg velocity can only be controlled to within a worst-case error cone. Uncertainty can be reduced using *compliant motion* where robot motion complies to the geometry of the environment. By projecting the velocity error cone backward from the goal, a geometrical region from which the motion is guaranteed to succeed – a *strong pre-image* – can be identified. The strong pre-image then becomes the goal for another motion. We can prove that a multi-step plan is guaranteed to reach the goal set using pre-images as pre-conditions in a Hoare-style logic. The approach bears some resemblance to dynamic programming [Bertsekas, 1987], where one finds an optimal plan by working backward from a desired outcome.

The correctness and completeness of the backchaining method was studied by Mason [1984]. Erdmann [1984] addressed computability issues and proposed a restricted class of computable pre-images. Natarajan [1986] and Canny [1987] analyzed the computational complexity of planning under this theory. Buckley [1986] implemented a motion planner based on the theory. Donald [1987] noted that guaranteed plans do not always exist and proposed a class of plans that fail gracefully. Brost [1988] used pre-images to find one-step grasping actions that are guaranteed to be stable.

Planning as a Game with Nature

Erdmann and Mason [1986] proposed the class of *tray-tilting* actions, where an object is oriented by tilting a four-sided tray. An action is specified by a tilting angle. They used pre-images to generate a sequence of open-loop tilting actions guaranteed to orient a part. They noted that searching forward for a guaranteed plan can be viewed as searching through a tree, where each node in the tree corresponds to a *set* of possible states. Actions define links between nodes. A guaranteed open-loop plan corresponds to a path from the root node (containing the initial set of states) to a node containing a single goal state.

This approached was generalized to include finite sensors (where the set of possible sensor values is finite) in Buckley [1986], Taylor, Mason, and Goldberg [1987] and Mason, Goldberg, and Taylor [1988]. With sensors, the tree is an And/Or tree where each sensor-outcome defines an AND-link between nodes. A guaranteed plan is a subtree where each sensor link leads to a goal node. Taylor, Mason and Goldberg noted that planning in this framework could be viewed as searching for a strategy in a *game against nature*, where a guaranteed plan includes subplans for all combinations of sensor data.

1.1.5 Binary Metrics For Plans

The emphasis on guaranteed plans amounts to a binary metric that classifies plans as either guaranteed or not guaranteed. This metric relies on a worst-case analysis that implicitly uses binary models of uncertainty and cost.

This binary model of uncertainty is related to the approach used in mechanical design, where worst-case *tolerance margins* are specified for each component such that performance is guaranteed when these tolerances are maintained [Requicha, 1983, Lange, 1984]. Taylor [1976] developed a plan checker that propagates numerical tolerances through a multi-step assembly plan. Brooks [1982] extended Taylor's approach to handle symbolic tolerances and showed that backward propagation can be used to give initial tolerances that will guaranteed a desired set of final tolerances. Hutchinson and Kak [1990] developed an assembly planning system that propagates tolerance margins. To identify guaranteed plans, the geometric theory of manipulation identifies a subset of state space that encloses all possible states that the physical system can be in. The complement of this subset is the set of states that the physical system cannot be in. This constitutes a binary model of uncertainty.

Under the geometric theory of manipulation, we identify a subset of state space that we label the *goal set*. All states included in this goal set are acceptable, and all other states are not. In the terminology of utility theory, this is a binary cost function.

Combining a binary model of uncertainty with a binary model of cost, we can identify plans such that even with worst-case uncertainty, the final state will be classified as a goal state. Such

plans are guaranteed to succeed. All plans that do not meet this criterion are lumped together as plans that are not guaranteed to succeed.

This binary metric for plans presents two fundamental limitations.

1. "It is simply not always possible to find plans that are guaranteed to succeed" [Donald, 1987]. The initial subset of state space may be such that no sequence of available commands can map it into the goal set. In such cases there is no way to choose among plans because all plans are equal under the binary metric.

2. A guaranteed plan may be extremely costly in terms of time or resources. With the binary metric, all guaranteed plans are equal; there is no way to compare the cost of plans.

In response to the first limitation, Donald proposed the class of *Error Detection and Recovery (EDR)* plans. EDR plans are allowed to fail, but must fail recognizably. Donald formalized geometric conditions for recognizing failure and developed methods for recovering once failure is detected. Donald's formalism for addressing failure greatly expands the scope of the geometric theory of planning. However this formalism still relies on binary models of uncertainty and cost – an EDR plan is *guaranteed* to either succeed or fail recognizably.

1.1.6 Real-Valued Metrics for Plans

Another approach to cases where a guaranteed plan does not exists is to use *randomized* plans. In his thesis, Erdmann [1989] explored many cases where randomization can be useful for robot planning. For example, in order to mesh two gears together, it may be far more efficient to randomly jiggle the gears until they mesh rather than trying to precisely align and mate them. Erdmann noted that as long as a randomized plan has nonzero probability of success, then by iterating the plan until it succeeds, we can say that the iterative plan will succeed with probability one.

To analyze the probability that a plan will succeed, we need a probabilistic model of uncertainty. This is a real-valued version of the binary model of uncertainty; instead of classifying states as possible or impossible, we can express degrees of likelihood. Probabilistic models are often used to describe uncertainty in mechanical systems.

> "No matter how many times we flip a coin and observe the outcome, we are unable to predict exactly the outcome of the next flip. The reasons for our inability to predict are varied: we may not know all of the causal forces at work; we may not have enough data about the initial conditions in the problem; the forces at work may be so complicated that a calculation of their combined effect is not feasible; or there actually may be some basic indeterminacy at work in the problem at hand. We shall call such unpredictable phenomena *random phenomena*." [Davenport, 1970]

Probability theory offers well-established methods for estimating the expected behavior of unpredictable phenomena.

The second limitation of the guaranteed plan metric, concerning the cost of plans, is addressed by introducing a cost metric on actions. We might associate a temporal cost with each action and search for the fastest plan. Similarly, we can introduce a cost metric on final states; rather than the binary distinction between goal and non-goal, we can relate cost to a real-valued distance metric. This is often appropriate in the context of robot planning, for example we want to move a block to location θ. If there is too much uncertainty to guarantee location θ, we'd like to get the block as close to θ as possible. We can capture this naturally by relating the cost of the outcome to the distance from the goal. We can use this measure of success to formalize the idea of "failing gracefully"; instead of thinking only in terms of success or failure, we can indicate degrees of success.

By assigning a scalar cost function to actions, we can allow tradeoffs between effort and outcome. For example, we can compare a fast plan that comes within a foot of the goal with a slower plan that comes within an inch. This also allows us to define tradeoffs between multiple goals.

Cost and probability metrics for plans address two of the limitations of the binary metric, but how do we combine cost and probability? Intuition suggests that there is a tradeoff between cost and the probability of success. For example, the null plan – do nothing – is inexpensive but may be unlikely to achieve a desired state. An approach to combining cost and probability metrics into a single real-valued metric for plans can be found in the theory of games and decisions.

1.2 The Theory of Games and Decisions

The problem of choosing among strategies appears in many contexts: e.g. game theory, decision theory, and control theory. A cost function (payoff matrix, loss function, performance index) is used to rank outcomes. Cost is simply negative utility. Decision theory ranks strategies on the basis of *average performance*, where the average is based on a probability distribution over the state space. *Bayesian* decision theory refers to approaches where the probability distribution is based on *a priori* information [Berger, 1985]. These theories have been applied to problems in robot sensing.

1.2.1 Estimation and Sensor Planning

Probabilistic techniques have been applied to problems in robot vision and navigation with good results. Optimal estimation has a long history going back to the least-squares and maximum-

likelihood estimators ([Gauss, 1809] and [Fisher, 1912]); see Gelb [1986] for an overview. For robot navigation, Smith and Cheeseman [1986] and Ayache and Faugeras [1987] applied a maximum likelihood estimator to propagate first and second moments using a sequence of noisy observations.

Bayesian decision theory has been successfully applied to problems in robot sensing. Durrant-Whyte [1988] used it to develop a unified approach to combining and transforming uncertain geometric models. Hager [1988] applied both Minimax and Bayesian decision theory to a general class of sensor fusion problems. Given a class of part models, a prior distribution, sensor models, noise models and a utility function, an optimal sensing strategy can be defined. When computing the posterior distribution from a single observation, Hager showed that minimum mean squared error estimation is not robust to variations in a non-linear sensor model. He then developed a finite-element implementation of Bayes' theorem that is computationally expensive but suited to parallel implementation. One application is estimating the width and orientation of a rectangular part with a mobile camera. Time cost is attached to each observation and the strategy terminates when the expected gain from the next (optimal) observation falls below a predetermined threshold. See also Hager and Mintz [1989].

Szeliski [1988] has applied Bayesian decision theory to low-level vision, using a Markov Random Field to capture smoothness constraints in array of depth estimates. He considered a variety of loss (energy) functions based on surface models. Matthies, Szeliski, and Kanade [1987] developed a generalization of the Kalman filter for incrementally updating dense estimates and used it to extract depth estimates from a mobile camera. Other Bayesian approaches to sensor noise can be found in papers by Bolle and Cooper [1986], Matthies and Shafer [1987], Buckley [1988], Moravec [1988] and Cameron and Durrant-Whyte [1989].

Decision theory has also been used to generate stochastically optimal image-processing strategies for determining part orientation based on visual data [Hong, Ikeuchi, and Gremban , 1990]. The set of possible initial orientations can be split different ways using alternative visual operators, each with an associated cost. The authors search for a sequence of operators that will determine the final orientation in minimal expected time, where the expectation is based on a uniform probability distribution.

1.3 This Thesis

The connection between decision theory and robot sensing brings us full circle, back to uncertainty. Our review began by identifying two approaches to coping with uncertainty: sensing and mechanical compliance. We described an existing geometric approach to the latter and noted two limitations due to the binary metric for plans. The binary metric arises from a binary model of uncertainty and a binary model of cost. This led us to consider the theory of games and decisions which combines utility and probability to produce a real-valued metric for choosing

among strategies. It turns out that decision theory has been successfully applied to planning with sensors, and so it is natural to apply this theory to planning open-loop actions.

1.3.1 Relation between Sensing and Acting

When uncertainty is represented with a probability distribution on the state space, sensors and actions both have the effect of *transforming* the probability distribution.

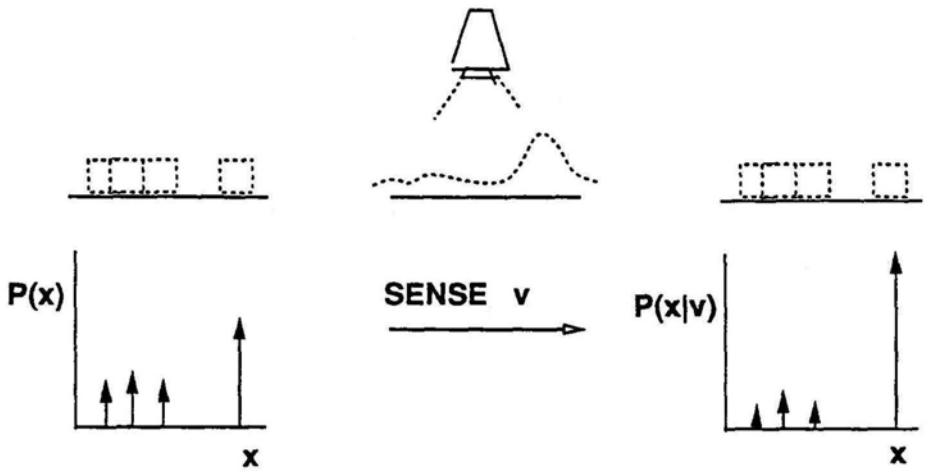


Figure 1.2: A sensor reading provides a mapping from one probability distribution to another.

Consider a block lying on a table where the position of the block is uncertain. Say we know that the block is in one of 4 positions, as shown with four "virtual" blocks in the upper left of Figure 1.2. Let each of the four positions have associated probabilities as shown in at the left..

Sensing changes the probability distribution. For example consider a camera that returns some noisy image v, as shown with dotted lines, that implies that the block is more likely to be to the right. Given a probabilistic model of the camera, the new probability distribution (as specified by Bayes' Law) might be the one shown at the right of Figure 1.2.

Now consider the same situation as before, but instead of sensing with a camera, a robot gripper reaches down over the set of virtual blocks and closes *without sensing* as shown in Figure 1.3. The action has the effect of "scooping up" two of the virtual blocks so that the block is more likely to be at the right. The resulting probability distribution shows the effect of adding the appropriate probabilities. Thus the open-loop action also has the effect of changing the probability distribution.

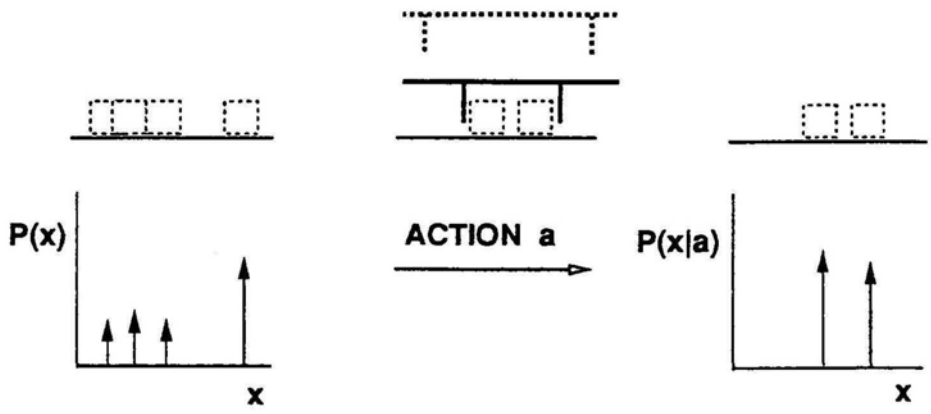


Figure 1.3: Similarly, an open-loop action provides a mapping from one probability distribution to another.

Both sensing and open-loop actions can be used to reduce uncertainty in a system. Viewed in terms of probability, both have the effect of transforming a probability distribution on the state space. Stochastic approaches have been useful for planning sensing strategies. This thesis explores the idea that a stochastic approach can be useful for planning open-loop strategies.

1.4 Other Work

In this section we describe several other areas of research that are relevant to the subject of stochastic planning.

1.4.1 Active Sensing

Bajcsy *et al.* [1989] used the term *active sensing* to refer to systems where sensors are servoed so as to acquire data in an efficient manner [Allen, 1987, Stansfield, 1987] Examples are data-driven *exploratory* motions, such as contour following [Goldberg and Bajcsy, 1984]. Koutsou [1988] used a parallel-jaw gripper equipped with force and tactile sensors to explore the weight, hardness, and shape of unknown parts.

This approach requires that the next sensor probe be chosen based on current data. In a factory environment where plans will be repeated many times, it may be worthwhile to generate an optimal sensor plan off-line. We discuss a stochastic approach to sensor planning in chapter 7.

1.4.2 Domain-Independent Planning in A.I.

Planning is also studied outside the geometric context of robotics. Feldman and Sproull [1977] published an early article on planning that advocated the use of decision theory with numerical models rather than the purely symbolic (logical) techniques that still dominate most work in domain-independent planning [Genesereth and Nilsson, 1987]. Cheeseman [1985] argues that probability theory subsumes many recent approaches to uncertainty in the AI literature. Stochastic approaches to domain-independent planning have been described in [Russell and Wefald, 1988, Pearl, 1988, Drummond and Bresina, 1990, Hansson *et al.*, 1990],

1.4.3 Stochastic Optimal Control Theory

Strategies that optimize an expected performance criterion are encountered in stochastic optimal control theory [Stengel, 1986, Bertsekas, 1987] where low-level systems are described by a set of continuous differential equations. Stochastic optimal control theory treats state uncertainty as additive noise, often Gaussian, as in the well-studied class of L-Q-G problems (Linear system, Quadratic loss function, Gaussian noise).

1.4.4 Algorithmic Complexity Theory

Robot plans can be viewed as algorithms. Recall that guaranteed plans do not always exist and when they do, they may be extremely costly. Searching for guaranteed plans is related to a *worst-case* analysis of algorithms. The worst-case analysis proceeds by imagining "an infinitely intelligent adversary who knows the structure of the algorithm and chooses inputs that will embarrass it to the maximal extent" [Karp, 1986]. Algorithms with acceptable worst-case performance, when they exist, may be inefficient for average-case problems.

Stochastic plans are related to algorithms with good average-case performance. An example of such a probably-fast algorithm is the widely-used Simplex algorithm for linear programming that requires exponential time to solve pathological problems but requires only quadratic time for "average" problems.

Stochastic plans are also related to fast algorithms that are allowed to fail with some small probability, known as probably-correct, or *Monte Carlo* algorithms. Perhaps best known is Rabin's probably-correct algorithm to test primality [Rabin, 1980].

Probabilistic algorithms have received a good amount of attention in the recent literature [Brassard and Bratley, 1988]. Average-case analysis is sometimes criticized on the grounds that its prior models are unrealistic. One response is to inject randomness into an algorithm to justify a probabilistic analysis. For example, Quicksort uses randomized pivots so the average-case running time is $O(n \log n)$ regardless of the distribution of inputs [Knuth, 1973]. In Chapter

4 we choose a grasp angle at random to justify a probabilistic analysis of stochastic grasping plans.

1.4.5 Statistical Decision Theory

Robot plans can be viewed as decision procedures. The branch of statistics known as *decision theory* is concerned with finding decision procedures that are statistically optimal with respect to a given cost measure. To decide between two competing processes for etching integrated circuits, for example, we could compare them on the basis of expected cost per functioning chip. In decision theory there are three approaches to choosing an optimal strategy:

1. The *classical* approach looks for a strategy that is superior to all other strategies for all inputs. For many problems, such a strategy does not exist.

2. The *minimax* approach looks for a strategy that minimizes *worst-case* performance. This approach is conservative: "the one situation where this is clearly appropriate is when the [input] is determined by an intelligent opponent who desires to maximize your loss" [Berger, 1985, p.308]. Minimax approaches are particularly relevant in game-playing situations.

3. The *Bayesian* approach looks for a strategy that minimizes *average-case* performance where the average is based on a prior distribution.

Guaranteed plans are related to minimax strategies in that both are evaluated on the basis of worst-case performance. One difference is that the minimax approach often uses a real-valued cost function to measure worst-case performance. The guaranteed planning framework uses a binary success/fail metric so that there is no way rank plans with the same worst-case performance. That is, if two plans fail in the worst case, there is no basis for choosing the lesser evil. Stochastic planning is closely related to Bayesian decision theory in that both require a probabilistic model to evaluate average-case performance. See Appendix A for more on statistical decision theory.

1.5 Overview of the Thesis

In this chapter we introduced stochastic plans and reviewed related work. In Chapter 2 we formalize a framework for ranking plans and define optimal single- and multi-step plans. In Chapter 3 we introduce the problem of stable grasping with a parallel-jaw gripper and show that guaranteed plans do not exist when orientational uncertainty exceeds a well-defined bound. This motivates us to consider probabilistic models and the random grasp. We derive a lower

bound on the probability that a grasp is stable as a function of the coefficient of friction between the part and the gripper. Noting that this probability goes to one as the coefficient of friction goes to zero, we introduce the frictionless gripper.

In Chapter 4 we focus on the issue of state uncertainty and consider multi-step plans for feeding polygonal parts. We use part geometry to rank plans on the basis of expected throughput and search for plans that are stochastically optimal. Chapter 5 introduces a fast backchaining algorithm for finding stochastically optimal parts-feeding plans. The algorithm leads to a proof that every polygonal part can be oriented up to symmetry.

Chapter 6 explores the use of empirical estimation to model control uncertainty. We consider the problem of orienting a polygonal part by tilting it in a tray through a sequence of angles. Control uncertainty arises from several sources including friction and impact, making it difficult to analytically model the trajectory of the part. We show how a Bayesian estimator can be used to build an empirical model based on physical experiments. We apply the framework of Chapter 2 to find stochastically optimal plans for orienting the part. In Chapter 7 we discuss some limitations to stochastic planning and show how the stochastic framework can be extended to incorporate sensors.

Appendices A-H provide: an overview of statistical decision theory, a physical implementation for a frictionless gripper, a tighter lower bound on the probability that a random grasp is stable, an $O(n \log n)$ algorithm for computing the diameter function, an algorithm for planning a sequence of *push-grasp* actions, a method for extending the backchaining algorithm to cases where the prior is non-uniform, examples of the Dirichlet distribution, and a list of notation.

Chapter 2

Stochastic Framework

> "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*?" [Donald, 1987].

In this chapter we propose a framework for planning open-loop robot manipulation sequences when there is uncertainty in state and control. The framework provides a formalism for ranking plans on the basis of *cost* and *probability* so that we can relax the restriction that plans must be guaranteed to succeed. The framework is based on well-established methods for the control of Markov processes [Howard, 1971, Dynkin and Yushkevich, 1979]. We focus on cases where the state and action spaces are finite and there is no observation of state during plan execution.

In a physical manipulation problem we are uncertain about the initial state of the system. We have a set of actions to choose from, however we are uncertain about the effect of these actions. Given a desired final state, what is an optimal sequence of actions to perform?

Suppose we treat the state of a system as a random variable that changes after an action. The random variable that results from an action is determined by transition probabilities. The sequence of random variables is a *stochastic process*. If the probability distribution for each variable depends only on the action and previous variable, then the sequence is a *Markov* chain. A Markov chain can be *controlled* by the choice of actions. When we do not know the state of the system after each action, we say that the Markov chain is *unobserved*.

When the following conditions are satisfied, a manipulation planning problem can be related to finding an optimal control policy for an unobserved Markov chain.

1. The set of states is finite.
2. We can describe the initial state of the system with a probability distribution.
3. The set of actions is finite.

4. We can describe the effect of an action with a conditional probability distribution that depends only on the current state.

5. We can characterize the cost of actions and a preference among outcomes with a scalar cost function.

6. We cannot sense the state of the system during execution of the plan.

Conditions 1 and 3 require that the state and action spaces be finite. Since the state and action spaces for manipulation problems are often related to the continuous spaces of translations and rotations, we must partition these spaces into a finite collection of subsets. This can be done with an arbitrary uniform grid or preferably, with a partition that depends on the geometry and mechanics of the problem.

We first consider one-step plans and then generalize to multi-step plans.

2.1 One-Step Plans

A one-step stochastic planning problem is a five-tuple $(\Theta, \lambda_0, \mathcal{A}, \mathcal{P}, C)$, specified as follows.

1. State Space, $\theta \in \Theta$
2. Prior Probability Distribution, $\lambda_0 \in \Lambda$
3. Action Space, $a \in \mathcal{A}$
4. Set of Transfer Matrices, $\mathcal{P} \ni P_a : \Lambda \mapsto \Lambda$
5. Cost (Loss) Function, $C : \mathcal{A} \times \Lambda \mapsto \Re$

The state θ is a vector describing the system that we wish to manipulate, e.g. the position and orientation of an object. The finite set of states is the state space, Θ. Let $n = |\Theta|$.

The prior probability distribution expresses uncertainty about the initial state of the system. A discrete probability distribution is specified by assigning a real number to each state; let p_i be the probability that the system is in state θ_i. Let λ refer to a vector of probabilities $[p_1, p_2, ...p_n]$ such that $p_i \geq 0$ for $i = 1, ..., n$, and $1 = \sum_{i=1}^{n} p_i$. Following the terminology of adaptive control theory, we refer to a probability distribution on Θ as a **hyperstate** [Astrom, 1987]. Let λ_0 refer to the initial hyperstate and let Λ represent the set of all hyperstates.

The action space is the finite set of actions that we can choose from. The effect of an action is expressed with a *stochastic transition matrix*, P, where p_{ij} is the probability that the system will be in state θ_j after the action is applied to state θ_i. Note that this probability depends only

on the current state and the applied action – this is known as the Markov property. The result of applying action a to λ is given by post-multiplying λ by P_a,

$$\lambda' = \lambda P_a. \tag{2.1}$$

Every action can be viewed as a mapping between hyperstates. This is a vector form of the equation

$$P(\theta_j|a) = \sum_{i=1}^{n} P(\theta_i)P(\theta_j|\theta_i, a). \tag{2.2}$$

Of course optimality must be defined with respect to some measure of performance. A scalar cost function allows us to rank one-step plans by considering both the cost of the action and the cost (negative utility) of the outcome. We express preferences between outcomes with a cost function that measures the "distance" from a desired final state. For example, we can express a strong desire for a particular goal state with a binary cost function that assigns zero cost to the goal state and unit cost to all other states.

The result of applying an action is a probability distribution on the set of states. Since the true final state is uncertain, we define the cost function on Λ, the set of hyperstates. We might assign a cost to individual states and define the cost of a hyperstate to be its average, or expected, cost. For example, when the objective is accuracy, the cost of a hyperstate could be related to its expected distance from a desired state. For succinctness, we combine the cost function for actions and the cost function for hyperstates into a single cost function.

Given the 5-tuple, $(\Theta, \lambda_0, \mathcal{A}, \mathcal{P}, C)$, we want to find an optimal plan. To compare plans, we compare their costs. For a given initial hyperstate λ_0, the cost of executing action a would be $C(a, \lambda_0 P_a)$, that is, the cost of the action itself and the cost of its outcome. An optimal one-step plan, a^*, is an action with minimal cost:

$$a^* = \arg\min_{\mathcal{A}} \; C(a, \lambda_0 P_a). \tag{2.3}$$

2.2 Multi-Step Plans

In this section we extend the framework to plans with more than one action. A multi-step plan is a sequence of actions. Since we have no sensory information about the state of the system during the plan, the sequence of actions is fixed (open-loop). As in the one-step case, a multi-step stochastic planning problem is a five-tuple $(\Theta, \lambda_0, \mathcal{A}, \mathcal{P}, C)$, specified as follows.

1. State Space, $\theta \in \Theta$

2. Prior Probability Distribution, $\lambda_0 \in \Lambda$

3. Action Space, $a \in \mathcal{A}$
4. Set of Transition Matrices, $\mathcal{P} \ni P_a : \Lambda \mapsto \Lambda$
5. Cost (Loss) Function, $C : \mathcal{A} \times \Lambda \mapsto \Re$

The state space and prior probability distribution are the same as for one-step plans, but the space of plans is the set of all sequences of actions. Since the set of actions is finite, the set of plans is countable.

The transition matrix for a plan is the product of the transition matrices of its actions. That is, for a plan $\rho =< a1, a2, a3 >$, the final hyperstate would be

$$\lambda' = \lambda_0 P_\rho = \lambda_0 P_{a1} P_{a2} P_{a3}. \tag{2.4}$$

Each plan is a mapping between hyperstates.

The cost function for multi-step plans is similar to that for one-step plans. To take into account the effort expended in performing a plan, the cost is usually the sum cost of all actions in the sequence plus the cost of the final hyperstate.

To compare plans, we compare their costs. For a given initial hyperstate λ_0, the cost of executing plan ρ would be $C(\rho, \lambda_0 P_\rho)$, that is, the cost of the actions plus the cost of the final hyperstate. An optimal plan, ρ^*, is a plan with minimal cost:

$$\rho^* = \arg\min_\rho \; C(\rho, \lambda_0 P_\rho). \tag{2.5}$$

2.3 Iterative Plans

If at first you don't succeed, try try again.

Often we want to achieve a desired outcome. In cases where we have a binary test that accepts a desired subset of final states and rejects all others, we can consider plans that loop until the system succeeds in passing the test. We call these *iterative plans*. If each iteration has some probability of being successful, the interactive plan will succeed with probability one.

One way to test for success is with a binary sensor. An alternative method is to design a mechanical loop, for example a binary mechanical filter that causes parts to fall off a conveyor belt unless they are in a desired orientation. Parts that fall off are then recycled until they pass the filter. With this approach no sensors are required.

How many iterations will the plan execute until success is achieved? Consider an iterative plan where the probability of achieving a desired state on any iteration is p. It is a well-known

result from probability that in a sequence of independent and identically distributed iterations, the expected number of iterations until the desired state is achieved is

$$\begin{aligned} E(n) &= p1 + p(1-p)2 + p(1-p)^2 3 + \ldots \\ &= p(1 + 2(1-p) + 3(1-p)^2 + \ldots) \\ &= p/p^2 \\ &= 1/p. \end{aligned}$$

For example, when rolling an unbiased die, the probability of rolling a 3 is $1/6$. *On average* it takes 6 independent rolls to observe a 3. This assumes that each iteration is independent, that is, there is no coupling between the state of one iteration and the next. We can justify this assumption by *randomizing* the state between iterations, perhaps by intentionally jumbling the part. By defining a cost function that is related to the expected number of iterations, we can define stochastically optimal iterative plans. This will be explored in Chapter 4.

2.4 Finding an Optimal Plan

Finding an optimal plan may require substantial computation. The brute force method is to compute the cost of every plan and choose one with minimal cost, breaking ties arbitrarily. In a manufacturing setting, the optimization is performed off-line allowing us to amortize computation time over hundreds of execution cycles.

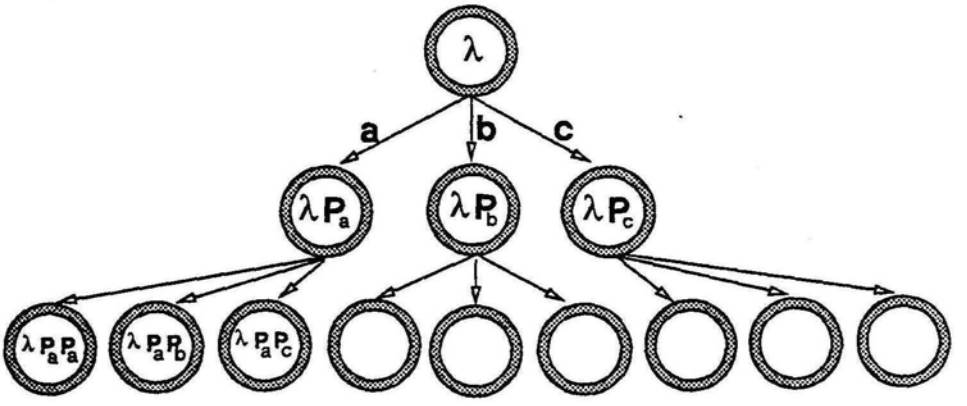


Figure 2.1: We can represent the set of open-loop plans with a tree.

We can visualize the search for an optimal plan as a search through a tree or graph where each node represents a hyperstate. The root node represents the initial hyperstate. Each action provides a link to another hyperstate and so on. Every path in the tree corresponds to a plan. The cost of a plan depends on the path and its terminal node.

To find the best multi-step plan, we cannot consider all plans because plans can be of arbitrary length – there is a (countably) infinite number of plans to consider. If the cost function for states and actions is additive and bounded from below, we can bound the length of plans since after a certain point, the cost of an additional action will outweigh any possible improvement in the hyperstate.

We could reduce planning time in many cases with a best-first branch-and-bound method as used by Hong, Ikeuchi, and Gremban [1990], where we avoid expanding paths that can't possibly be better than the current best path. Under this method, we expand the last node in the best (cheapest) current path. We continue until we reach a leaf node (from which we can't improve the hyperstate) and continue searching until all incomplete paths are at least as expensive as the minimum cost to a leaf. The speedup over exhaustive search depends on the ordering of nodes and the worst-case running time is the same. This method is closely related to the well-known Alpha-Beta search method for searching game trees [Pearl, 1984].

There are some theoretical lower bounds on the worst-case computational complexity of finding an optimal control policy for an unobserved Markov chain. When system state is perfectly observed at each stage an optimal policy can be found in polynomial time by dynamic programming techniques [Bertsekas, 1987] or linear programming. For cases where the system state is only partially observed (in a known element of a partition) at each stage, the problem is *PSPACE-Hard*. For cases such as ours where the system state is unobserved at each stage, the problem can be shown to be *NP-Complete* [Papadimitriou and Tsitsiklis, 1987], suggesting that an algorithm with good worst-case running time may not exist. See Section 7.1.2 for more on the cost of planning.

2.5 Discussion

In this section we describe how the framework in this chapter is related to planning with perfect information, Bayesian decision theory, and finding guaranteed plans. We also comment on the challenge of extending the framework to continuous state and action spaces.

2.5.1 Planning with Perfect Information

When we have perfect information – there is no uncertainty in state or control – the probabilities become degenerate. We can represent this in the same framework by treating the probability vector as a binary vector with a single one corresponding to the initial state. If we can perfectly predict the effect of an action, then its transition matrix will be binary. We refer to such actions as *deterministic*, since the next state is perfectly determined. In this case every plan is mapping between states and we want a plan that maps the initial state into a desired final state. Note that finding such a plan may still be non-trivial.

2.5.2 Relation to Decision Theory

The framework is related to Bayesian decision theory. Decision theory is concerned with the problem of making decisions, or choosing between alternative actions assuming that the action will have no effect on the state.

The cost function is defined over each pair of states and actions – once we know the state, we can choose the best action. We are uncertain about the exact state but can treat it as a random variable. In this case we can use its probability distribution to calculate an *expected* cost for each action and choose the action that will maximize expected cost. An idea added by *statistical* decision theory is that we can make the action depend on the result of some statistical experiment, referred to as the *data*. Problems without this complication are referred to in statistical decision theory as *no-data* problems [Berger, 1985].

Bayesian decision theory says that we can combine a prior model of state with statistical data to determine a posterior model of state. Let θ refer to any state of nature from a set of possible states Θ and $\lambda(x|\theta)$ be the probability of observing data x given that the true state of the system is θ. We use Bayes' Theorem to derive $\lambda(\theta|x)$, the probability that the state is θ if we observe data x. Let a refer to an action from a set of possible actions $\mathcal{A}$. Let $C(a,\theta)$ be the cost associated with a particular combination of action and state. An optimal action, a^*, is an action that minimizes expected cost:

$$a^* = \arg\min_{a \in \mathcal{A}} \sum_{\theta \in \Theta} C(a,\theta)\ \lambda(\theta|x). \tag{2.6}$$

To compare the framework for planning open-loop manipulation plans with the Bayesian framework, consider a manipulation planning problem where the cost of a hyperstate is its expected cost,

$$C(a,\lambda) = \sum_{\theta \in \Theta} C(a,\theta)\ \lambda(\theta|a), \tag{2.7}$$

where $\lambda(\theta|a)$ is the probability of state θ given action a. Substituting (2.7) into (2.3),

$$a^* = \arg\min_{a \in \mathcal{A}} \sum_{\theta \in \Theta} C(a,\theta)\ \lambda(\theta|a). \tag{2.8}$$

Comparing equations (2.6) and (2.8) we see that the optimal action is similar in both cases. The difference is that the posterior distribution depends on the *data* in the decision theory problem and on the *action* in the manipulation planning problem. Jeffrey [1965] discusses how decision theory can be reformulated to include the effect of actions on the posterior distribution.

2.5.3 Guaranteed Plans as a Special Case

We can use the same framework to find guaranteed plans by letting the probability be uniformly distributed among all possible initial states and letting the transition matrices map the probability

evenly among all possible outcomes. If we define the cost function to be zero for goal states and infinity for all other states, and let the cost of actions be zero, then any plan with any chance of failure will have infinite cost. Any plan with zero cost is guaranteed to reach the goal. It may be the case that there is no plan with zero cost, in which case all plans have infinite cost and are equivalent.

2.5.4 Extension to Continuous State and Action Spaces

We assume that state and action spaces are discrete. The underlying state space may be continuous, but we use geometric methods to partition this space into a finite collection of subsets and assign a probability to each subset.

Generalizing the framework to continuous spaces introduces a variety of measure-theoretic issues. For example for each action we must define a *probability space* including a countable set of events and a probability measure over this set of events. We must take care in defining conditional probabilities and the cost must be measurable function from the probability space into the real line [Bertsekas, 1987, Gray and Davisson, 1986]. Dynkin and Yushkevich [1979] extends the control framework to semi-continuous and Borel processes.

In this chapter we specify a stochastic framework for manipulation planning based on the theory of Markov chains. We define a planning problem as a five-tuple consisting of a state space, prior probability distribution, action space, set of transfer matrices, and a cost function. Planning is accomplished by propagating probability distributions on the state space (hyperstates). For cases where plans either succeed or fail, we define iterative plans and propose that iterative plans be evaluated based on expected throughput, which in turn depends on the probability of success in a single iteration. The probability of success is based on a plan's final hyperstate.

To be useful, the framework requires a good probability model. In the next chapter we give an example where a probability model can be derived from part geometry.

Chapter 3

Random Grasping with Friction

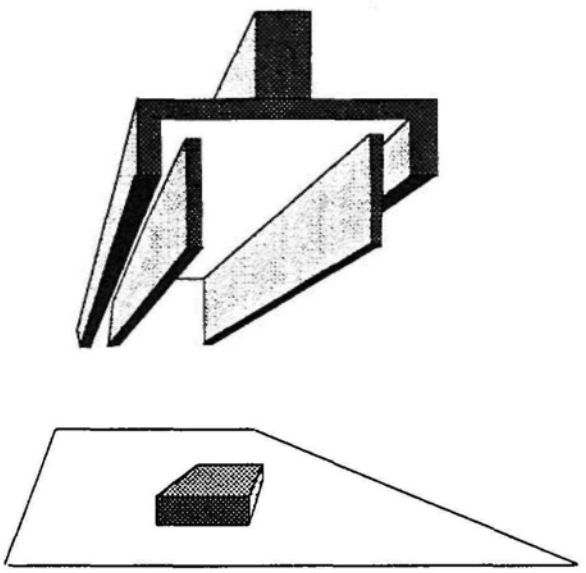


Figure 3.1: Schematic of a parallel-jaw gripper.

3.1 Introduction

In this chapter we apply a stochastic model to a simple grasping problem where a desired outcome cannot be guaranteed. Consider a polygonal part and a parallel-jaw gripper as in Figure 3.1. The frictional coefficient between part and gripper is given. We want to find a single-step plan (an orientation for the gripper) such that the final grasp configuration is stable (where at least one edge of the part is aligned with the gripper). The catch is that the part's exact initial orientation cannot be predicted.

Brost [1988] developed a geometric method to address this problem where uncertainty in part orientation is described with a tolerance interval (*e.g.* $\pm 10°$). This method finds gripper orientations that are guaranteed to produce a stable grasp configuration when the part's initial orientation is within the tolerance interval as in figure 3.2. Brost noted that if the tolerance interval is sufficiently large, then it may be impossible to guaranteed a stable grasp as in figure 3.3.

Figure 3.2: Top view of the parallel-jaw gripper grasping a part (shaded square). Small variations on the part's initial orientation are cancelled out during grasping so that the final orientation is guaranteed to be as shown on the right.

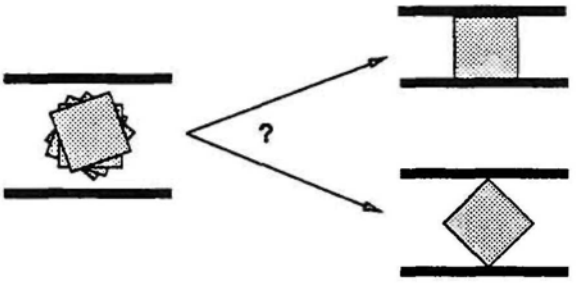


Figure 3.3: In this case large variations in the part's initial orientation make it impossible to guaranteed the final orientation of the part.

For such cases we extend Brost's method by treating part orientation as a *random variable*. We derive a lower bound on the probability that a *random grasp* will be stable. This lower bound is a function of part geometry and the coefficient of friction. We find that this lower bound goes to one as the coefficient of friction goes to zero, which suggests a modification to the parallel-jaw gripper that we call the *frictionless gripper*.

In the next section we describe related research on the mechanics of manipulation. We then specify assumptions and define terms. In section 3.5 we show how control uncertainty arises from friction. In section 3.6 we review Brost's method and find an upper bound on the orientational uncertainty that can be tolerated in order to guarantee a stable grasp. When this upper bound is exceeded we cannot *guarantee* a stable grasp but we can find a lower bound on the *probability* that a grasp will be stable. In sections 3.8 and 3.9 we introduce a probabilistic

model of the initial state and use it to find a lower bound on $P(S)$, the probability that a grasp will be stable. In section 3.10 we report this lower bound for a variety of part shapes.

3.2 Related Work

There is a substantial literature on the subject of grasping. See Grupen *et al.* [1989] and Pertin-Troccaz [1989] for reviews.

3.2.1 Grasp Stability

The *stability* of a grasp configuration can be defined in several ways. Hanafusa and Asada [1977] defined a grasp configuration to be stable when a small deviation from the configuration produces restoring forces. They used a potential function to locate stable configurations for three spring-loaded frictionless point fingers surrounding a planar part. This idea was extended by Baker, Fortune, and Gross [1985], who showed that a stable three-fingered grasp for any n-sided polygon can be found in time $O(n \log n)$ by finding the maximum inscribed circle. Jameson [1985] stated that a grasp has *second-order stability* if the part will return to an equilibrium position after a perturbation of its position with respect to the gripper. Nguyen [1989] generalized Hanafusa and Asada's potential function and showed that a stable grasp must have a positive definite stiffness matrix. Other definitions of grasp stability can be found in [Salisbury, 1982] [Kerr, 1984] [Baker, Fortune, and Gross , 1985], [Abel *et al.*, 1985], [Mishra, Schwartz, and Sharir , 1987], [Nguyen, 1988], and [Cutkosky, 1985]. In this chapter we follow Brost [1988] in defining a parallel-jaw grasp as stable when at least one edge of the part is aligned with the gripper jaws.

Other measures of grasp quality have been suggested. Markenscoff and Papadimitriou [1989] suggested a quality measure for grasping based on the gripping force needed to balance the weight of a part. Barber *et al.* [1986] and Demmel and Lafferriere [1989] suggested a quality measure based on the coefficient of friction needed to maintain stability under a given force and torque. Li and Sastry [1988] proposed three quality measures: a worst-case measure of stability based on the smallest eigenvalue of the grasp matrix, a volumetric measure based on the product of eigenvalues, and a task-oriented measure based on the volume of the largest *task ellipsoid* that can be embedded in the force domain of a grasp.

3.2.2 Grasp Mechanics

To *achieve* a desired grasp configuration, we must control interactions between the part and the gripper. Sensors can be used to control part state by actively servoing gripper motion in response

to sensor data. This approach is known as *active compliance* or *closed-loop control*. Inoue [1974] considered the use of force feedback to eliminate uncertainty during grasping. Salisbury [1982] introduced the *grip Jacobian* and stiffness control for grasping. Active compliance for robots was also studied by Mason [1978] and Hogan [1984].

Another way to control part state is to use *passive compliance* or *open-loop control*, where part motion is constrained mechanically without the use of sensors. For example, Pingle *et al.* [1974] demonstrated that *pushing* can be used to eliminate uncertainty while grasping. Mason [1982] formalized the role of pushing in robot manipulation. The difficulty is that the motion of a pushed part on a planar surface with friction depends on the distribution of pressure between the part and the planar surface. In general, the distribution of pressure depends on the microscopic contact topology and will change as the workpiece moves. For example, a single grain of sand beneath the part's centroid can dramatically affect its rotational motion. In his dissertation, Mason showed that the sign of the rotation rate is independent of the distribution of pressure and depends only on the relative position of the part's center of mass. Erdmann [1984], Rajan, Burridge, and Schwartz [1987], and Brost and Mason [1989] developed graphical methods for determining the set of all possible forces that can arise due to Coulomb friction. Peshkin [1986] extended Mason's result using a minimum-work principle to bound the set of all possible rotations of a pushed part.

Whitney [1982] analyzed the forces that can arise during peg-in-hole insertion and developed the *remote center compliance* device that passively directs these forces toward insertion. ([Whitney and Junkel, 1982] incorporated a stochastic sensing model into the control of an instrumented *RCC*). The use of passive compliance for grasping was also studied by Fearing [1983], Chiu [1985] and Trinkle and Paul [1988]. The kinematics of grasping was studied by Kerr and Roth [1986], Cai and Roth [1987], and Montana [1988].

The application in this chapter builds directly on the work of Brost [1988], who developed a systematic approach to plan single-step grasping operations. He defined a grasp to be stable when small perturbations in orientation are cancelled with further squeezing. Starting with an interval of possible part orientations, Brost divided the grasping process into pushing and squeezing phases and applied Mason's results on the mechanics of pushing to select a grasp angle that is *guaranteed* to produce a stable grasp.

3.3 Assumptions and Problem Definition

In this chapter we consider two problems. In the first, we are given a list of vertices representing a polygonal part, the part's center of mass, the coefficient of friction, and a range of initial part orientations. We are asked asked to find a grasp angle that is guaranteed to produce a stable grasp for all orientations in the range. We show that a solution to this problem does not always

exist. We then consider a related problem: given the same input, compute a lower bound on the probability that a random grasp will be stable.

Definition 3.1 *A* **grasp** *consists of orienting the gripper around the part and then closing the jaws as far as possible.*

Definition 3.2 *A* **random grasp** *is a grasp where the gripper's orientation is selected from a uniform distribution on the set of planar rotations.*

Figure 3.4: We define the configuration on the left as *stable* and the configurations on the right as *wedged.*

The final configuration of the part within the gripper can be either *stable* or *wedged* (see Figure 3.4).

Definition 3.3 *The set of* **wedged** *orientations,* Θ_w*, includes all orientations where the part is cocked between exactly two vertices.*

Definition 3.4 *The set of* **stable** *orientations,* Θ_s*, includes orientations where at least one edge of the part is aligned with the gripper.*

Wedged configurations arise due to friction between the gripper and the part. Note that the set of wedged orientations is infinite and depends on the coefficient of friction. The set of stable orientations is finite (at most one per edge). Our objective is to achieve a stable orientation. Let $\Theta_f = \Theta_w \cup \Theta_s$.

We assume that:

1. The gripper has two linear jaws arranged in parallel.
2. The direction of gripper motion is orthogonal to the jaws.
3. The part is a rigid planar polygon of known shape.
4. The part's initial position is unconstrained as long as it lies somewhere between the two jaws. The part remains between the jaws.

5. All motion occurs in the plane and is slow enough that inertial forces are negligible. The scope of this *quasi-static* model is discussed in [Mason, 1986] and [Peshkin, 1986].

6. Once contact is made between a jaw and the part, the two surfaces remain in contact throughout the grasp. A grasp continues until further motion would deform the part.

7. The part's center of mass is known.

8. Friction between part and the jaws can be modelled with Coulomb's Law: $F = \mu N$, where the coefficient of friction is known and independent of velocity.

Only assumptions 2 differs from those made by Brost [1988]. Assumption 2 restricts the action space to the one-dimensional space of gripper angles (Brost treats the two-dimensional action space that also includes the direction of gripper motion). This restriction offers three advantages: the pushing analysis is independent of the coefficient of friction, the space of actions is easier to search, and the resulting motion is simple to implement on existing grippers, where jaw motion is rarely coupled to arm motion.

3.3.1 Coordinate Frames

Let ${}^W_p[x, y, \theta]_1$ describe the initial position and orientation of the *part* with respect to a fixed *world* frame. Let ${}^W_g[x, y, \theta]$ describe the position and orientation of the *gripper* in the same frame. Let $\theta_1 = ({}^W_g\theta - {}^W_p\theta_1)$ be the initial orientation of the part with respect to the gripper. That is, $\theta_1 = 0$ when the part's reference edge is aligned with the jaws. All angles are evaluated modulo 2π. Variables are defined in the gripper frame unless an explicit superscript appears.

As stated above, we assume that the part's initial position is somewhere between the two jaws and that the part remains between the jaws throughout grasping. The grasp proceeds by monotonically closing the jaws until further motion would deform the part. Thus a gripper angle defines a one-step grasp strategy.

3.4 Pushing and Squeezing Phases of Grasping

Grasp motion can be divided into two phases. First is the *pushing* phase, where the part moves due to contact with one jaw. This phase continues until the second jaw makes contact. Next is the *squeezing* phase, when the part moves due to contact with both jaws until it reaches its final orientation. Let θ_2 be the orientation of the part when the second finger makes contact, and θ_f be the part's final orientation.

For a given polygon, coefficient of friction, and gripper angle, both phases of the grasp define a mapping. The outcome of the pushing phase is described by the function

$$\theta_2 = G_p(\theta_1) = \theta_1 + \int_{\Delta y} \theta_1' dy \tag{3.1}$$

where θ_1 is the part's initial orientation, $\theta_1' = d\theta_1/dy$ is the part's rate of rotation relative to the jaw separation during the pushing phase, and Δy is how far the jaw travels during the pushing phase.

We define the effect of the squeezing phase with the function

$$\theta_f = G_q(\theta_2) = \begin{cases} \theta_2 & \text{if part is wedged,} \\ \theta_x & \text{otherwise.} \end{cases} \tag{3.2}$$

where θ_x is the next stable orientation as the part rotates due to squeezing.

3.5 Sources of Uncertainty in the Transfer Function

The rotation rate in equation (3.1) depends on the (time-varying) distribution of pressure at the contact between the part and the supporting surface and on the part's orientation. In practice, the exact rotation rate is almost impossible to predict [Mason, 1982], [Peshkin, 1986]. The amount of jaw travel during of the pushing phase is another source of uncertainty. It depends on the initial position of the part between the jaws.

We can compose G_p with G_q to define a mapping for the entire grasp: $G = G_q \circ G_p : \Theta_1 \rightarrow 2^{\Theta_f}$. G maps an initial orientation to a *set* of possible final orientations: for any particular trial with initial orientation θ_1, the outcome will be some $\theta_f \in G(\theta_1)$. The set of possible final orientations is the union over all possible values of Δy and θ_1'.

$$\Theta_f = G(\theta_1) = \bigcup_{\Delta y} \bigcup_{\theta_1'} G_q\left(\theta_1 + \int_{\Delta y} \theta_1' dy\right), \tag{3.3}$$

We desire a grasp angle such that $G(\theta_1) \subseteq \Theta_s$.

3.6 Review of Brost's Method

To motivate the probabilistic treatment in subsequent sections, we review a method reported by Brost [1988] for selecting a grasp angle such that the final grasp is guaranteed to be stable. His method assumes that the initial orientation of the part is known to within a tolerance interval. In this section we review Brost's method and find a bound on the maximal tolerance interval required to guarantee a stable outcome – when this bound is exceeded, we cannot guarantee a stable grasp and thus consider the *probability* of a stable grasp.

3.6.1 Analysis of Pushing Phase

Mason showed that when motion is orthogonal to the jaws, the sign of the rotation rate depends only the position of the center of mass with respect to the line parallel to the jaw velocity and passing through the contact point [Mason, 1982]. To treat uncertainty in the pushing phase, Brost incorporated Mason's result into the *push-stability* diagram: given part orientation and direction of pushing, the part's direction of rotation is explicit in the diagram.

The horizontal axis of the diagram is θ_1, the angle of the part with respect to the gripper. The push-stability diagram is formed by placing $\times$'s at each angle corresponding to an edge of the part, and $\diamond$'s at angles perpendicular to each angle where a vertex is aligned with the part's center of mass, taking care to delete $\times$'s that are unstable and $\diamond$'s when a neighboring edge makes it geometrically impossible to push toward the center of mass at a vertex. The direction of motion between any two marks is always toward the $\times$ (see Figure 3.5.) The push-stability diagram depends on part geometry and center of mass and is independent of the friction angle.

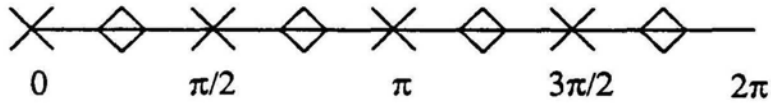


Figure 3.5: The push-stability diagram for a square. The orientation of the square wrt the gripper, θ_1, runs along the horizontal axis. The $\times$ at 0 radians corresponds to a stable pushing orientation. The $\times$ at the stable orientations is meant to suggest two converging arrows: a perturbation in orientation will produce rotation back toward this orientation. The $\diamond$ at $\pi/4$ corresponds to a jaw aligned with the diagonal – an unstable pushing equilibrium. Each $\diamond$ is mean to suggest two diverging arrows: a perturbation will rotate the square away from this orientation.

3.6.2 Analysis of Squeezing Phase

To treat the squeezing phase, Brost introduced the *squeeze-stability* diagram: the part's direction of rotation can be determined from the orientation of the part when both jaws make contact. Due to friction between the jaws and the part, the part will wedge between two vertices if the angle between opposing vertices and the perpendicular to the jaw is less than the friction angle, $\alpha = \arctan \mu$. The center of each wedging interval corresponds to the angle where two vertices form a perpendicular to the jaws. The width of each interval is determined by the friction angle (see Figure 3.6). The squeeze-stability diagram depends on part geometry and the friction angle and is independent of the part's center of mass.

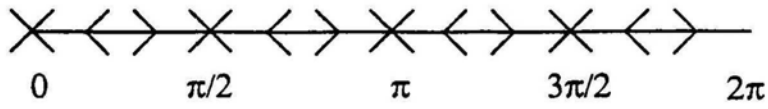


Figure 3.6: The squeeze-stability diagram for a square with $\mu = .25$. The $\times$ at 0 radians corresponds to the jaw aligned with an edge – a stable squeezing orientation. A perturbation in the orientation of the square will produce rotation back toward this orientation. The $<$ and $>$ between 0 and $\pi/2$ delimit a *wedging interval*: a set of orientations where the square can be cocked between a pair of diagonal vertices. The width of the wedging interval is $2 \arctan \mu$. Outside the wedging interval, the part will always rotate toward the nearest $\times$.

Brost combined these diagrams into the *squeeze-grasp* diagram (Figure 3.7). From this diagram one can find all the initial orientations of the part that are guaranteed to rotate into a stable final orientation. By choosing the orientation of the gripper so that the relative orientation of the part is one of these initial orientations, we can insure that the final orientation will be stable.

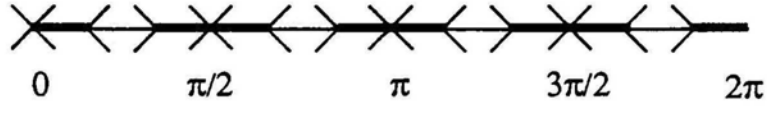


Figure 3.7: The squeeze-grasp diagram for a square with $\mu = .25$. Consider the stable orientation at $\pi/2$. The bold line surrounding this orientation is the *strong pre-image* of $\theta_f = \pi/2$. If θ_1 is anywhere in this interval, then the resulting grasp is guaranteed to be stable.

3.7 Necessary Conditions to Guarantee a Stable Grasp

In this section we define images and pre-images for the grasping problem. We will then relate the size of a strong pre-image to an upper bound on the orientational uncertainty required to guarantee a stable grasp.

3.7.1 Strong and Weak Pre-images of Θ_s

Recall that G is the mapping from initial orientation to the set of all of all possible final orientations. We define the *weak pre-image* of Θ_f as the set of all part orientations that *might* rotate into Θ_f,

$$G^{-1}(\theta_f) = \{\theta_1 | \theta_f \in G(\theta_1)\}. \tag{3.4}$$

The *strong pre-image* of orientation θ_f is the set of initial orientations θ_1 that are *guaranteed* to rotate into final orientation θ_f. We denote this set by

$$\Pi(\theta_f) = \{\theta | G(\theta) = \{\theta_f\}\} \tag{3.5}$$

See Figure 3.7.

Definition 3.5 *The strong pre-image of a set Θ_s is the set of all initial orientations that are guaranteed to rotate into some element of Θ_s:*

$$\Pi(\Theta_s) = \{\theta | G(\theta) \subseteq \Theta_s\}. \tag{3.6}$$

That is, if we begin with any orientation in $\Pi(\Theta_s)$ we are guaranteed to finish with some orientation in Θ_s.

Note that for any final orientation, its strong pre-image is always a subset of its weak pre-image: if θ_1 is guaranteed to rotate into θ_f, then certainly θ_1 *may* rotate into θ_f.

3.7.2 Upper Bound on Uncertainty in θ_1

Given uncertainty in part orientation (or equivalently, in gripper orientation) expressed as an initial tolerance margin for the relative orientation of the part, Brost showed that we can still guarantee a stable outcome by shrinking the strong pre-image by the size of the tolerance margin.

Let the tolerance margin be defined with an error radius, ϵ such that the initial orientation of the part is $\theta_1 \pm \epsilon$. As long as the strong pre-image is wider than 2ϵ, we can insure a stable grasp by choosing a gripper orientation so θ_1 is in the center of the interval.

We can use the squeeze-grasp diagram to determine an *upper bound* on the tolerance margin for a stable grasp to be guaranteed. Let Θ_{max} be the largest contiguous interval in the strong preimage. As long as

$$\epsilon \leq \frac{|\Theta_{max}|}{2}, \tag{3.7}$$

we can insure a stable grasp as above. What if uncertainty exceeds this amount? In particular, if we know nothing at all about the part's orientation, so that $\epsilon = \pi$, we cannot guarantee a stable outcome in the presence of friction.

3.8 Stochastic Analysis

For cases where we cannot guarantee a stable outcome, we might consider the *probability* of a stable outcome. We can treat each grasp as a trial where the outcome is either stable or wedged. We denote the sample space with the symbols $\{S, W\}$. Our results will be in terms of $P(S) = 1 - P(W)$, the probability of achieving a stable grasp.

To cope with the uncertainty in initial orientation, we treat θ_1 as a continuous random variable with probability density function (p.d.f.) $f_{\theta_1}(\theta_1)$. We find a lower bound on $P(S)$ by integrating $f_{\theta_1}(\theta_1)$ over the strong pre-image of Θ_S.

3.9 Lower Bound on $P(S)$

To determine a lower bound on the probability that a grasp will be stable, consider a case when the initial orientation happens to fall in the strong pre-image of Θ_S. By definition of the strong pre-image, this is a sufficient condition for a stable outcome. The probability that $\theta_1 \in \Pi(\Theta_s)$ is thus a lower bound on $P(S)$. Formally,

$$P(S) \geq \int_{\Pi(\Theta_s)} f_{\theta_1}(\theta_1)\, d\theta. \tag{3.8}$$

Another way to think of this is to consider the set of all initial orientations that might end up in a wedged final orientation: the weak pre-image of Θ_w. In the worst case all of these end up wedged, so the maximum $P(W)$ is $|G^{-1}(\Theta_w)|/2\pi$. Equation (3.8) follows by observing that the orientations guaranteed to succeed are precisely those that cannot end up wedged: $\Pi(\Theta_s) = \overline{G^{-1}(\Theta_w)}$.

Random Grasping

If nothing is known about the initial orientation of the part in the *world* frame it is tempting to assume a noninformative prior where all angles have the same probability, *i.e.* the uniform density $f_{{}^W\theta_1}({}^W\theta_1) = \frac{1}{2\pi}$. However the true prior may be biased and highly non-uniform.

An alternative is to assume that all orientations of the part in the *gripper* frame are equally likely. We can in fact justify this assumption by commanding a random twist of the gripper prior to grasping, *i.e.* a random grasp. To see that a random grasp gives rise to a uniform probability density, recall that the orientation of the part in the gripper frame is $\theta_1 = ({}^W\theta_g - {}^W\theta_1)$ (all angles are taken modulo 2π). Let X, Y, Z be random variables corresponding to ${}^W\theta_g, {}^W\theta_1, \theta_1$ respectively, *i.e.* $Z = X - Y$.

Theorem 3.1 *If X and Y are two independent random variables defined over the set S^1 and the pdf for X is uniform, then the random variable $Z = Y - X \bmod 2\pi$ has a uniform pdf.*

Proof:

$$p(z) = \int p(z,y)\, dy \tag{3.9}$$

$$= \int p(z|y)\, p(y)\, dy \tag{3.10}$$

$$= \int \frac{1}{2\pi}\, p(y)\, dy \tag{3.11}$$

$$= \frac{1}{2\pi}. \tag{3.12}$$

The conditional probability in the third step is uniform for all y since X has a uniform distribution and X and Y are independent. From Theorem 3.1, a random grasp insures that the initial orientation of the part in the gripper frame has a uniform probability density.

For a random grasp, equation 3.8 becomes

$$P(S) \geq \frac{|\Pi(\Theta_s)|}{2\pi}. \tag{3.13}$$

where $\Pi(\Theta_s)$ depends on part geometry, center of mass, and the coefficient of friction.

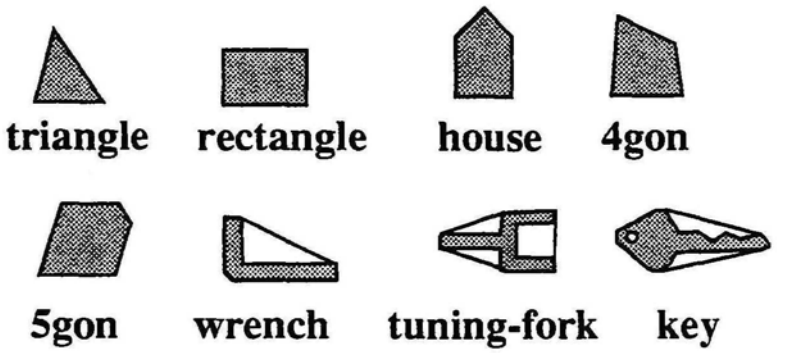
3.10 Examples of Lower Bound on $P(S)$

As stated earlier, given a list of vertices representing a polygonal part, the part's center of mass, and the coefficient of friction, we want to find a lower bound on the probability that a random grasp will be stable. To find this lower bound, we implemented Equation 3.13 in Common Lisp, representing vertices with rational numbers so that the angles can represented exactly using rational complex numbers. Examples are shown in Table 3.1.

A trivial lower bound of zero occurs when the strong preimage shrinks to a point. Recall that the size of the strong preimage depends on part geometry and the coefficient of friction. For the triangle, 4gon, wrench and key, it is almost always possible to wedge depending on which jaw makes contact first. Since we don't know which jaw will make contact first, almost any orientation can lead to wedging. For the other objects, note that a stable grasp becomes more likely as friction is reduced – with less friction it is harder to achieve a wedged orientation.

For uniform-density regular n-gons:

$$P(S) \geq \begin{cases} 0 & \text{if } n \text{ odd or } n \geq \lfloor \frac{\pi}{\alpha} \rfloor \\ 1 - n\alpha/\pi & \text{otherwise} \end{cases} \tag{3.14}$$



part shape	sides	$\mu = .25$	$\mu = .10$	$\mu = .05$
triangle	3	0.00	0.00	0.00
rectangle	3	0.69	0.87	0.94
house	5	0.34	0.44	0.47
4gon	4	0.00	0.00	0.00
5gon	5	0.60	0.76	0.78
wrench	6	0.04	0.06	0.07
tuning-fork	6	0.69	0.76	0.76
key	11	0.00	0.00	0.00

Table 3.1: Lower bound on $P(S)$ varying shape and coefficient of friction. (All parts have uniform mass density).

where $\alpha = \arctan \mu$. As we increase the number of edges the probability of a stable grasp decreases since there are more opportunities for wedging. Similarly, the probability of a stable grasp decreases as we increase the coefficient of friction. When n is odd, we get a trivial lower bound of zero since each edge always has an opposing vertex. If the jaws touch the vertex first, the object may be pushed into a wedged orientation so the strong preimage shrinks to a point. A tighter lower bound on the probability of a stable grasp requires a stochastic model of the pushing phase. We explore this extension in Appendix C.

Regarding an *upper* bound on $P(S)$, assume that whenever it is possible to rotate to a stable edge while pushing, the part will do so. Given a sufficiently fast rotation rate, the part can always rotate to a stable edge during the pushing phase, so the best we can say is $P(S) \leq 1$.

3.11 The Frictionless Gripper

Merrick Furst pointed out that since the probability of a stable grasp goes to one as the coefficient of friction goes to zero, it may be desirable to eliminate friction mechanically. One approach

is to cover the jaws with grease, but this has the disadvantage that the part will slip when the gripper is lifted out of the plane. We desire low friction in the plane of the part, and high friction out of the plane. This can be achieved by adding a linear bearing to one of the jaws; the *frictionless gripper* is discussed in Appendix B.

3.12 Discussion

In this chapter we considered the problem of achieving a stable grasp with a parallel-jaw gripper. We showed that a stable grasp cannot be guaranteed in the presence of friction and derived a lower bound on the probability that a random grasp will be stable. We found that this lower bound goes to one as the coefficient of friction goes to zero (Equation 3.14), which suggests a modification to the parallel-jaw gripper that we call the frictionless gripper.

The frictionless gripper insures that the part is in one of a finite number of orientations. In the next two chapters we employ the frictionless gripper and refine the sample space to distinguish between stable orientations. We derive a probability distribution over the set of stable orientations and use this to find *multi-step iterative* plans for orienting a part in minimal expected time.

Chapter 4

Stochastically Optimal Parts Feeding

4.1 Introduction

We are now ready to apply the stochastic planning framework to a concrete example. In this chapter and the next, we consider the problem of *parts feeding*: orienting parts that are initially jumbled.

The *vibratory bowl feeder* is a well-known mechanism for feeding industrial parts. As a bowl is vibrated with a rotary motion, parts climb a helical metal track. Parts in undesirable orientations are deflected back into the bowl so that only parts with a desired orientation emerge from the feeder [Boothroyd, Poli, and Murch , 1982]. The difficulty with this method is that the track must be designed by trial-and-error; there is currently no systematic theory for generating tracks from part geometry.

We explore the idea of a *programmable* parts feeder: a mechanism that can be reprogrammed rather than physically modified for new part geometries. The problem of converting part geometry into a parts feeding program is an example of manipulation planning in the presence of uncertainty.

We consider the class of polygonal parts, that is, two-dimensional parts with linear edges. We propose that the programmable mechanism consists of a frictionless parallel-jaw gripper that can be oriented, opened and closed. We define a *squeeze action* as the combination of orienting the gripper, closing the jaws as far as possible, and then opening the jaws. An iterative plan is a sequence of such actions followed by a binary filter, some mechanism that recycles all but one desired orientation of the part, for example a silhouette trap. Although many different performance measures are possible depending on the application, we apply the framework of Chapter 2 to define iterative plans that maximize *expected feedrate*.

---

[0]Parts of this chapter have appeared in [Goldberg and Mason, 1990].

Example

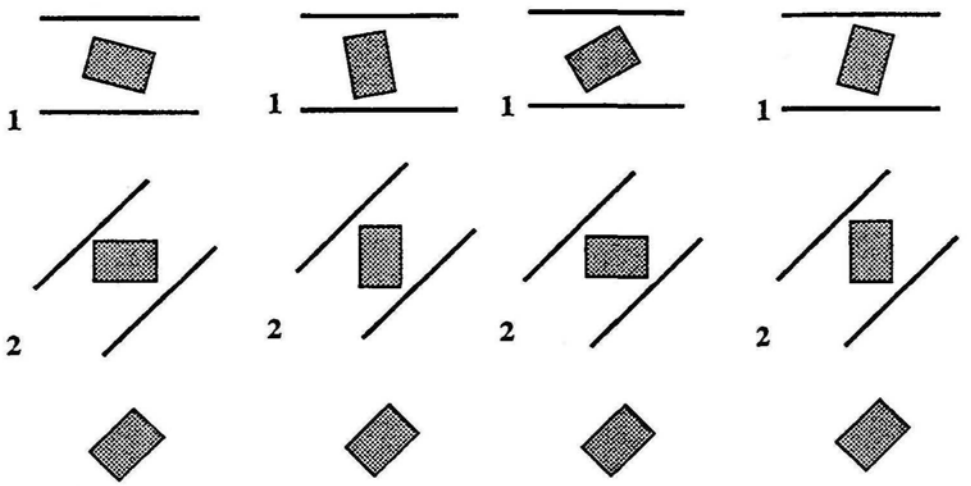


Figure 4.1: Four traces of a two-step plan for orienting a rectangular part using the frictionless parallel-jaw gripper. Each trace runs from top to bottom showing the orientation of both gripper and part after each squeeze action. Although the part's *initial* orientation is different in each trace, its *final* orientation is the same. Note that gripper motion for each case is the same. The plan is open-loop: it does not depend on sensor data.

Consider a rectangular part whose initial orientation is unknown. A sequence of two squeeze actions will insure that the part's major axis is aligned with the gripper regardless of the part's initial orientation. See Figure 4.1.

Now consider a one-step plan that grasps only once. The one-step plan will align the major axis with the gripper unless the part's major axis is initially almost orthogonal to the jaws. If we had a probabilistic model of the part's initial orientation, then we could compute the *probability* that the major axis will be aligned with the gripper after one step. If this probability were, say, 0.8, then we may be willing to accept the one-step plan rather than the more conservative two-step plan.

How can we compare a one-step plan that succeeds with probability 0.8 and a two-step plan that succeeds with probability 1? Consider augmenting the one-step plan with a binary filter that rejects parts that are not aligned with the gripper. Rejected parts are randomized and the plan is repeated until the part is correctly oriented. We expect that, on average, we will have to execute the one-step plan $1/0.8 = 1.25$ times until it succeeds. On the other hand, we only have to execute the two-step plan once to succeed.

If every step in the plan requires one time unit then the *expected time* for the one-step plan is

1.25 units and the expected time for the two-step plan is 2.0 time units. Under these conditions we might prefer the one-step plan. Of course we must also consider the time required for the filter stage.

We have compared two plans; there are an infinite number of plans for orienting a rectangular part. How can we identify the best plan? Where do the probabilities come from? Can we develop a method for optimally orienting polygonal parts that is not completely *ad hoc*?

In the next section we describe related work. We then specify assumptions. In section 4.4 we define a transfer function based on part geometry and use it to specify the elements of a stochastic planning problem. We then define a stochastically optimal parts-feeding plan and give examples.

4.2 Related Work

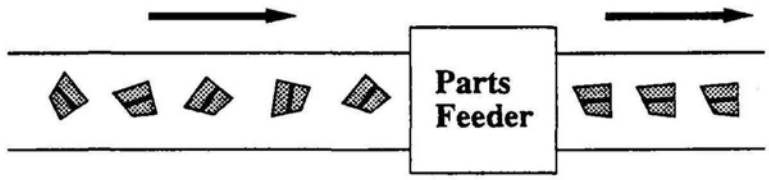


Figure 4.2: A feeder orients parts as they arrive on the left-hand conveyor belt.

A schematic illustration of a parts feeder is shown in Figure 4.2. There are many designs for parts feeders that use sonic or optical sensing such as the one reported by Suzuki and Kohno [1981]. Once the orientation of a part is sensed the part can be mechanically reoriented with an actuator. Since mechanical actuation is unavoidable, it may be less expensive to eliminate sensing and rely only on passive compliance.

An excellent introduction to mechanical parts feeders can be found in [Boothroyd, Poli, and Murch , 1982]. Boothroyd, Poli and Murch describe vibratory bowl feeders in detail as well as non-vibratory feeders such as the magnetic and revolving hook feeders. They note that the feedrate for a parts feeder is related to the probability that parts are aligned correctly when they encounter a mechanical filter and give probabilistic models for the orientation of rectangular and cylindrical parts dropped at random.

In his master's thesis, Singer [1985] noted that a hard-tooled bowl feeder requires several months of tooling. He identified criteria for a parts feeder that included changeover time for new parts, ability to handle a wide variety of parts, and feed rate. He proposed several designs for programmable parts feeders, one using impact and another where programmed vibration was used to actively excite parts into a stable orientation. See also Singer and Seering [1987]. The

Sony Corporation has recently begun to market a parts feeder that uses programmed vibration [Hitakawa, 1988].

The following four designs for programmable parts feeders use binary models of uncertainty and cost, *i.e.* guaranteed plans. Mani and Wilson [1985] developed plans to orienting polygonal parts using a sequence of *pushing* actions. They used heuristics to find a plan that maps all possible initial orientations into a single final orientation. Peshkin and Sanderson [1988] considered the related problem of designing an arrangement of planar "fences" such that polygonal parts on a conveyor belt are oriented as they slide along the fences. To plan, they define a *configuration map*, C_a, for each fence angle to be a boolean function on the space $\Theta \times \Theta$ such that $C_a(\theta, \theta') = 1$ if it is possible to reach state θ' from state θ by passing by a fence at angle a. Using a uniform discretization of the space of fence angles, they performed a depth-first search through the space of fence arrangements. For most of the parts they considered, they were able to find an arrangement of fences guaranteed to orient a given part. Due to the discretization of fence angles, this search procedure neglects fence arrangements using intermediate angles. Erdmann and Mason [1986] developed multi-step plans to orient parts with a sequence of tray-tilting actions. Taylor, Mason, and Goldberg [1987] and Mason, Goldberg, and Taylor [1988] considered guaranteed multi-step plans to orient parts using a sequence of grasps with a parallel-jaw gripper. We found that it was impractical to search the continuous action space that resulted from frictional contacts.

Each of the feeders described in the previous paragraph used mechanical compliance to constrain the state of the part. In this chapter we propose another feeder design that uses mechanical compliance. In contrast to the preceding approaches, we use a stochastic framework to define optimal plans.

4.3 Assumptions

We assume that:

1. The gripper has two linear jaws arranged in parallel.
2. The direction of gripper motion is orthogonal to the jaws.
3. The part is a rigid planar polygon of known shape.
4. The part's initial position is unconstrained as long as it lies somewhere between the two jaws. The part remains between the jaws throughout grasping.
5. All motion occurs in the plane and is slow enough that inertial forces are negligible. The scope of this *quasi-static* model is discussed in [Mason, 1986] and [Peshkin, 1986].

6. Both jaws make contact simultaneously (pure squeezing).

7. Once contact is made between a jaw and the part, the two surfaces remain in contact throughout the grasp. A grasp continues until further motion would deform the part.

8. There is zero friction between part and the jaws.

9. We can design a binary filter that accepts a particular orientation of the part and rejects all others.

These assumptions are similar to those made by Brost [1988], Taylor, Mason, and Goldberg [1987], and Mason, Goldberg, and Taylor [1988]. Assumptions 2, 6, and 8 simplify the analysis and improve the combinatorics of the search. By restricting gripper motion to be orthogonal to the jaws (assumption 2), we obtain a one-dimensional action space. Using a frictionless gripper (assumption 8) insures that the state space is the *finite* set of stable part orientations (see Appendix B for a physical implementation). Assuming simultaneous contact (assumption 6) greatly simplifies the mechanical analysis. In Appendix E we show how assumption 6 can be relaxed.

Under the framework of Chapter 2, a manipulation planning problem is a 5-tuple, $(\Theta, \lambda_0, \mathcal{A}, \mathcal{P}, C)$: a finite state space, initial probability distribution, finite action space, a transition matrix for each action, and a cost function. In the next section we introduce a transfer function and use it to specify the elements of the planning problem.

4.4 The Transfer Function

The transfer function for a squeeze action provides a mapping from an initial orientation of the part to a final orientation of the part. Each squeeze action corresponds to an orientation of the gripper. Since rotating the gripper is equivalent to rotating the part, we define the transfer function in terms of the part's orientation with respect to the gripper. The transfer function depends on part geometry. For a given part, we derive the *squeeze* function based on another function that we call the *diameter* function.

4.4.1 The Diameter Function

When a part is grasped with the frictionless gripper, it assumes one of a finite number of stable orientations. These stable orientations correspond to local minima in the diameter function defined as follows.

Let a two-dimensional part be described with a continuous curve, C, in the plane. The distance between two parallel tangent lines varies with the orientation of the lines.

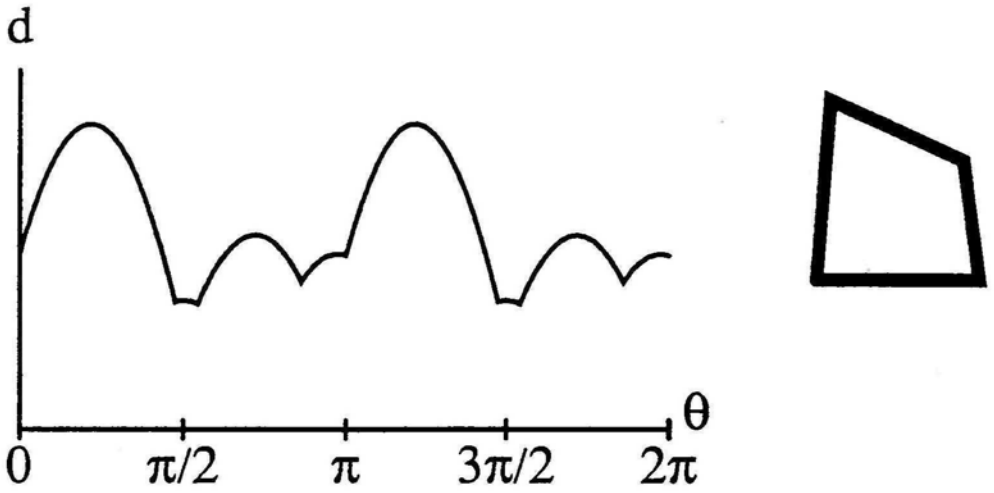


Figure 4.3: The diameter function for the four-sided part shown at the right in its zero orientation. During a squeeze, the part rotates so as to reduce the diameter, terminating when the diameter reaches a local minimum.

Definition 4.1 *The diameter function, $d : S^1 \rightarrow \Re$, is the distance between two parallel tangents at angle θ.*

For polygonal parts, the diameter function is piecewise sinusoidal as shown in Figure 4.3. An $O(n \log n)$ algorithm for computing the diameter function for an n-sided polygon is given in Appendix D.

When a part is grasped between the jaws of the gripper, the distance between the jaws corresponds to the diameter. Closing the jaws changes the diameter and thus the relative orientation of the part. The jaws continue closing until the diameter is at a local minimum that also defines a stable orientation of the part. The diameter function can be viewed as a potential energy function for a conservative system (see appendix D).

4.4.2 The Squeeze Function

During a squeeze, part motion is determined by the diameter function. That is, given an initial orientation of the part with respect to the gripper, the part's final orientation can be determined from the diameter function. A transfer function, relating initial orientations to final orientations, can be represented with a piecewise constant function that we call the *squeeze function*, $s : S^1 \rightarrow S^1$.

We define the squeeze function such that if θ is the initial orientation of the part with respect to the gripper, $s(\theta)$ is the orientation of the part with respect to the gripper after squeezing. The squeeze function can be derived from the diameter function. All orientations that lie between a

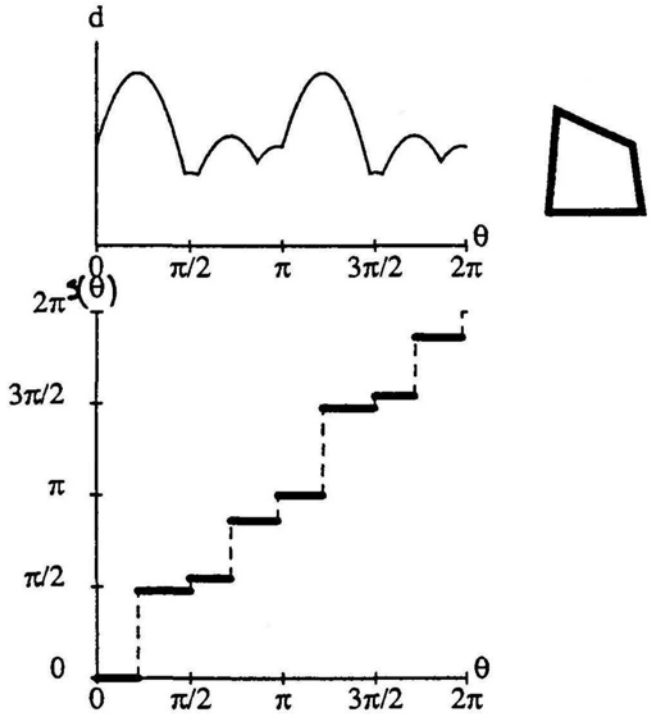


Figure 4.4: The diameter and squeeze function for the four-sided part.

pair of adjacent local maxima in the diameter function will map into the same final orientation. The squeeze function is constant over this interval of orientations. Each local maximum in the diameter function corresponds to a discontinuity in the squeeze function. In order for the function to be single-valued, we assume that all steps are closed on the left. See figure 4.4.

Since the squeeze function is piecewise constant, its range is actually the finite set of stable states, Θ_s. This set includes up to $2n$ states, where n is the number of edges on the polygon (fewer if some polygon edges are unstable). Note that more than one polygon can yield the same squeeze function. E.g., all triangles with obtuse angles give rise to the same squeeze function.

4.4.3 The State Space

The part's initial orientation in the world frame is the uncountable set of planar rotations. After the first action, the part's orientation relative to the gripper will be one of the stable orientations, $\theta_s \in \Theta_s$. By defining the state to be the part's orientation relative to the gripper *after an action*, the state space of the system becomes the finite set Θ_s. To derive a prior probability distribution on this state space, we must consider the part's probability distribution in the world frame.

4.4.4 Prior Probability Distribution

The part's initial orientation in the world frame is uncertain. Say that we can describe this orientation with a continuous random variable with probability density $f(\theta)$. Since the state space of the system is defined as the part's orientation relative to the gripper after an action, we must convert this density function into a discrete probability distribution on the set Θ_s. After the first action the probability that the part is at orientation θ' is related to the probability that the part was initially in some orientation θ such that $s(\theta) = \theta'$. Accordingly, we can compute the first probability distribution (hyperstate) by integrating the probability density between discontinuities in the transfer function.

The first squeeze action in the plan is a random grasp, where gripper orientation is chosen from a uniform probability distribution on the interval $[0, 2\pi)$. Theorem 3.1 shows that this is equivalent to assuming that the initial part orientation in the world frame has a uniform probability density, $f(\theta) = \frac{1}{2\pi}$. In this case the initial hyperstate can be computed based on the *width* of steps in the transfer function.

An ambiguity arises when the part's initial orientation is exactly at a local maximum in the diameter function corresponding to an unstable equilibrium. However if we assume that the prior probability density for orientations is continuous, the initial squeeze will encounter an unstable equilibrium with probability zero. Thereafter we avoid ambiguous actions. Next we consider the space of actions.

4.4.5 The Action Space and Transition Matrices

We specify an action by the orientation of the gripper.

Definition 4.2 *A* **squeeze action** *(at angle θ) consists of orienting the open gripper at angle θ around the part, closing the jaws as far as possible, and then opening the gripper.*

Let θ be the orientation of the part in the world frame. If the gripper is oriented at angle θ_g, then the relative angle of the part is $\theta - \theta_g$. After the squeeze action, the final orientation of the part is $s(\theta - \theta_g)$ in the gripper frame and $s(\theta - \theta_g) + \theta_g$ in the world frame. For an action occurring at angle θ_g, we define

$$s_{\theta_g}(\theta) = s(\theta - \theta_g). \tag{4.1}$$

The space S^1 is uncountably infinite, so there is an uncountable number of squeeze actions. Fortunately, the continuum of squeeze actions can be partitioned into a finite number of equivalence classes.

We treat two actions as equivalent if they have the same transition matrix. Recall from chapter 2 that the transition matrix for an action defines a mapping between hyperstates. For squeeze actions there is no control uncertainty – the part's final orientation is completely

determined by its initial orientation – so the transition matrix for each squeeze action is a binary matrix.

After an action, the part is in one of its stable states. The next action defines a mapping between stable states. Consider what happens if the part is at orientation θ_s (relative to the gripper) and the gripper is rotated by angle θ_g. After the gripper is closed, the final orientation of the part relative to the gripper will be $s(\theta_s - \theta_g)$. Since the squeeze function is piecewise constant, a range of gripper angles gives rise to the same outcome. As we rotate the gripper the outcome changes when $\theta_s - \theta_g$ crosses a discontinuity in the squeeze function.

To take into account all stable initial orientations, we must consider all pairs of initial states and discontinuities in the squeeze function. One way to do this is to imagine two copies of S^1, the space of planar rotations. In the first copy we mark all stable angles. In the second copy we mark all discontinuities in the transfer function. As we rotate the first copy of S^1 across the second copy, we generate a new transition matrix every time a mark in the first copy crosses a mark in the second copy. The relative angle between copies defines the associated action. For each transition matrix, we choose an action in the center of the equivalence class to be the representative action. Since there are $O(n)$ marks in each copy, there are $O(n^2)$ unique actions.

A plan is a composition of actions. Consider a two-step plan, ρ, with actions at angles θ_1 and θ_2 respectively. Let

$$\rho(\theta) = s_{\theta_2} \circ s_{\theta_1}(\theta) = s(s(\theta - \theta_1) - \theta_2). \tag{4.2}$$

Orienting a Part up to Symmetry

We noted that the squeeze function has period π due to symmetry in the gripper; rotating the gripper by 180 degrees produces a symmetric arrangement that preserves the diameter. Rotational symmetry in the part also introduces periodicity into the squeeze function. In general the transfer function has period T such that

$$s(\theta + T) = (s(\theta) + T) \bmod 2\pi. \tag{4.3}$$

For polygons with r-fold rotational symmetry, the squeeze function will have period $T_r = 2\pi/r(1 + r \bmod 2)$. For example, a part with no rotational symmetry ($r = 1$) produces a squeeze function with period π. An equilateral triangle has 3-fold rotational symmetry; its squeeze function has period $\pi/3$. A square has 4-fold rotational symmetry; its squeeze function has period $\pi/2$.

Periodicity in the squeeze function gives rise to *aliasing*, where the part in orientation θ behaves identically to the part in $\theta + T$. Any sequence of actions that maps θ to θ' will map $\theta + T$ to $\theta' + T$. This implies that there is no sequence of squeeze actions that can map orientations θ and $\theta + T$ into a single final orientation.

Definition 4.3 *For a given part, let T be the smallest period in its transfer function. We say that a plan orients a part* **up to symmetry** *if the set of possible final states includes exactly $2\pi/T$ states that are equally spaced on S^1.*

For example, for a part with no rotational symmetry, a squeeze plan orients the part up to symmetry if the plan that yields exactly two final states π radians apart. A part with 3-fold rotational symmetry can be oriented up to symmetry with a squeeze plan that yields six possible final states each $\pi/3$ radians apart.

4.4.6 The Cost Metric

The cost function depends on the application. To illustrate the approach, we relate a cost function to the time required to orient a part (we consider a cost function based on stability in section D.2). Let c_a be the cost of a grasping action and c_f be the cost of passing the part through the binary filter. Consider a i-step plan ρ_i. The cost for one iteration of ρ_i is $ic_a + c_f$.

Recall that the binary filter rejects all but one orientation. The plan tries to make this orientation likely so that parts go through as often as possible. As described in Section 2.3, the expected number of iterations needed to achieve this orientation is the reciprocal of the probability of achieving the orientation on any single iteration of the plan. For a given hyperstate, we can identify a *most-likely state* (breaking ties arbitrarily),

$$\theta^*(\lambda) = \arg\max_{\Theta_*} P(\theta), \tag{4.4}$$

where $P(\theta)$ is the probability of θ in hyperstate λ. The hyperstate that results from executing plan ρ is $\lambda_0\ P_\rho$. For a given combination of initial hyperstate and plan, let $\theta^* = \theta^*(\lambda_0\ P_\rho)$. After executing the plan, the gripper aligns the part over the filter so that only parts in orientation θ^* are accepted.

As derived in Chapter 2, we expect that *on average*, plan ρ will go through $1/P(\theta^*)$ iterations until θ^* is achieved. Since the cost per iteration for a i-step plan is $ic_a + c_f$, the expected cost for the iterative plan is

$$\bar{C}(\rho_i) = \frac{ic_a + c_f}{P(\theta^*)}. \tag{4.5}$$

And an optimal i-step plan is one with minimal expected cost,

$$\rho_i^* = \arg\min_{\rho_i} \bar{C}(\rho_i). \tag{4.6}$$

4.5 Definition of Stochastically Optimal Plan

Given part geometry as a list of vertices, we use the analysis in the preceding section to obtain the 5-tuple, $(\Theta, \lambda_0, \mathcal{A}, \mathcal{P}, C)$. Equation 4.6 defines an optimal iterative i-step plan for each i. Over all plans, the minimal expected cost is

$$\bar{C}^* = \min_i \bar{C}(\rho_i^*) \tag{4.7}$$

Definition 4.4 *A stochastically optimal parts-feeding plan, ρ^*, is a plan that orients a part with minimal expected cost, $\bar{C}(\rho^*) = \bar{C}^*$, or equivalently, with maximal expected feed rate.*

By this definition there may be more than one stochastically optimal plan due to ties. Thus to be precise we should refer to "an" optimal plan rather than "the" optimal plan. In the text I may occasionally refer to "the optimal plan" with the tacit understanding that it may not be unique.

4.6 Theorem: Stochastically Optimal Plans are of Length $O(n)$

Theorem 4.1 *A stochastically optimal plan requires no more than $2n(\frac{c_f}{c_a} + 1)$ steps, where n is the number of edges, c_f is the filter cost, and c_a is the cost for a single action.*

Proof: We find an upper bound for the expected cost of an optimal 1-step plan. This provides an upper bound on the number of steps in the optimal plan.

Let ρ_1^* be the optimal 1-step plan. Its expected cost is

$$\bar{C}(\rho_1^*) = \frac{c_a + c_f}{p_1^*}, \tag{4.8}$$

where c_a is the cost for one action, c_f is the cost for the filter, and p_1^* is the probability associated with the most likely outcome. Since the probability mass of 1.0 must be distributed over no more than $2n$ states, the most likely outcome must have probability at least

$$p_1^* \geq \frac{1.0}{2n}, \tag{4.9}$$

Combining Equations 4.8 and 4.9, we get an upper bound on the expected cost of the optimal 1-step plan:

$$2n(c_a + c_f) \geq \bar{C}(\rho_1^*). \tag{4.10}$$

Say the optimal plan has length k. Let ρ_k^* be the optimal plan. Then ρ_k^* costs no more than ρ_1^*.

$$2n(c_a + c_f) \geq \bar{C}(\rho_1^*) \geq \bar{C}(\rho_k^*). \tag{4.11}$$

Even if the k-step plan succeeds with probability one, its expected cost is

$$\bar{C}(\rho_k^*) \geq kc_a. \tag{4.12}$$

Combining inequalities,

$$p_k^* \geq \frac{kc_a + c_f}{2n(c_a + c_f)}. \tag{4.13}$$

$$2n(c_a + c_f) \geq kc_a. \tag{4.14}$$

Rewriting,

$$k \leq 2n(\frac{c_f}{c_a} + 1). \tag{4.15}$$

■

4.7 Problem Definition

We define a stochastic parts-feeding problem as follows:

- Given a list of n vertices describing a planar part and the ratio $c = c_f/c_a$, where c_f is the cost of the filter step and c_a is the cost of a single squeeze action.
- Find a list of gripper angles corresponding to a plan for orienting the part with minimal expected cost (a stochastically optimal plan).

We represent vertices with rational numbers so that gripper angles can be represented exactly using rational complex numbers. In the examples below, $c = 1$.

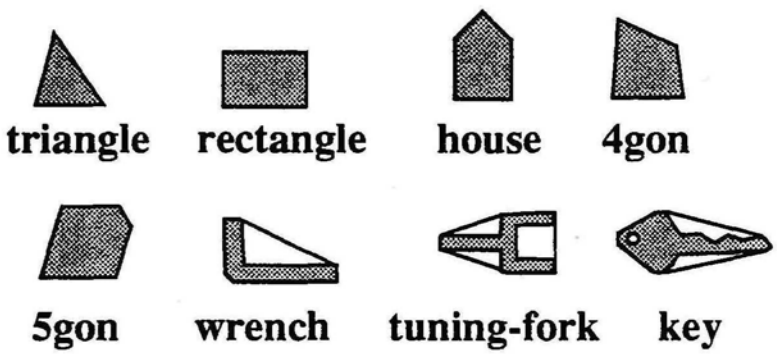
4.8 Implementation

One way to find an optimal plan is to use breadth-first search as follows. Recall that the space of plans defines a tree where each node is a hyperstate and each link corresponds to an action. The root is the hyperstate resulting from the first action. The first level of nodes corresponds to the outcome of one-step plans. The second level of nodes corresponds to the outcome of two-step plans. We consider all hyperstates at each level. A path from the root to the min-cost

hyperstate at level i corresponds to a stochastically optimal i-step plan. Theorem 4.1 shows that we only need to search to a fixed depth bound to find the optimal plan.

We implemented the breadth-first search in Common Lisp and tested it on several familiar shapes as well as on a test set of 2000 randomly generated polygons (We generate 10 random points in the unit square and find their convex hull; the average number of sides was 5.9). If the planner does not find an optimal plan after generating 1000 nodes, it halts and reports failure.

The planner found an optimal plan for 99% of the polygons in the test set, running each case in under a minute on a Sun 3/60. For the remaining 1% of the polygons in the test set, the breadth-first search terminated after searching 1000 nodes. Table 4.1 shows some typical results.



part shape	# sides	1-step	2-step	3-step	4-step
triangle	3	9.5	8.2	**8.0**	10.0
rectangle	4	6.4	**6.0**	8.0	10.0
house	5	**5.0**	6.0	8.0	10.0
4gon	4	14.3	12.0	16.0	**10.0**
5gon	5	8.6	**6.5**	8.0	10.0
wrench	6	9.1	**7.4**	8.0	10.0
tuning fork	6	**4.9**	6.0	8.0	10.0
key	11	9.0	**6.2**	8.2	10.0

Table 4.1: Expected time for grasping plans. Values are in time units where each squeezing step and the filter step takes one unit. Stochastically optimal plan for each part is indicated in boldface.

Figure 4.5 shows a typical plan with the evolution of the probability distribution for the 5-sided house-shaped part as it goes through the plan. Due to symmetry in the diameter function, there is no sequence that can orient the part beyond a 180° ambiguity. In this case we might use the filter to only accept parts in the orientation shown at the lower left.

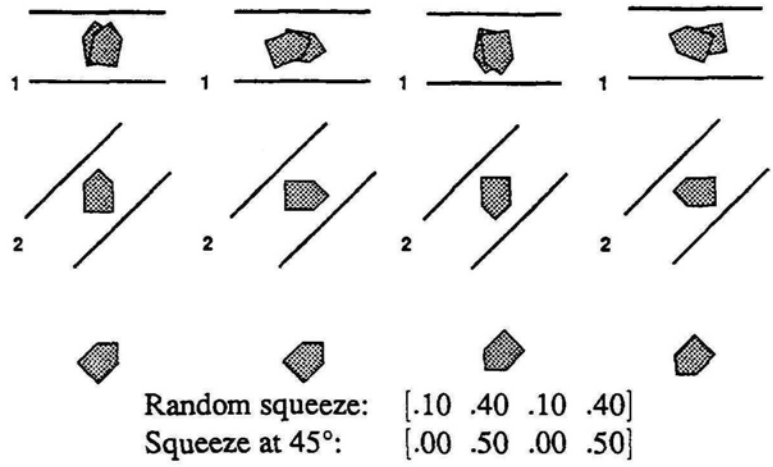


Figure 4.5: Four traces of open-loop plan for the house shape: squeeze at random orientation (here normalized to 0°), then squeeze at 45°. Below: evolution of the probability distribution as the plan proceeds.

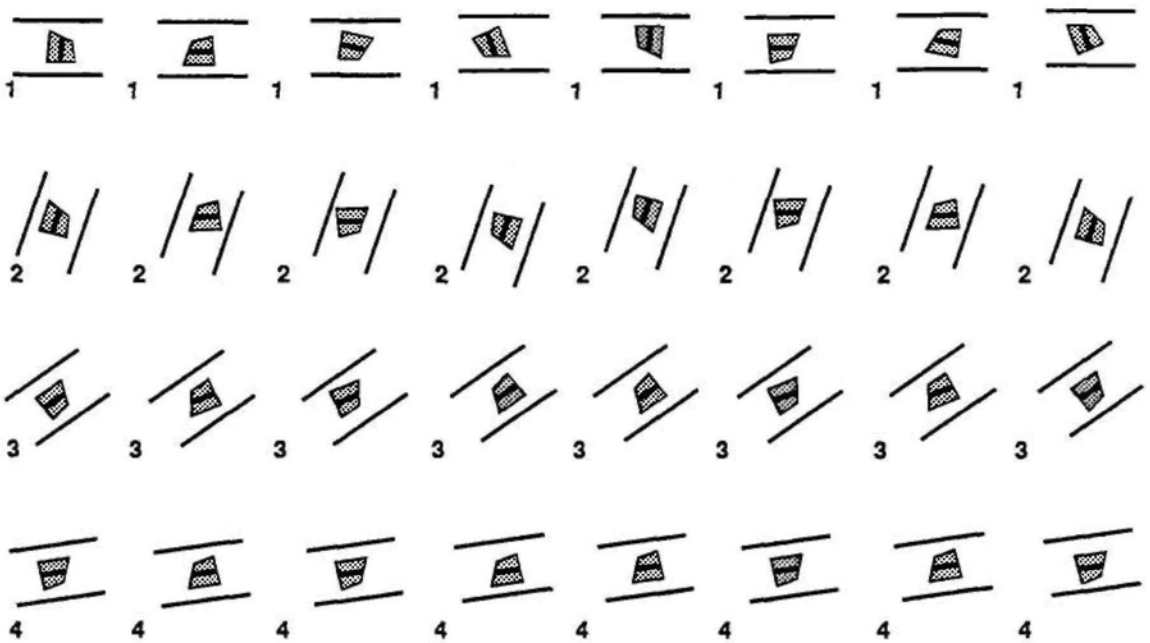
The cost per iteration of the one-step plan is 2 time units, and we expect to go through $1/.40 = 2.5$ iterations on average until achieving a desired orientation, so the expected cost for the one-step plan is 5.0 time units. The cost per iteration of the two-step plan is 3 time units, and we expect to go through $1/.50 = 2.0$ iterations on average until achieving a desired orientation, so the expected cost for the one-step plan is 6.0 time units. For this part geometry, a one-step iterative plan is stochastically optimal.

Figure 4.6 shows a typical plan. In this case the four-step plan is stochastically optimal.

4.9 Discussion

In this chapter we propose a design for a programmable parts feeder using a frictionless parallel-jaw gripper and binary filter. Any two-dimensional part can be analyzed by computing its diameter function. We give an algorithm for automatically generating stochastically optimal open-loop iterative plans – i.e. programs – for the parts feeder. The purpose of this chapter is to illustrate the framework of chapter 2 with a specific application where:

- the mechanical behavior of the part can be related to its geometry,
- the state and action spaces can be discretized based on mechanics.
- a discrete probability distribution can derived by integrating a prior probability density over regions of state space,



Random squeeze (0°):	[.12	.14	.11	.13	.12	.14	.11	.13]
Squeeze at 71°:	[.00	.25	.00	.25	.00	.25	.00	.25]
Squeeze at 35°:	[.00	.25	.25	.00	.00	.25	.25	.00]
Squeeze at 8°:	[.00	.50	.00	.00	.00	.50	.00	.00]

Figure 4.6: Above: Eight traces of plan to orient part (4gon) up to symmetry. This is the stochastically optimal plan. Below: Evolution of the probability distribution as the plan proceeds.

- a cost model for can be related to throughput.

The last two items comprise the real-valued metric used to compare plans.

4.9.1 Randomizing to Justify Prior Probabilities

We justify the assumption that the initial orientation of the part in the world frame has a uniform prior distribution by *injecting* a random twist of the gripper prior to the first grasping operation. If we had an accurate model of the underlying probability distribution, we might be able to generate more efficient plans. One way to approach this is to estimate the prior distribution

empirically, that is, by observing the initial state in a number of trials and replanning based on the new estimates. Since it may be difficult to observe the initial state, we could monitor the rejection rate of parts going through the filter. This data would give partial information about the initial distribution – that is, a rejection tells us only that the part's initial orientation was in some subset of the discrete state space. Statistical methods for estimating the prior distribution from such data are discussed in [Dickey *et al.*, 1987, Shefrin, 1981].

Monitoring the state of the system can also be useful for modifying the plan on-line, i.e. branching based on sensory data. Of course the resulting plans are no longer open-loop. As mentioned in chapter 1, stochastic methods can be very powerful for treating noisy sensor data. In chapter 7 we outline how the stochastic framework can be extended to incorporate sensor data.

4.9.2 Control Uncertainty

In this chapter we assumed that both jaws make contact simultaneously so that there is no control uncertainty. However if one jaw makes contact first and *pushes* the part, then there is control uncertainty since to predict the final orientation of the part we need to know the rotation rate and the pushing distance, which in turn depends on the microscopic interface at the support surface as discussed in chapter 3. We can treat this form of uncertainty by modelling the transfer function with a conditional probability distribution. Alternatively, we can consider the class of *push-grasp* motions [Brost, 1988], that push the part with one jaw for a sufficient distance so that the part rotates into a stable orientation prior to contact with the second jaw. This extension is treated in appendix E.

4.9.3 Worst-Case Computational Complexity

A nagging difficulty with the reported algorithm is its worst-case running time. The breadth-first search reported in section 4.5 has branching factor $O(n^2)$, where n is the number of edges on the part. Although Theorem 4.1 provides a linear bound on the depth of the search tree, the resulting search may require exponential time. In the next chapter we present a planning algorithm that is guaranteed to run in time $O(n^2)$.

Chapter 5

Backchaining Algorithm

5.1 Introduction

Stochastic methods are commonly criticised for requiring exhorbitant amounts of computation. In the previous chapter we defined stochastically optimal plans for a programmable parts feeder. To find optimal plans, we proposed a breadth-first search through the space of all plans; in the worst-case, planning time is exponential in the number of part vertices.

In this chapter we present an efficient algorithm that uses backchaining to find stochastically optimal parts-feeding plans. The planner runs in two phases. Phase I finds a series of plans based on part geometry: a one-step plan, a two-step plan, and so on up to a fixed length limit. Phase II applies cost and probability models to select a globally optimal plan. We prove that the algorithm is correct, complete, and runs in time $O(n^2)$.

5.2 Related Work

Phase I of the algorithm works by propagating *sets* of states backward from a unique final state. It is thus closely related to the preimage backchaining approach described in section 1.1.4. In that approach, if Θ is a subset of state space, we can consider the *image* of Θ under action a. Similarly, we can consider a *preimage* of Θ under action a to be a subset of state space whose image under action a is Θ. Given an initial and desired subset of state space, we can work backwards from the desired subset, chaining preimages until we find a preimage that contains the initial subset of state space. The resulting sequence of actions is guaranteed to yield a desired state. Finding a strategy using the preimage-backchaining approach can require extensive computation. The general problem of planning a sequence of compliant motions to

reduce uncertainty was shown to be PSPACE-hard by Natarajan [1986] and non-deterministic exponential time hard by Canny [1987].

Natarajan [1989] considered several computational problems related to the design of parts feeders. He considered the following problem where there is no control uncertainty.

> Given k functions $f_1, f_2, ..., f_k : \Theta \rightarrow \Theta$, find a mapping f_0 that is a composite of the f_i's such that f_0 is a constant function on Θ, i.e., $|f_0(\Theta)| = 1$.

The set Θ corresponds to a discrete set of part states. The functions $f_1, f_2, ..., f_k$ correspond deterministic transfer functions that change the state of a part. A parts feeder that will accept parts in any state and output parts in a unique state can be designed based on the composite function f_0. We say that the function f_0 *collapses* the set Θ. Natarajan neglected the mechanics of transfer functions and focussed on the abstract problem of combining a given set of functions.

Natarajan presented an algorithm to solve this problem in time $O(kn^4)$, where k is the number of functions and n is the number of states. The algorithm reduces the size of the initial state set by searching a graph where each node contains a pairs of states. Each edge between nodes corresponds to some f_i that provides a mapping between pairs. For any pair of states in the initial set, the algorithm uses a depth-first search to find a sequence of functions that will merge this pair into a single orientation. If no such pair exists then the algorithm halts and reports failure. This greedy algorithm works because the problem satisfies the *inclusion property*: if there exists a composite function that collapses Θ, then there exists a composite function that collapses any subset of Θ. That is, no matter which pair we choose to reduce at each stage, it is never necessary to backtrack.

Note that there may be a sequence that collapses a subset $\Theta' \subset \Theta$ even though there is no sequence that collapses Θ. Natarajan cited a result by Kozen [1977] as evidence that solving the problem for an arbitrary initial subset is PSPACE-Complete. This lead Natarajan to consider a restricted class of transfer functions. Say that the states have a cyclic ordering $\theta_1 \preceq \theta_2 \preceq ... \preceq \theta_n \preceq \theta_1$, as is the case for the set of planar orientations. The function f is *monotonic* if the sequence $f(\theta_1), f(\theta_2), ...$ also has cyclic order. Recently, Eppstein [1990] reported an optimal $O(kn^2)$ algorithm for the restricted problem where all functions are monotonic.

In chapter 4 we analyzed the mechanics of grasping with a frictionless parallel-jaw gripper and identified a physically realizable class of actions, the squeeze actions, that yield a monotonic transfer function. Since we showed in the previous chapter that the number of unique actions is $O(n^2)$, Eppstein's algorithm would allow us to find, in time $O(n^4)$, a plan that collapses the state set. In this chapter we exploit the fact that the transfer functions due to grasping arise from cyclic shifts of the diameter function. We present an algorithm that finds, in time $O(n^2)$, a stochastically optimal plan of length $O(n)$ that collapses the state set.

5.3 Assumptions and Problem Statement

As in Section 4.7, we define a stochastic parts-feeding problem as follows:

- Given a list of n vertices describing a planar part and the ratio $c = c_f/c_a$, where c_f is the cost of the filter step and c_a is the cost of a single squeeze action.
- Find a list of gripper angles corresponding to a plan for orienting the part with minimal expected cost (a stochastically optimal plan).

In this chapter we randomize the initial gripper orientation to insure that the initial distribution of states is uniform. In Appendix F we show how the algorithm can be extended to non-uniform probability density functions.

We assume that all steps in the transfer function are closed on the left. Let T be the smallest period in the transfer function.

5.3.1 Images and Preimages

We define the transfer function for a *set* of angles as

$$\Theta' = s(\Theta) = \{s(\theta)|\theta \in \Theta\}. \tag{5.1}$$

That is, if the initial orientation is in Θ, then after the action the orientation will be in Θ'. We say that Θ' is the *image* of Θ. Similarly, we define the inverse operation:

$$\Theta = s^{-1}(\Theta') = \{\theta|s(\theta) \in \Theta'\}. \tag{5.2}$$

If the orientation after the action is in Θ', then the orientation before the action must have been in Θ. We say that Θ is the *preimage* of Θ'.

5.4 The Algorithm

The algorithm for finding a stochastically optimal parts feeding plan proceeds in two phases. Phase I uses part geometry to find a series of geometrically optimal plans. Phase II applies cost and probability models to the geometrically optimal plans to select a stochastically optimal plan.

1. Compute the transfer function for squeezing. Let T be its period.
2. Find the widest single step and define Θ_0, Θ_1 such that $\Theta_0 = s(\Theta_1)$. Let $h_1 = |\Theta_1|$. Let $N = \frac{2\pi}{h_1}(c+1)$. Let $i = 2$.
3. Let Θ_i be the widest interval such that $|s(\Theta_i)| < h_{i-1}$. Let $h_i = |\Theta_i|$.
4. If $i > N$ or $h_i = T$, let $k = i$ and goto step 5. Otherwise, increment i and goto step 3.
5. Return the list $(\Theta_0, \Theta_1, ..., \Theta_k)$.

Figure 5.1: Phase I

5.4.1 Phase I

Phase I, given in figure 5.1, works backward from a unique final orientation – the set Θ_0 contains only one point.

Figures 5.2 through 5.4 illustrate how the algorithm proceeds for a rectangular part. All orientations in Θ_1 map into the single orientation in Θ_0 when the frictionless gripper is closed. So Θ_0 is the image of Θ_1, and Θ_1 is the *preimage* of Θ_0. Step 3 of phase I searches for the widest interval whose image is smaller Θ_{i-1}. We can implement step 3 geometrically using a square box of dimension h_{i-1}. We position the box over the step function such that the range of output angles contained in the box is smaller than the range of input angles. That is, the function must enter on the box's left-hand edge and exit on the box's right-hand edge as illustrated in Figure 5.4.

Termination Condition

Phase I terminates when either $i > N$, where N is such that the cost of further actions outweighs any possible improvement in the probability of success, or $h_i = T$, when the object is oriented up to symmetry. To determine N, consider that we have a one-step plan that succeeds with probability $h_1/2\pi$, so the expected cost of this one-step plan is $\frac{2\pi}{h_1}(c_a + c_f)$. A better N-step plan must have lower expected cost. But the expected cost for an N-step plan is at least Nc_a. This is better when $N < \frac{2\pi}{h_1}(c+1)$. Thus there is no need to consider plans longer than this bound (see also Theorem 4.1).

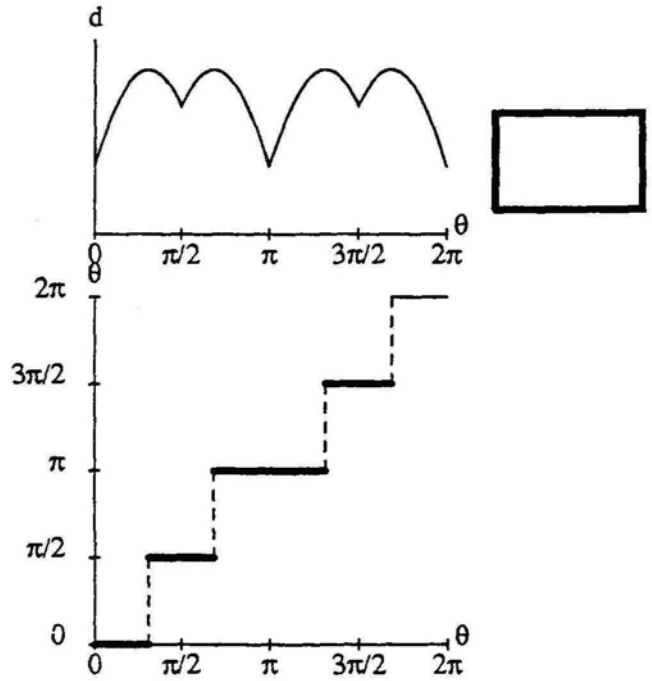


Figure 5.2: For the rectangular part shown at the right, The first step is to compute the squeeze function as shown.

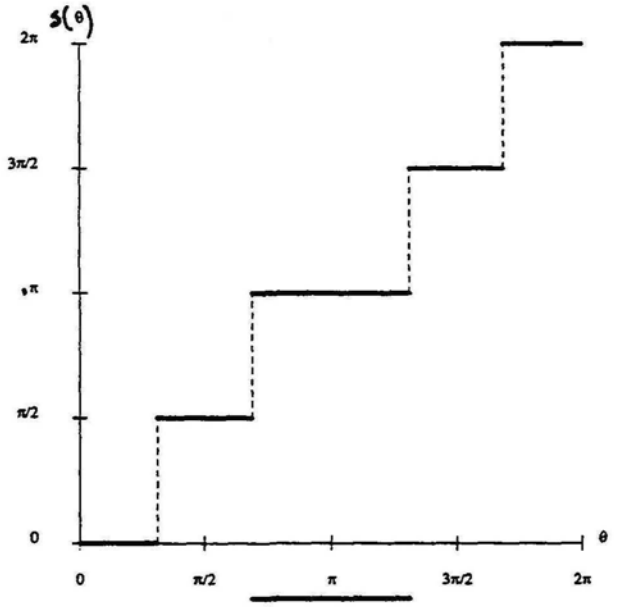


Figure 5.3: In step 2, the widest single step in the transfer function is identified. All the orientations in Θ_1 (horizontal bar at bottom), map into the single final orientation: Θ_0 (dot at π).

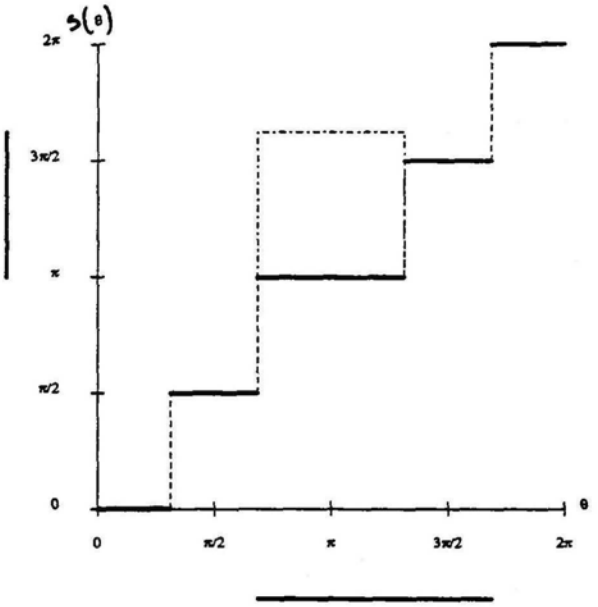


Figure 5.4: In step 3, we identify the largest interval whose image is smaller than $h_1 = |\Theta_1|$. This can be visualized by left-aligning a box of dimension h_1 with each step. If the squeeze function emerges from the right edge of the box, then the corresponding image is smaller than h_1. The largest such interval in this case is Θ_2 (shown at bottom).

Recovering Plans from a Sequence of Preimages

As the algorithm proceeds, the interval of orientations corresponding to each preimage is shifted to fit within the following preimage. Shifting the set of part orientations is equivalent to rotating the gripper. Thus the relative position of adjacent preimages defines a gripper orientation.

The relative position of the preimages Θ_0, Θ_1 defines a one-step plan that collapses all orientations in Θ_1 to the single orientation in Θ_0. The set Θ_1 becomes the image for Θ_2 such that there is a two-step plan for collapsing all orientations in Θ_2 to the single orientation in Θ_0.

The sequence of images $\Theta_0, \Theta_1, ..., \Theta_k$ defines a sequence of plans. To recover the plans we work backward from the final image, Θ_0 (a point). Let θ_i be the leftmost point in image Θ_i: $\Theta_i = [\theta_i, \theta_i + h_i)$. The relative orientation of Θ_1 with respect to Θ_0 is $\sigma_1 = \theta_1 - \theta_0$. An action applied at angle σ_1 will cause all orientations in Θ_1 to be aligned with the gripper.

Next we relate the relative orientation of Θ_2 to that of Θ_1. This is $\sigma_2 = \theta_2 - \theta_1$. That is, by rotating the gripper by σ_2 radians and then squeezing, we convert all orientations in image Θ_2 to orientations in image Θ_1. We proceed backwards until we reach Θ_i. Then by reversing

and negating the sequence: $[-\sigma_i, -\sigma_{i-1}, ..., -\sigma_1]$ we have a set of gripper angles that defines an i-step plan for converting all orientations in Θ_i to the single orientation θ_0.

Note that at each step we align the images precisely against their left edge. We can allow some error margin in gripper angle by noting the relative difference in size between neighboring intervals in the sequence. For example Θ_1 is smaller than Θ_2 so we can adjust the gripper angle by half the difference in size. Let $\epsilon_i = (|\Theta_i| - |\Theta_{i-1}|)/2$.

Each i-step squeezing plan is given by

$$\rho_i = [-\sigma_i + \epsilon_i, -\sigma_{i-1} + \epsilon_{i-1}, ..., -\sigma_1 + \epsilon_1]. \tag{5.3}$$

Phase I returns N preimages corresponding to a 1-step plan, a 2-step plan, and so on up to an $N-1$-step plan. For each i, let ρ_i be the corresponding i-step plan. Before describing Phase II, we prove that each plan generated by Phase I is *geometrically optimal*.

5.4.2 Phase I Finds Geometrically Optimal Plans

Definition 5.1 *An i-step plan is* **geometrically optimal** *if there is no i-step plan that collapses a larger set.*

Definition 5.2 *We say that plan ρ_i* **collapses** *the subset Θ_i if*

$$\rho_i(\Theta_i) = \Theta_0. \tag{5.4}$$

where Θ_0 includes only a single point in state space.

We prove that each subplan found by Phase I is geometrically optimal in three steps. First we show that any plan that collapses a discontiguous subset also collapses a larger contiguous subset. Then we shown that the maximal subset that can be collapsed by any plan is contiguous. Then we show that ρ_i collapses a maximal contiguous subset. The proof of the first part relies on the assumption that the transfer function is monotonically non-decreasing.

Definition 5.3 *The* **closure** *of set Θ, denoted by $C\Theta$, is the smallest contiguous set of state space that contains Θ.*

(This is not to be confused with the closure of an open set). If Θ is not contiguous, $|\Theta| < |C\Theta|$.

Lemma 5.1 *For any plan ρ, if ρ collapses Θ, then ρ also collapses the closure of Θ.*

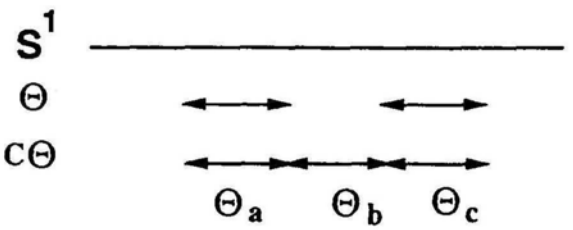


Figure 5.5: Θ is a discontiguous subset of the space of orientations, S^1. The figure illustrates the closure of Θ as the union of a pair of contiguous subintervals, Θ_a, Θ_c, and the interval between them, Θ_b.

Proof: If Θ is contiguous the lemma is obviously true. Say Θ is not contiguous. Let Θ_b be a gap in Θ as shown in figure 5.5 such that such that $\Theta_a \preceq \Theta_b \preceq \Theta_c$, $\Theta_a \cup \Theta_c \subseteq \Theta$, and $\Theta_a \cup \Theta_b \cup \Theta_c \subseteq C\Theta$.

We show that Θ_b must be collapsed by the plan that collapses its neighbors. For any interval Θ_x, let $\Theta_{x,i-j}$ be the image of interval Θ_x after j steps of plan ρ_i. For example $\Theta_{a,i} = \Theta_a$, the image after zero steps. And $\Theta_{a,0} = \Theta_0$, the image after i steps.

Say that $\rho(\Theta) = \Theta_0$, that is $\Theta_{a,0} = \Theta_{c,0} = \Theta_0$. We show that it must be the case that $\Theta_{b,0} = \Theta_0$.

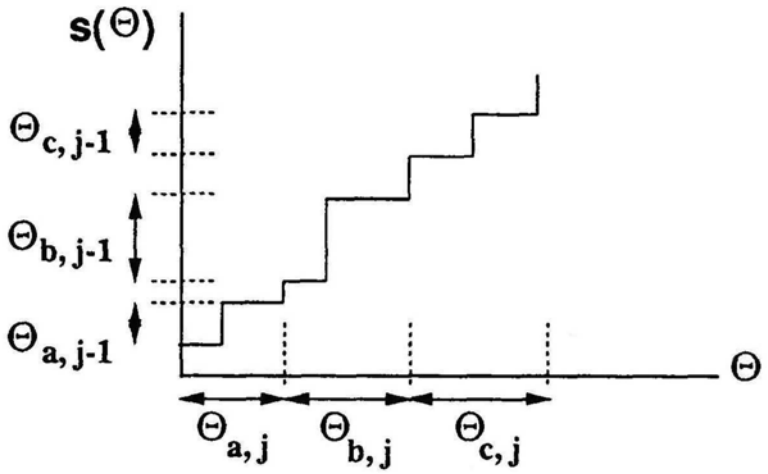


Figure 5.6: A portion of the step function showing Θ_a, Θ_b and Θ_c before and after step j in the plan. Note that if $\Theta_{a,j} \preceq \Theta_{b,j} \preceq \Theta_{c,j}$, then $\Theta_{a,j-1} \preceq \Theta_{b,j-1} \preceq \Theta_{c,j-1}$.

We show by induction that for all $j \leq i$,

$$\Theta_{a,j} \preceq \Theta_{b,j} \preceq \Theta_{c,j}. \tag{5.5}$$

Basis: It is true for $j = i$ since $\Theta_a \preceq \Theta_b \preceq \Theta_c$. Assume it is true for j. Then it must be true for $j - 1$ due to the monotonicity of the step function. This is shown graphically in figure 5.6. Thus it must be the case that

$$\Theta_{a,0} \preceq \Theta_{b,0} \preceq \Theta_{c,0}. \tag{5.6}$$

Since $\Theta_{a,0} = \Theta_{c,0} = \Theta_0$, it must be the case that $\Theta_{b,0} = \Theta_0$. We can argue similarly for other gaps to show that all points in the closure will be collapsed by the plan. ■

Lemma 5.2 *The maximal subset that can be collapsed with an i-step plan is contiguous.*

Proof: By contradiction. Assume Θ is not contiguous and is the maximal subset that can be collapsed with an i-step plan, and let ρ_i be the i-step plan that collapses it. By Lemma 5.1, ρ_i collapses $C\Theta$. Since $|\Theta| < |C\Theta|$, Θ is not the maximal subset that can be collapsed.

Lemma 5.3 *The set Θ_i found by Phase I at stage i is a maximal contiguous subset that can be collapsed with an i-step plan.*

Proof: By induction. It is obviously true for $i = 1$ by the definition of Phase I. If we assume that Θ_{i-1} is the maximal contiguous subset, then by the definition of Phase I, Θ_i will be the maximal contiguous subset. ■

Theorem 5.1 *Each ρ_i found by Phase I is geometrically optimal.*

Proof: Follows from Lemmas 5.2 and 5.3. ■

5.4.3 Orienting a Part up to Symmetry

If we let Phase I run until $h_i = T$, then the associated i-step plan will orient the part up to symmetry. Recall from section 4.4.5 that a plan orients a part *up to symmetry* if T is the smallest period in the part's transfer function and the plan's set of possible final states includes exactly $2\pi/T$ states that are equally spaced on S^1. Equivalently, we can say that a plan orients a part up to symmetry if T is the smallest period in the part's transfer function and the plan collapses some interval of angles, Θ, into a single orientation such that $|\Theta| = T$. To see this, note that if $s(\theta + T) = s(\theta) + T$, it follows that for any plan ρ, $\rho(\theta + T) = \rho(\theta) + T$. Consider a 2-step plan with actions at angles θ_1, θ_2 respectively.

$$\rho(\theta + T) \;=\; s(s(\theta + T - \theta_1) - \theta_2)$$

$$
\begin{aligned}
&= s(s(\theta - \theta_1) + T - \theta_2) \\
&= s(s(\theta - \theta_1) - \theta_2) + T \\
&= \rho(\theta) + T.
\end{aligned}
$$

If a plan collapses all angles in an interval to a single angle, then it will also collapse all angles in the interval offset by T to a single angle at offset T. The result will be a set of exactly $2\pi/T$ final angles equally spaced on S^1.

Theorem 5.1 implies that

Corollary 5.1.1 *If Phase I finds a plan to orient the part up to symmetry, it is the shortest such plan.*

5.4.4 Plans as Funnels

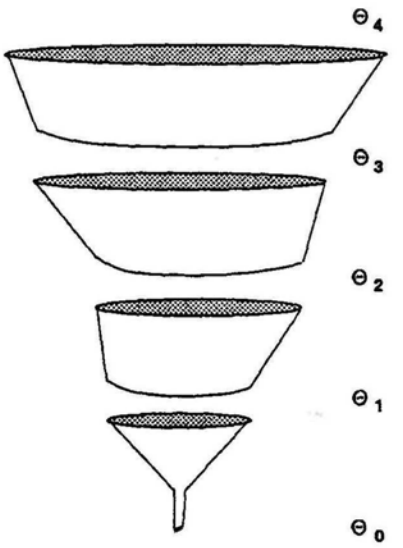


Figure 5.7: We can view the forward plan as a sequence of "funnels" that maps subsets of state space into each other.

We can visualize each plan ρ_i as a "funnel" that directs all states in Θ_i into Θ_0 (see figure 5.7). Note that we can re-position the "mouth" of the funnel, Θ_i, by rotating the gripper in the world frame as part of the first action in the plan. In the previous section we showed that each plan found by Phase I corresponds to a funnel with the widest possible mouth. Next we relate the size of the mouth to the expected cost of a plan.

1. For each subsequence of preimages found by Phase I, compute the expected cost of the associated plan: $\bar{C}(\rho_i) = 2\pi(i+1)/h_i$.
2. Return the plan with minimal expected cost as specified by Equation 5.3.

Figure 5.8: Phase II

5.5 Phase II

Phase II applies cost and probability models to the geometrically optimal plans to choose one with minimal expected cost. We relate geometric optimality with stochastic optimality to prove that the algorithm is correct.

5.6 Correctness

In section 4.5 we defined a stochastically optimal parts feeding plan to be a plan with minimal expected cost. If we assume that all actions have the same cost, then a geometrically optimal plan is stochastically optimal. To show this, we must relate the size of a plan's preimage to its expected cost.

Lemma 5.4 *For a uniform prior probability distribution, if an i-step plan is geometrically optimal then it is also stochastically optimal. (We relax the first condition in Appendix F).*

Proof: Let ρ_i be a geometrically optimal plan with preimage of length h. When the initial probability distribution is uniform, then the probability that plan ρ_i will succeed is $p = h/2\pi$, and the expected cost for the plan is $\bar{C}(\rho_i) = (i+1)/p$.

Let ρ_i' be any other plan with preimage of length h'. Note that $h' \leq h$, since h corresponds to a geometrically optimal plan. The probability that plan ρ_i' will succeed is $p' = h'/2\pi$. Since $h' \leq h$, $\bar{C}(\rho_i') = (i+1)/p' \geq \bar{C}(\rho_i)$. Thus there is no other plan that has lower expected cost and plan ρ_i is stochastically optimal. ■

Theorem 5.2 *The algorithm finds a stochastically optimal plan.*

Proof: Follows from Theorem 5.1 and Lemma 5.4.

Note that the position of the mouth in the world frame is irrelevant when the prior distribution is uniform. However, in cases where the prior distribution is non-uniform, we want to position the mouth so as to capture as much probability as possible. Appendix F discusses how the algorithm can be generalized to cases where the prior is non-uniform.

5.7 Completeness

To show that the algorithm is complete, that is, that it finds a stochastically optimal plan for any polygon, we must show that for any polygon, we can continue to grow the preimage until Phase I terminates. It is sufficient to show that we can continue to grow the preimage until $h_i = T$, or equivalently,

Theorem 5.3 *For any polygonal part, we can always find a plan to orient the part up to symmetry.*

Proof: As described earlier, any polygonal part will generate a squeeze function where all step widths have positive measure and $s(\theta + T) = s(\theta) + T$. All values are taken modulo 2π. We prove the claim by showing that for any squeeze function, we can always find a sequence of sets $< \Theta_0, \Theta_1, ..., \Theta_k >$ such that: the first set contains only a single point, each set has an image that is smaller than the previous set, and the last set corresponds to a period of symmetry in the step function.

The trick is to show that for any set we can always find a way to generate a larger set (unless the set corresponds to a period of symmetry in the step function). Since we are primarily concerned with the size of the sets, let $h_i = |\Theta_i|$.

Let the first two sets Θ_0, Θ_1 correspond to any step in the squeeze function. That is, all points in interval Θ_1 map into the single point Θ_0. As described earlier, we say that Θ_0 is the preimage of Θ_1. We must now show that we can find a set larger than Θ_1 whose image is no larger than Θ_1. Call this Θ_2. Then we need a set larger than Θ_2 whose image is no larger than Θ_2, and so on. We want to show that for any squeeze function and any h,

Either we can find a larger preimage:

$$\exists \theta, s(\theta + h) - s(\theta) < h, \tag{5.7}$$

Or h is a period of symmetry in the squeeze function:

$$\forall \theta, s(\theta + h) = s(\theta) + h, \tag{5.8}$$

where the quantifiers range over the interval $[0, T)$.

To understand formula 5.7, consider that we've reached a point in the algorithm where the current set is Θ of size h. Formula 5.7 says that there is some set, $[\theta, \theta + h)$, equal in size to Θ, whose image, $[\ s(\theta), s(\theta + h)\ )$, is strictly smaller than Θ. This means that we can expand $[\theta, \theta + h)$ to get a set larger than Θ whose image is equal to Θ. We can also interpret this with reference to figure 5.4. Formula 5.7 says that we can always find a position for the lower left hand corner of the box such that the squeeze function enters on the left edge of the box and exits on the right edge.

To show that for any squeeze function and any h, either formula 5.7 or formula 5.8 must hold, consider the integral of the function $s(\theta + h) - s(\theta) - h$ over the domain $[0, T)$.

$$\begin{aligned}
\int_0^T [s(\theta + h) - s(\theta) - h]d\theta &= \int_h^{T+h} s(\theta)d\theta - \int_0^T s(\theta)d\theta - hT \\
&= +\int_0^T s(\theta)d\theta - \int_0^h s(\theta)d\theta + \int_T^{T+h} s(\theta)d\theta - \int_0^T s(\theta)d\theta - hT \\
&= -\int_0^h s(\theta)d\theta + \int_T^{T+h} s(\theta)d\theta - hT \\
&= -\int_0^h s(\theta)d\theta + \int_0^h [s(\theta) + T]d\theta - hT \\
&= -\int_0^h s(\theta)d\theta + \int_0^h s(\theta)d\theta + hT - hT \\
&= 0.
\end{aligned}$$

Since this integral is zero, then by the mean value theorem, either the function is uniformly zero (formula 5.8, i.e. $h = T$) or there is some point in the domain where the function is less than zero (formula 5.7).

Hence we can always continue to find larger sets until we reach a period of symmetry in the step function. We have shown earlier that we can transform this sequence of sets into a plan to orient the part up to symmetry. This proves the claim. ■

5.8 Worst-Case Computational Complexity

We define the complexity of the algorithm as a function of the geometric complexity of the input, in this case the number of part vertices, n. We neglect the numerical complexity of representing vertices and angles as rational numbers.

Step 1 of Phase I, computing the squeeze function, can be performed in time $O(n \log n)$ (see Appendix D). Steps 2 and 3 of Phase I run in time $O(n)$. To see this, note that we only need to consider positioning the box flush with each step and there are $O(n)$ steps. We only need $O(n)$

iterations of Step 3, since h_1 corresponds to the largest single step and there are no more than $2n$ steps in the transfer function, $h_1 \geq 2\pi/2n$ and so N is $O(n)$ as stated in Theorem 4.1.

Phase II runs in time $O(n)$, since there will be at most $O(n)$ plans and it takes time $O(1)$ to compute the expected cost for each plan.

Theorem 5.4 *The algorithm runs in time $O(n^2)$ and finds plans of length $O(n)$.*

See Appendix F for the complexity when the algorithm is modified to handle non-uniform prior densities.

As stated earlier, if we modify the termination condition to stop only when $h_i = T$, the algorithm will find a plan to orient the part up to symmetry using only part geometry. We call this the purely geometric algorithm. Although the $O(n^2)$ time bound applies to the stochastically optimal plan, the best we have been able to show is that the purely geometric algorithm runs in time $O(n^3)$: there are $O(n)$ steps in the squeeze function and so there are only $O(n^2)$ intervals, so the required number of iterations of Step 3 is $O(n^2)$.

Conjecture 5.4.1 *The purely geometric algorithm finds a plan to orient the part up to symmetry in time $O(n^2)$.*

5.9 Implementation

We implemented the algorithm in Common Lisp using exact (rational) arithmetic to express vertices and angles. In the examples below, $c = 1$. For 1000 random parts, we compared the plans generated by the algorithm to the plans found by exhaustive breadth-first search. In all cases where the exhaustive search was able to run to completion, the plans found by the two planners were equivalent. Examples of the geometric analysis are shown in Figures 5.9 through 5.11.

5.10 Discussion

In this chapter we developed a backchaining algorithm to find stochastically optimal parts-feeding plans assuming that the prior probability density is uniform. We prove that the algorithm is correct by relating stochastically optimality to geometric optimality and show that the worst-case computational complexity of the algorithm is $O(n^2)$, where n is the number of edges on the part. Finally we show that the algorithm is complete for all polygons by showing that we can orient any polygonal object up to symmetry in its transfer function. In Appendix F,

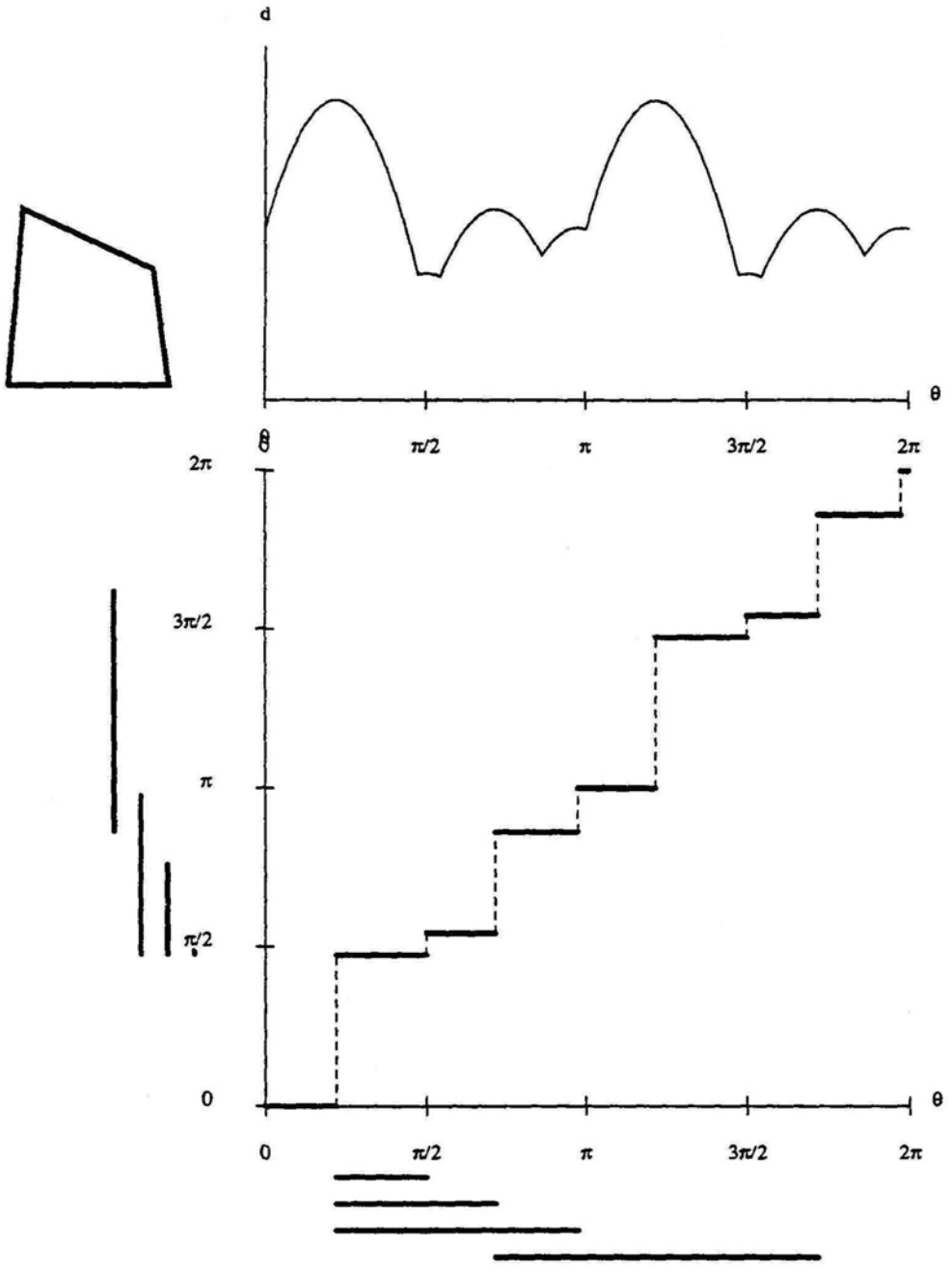


Figure 5.9: Geometric analysis of 4-sided part (4gon), showing diameter and squeeze functions. The horizontal bars show the preimage at each stage. The corresponding plan is shown in Figure 4.6.

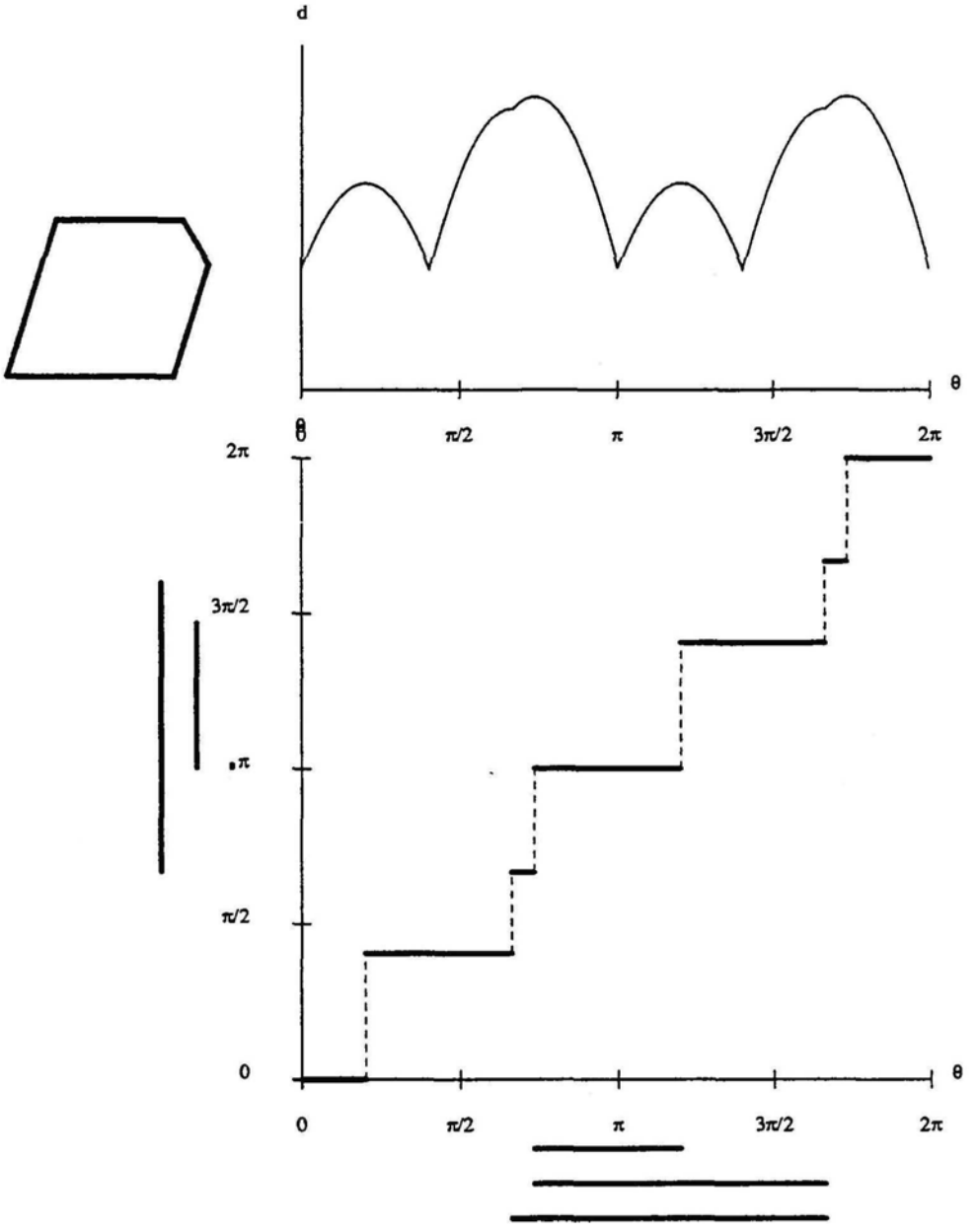


Figure 5.10: Geometric analysis of 5-sided part (5gon).

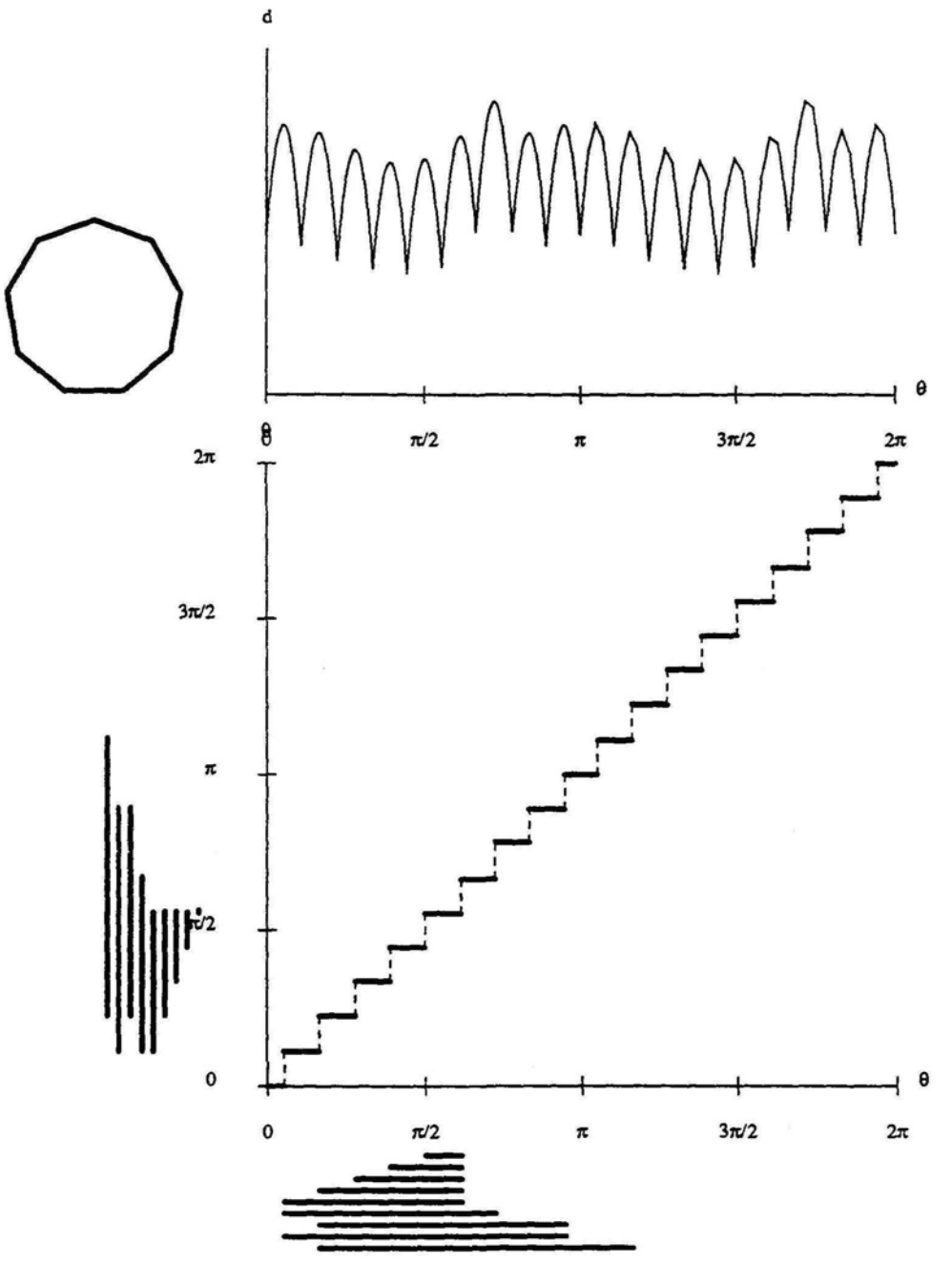


Figure 5.11: Geometric analysis of 9-sided part.

we explore how the algorithm can be generalized to cases where the prior probability density is non-uniform.

The analysis in this chapter relies on the assumption that action space can be described by cyclic shifts of a transfer function, $s : S^1 \times S^1$, that is piecewise constant and monotonically non-decreasing on a closed interval. In addition to the class of *squeeze* actions the class of *push-grasp* actions meet these criteria. Push-grasp actions are explored in Appendix E. Perhaps we can find other classes of actions that meet these criteria. It would be interesting to extend this algorithm to actions that are not deterministic, such as the class of fence-push actions considered by [Peshkin and Sanderson, 1988]. In the next chapter we consider the class of tray-tilting actions proposed by [Erdmann and Mason, 1986]. Since these actions are not deterministic, we use forward search rather than the backchaining algorithm to find optimal plans.

Chapter 6

Empirical Model of Control Uncertainty

> When the mechanics are non-deterministic...it will be necessary to assign probabilities to the alternative events. This might be done empirically...[Taylor, Mason, and Goldberg , 1987]

6.1 Introduction

We now turn our attention to control uncertainty – where the outcome of an action cannot be determined exactly. In this chapter we derive a stochastic control model based on empirical observations. (In Appendix C, we consider control uncertainty in the context of pushing with friction where we assume a uniform distribution to derive a stochastic control model analytically.)

Control uncertainty arises from many sources including friction, backlash, dynamics, impact, and sampling effects [Goldberg and Pearlmutter, 1988]. To model such effects, we often assume that uncertainty can be modeled with a one- or two-parameter probability model. For example, when control uncertainty can be characterized by zero-mean additive Gaussian noise, then for linear systems with quadratic cost functions, it can be shown that the optimal plan depends only on the mean of the Gaussian distribution [Gelb, 1986]. When we don't know the parameters of the probability model, we can *sample* the physical system to estimate the parameters.

In this chapter we apply this approach to a different mechanism for feeding parts. We use the experimental setup developed by Alan Christiansen in our laboratory at Carnegie Mellon. He is interested in machine learning; how can a machine automatically generate a model of robot control based on empirical observations? [Christiansen, 1991]. To collect data, he developed a

---

[0]Parts of this chapter have appeared in [Christiansen and Goldberg, 1990].

system including robot arm and CCD camera to automatically perform experiments and record the results. In this chapter we derive a stochastic control model based on this data and use it to generate optimal plans.

6.2 The Domain: Tray-Tilting

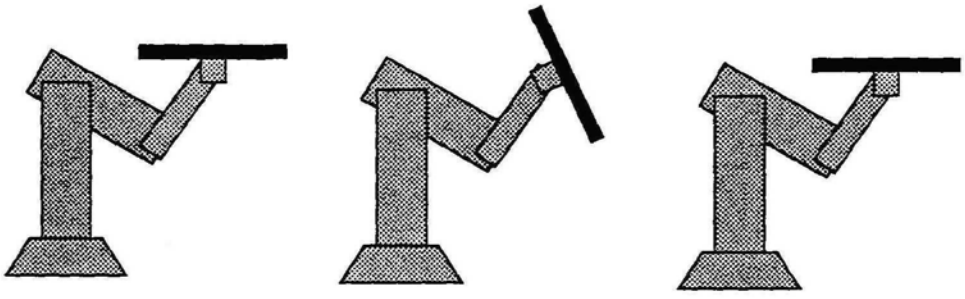


Figure 6.1: Three side views of the robot holding the tray. In the center the robot tilts the tray allowing gravity to move the part.

We consider a problem in robot manipulation where our objective is to orient a two-dimensional part in a tray though a sequence of *tilting* motions as in Figures 6.1 and 6.2. Tray-tilting has been studied before, both analytically [Erdmann and Mason, 1986, Taylor, Mason, and Goldberg , 1987] and empirically [Christiansen *et al.*, 1990]

The state is the position and orientation of a polygonal part that is free to slide in a square tray. Each action is defined by an angle at which the robot tilts the tray, causing the object to slide and roll into a stable state along some wall of the tray. We uniformly quantize state and action spaces. We classify stable states of the rectangular part into 12 equivalence classes as shown in Figure 6.2: the part can be in one of the four corners or the four sides and it can be oriented either horizontally or vertically. These are the canonical states of the system. The tray is initially held orthogonal to the gravity field. Tilting the tray is performed by rotating the tray about an axis in the plane of the tray, waiting for a fixed time until the object settles, and then returning the tray to its original orientation. Tilting actions are specified by the direction of the steepest vector pointing into the gravity field. We consider twelve tilt angles at 30 degree increments.

Figure 6.3 gives an example of control uncertainty. Although the tilting direction fairly repeatable, the true initial position and orientation of the part may vary quite a bit within canonical state 7. That is, our model of the state space cannot distinguish between positions of the part within canonical state 7. This confusion might be termed "model aliasing" [Shafer, 1990].

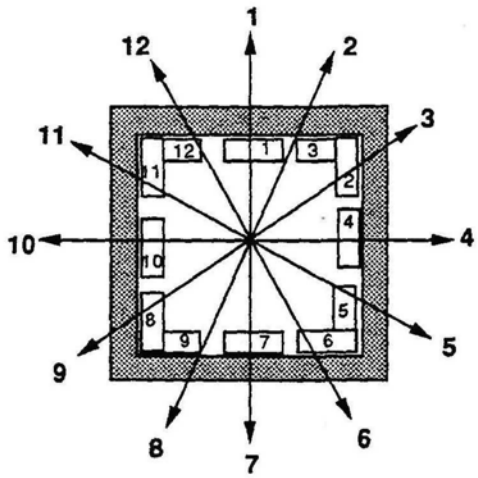


Figure 6.2: Looking down into the tray. The rectangular part is shown in its 12 canonical states. The arrows indicate the 12 tilting actions.

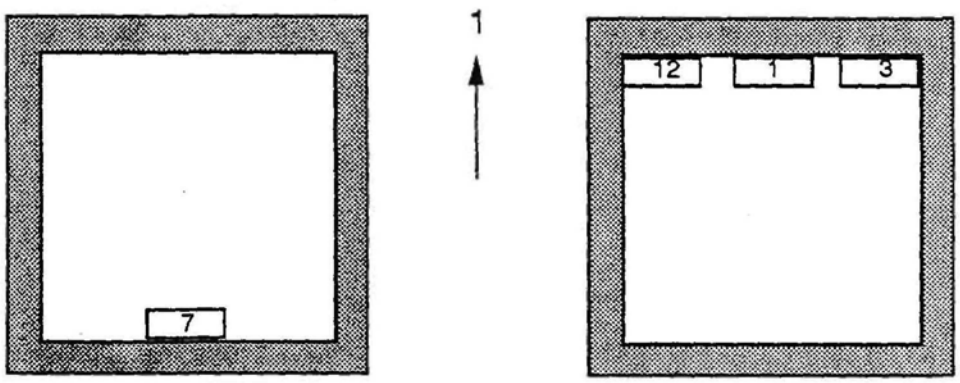


Figure 6.3: On the left is the tray in its initial state with the rectangular part in state 7. On the right is the tray after tilting the tray so that the arrow corresponding to action 1 points into the gravity field. After observing multiple trials, we conclude that there are three possible outcomes: the final state is either 1, 3, or 12.

Since we can't predict the outcome of actions exactly due to model aliasing, we propose a probabilistic model of control. We assume that actions have the Markov property, i.e. where the mapping depends only on the current hyperstate. For each action and initial state, we want a probability distribution over the set of final states. This can be represented with a stochastic transition matrix for each action as described in Chapter 2. Given the initial state of the system and a desired final state, we want to find an open-loop sequence of tilting actions that will reach this state with maximal probability.

Where do we get the numbers for the transition matrices? The effects of friction and impact in this domain make it particularly difficult to derive these matrices analytically. Hence we estimate the matrices through observation.

6.3 Developing a Stochastic Control Model from Observations

Given a set of observations from physical experiments, what is an appropriate stochastic model of actions? We represent each tilting action a with a stochastic transition matrix, $\mathbf{P}_a$, where p_{ij} is the (conditional) probability that the system will be in state j after action a is applied to state i.

In the physical experiments, each observation consists of an initial state, a tilting action, and a final state. For each tilting action a, consider the matrix $\mathbf{X}_a$, where x_{ij} is the number of observations with initial state i and final state j. Given an observation matrix $\mathbf{X}_a$, how do we generate a stochastic transition matrix $\mathbf{P}_a$?

One idea is to use the observed frequencies (the maximum likelihood estimator). The difficulty is that we haven't necessarily observed the outcome of every pair of initial states and actions; for such pairs the frequency is undefined.

6.3.1 Bayesian Estimation of Multinomial Parameters

Consider an experiment where the outcome is uncertain. We might treat the outcome as a random variable with some probability distribution. Call this the *state* distribution. Say this distribution has a finite set of parameters. When we don't know the parameters, one approach is to treat the vector of parameters as a second random variable with its own probability distribution. Call this the *parameter* distribution. We use trials of the experiment to estimate the mean of the parameter distribution. As the mean of the parameter distribution converges to the true value of the parameters, we are better able to predict the outcome of future experiments. In this section we review a Bayesian method for estimating the parameters of a Multinomial distribution based on observations.

Let $\mathbf{x}$ be a vector of observed data. The prior probability density of the data is $f(\mathbf{x}|\mathbf{p})$, where $\mathbf{p}$ is a vector of parameters for the distribution. That is, we can use $\mathbf{p}$ to estimate the probability of $\mathbf{x}$. But we're not certain about the value of $\mathbf{p}$. In Bayesian inference we turn $f(\mathbf{x}|\mathbf{p})$ around to get a probability distribution for the parameters given the data:

$$f(\mathbf{p}|\mathbf{x}) = \frac{f(\mathbf{x}|\mathbf{p})\ f(\mathbf{p})}{f(\mathbf{x})}. \tag{6.1}$$

where $f(\mathbf{p})$ is the prior probability density for $\mathbf{p}$ and $f(\mathbf{x})$ is a normalizing factor: $f(\mathbf{x}) = \int \mathbf{f}(\mathbf{x}|\mathbf{p})\mathbf{f}(\mathbf{p})\mathbf{dp}$. Note that we assume here that $f(\mathbf{x})$ is non-zero.

In Bayesian *estimation* we estimate the parameter vector by minimizing some loss function (usually mean squared error). A *Bayesian estimator*, $\hat{\mathbf{p}}(\mathbf{x})$, is a function that maps the data into an optimal estimate of $\mathbf{p}$.

There are three parts to a Bayesian estimation problem: a prior distribution for the data, a prior distribution for the parameter vector, and a loss function. We consider a specific problem: estimating the parameters for a multinomial distribution.

A Prior Distribution for the Data: the Multinomial

Our initial predictions of the data will be based on the multinomial distribution, which is the multivariate counterpart to the binomial.

Consider an urn filled with balls of k different colors ($k \geq 2$) where the proportion of balls that are of color i is p_i. Note that $p_i > 0$ and $\sum p_i = 1$. Now suppose that we draw n balls with replacement. Let X_i be the observed number of balls of color i. Then the random vector $\mathbf{X} = (X_1, ..., X_k)$ has a *multinomial* distribution with parameters n and $\mathbf{p} = (p_1, ..., p_k)$,

$$f(\mathbf{x}|n, \mathbf{p}) = \frac{n!}{x_1!...x_k!} p_1^{x_1} ... p_k^{x_k}. \tag{6.2}$$

For example, if there are 3 red balls, 2 black balls, and 5 white balls and we draw 2 balls with replacement, the probability of drawing one red and one black ball is

$$f[(1,1,0)|2,(.3,.2,.5)] = \frac{2!}{1!1!0!}.3^1.2^1.5^0 = 0.12 \tag{6.3}$$

Note that the expected value for any dimension of the random vector is $E(X_i) = np_i$. Similarly, $Var(X_i) = np_i(1 - p_i)$ and $Cov(X_i, X_j) = -np_ip_j$.

Our predictions about the data will be based on a multinomial distribution with parameter vector $\mathbf{p}$. We don't know the value of $\mathbf{p}$ exactly, but we can model this uncertainty with another probability distribution.

A Prior Distribution for the Parameters: the Dirichlet

Let the parameter space, Ω, be the set of vectors $\{\mathbf{p}|p_i > 0 \text{ and } \sum p_i = 1\}$. A convenient class of probability distributions on this space is the *Dirichlet*, the multivariate counterpart to the beta distribution (see Appendix G).

A random vector $\mathbf{p} = (p_1, ..., p_k)$ has a Dirichlet distribution with parametric vector $\alpha = (\alpha_1, ..., \alpha_k)$, where $\alpha_i > 0$. For any point in Ω,

$$f(\mathbf{p}|\alpha) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)...\Gamma(\alpha_k)} p_1^{\alpha_1 - 1} ... p_k^{\alpha_k - 1}. \tag{6.4}$$

where $\alpha_0 = \sum \alpha_i$, and the Gamma function is the continuous counterpart to the factorial.

Note that the expected value along any dimension of the random vector is $E(p_i) = \alpha_i/\alpha_0$. Similarly, $Var(p_i) = \alpha_i(\alpha_0 - \alpha_i)/\alpha_0^2(\alpha_0 + 1)$ and $Cov(p_i, p_j) = -\alpha_i\alpha_j/\alpha_0^2(\alpha_0 + 1)$.

The Dirichlet is a continuous distribution over a $(k-1)$-dimensional simplex, so it can be hard to visualize. See Appendix G. We can represent a uniform distribution over the parameter space by setting $\alpha_1 = ... = \alpha_k = 1$.

A Posterior Distribution for the Parameter: the Dirichlet!

Fortunately, the Dirichlet is a *conjugate family* for samples from a multinomial distribution. That is, after sampling the data, the posterior distribution for $\mathbf{p}$ will also be a Dirichlet. From equation 6.1 we see that the posterior distribution is proportional to the product of a multinomial density and a Dirichlet density.

$$f(\mathbf{p}|\mathbf{x}) \propto f(\mathbf{x}|\mathbf{p})\, f(\mathbf{p}). \tag{6.5}$$

The multinomial density has the form

$$f(\mathbf{x}|\mathbf{p}) \propto p_1^{x_1} ... p_k^{x_k}, \tag{6.6}$$

and the Dirichlet density has the form

$$f(\mathbf{p}) \propto p_1^{\alpha_1 - 1} ... p_k^{\alpha_k - 1}. \tag{6.7}$$

Multiplying, we see that the posterior distribution has the form of a Dirichlet,

$$f(\mathbf{p}|\mathbf{x}) \propto p_1^{\alpha_1 + x_1 - 1} ... p_k^{\alpha_k + x_k - 1}. \tag{6.8}$$

That is, the posterior distribution for $\mathbf{p}$ after observing vector $\mathbf{x}$ is a Dirichlet distribution with parametric vector $\alpha = (\alpha_1 + x_1, ..., \alpha_k + x_k)$.

A Loss Function: Squared Error

We now want to use the posterior distribution for the parameter to find the "best" estimate of the parameter. We would like our estimate to be close to the actual value of the parameter. We can formalize this by choosing a value for the parameter that minimizes the mean square difference between the estimate and the true value.

Minimizing Mean Square Error

Suppose $\mathbf{P}$ is a random variable with mean μ. We want to predict the value of an observation. One way is to find the value $\hat{\mathbf{p}}$ such that the Mean Square Error (MSE), $E[(\mathbf{P} - \hat{\mathbf{p}})^2]$, will be a minimum. For any value of $\hat{\mathbf{p}}$,

$$E[(\mathbf{P} - \hat{\mathbf{p}})^2] = E[\mathbf{P}^2] - 2\hat{\mathbf{p}}\mu + \hat{\mathbf{p}}^2. \tag{6.9}$$

To minimize this quantity, we differentiate with respect to $\hat{\mathbf{p}}$ and set the result equal to zero; the value of $\hat{\mathbf{p}}$ that gives a minimum is the mean, $\hat{\mathbf{p}} = \mu$. So for any distribution the MSE estimate is the mean.

See [Berger, 1985, Section 4.4.2] for a discussion of alternative loss functions. For example, it can be shown that the Bayesian estimator for the *absolute* error loss criterion is the *median* of the posterior.

A Bayesian Estimator for the Multinomial

We are now ready to put together the three pieces: If the prior distribution of the data is a multinomial with parameter $\mathbf{p}$ and the prior distribution of $\mathbf{p}$ is a Dirichlet with parametric vector $\alpha = (\alpha_1, ..., \alpha_k)$, then the posterior distribution of $\mathbf{p}$ after observing $\mathbf{x}$ is a Dirichlet with parametric vector $\alpha = (\alpha_1 + x_1, ..., \alpha_k + x_k)$. For the mean square error loss criterion, then the Bayesian estimator is the mean of the posterior,

$$\hat{p}_i(\mathbf{x}) = \frac{\alpha_i + x_i}{\alpha_0 + n}, \tag{6.10}$$

The variance associated with this estimate is

$$Var(\hat{p}_i(\mathbf{x})) = \frac{(\alpha_i + x_i)(\alpha_0 + n - \alpha_i - x_i)}{(\alpha_0 + n)^2(\alpha_0 + n + 1)}. \tag{6.11}$$

Note that the variance decreases as sample size, n, is increased.

The Maximum Likelihood Estimator for the Multinomial

The Bayesian estimator is not the only way to skin a cat. The *maximum likelihood* approach is to estimate $\mathbf{p}$ by finding the value of $\mathbf{p}$ that maximizes the function $f(\mathbf{x}|\mathbf{p})$, where $\mathbf{x}$ is the observed vector. That is, what value of $\mathbf{p}$ makes it most probable that we observe what we observed? To find the most likely value, we differentiate $f(\mathbf{x}|\mathbf{p})$ with respect to $\mathbf{p}$ and set the result equal to zero.

It can be shown that the maximum likelihood estimator (MLE) for the parameter of a multinomial distribution after observing a vector of n outcomes is

$$\hat{p}_i(\mathbf{x}) = \frac{x_i}{n}. \tag{6.12}$$

Maximum Likelihood vs. Bayes'

The maximum likelihood method does not depend on either a prior distribution of $\mathbf{p}$ or on the loss function. However the MLE is not always unique, e.g. for samples from a uniform distribution [DeGroot, 1975, p288] and it is undefined when there are no observations: $n = 0$. When the sample size approaches infinity, it can be shown that both the MLE and the Bayes' estimators converge to the true parameters of a multinomial distribution.

Bayes' Estimator for Tilting Actions

For the tray-tilting problem, we use the following estimator for each tilting action corresponding to $\mathbf{P_a}$,

$$\hat{p}_{ij}(\mathbf{x}) = \frac{\alpha_{ij} + x_{ij}}{\sum_j \alpha_{ij} + x_{ij}}, \tag{6.13}$$

where the numbers α_{ij} for $i = 1, 2, ..., 12$ are Dirichlet parameters based on *a priori* assumptions and the x_{ij} correspond to elements fo the observation matrices.

We could set $\alpha_{ij} = 1.0$ to represent the prior assumption that the conditional probability distribution is uniform: after an action is applied, the system is as likely to be in any one state as in any other. For the tray tilting problem, we set $\alpha_{ij} = .01$ to represent our prior assumption that the conditional probability distribution for each action will be skewed toward some subset of states. See Appendix G for more on the Dirichlet parameters.

6.3.2 Physical Observations

To build a model of tray-tilting actions, the robot performed a sequence of trials. The part was initialized in some position and orientation in the tray. This initial state was detected by the

0.9901	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.000
0.0006	0.9932	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.000
0.0005	0.0005	0.9945	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.000
0.0020	0.9785	0.0020	0.0020	0.0020	0.0020	0.0020	0.0020	0.0020	0.0020	0.0020	0.002
0.0003	0.7394	0.2574	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.000
0.0528	0.0005	0.9419	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.000
0.7401	0.0012	0.1244	0.0012	0.0012	0.0012	0.0012	0.0012	0.0012	0.0012	0.0012	0.124
0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.9214	0.071
0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.0006	0.993
0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.0009	0.9901	0.000
0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.9922	0.000
0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.0008	0.991

Figure 6.4: Here is the transition matrix, $\mathbf{P}_1$, associated with tilting the tray so that the vector 1 is pointing into the gravity field. Row 7 gives the probabilities associated with each final state, i.e. the value in element $p_{7,1}$ says that if the part is initially in state 7, its probability of moving to state 1 is 0.7401.

vision system. The robot then generated a random number from zero to twelve and tilted the tray along the corresponding azimuth. The initial state, action and final state was recorded by incrementing the corresponding element in an observation matrix. Say the part was initially in state 5, the robot randomly chose action 2, and the final state was 3. The system would increment element x_{53} in matrix $\mathbf{X}_2$.

We assumed that each observation was independent. The robot performed 2000 trials with the physical robot system to yield the $\mathbf{X}_a$ matrices [1]. We used Equation 6.13 to generate the corresponding stochastic transition matrices. A typical transition matrix is shown in figure 6.4.

6.4 Finding Stochastically Optimal Plans

For the tray tilting task we are given a known initial state and desired final state. In this case the initial hyperstate is a vector with a 1 corresponding to the initial state and zeros elsewhere. Each action (and hence plan) defines a final hyperstate using the stochastic transition matrices described in the previous section. To compare plans, we compare their final hyperstates.

We express our desire for a particular outcome with a cost function on the set of hyperstates. Typically, we assign a cost to individual states and define the cost of a hyperstate to be the

[1] Empirical data used in this chapter was generously provided Alan Christiansen.

weighted average (expected cost) of its components. For the tray tilting task the cost function depends on the desired final state. Say we want to reach state i. Let

$$C(\lambda) = -p_i,$$

so that the minimum cost hyperstate corresponds to the highest probability that the system is in state i. There may be more than one minimum-cost hyperstate. Note that this cost function does not depend on the number of actions.

To find the best plan, we consider all plans and find one with minimum cost. The difficulty is that there is an infinite number of plans to consider. We set a cutoff threshold depending on how much time we have and how fast we can evaluate plans, which in turn depends on how fast we can multiply matrices. Say we consider all plans up to some length limit, k. Let n be the number of states and m be the number of actions. There are m^k k-step plans. We can visualize the search for an optimal strategy as proceeding through a tree, where the root node contains the initial hyperstate and has a branch for each action in the action space. Each branch leads to a new hyperstate which in turn has branches for each action. We expand the tree to some fixed depth (horizon) and select the optimal path. To generate each node in the tree we must perform $O(n^2)$ multiplications. The total time for finding the best k-step plan is $O(n^2m^k)$. In the present case we considered all plans with length ≤ 3 and found a plan with minimal cost. With 12 actions available at each stage, a 3-ply search considers $12^3 = 1728$ hyperstates.

For twelve possible tile configurations, there are 144 pairs of configurations defining start state and goal state. Let us refer to each of these pairs as an *instance* of the planning problem. There are 132 non-trivial instances. (The twelve instances with start state and goal state equal to each other are trivial, since a null plan always solves the problem.) We ran the planner on each of the 132 non-trivial instances. Table 6.1 lists some of the resulting plans.

The planner was written in Common Lisp and run on a Sun 3/280. Over the 132 instances, the planner took an average of 62 seconds real time per instance, with a standard deviation of 2.8 seconds. The average length of plans was 2.4 tilts with a standard deviation of 0.75 tilts.

To test the resulting plans, we executed the last two instances of Table 6.1 on the robot in 400 trials. The first plan performed very close to the predicted probability. The second plan performed slightly better than predicted. The prediction may have been low because there were not enough observations to produce an accurate transition matrix. In [Christiansen and Goldberg, 1990], we compare these plans against plans generated by an alternative planning algorithm.

6.5 Discussion

In previous chapters we studied the problems of grasping and orienting parts with a parallel-jaw gripper. In this chapter we study an alternative method for orienting parts: tray-tilting. We use

Instance			
Start State	Goal State	Plan	Estimated Probability
1	3	(4)	.98
1	2	(9 7 3)	.97
1	4	(5)	.97
1	5	(6 2 7)	.98
1	6	(4 7 4)	.98
1	7	(9 6)	.97
1	9	(10 10 7)	.97
1	8	(9 7)	.98
1	10	(9)	.99
1	11	(9 1)	.98
1	12	(10 2 10)	.97
		...	
8	1	(5 11 2)	.62
		...	
2	1	(10 2 3)	.67
2	3	(11 4)	.98
		...	

Table 6.1: Sample plans.

experimental data to estimate stochastic transition matrices and then use these matrices to plan stochastically optimal tray-tilting strategies.

This chapter serves several purposes. Tray-tilting gives another example where control uncertainty arises in a manipulation problem. Also, since control uncertainty is particularly difficult to model analytically due to friction and dynamics, the tray-tilting problem justifies the use of empirical data to estimate the control model. Last, tray-tilting illustrates another application of the stochastic framework for manipulation planning.

Start State	Goal State	Plan	Estimated	Measured
2	3	(11 4)	.98	.99
2	1	(10 2 3)	.67	.85

Table 6.2: Summary of 400 execution trials for two plans showing estimated and measured success ratios.

Chapter 7

Discussion and Future Work

We have formalized a stochastic framework for robotic manipulation planning and developed two examples. In this chapter we discuss limitations and extensions to the stochastic framework.

7.1 Limitations of Stochastic Planning

Two common objections to stochastic methods are (i) "There is little justification for the required probability and cost models." and (ii) "It is impractical to find optimal solutions." We cannot dismiss either of these objections. But there is hope.

7.1.1 "Probability and Cost Models are Ad-Hoc."

If the probability and cost models used for stochastic planning are not well founded, then we certainly can't expect much from the resulting plans: garbage in, garbage out. One aim of this thesis is to illustrate how probability and cost models can be justified in geometric manipulation problems.

One advantage of guaranteed plans is that we can avoid making probabilistic assumptions. All we need to specify is the set of possible states that can result from an action. State sets may be simpler to express than probability distributions, especially when there is no closed-form expression for the probability distribution that results from an action. When guaranteed plans exist, we can consider a hybrid approach that uses a real-valued cost metric to rank guaranteed plans. The best plan might be called a *minimax plan*, in that it minimizes the worst-case cost. For example, a cost model is implicit when breadth-first search is used to find a shortest plan. But for cases where there is no guaranteed plan, however, probabilistic models may be a necessary evil.

In this thesis we derived prior probability models *analytically* from part geometry (chapters 3 and 4) and *empirically* using observations of system behavior (chapter 6). We also related cost to the probability of success for iterative plans. Empirical methods can also be used to derive cost models. Automated methods for building probability and cost models from empirical observations may be considered as examples of *machine learning* [Narendra and Thathachar, 1989].

7.1.2 "Stochastic Planning is Intractable."

The brute force approach to stochastic planning is indeed intractable. For an n-dimensional state space, each action is represented with an $n \times n$ matrix, and so we require $O(n^2)$ multiplications to determine the result of any single action. If there are m actions, the brute force approach requires $O(m^d n^2)$ time to consider all plans with d or fewer actions. We note here that the brute force approach can be trivially parallelized.

There is often a tradeoff between planning time and plan quality: in general the longer we have to plan, the better plans we find. We can reduce planning time by accepting the first plan that achieves some cost threshold. That is, rather than looking for an optimal plan, we look for a *satisfactory* plan. There is some evidence that humans plan in this manner [Simon, 1955]. A way to speed up this type of planning would be to *search* probabilistically, for example by choosing the next node randomly [Barraquand and Latombe, 1990]. Such heuristics might work well when there are many satisfactory plans.

Christiansen and Goldberg [1990] compared exact and heuristic methods for stochastic planning in the tray-tilting domain. The heuristic used a shortest-path algorithm to minimize a lower bound on the probability that a plan would reach the goal. While the exact method requires computation time exponential in the size of the action space, the heuristic method runs in polynomial time. When we compared methods, we discovered that there was little difference in plan quality. It would be interesting to apply the shortest-path heuristic to other domains.

Another approach is to incorporate planning time into the cost function to optimize the tradeoff between plan time and execution time [Kanazawa and Dean, 1989, Etzioni, 1989, Tan, 1990]. A similar tradeoff exists in the design of optimizing compilers where time spent on optimizing code must be balanced against execution savings. When plans must be generated on-line, we are often interested in the best plan that can be found before some deadline [Drummond and Bresina, 1990]. Plan time is less critical if planning is done off-line. This is especially true in a factory environment where we can amortize plan time over thousands of execution cycles.

In this thesis we demonstrated an efficient algorithm for finding stochastically optimal plans in the special case where the set of actions can be described with cyclic shifts of a piecewise constant monotonic transfer function.

7.2 Future Work: Stochastic Plans with Sensors

> Recently, Mason and his colleagues have begun to investigate the trade-offs between (i) a sequence of [open-loop actions] such as allowing a part to collide with the sides of a tray, and (ii) using a sensor to constrain the configuration of the part. Such analyses have a great potential throughout robotics. *The heart of the problem is to relate the constraint on the state of an object as a result of a sensing step with the result of an [open-loop action].* [Brady, 1989, italics mine.]

In chapter 1 we identify two methods for reducing uncertainty: sensing and open-loop actions. The thesis concentrates on the latter, showing that expected performance can be used as a real-valued metric for evaluating open-loop plans. What about *closed-loop* plans, that is, plans that use sensors?

In this section, we consider how the stochastic planning framework can be extended to planning with sensors. Recall that a sensing operation transforms a prior probability distribution into a posterior probability distribution. This is also the way we model open-loop actions. Thus we can compare the effect of receiving sensory data with the effect of an open-loop action based on their respective posterior hyperstates. Since we don't know the sensor data at plan time, we can compute the hyperstate for each possible sensor value and take a weighted average to find the expected cost of sensing vs. the expected cost of an action. In this way we can compare sensing with acting.

7.2.1 Sensor Data Transforms a Hyperstate

Consider a system with finite state space, Θ. Since we don't know the exact state of the system, we describe it with a hyperstate, a probability distribution on Θ that gives $P(\theta)$ for each state $\theta \in \Theta$. Now consider a sensor that can return data, $x \in X$, where X is a finite set. The probability that x will be measured, given that the system is in state θ, is given by a conditional probability, $P(x|\theta)$. This allows us to represent *imperfect* sensors, where the sensed value may be corrupted by noise. Now if we apply the sensor to the system and measure x, what is the new hyperstate?

Using Bayes' Theorem, the new hyperstate, also known as the posterior distribution given data x, is

$$P(\theta|x) = \frac{P(x|\theta)\ P(\theta)}{m(x)}, \tag{7.1}$$

where $m(x)$ is a normalizing factor:

$$m(x) = \sum_{\Theta} P(x|\theta)\ P(\theta). \tag{7.2}$$

For a given hyperstate, $m(x)$ gives the probability that x will be measured; $m(\cdot)$ is known as the *marginal distribution.*

We can implement Bayes' Theorem with matrices. As in chapter 2, we represent the initial hyperstate with a vector λ. We represent the conditional probability distribution $P(x|\theta)$ with a diagonal matrix, $\mathbf{P_x}$ such that

$$p_{i,j} = \begin{cases} P(x|\theta_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{7.3}$$

The hyperstate resulting from measuring x is $\lambda\mathbf{P_x}$, normalized so that its elements sum to 1.0.

7.2.2 Plans with Sensors

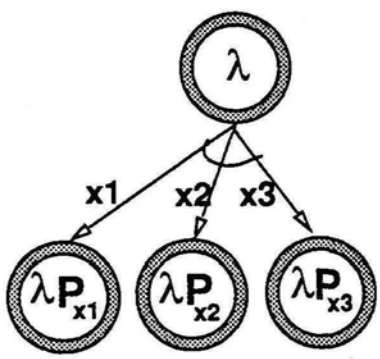


Figure 7.1: Representing a single-step sensor plan with a tree.

Consulting a sensor introduces *branching* into a plan, since the course of action will depend on the data returned by the sensor. A sensor plan can be represented as a tree where each node represents a hyperstate (figure 7.1). The tree has branches for each value of sensor data. A complete plan specifies a course of action for all possible values of sensor data.

When treating open-loop plans, we compared plans based on their final hyperstates. For plans that use sensors, the final hyperstate depends on the sensed data, which we don't know at plan time. If we compute the cost for each possible hyperstate, however, we can use the marginal probability distribution to compute an *expected cost* for the plan. Thus we can compare closed-loop plans based on expected cost.

7.2.3 Example: Sensing Gripper Diameter

We can incorporate sensing in the parallel-jaw parts feeder by including a sensor that returns gripper diameter when the jaws are closed around a part [Taylor, Mason, and Goldberg , 1987].

Say we have a hyperstate describing the orientation of the part in the gripper. If we sense the gripper diameter, what is the resulting hyperstate?

If the sensor is perfect, we can determine the diameter for each state using the diameter function. Measuring value x, we could find the set of possible states using the inverse of the diameter function. Note that the diameter will not uniquely determine the state of the system, since the inverse of the diameter function is not single-valued.

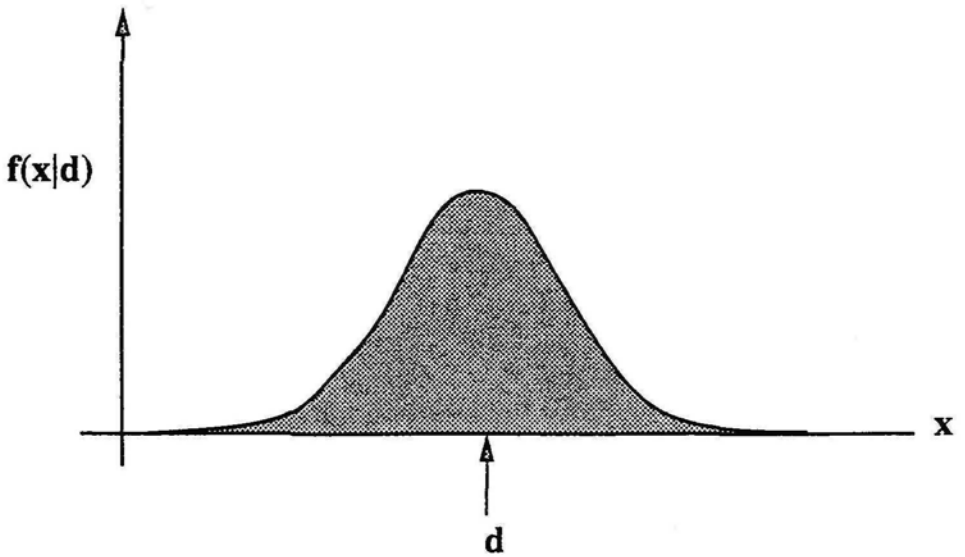


Figure 7.2: Probability density of sensor data x given jaw diameter d.

It is more realistic to treat the sensor as imperfect. That is, for a given state of the part, the sensor may return a *range* of possible values, where the exact value depends on jaw pressure, backlash, and the elasticity of the jaw surfaces. Let us model sensor noise with a probability distribution that depends on the true state of the system. To apply equation 7.1, we need to know $P(x|\theta)$ for each combination of data and state. That is, if the true diameter of the part in the gripper is d, then the sensor data might have a probability density as shown in figure 7.2

When planning, we consider a finite set of sensor outcomes. Say the sensor returns b bits of useful data, giving 2^b possible sensor outcomes. If the true diameter is d, we can assign probability to each outcome as shown in figure 7.3. Each stable orientation of the part in the gripper corresponds to a diameter $d(\theta)$. For each such diameter and each value of x, we can compute $P(x|\theta)$ using a Gaussian noise model to yield a set of matrices $\{\mathbf{P}_{\mathbf{x}_1}, \mathbf{P}_{\mathbf{x}_2}, \mathbf{P}_{\mathbf{x}_3}, \mathbf{P}_{\mathbf{x}_4}\}$. We could also determine the matrices empirically by sampling sensor data as we repeatedly grasp the object.

Consider a one-step iterative plan for orienting the part that consists of grasping the part with the jaws, consulting the diameter sensor, and, based on the sensor data, aligning the part

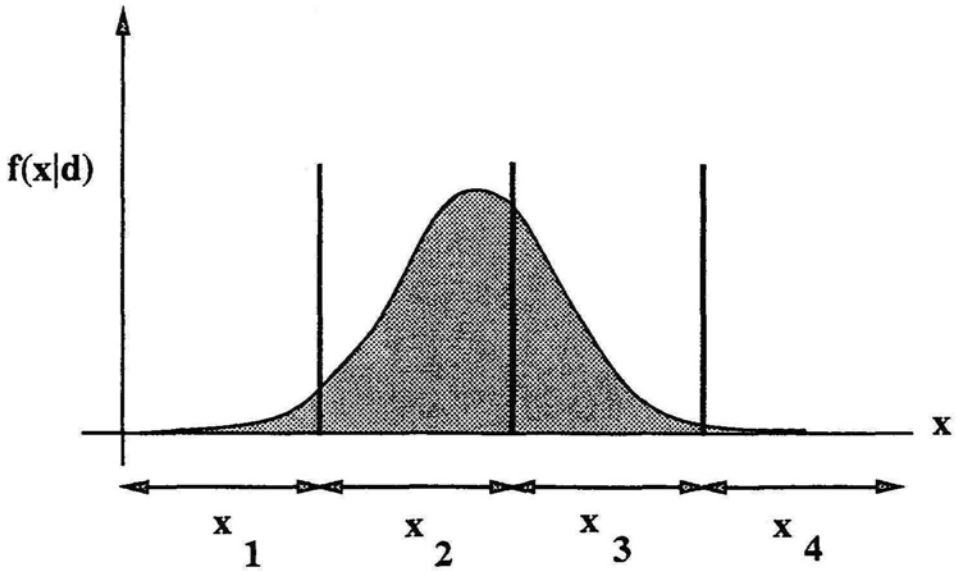


Figure 7.3: A 2-bit sensor returns one of 4 outcomes: x_1, x_2, x_3, x_4. If the true diameter is d, the probability of each sensor outcome can be determined by integrating the probability density over the associated range of sensor inputs.

over a filter that accepts one orientation and rejects all others. Let λ_0 be a prior probability distribution the set of stable part orientations when the jaws are closed around an object. We now consult the diameter sensor. Say it returns x_i. Then the posterior probability distribution is $\lambda_i = \lambda_0 \mathbf{P}_{x_i}$. As in section 4.4.6, we can identify a most-likely orientation in this hyperstate as

$$\theta^*(\lambda_i) = \arg\max_{\Theta} P(\theta), \tag{7.4}$$

where $P(\theta)$ is the probability of θ in hyperstate λ_i. After consulting the sensor, the gripper rotates to align the part over the filter so that only parts in orientation $\theta^*(\lambda_i)$ are accepted. Note: due to sensor noise, there is still some probability that the part is not in orientation $\theta^*(\lambda_i)$.

Let p_i be the probability that the part is in orientation $\theta^*(\lambda_i)$. If the sensor returns value x_i, it will take $1/p_i$ iterations until state $\theta^*(\lambda_i)$ is achieved. As in chapter 4, we relate the cost of actions to time. Say the time required to grasp, consult the sensor, and orient the part based on the sensor value is one time unit. Then the expected cost for orienting the part is $1/p_i$ units, assuming that the sensor returns value x_i.

At plan time, we don't know what value the sensor will return. But we can use the marginal probability distribution defined in equation 7.2 to determine the probability of each sensor

outcome. The expected cost for the one-step sensor plan is then

$$\sum_{i=1}^{2^b} \frac{m(x_i)}{p_i}. \tag{7.5}$$

Rather than submitting the part to the filter immediately after sensing, we could perform a sequence of open-loop grasping actions that depends on the sensor data to increase the probability of some state before testing part orientation with the filter.

7.2.4 Plans with Sensors and Actions

We can use both sensors and open-loop actions in one plan. We showed that sensor data can be combined with a prior hyperstate to yield a posterior hyperstate. We can treat the posterior hyperstate from each sensor value as the initial hyperstate for a new plan. Each of these plans becomes a subplan in a plan that consults the sensor and then executes the subplan corresponding to the measured sensor value. The expected cost for the larger plan can be determined by computing the expected cost for each subplan and weighting the expected costs by the marginal probabilities associated with each sensor value.

Alternatively, we can consider a plan that executes a sequence of open-loop actions and then consults a sensor. The expected cost of such a plan can be computed as if the hyperstate resulting from the sequence of actions is the initial hyperstate in a one-step sensor plan. The expected cost for any sequences of sensing and actions can be computed recursively.

When planning with only open-loop actions, we prefer actions that cause the probability distribution to converge. When sensors are available, we may prefer actions that cause the distribution to diverge so that we can improve the ability of a subsequent sensing operation to discriminate between possibilities.

7.2.5 Planning with Sensors

When planning, we want to compare plans. As in the previous subsection, consider plan A that senses first and then that executes a sequence of open-loop actions. Now consider plan B that executes a sequence of open-loop actions and then senses. Which is better? The real-valued metric of expected cost can be used to compare such plans.

In subsection 7.2.2, we noted that a sensor plan can be viewed as a tree with branches corresponding to each sensor value. Each such tree can be viewed as a subtree in a larger tree that represents all possible plans (figure 7.4). There are two types of branches in the large tree: branches corresponding to actions and branches corresponding to sensing. Recall that sensor branches come in bundles, since each branch corresponds to a sensor value; if we use a sensor,

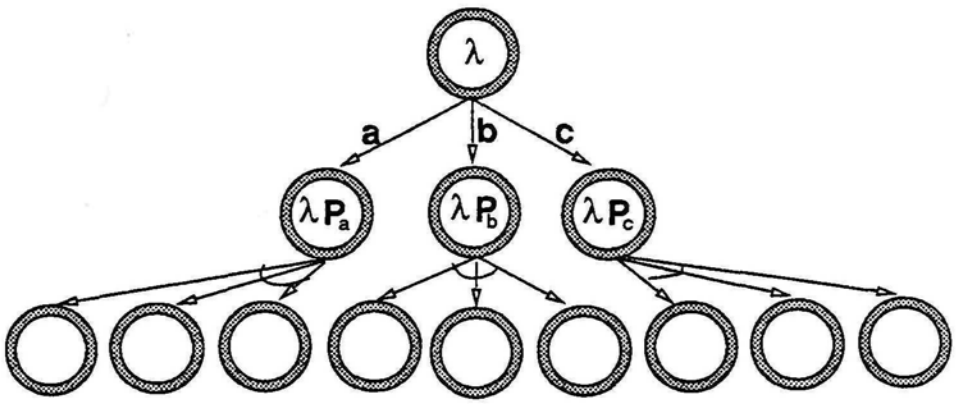


Figure 7.4: A tree that represents all possible plans.

then we must identify a subplan for each branch in the bundle. We can differentiate between branch types by putting an arc across a bundle of sensor branches to indicate that all branches in the bundle are part of one plan. This is the convention used in the well-known class of AND/OR trees [Nilsson, 1980].

To plan, we expand the tree from the root node corresponding to the initial hyperstate. We can compute an expected cost for each plan by propagating expected costs upward from leaf nodes. Branch-and-bound techniques may be helpful in searching the plan tree for an optimal plan. Note that all subplans need not be of the same length. We can truncate a subplan when we find a hyperstate with acceptable cost.

7.2.6 Related Work

A tree-based model for planning with sensors and actions was proposed by Taylor, Mason, and Goldberg (1987), who noted that plans using sensor data can be viewed as game-playing strategies, where sensor data is chosen by an adversary. In that paper, each node in the tree corresponded to a set of possible states. Without a probability measure, the objective is to find *guaranteed* plans, that is, plans that achieve a desired final state for all combinations of sensor data. Finding guaranteed plans is related to the class of chess problems where one is asked to find a sequence of moves that will guarantee checkmate in k moves for White, regardless of what Black does. In this framework plans are evaluated with a binary metric: either they are guaranteed to succeed or they are not.

The binary metric suffers from the limitations described in section 1.1.5: there is no way to compare plans when a guaranteed plan does not exist or when more than one guaranteed plan exists. Insisting on guaranteed strategies can be overly conservative and inefficient since one must consider all possibilities, however unlikely. Guaranteed plans are related to *minimax*

strategies in game theory in that both consider the worst-case. For iterative plans where failures can be recycled, it might be more efficient over the long run to plan for the *average* case. In game theory such situations are known as stochastic games [Luce and Raiffa, 1957].

Plans that use only sensors can be viewed as *sequential decision procedures* [Wald, 1947, Berger, 1985]. The problem of finding an optimal sequential decision procedure is called *experimental design*, where the cost of an experiment is related to how much state uncertainty it can eliminate [Fisher, 1942, Lindley, 1956, Good, 1969, DeGroot, 1970, Bernardo, 1979].

7.2.7 Other Applications

Taylor, Mason, and Goldberg [1987] proposed several applications of their approach to manipulation planning. These applications are also relevant when that approach is extended with cost and probability models.

We can use the stochastic planning framework to treat the problem of finding an optimal strategy for recognizing parts. In this case the state space is augmented with a dimension specifying the identity of the part so that we can recognize the part by reducing uncertainty in the state space. This is essentially Donald's idea for treating model error with a *generalized configuration space* [Donald, 1987].

So far, we have considered only one type of sensor. It is possible to plan with multiple sensors where each sensor has an associated data space and conditional probability distribution. If we consult two different sensors in sequence, we can compute the resulting hyperstate conditioned on both sensor readings. In this way we can combine information from multiple sensors (*sensor fusion*). We can also compare sensors and combinations of sensors and actions using the expected cost metric.

We can treat the problem of *sensor design* in this framework by planning with a set of proposed sensor designs. We implement those sensors required by the stochastically optimal plan. For example, consider the problem of finding an optimal sensor placement. Such a problem can be approached by treating each position of the sensor as if an open-loop action had occurred to locate the sensor. The action in a stochastically optimal plan will specify the sensor position.

We can also treat the problem of *sensor calibration* by including sensor parameters in the state space. A calibration plan is a sequence of actions and sensor probes, where each action and sensor datum changes a probability distribution on the parameter values. Plans that minimize uncertainty in the final hyperstate can be used to determine sensor parameters. For such problems the cost function should measure the uncertainty in the final hyperstate.

7.3 Future Work: Other Cost Functions

One way to measure the uncertainty in a probability distribution is with variance, $Var(\lambda) = \sum P(\theta)(\theta - \bar{\theta})^2$, where $\bar{\theta}$ is the mean of the distribution. For discrete distributions we can also use an "information" measure such as entropy.

Entropy and Expected Information

When the set of possible states at each stage of the plan is described by a discrete probability distribution, we can use *Shannon entropy* as a scalar measure of the uncertainty associated with the set. [Sanderson, 1984] used Shannon-entropy to measure uncertainty in robotic assembly, charting the change in entropy at each assembly stage.

The definition of Shannon-entropy comes from a logarithmic measure of information suggested by R. V. Hartley in 1928: Let E be an event that occurs with probability $P(E)$. If we are told that event E has occurred, then we can say that we have received $-\log P(E)$ units of information. If the logarithm is base 2, then the units of information are *bits*. [Shannon, 1948] defined the entropy of a discrete probability distribution as $H(\lambda) = -\sum P_i \log P_i$. For a hyperstate with uniform probability distribution, $H = \log n$, where n is the number of elements in the state space. For a hyperstate where all the probability is concentrated at one state, $P_i = 1$ for some i, then $H = 0$. ($0 \log 0$ is defined to be zero.)

We can use Shannon-entropy as a cost function for hyperstates when planning. Say $H(\lambda_0)$ is the entropy of the initial hyperstate, and $H(\lambda_1)$ is the entropy of the hyperstate resulting from an open-loop plan. We can use the change in entropy, $H(\lambda_0) - H(\lambda_1)$, to measure how much uncertainty is reduced by the plan. For a plan with sensors, we can compute the *expected entropy* for the set of final states using the marginal probabilities associated with each sensor value, and compute the average change in entropy that we expect from the plan. When entropy is measured in bits, we can measure the effectiveness of a plan in terms of how many bits of information it will produce. Shannon's First Theorem can be used to show that the average information yielded by any sensor plan is non-negative.

Entropy is not a Panacea

Relating the cost of a hyperstate to its entropy is not always appropriate. The definition of information adopted by coding theory discards any conception of the "quality" of the information, saying that the information gained from event E is simply proportional to $-\log P(E)$. This can produce some interesting conclusions. Consider a state space with two elements, such that the first corresponds to a desirable outcome and the second corresponds to an undesirable outcome. Now consider two probability distributions on this state space, $\lambda_1 = (.5 \quad .5)$ and

$\lambda_2 = (.01 \quad .99)$. Clearly λ_1 is to be preferred over a λ_2. If we use entropy as a cost measure, however, λ_2 will have much lower cost and will be preferred over λ_1. The hyperstate λ_2 is indeed more certain. Unfortunately, it is more certainly bad.

7.4 Conclusions

Since uncertainty is unavoidable in physical applications, we must anticipate multiple outcomes when planning. In this thesis we extend the geometric theory of manipulation with a stochastic framework that uses probability and cost models to rank plans on the basis of expected performance.

The bulk of the thesis is devoted to developing examples of stochastic planning. The examples focus on the problem of orienting planar parts with a programmable parts feeder. We propose a new design for such a feeder and use part geometry to derive cost and probability models. We also apply the stochastic framework to planning for a tray-tilting system where the stochastic transition matrices are determined empirically.

- We find a lower bound on the probability that a *random* grasp will be stable.
- Noting that this lower bound approaches one as the coefficient of friction goes to zero, we discover that a *frictionless* parallel-jaw gripper will achieve a stable orientation with probability one. We propose a practical mechanism to implement such a gripper.
- We give an algorithm for finding stochastically optimal parts-feeding plans and prove that this algorithm is correct, complete and runs in time $O(n^2)$, where n is the number of part vertices.
- We demonstrate how the stochastic model of control can be generated empirically by observing a physical *tray-tilting* system.

The stochastic framework proposed in this thesis uses the theory of Markov chains to provide a mathematical foundation for planning in the presence of uncertainty. The binary-valued metric used to evaluate guaranteed plans can be viewed as a special case of the real-valued metric used in the stochastic framework. We have shown how probability and cost models can be justified with system geometry or empirical observations. On this basis we conclude that the stochastic framework proposed in this thesis is well-founded, general, and applicable to robotic manipulation.

Appendix A

Overview of Statistical Decision Theory

Statistical decision theory can be a useful formalism for planning if we think of a plan as a sequence of decisions. In chapters 1 and 2 we discussed the relation between decision theory and manipulation planning. In this appendix we review the basic components of decision theory. There are also several excellent textbooks on decision theory [DeGroot, 1970, Berger, 1985]. Below we use Berger's notation.

Let $\theta \in \Theta$ be an unknown state of nature, $x \in X$. Decisions are based on noisy measurements (data) described by $f(x|\theta)$, the probability of measuring data x given state θ. Let $a \in A$ be an action. A *loss function*, L, is a real-valued function on $\Theta \times A$. A *decision rule*, $\delta \in \Delta$, is mapping from data to actions – a function from X to A. We define *risk*, R, as a real-valued function on $\Theta \times \Delta$:

$$R(\theta, \delta) = \int_X L(\theta, \delta)\, f(x|\theta)\, dx. \tag{A.1}$$

Given $(\Theta, X, f, A, L, \Delta)$, we want to find the optimal decision rule: one that minimizes R. The difficulty is that R is a *function* over the state space. It is not obvious how to define a maximum. There are three basic approaches.

The *Classical* approach is to select a "best" decision rule – one that will have a minimum risk for *all* values of θ. But often a "best" rule does not exist: a rule may minimizes R for some values of θ while another rule minimizes R for other values.

The *Minimax* approach is to find a rule that will minimize the worst-case risk, $\sup_\theta R$. This approach leads to conservative rules that may be far from optimal on average. Note that both the Classical and Minimax approaches are independent of the distribution of inputs.

The *Bayesian* approach is to find the rule that minimizes the *expected* risk using some prior distribution of inputs, $\pi(\theta)$. We can define the *Bayes'* risk as

$$r(\pi, \delta) = \int_\Theta R(\theta, \delta)\, \pi(\theta)\, d\theta \tag{A.2}$$

$$= \int_\Theta \int_X L(\theta, \delta)\, f(x|\theta)\, \pi(\theta)\, d\theta dx, \tag{A.3}$$

Note that for any δ, r is a scalar and so a maximum over all δ is well-defined. The resulting rule, δ^*, is called a *Bayes'* rule.

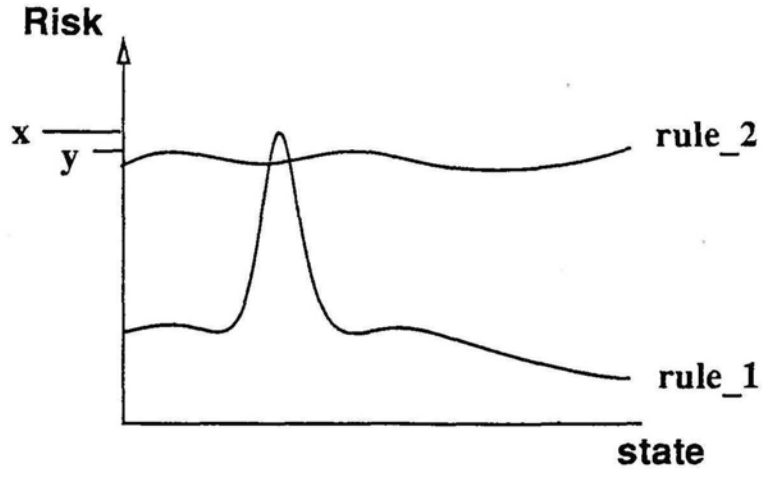


Figure A.1: The risk function for two decision rules, rule 1 and rule 2. We want to minimize risk. Rule 1 is superior (has lower risk) for the majority of states, however the Minimax approach would select rule 2 since it's worst-case risk (y) is lower than the worst-case risk for rule 1 (x). The Bayesian approach would use a probability distribution on the state space to compute a scalar expected risk for each rule and choose the rule with lower *expected* risk. Unless the probability were highly concentrated around its peak, rule 1 would be the better choice.

In the Bayesian approach it may be more appealing to think of the prior as being conditioned by the measurement. After measuring value x, let $\pi(\theta|x)$ be the resulting prior. Define the *Bayesian expected loss* to be real-valued function on $X \times A$:

$$\rho(\pi(\theta|x), a) = \int_{\Theta} L(\theta, a)\, \pi(\theta|x)\, d\theta. \tag{A.4}$$

A fundamental result from decision theory is that any rule that minimizes ρ for each x also minimizes r where the expectation is taken over $\pi(\theta|x)$ – either criteria produces a Bayes' rule. For some intuition behind this equivalence, compare Eq. (A.3) with Eq. (A.4). For a given decision rule, δ, Eq. (A.3) defines the risk over the joint distribution of measurements and states. Similarly, Eq. (A.4) defines the risk over the conditional distribution of states for each value of x. If we think of the latter as a function to be minimized for each x then it is less surprising that the two formulations are equivalent.

The Bayesian approach is sometimes criticized on the grounds that its models of the prior distribution are unrealistic. Bayesians like to point out that although the Minimax method avoids making assumptions about probabilities, it is in fact consistent with assuming that the worst-case is most probable. Thus minimax plans can be overly conservative and inefficient for the average case. Another response is to show that Bayesian strategies are robust to the choice of prior distribution [Berger, 1985].

Appendix B

The Frictionless Gripper

In this Appendix we describe an invention[1] that can improve the performance of the standard parallel-jaw gripper. The invention uses a passive linear bearing to reduce the effective frictional force between the gripper's jaws.

B.1 Background

Grippers are used to grasp and transport objects in robotics, teleoperation (master-slave control), and prosthetics (artificial limbs). See Kato, Chen [1982, 1982] for a review of existing designs. The parallel-jaw gripper (Figure B.1) is perhaps the most common, known as "the workhorse of the mechanical end-effectors" [Grupen *et al.*, 1989]. Parallel-jaw grippers are "universal" in that they can be used to pick up almost any part and hence are standard equipment on many commercial robot systems.

Tella *et al.* [1982] reports that a major disadvantage with the parallel-jaw gripper is that "it requires rotational alignment of the jaw with the opposing surface holdsite of a desired workpiece for reliable acquisition." For example, consider the two grasps shown in Figure B.2. On the left, the gripper has not been aligned with the gripping surfaces (holdsite) of the hexagonal workpiece; the resulting grasp is intuitively less stable (reliable) than the grasp on the right.

In what follows we will consider the two-dimensional projection of parts and refer to forces in the plane of the part. In reality of course, parts will be three-dimensional. The two-dimensional projection is particularly appropriate for the class of *extruded* parts.

---

[1]Carnegie Mellon University applied for a patent on this device in July 1989, listing the author and Professor Merrick Furst of the School of Computer Science as co-inventors.

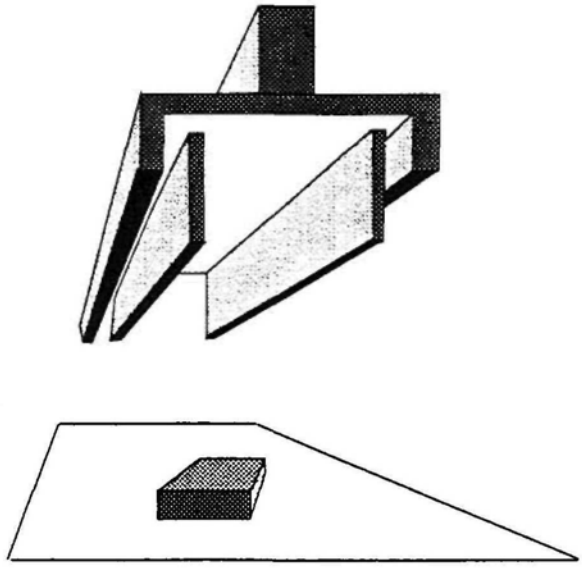


Figure B.1: The standard parallel-jaw gripper.

Several stability measures for grasping have been proposed. Hanafusa and Asada [1977] defined a *multi-fingered* grasp configuration to be *stable* when small deviations produce restoring forces. They developed a potential function for a gripper with three spring-loaded frictionless fingers and showed that local minima in this function correspond to stable grasps. Similarly, [Brost, 1988] defined a *parallel-jaw* grasp to be stable when at least one face of the (polyhedral) part is parallel to the jaws. This definition can be extended to smooth parts using local minima in a potential based on the diameter function as described in Section D.1. We will adopt this definition of stability although it is not the only definition.

One way to achieve a stable grasp is to *sense* the part's orientation and align with jaws prior to grasping. Analyzing the probability that a random grasp is stable led us to observe in Section

Figure B.2: Two-dimensional projection of a parallel-jaw gripper and a hex nut. The grasp on the right is stable in the sense that small perturbations in orientation produce restoring forces.

3.11 that a simple alternative is to eliminate friction in the plane of the part so that the part will passively rotate into a stable configuration as the jaws are closed. We require high friction between the part and the gripping jaws to counter the effect of gravity as the part is lifted and carried. We therefore desire low friction along one direction and high friction along a second direction.

B.2 The Invention

We can achieve low friction in the plane of the part but high friction orthogonal to the part by mounting a sliding plate (linear bearing) on one jaw. The inner surface of both jaws is covered with a high-friction material such as rubber. Since zero friction is impossible in a physical mechanism, we refer to the resulting device as a *low friction gripper*, see figure B.3. The low friction gripper works as shown in figure B.4.

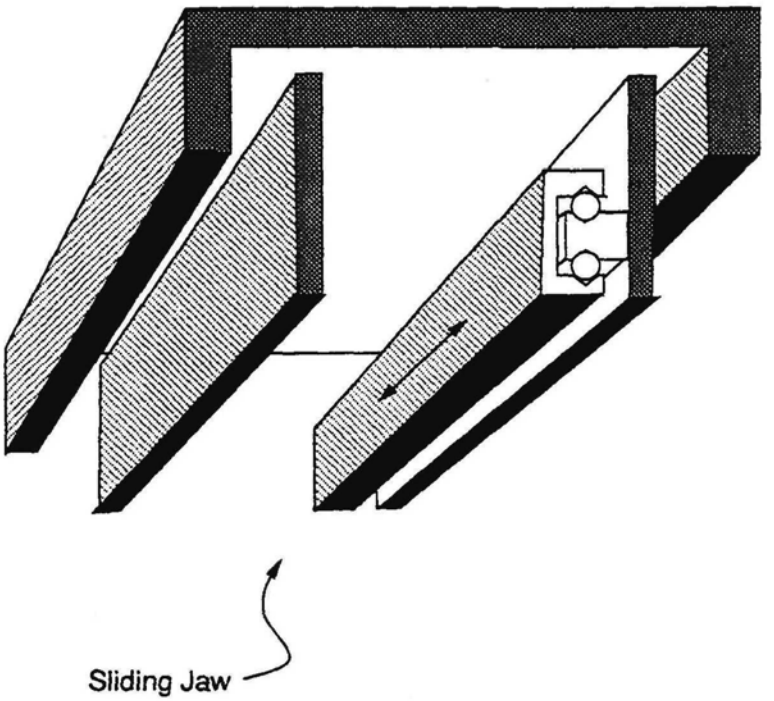


Figure B.3: The low-friction gripper. (Based on a drawing by Ben Brown).

When grasping, the jaws are closed with increasing force until the motor stalls (or maximum air-pressure is applied in the case of pneumatic actuation). The linear bearing operates passively,

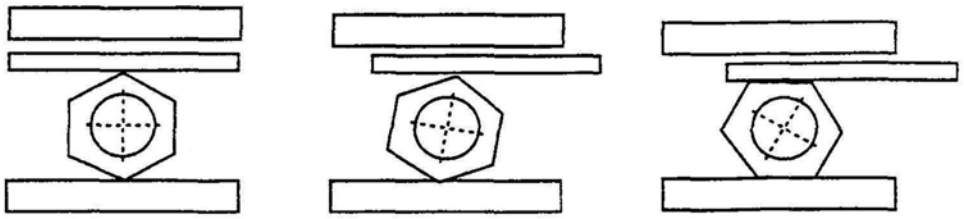


Figure B.4: Time-sequence of grasping with a low-friction gripper. (1) As the two outer jaws close over a typical part (hex nut), horizontal forces cause the sliding jaw to translate to the left (2) until the part is gripped in a stable configuration (3). (Based on a drawing by Ben Brown).

responding to mechanical forces that arise during grasping. This is an example of (passive) compliance [Mason, 1978].

We can add a spring to center the bearing between grasps without impeding the action of the gripper, since forces arising from the spring will be dominated by the gripping force as the jaws are closed.

We require a bearing on only one jaw. A single bearing is sufficient to reduce friction between the jaws so that the part will tend to rotate into a stable configuration. One fixed jaw insures that the part will not continue to translate after the part reaches a stable configuration. Note that the proposed gripper does not constrain lateral motion of circular parts.

We have built a prototype low friction gripper and verified its performance in experiments with several shapes. We grasped parts at random orientations in two sets of 250 trials. Without the sliding jaw, half the grasps were stable (126/250). With the sliding jaw, every grasp was stable. Although the bearing does exhibit some friction, we conjecture that vibration arising from the gripper drive is sufficient to dislodge unstable orientations, so the mechanism is effectively a frictionless gripper.

A frictionless gripper can simplify the problem of orienting industrial parts, since the workpiece is known to be in one of a finite number of stable orientations after gripping. A parts-orienting algorithm using the low friction gripper is described in Chapters 4 and 5.

Appendix C

Towards a Tighter Lower Bound on $P(S)$

In Chapter 3 we derived a lower bound on $P(S)$, the probability that a random grasp will be stable. The lower bound may be overly conservative; however a tighter bound requires a stochastic model of the pushing phase, which depends on the microscopic contact topology and the distance pushed. Peshkin [1986] found bounds on the set of all possible rotation rates. Simulation results led him to speculate that the rotation rate might be described by a unimodal probability distribution.

Uncertainty in the details of the load pressure is reminiscent of the uncertainty in the pressure distribution due to molecules in a gas, where aggregate behavior is successfully predicted using statistical mechanics. Probabilistic models of microscopic surface topology have been developed to justify the linear relationship between frictional force and the normal load [Onions and Archard, 1973], but so far no one has developed a probabilistic model to predict the motion of a pushed part.

For each initial orientation there is a set of possible orientations that can be produced by pushing. Since we have no basis for preferring one orientation over another, we might assume that every orientation in this set is equally likely. This is a very strong assumption and we use it only to illustrate how a probabilistic model of the pushing phase might be used to derive a tighter bound on $P(S)$.

Recall from Chapter 3 that θ_2 is the part orientation at the onset of squeezing – when the second jaw makes contact. If we can derive a probability density for θ_2 we can bound $P(S)$ with

$$P(S) \geq \int_{\Pi(\Theta_*)} f_{\theta_2}(\theta_2)\, d\theta. \tag{C.1}$$

Note that this is a lower bound since the integral is taken over the strong preimage; orientations in the weak preimage may also become stable as the grasp proceeds.

In general, θ_2 will depend on θ_1, how far the jaw pushes, and the rotation rate, which in turn depends on the time-varying distribution of pressure under the part which depends on the

microscopic contact topology. Intuitively, pushing motion tends to cause stable part edges to align with the gripper so that the part's orientation after pushing, θ_2, will be more likely to be at a stable orientation than an unstable one. If we had a stochastic model of rotation rate and pushing distance, we could use Eq. (3.1) to determine f_{θ_2}.

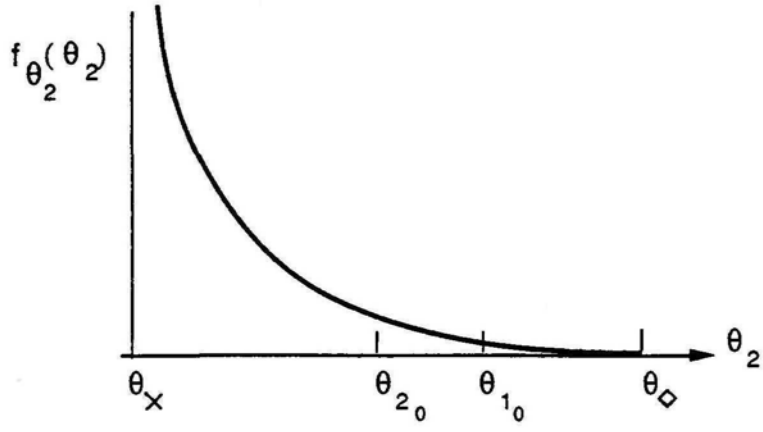


Figure C.1: The probability density function $f_{\theta_2}(\theta_2)$ over a subset of the push-stability diagram. Shown is a representative interval between a stable orientation $\theta_\times$ and an unstable equilibrium orientation $\theta_\diamond$. Direction of rotation is always to the left, toward the stable orientation.

An alternative approach is to bound the possible values of θ_2 and assume that the distribution of angles within this range is uniform. Assume for the moment that we know which jaw makes contact first. Consider a typical subset of the push-stability diagram as shown in Figure C.1. Let $\theta_\diamond$, $\theta_\times$ be the orientation of the surrounding $\diamond$ and $\times$ marks.

Assume that the direction of rotation is clockwise, or right-to-left on the push-stability diagram (we can reverse signs to handle the symmetric case). The amount of rotation that occurs during a particular trial is $\theta_1 - \theta_2 \bmod 2\pi$. Note that θ_2 can range from θ_1 (no rotation) to $\theta_\times$ (the part rotates all the way to the nearest stable edge). Without a good reason to prefer any value in this interval, we assume that the probability of any θ_2 is uniformly distributed on the interval: $f(\theta_2|\theta_1) = \frac{1}{\theta_1 - \theta_\times}$.

The joint distribution is simply $f(\theta_2, \theta_1) = f(\theta_2|\theta_1) f_{\theta_1}(\theta_1)$. We can compute $f_{\theta_2}(\theta_2)$, by integrating the joint distribution over all initial orientations that may end up at angle θ_2.

For example, consider θ_{2_0} shown in Figure C.1. Recall that motion is always in the direction from $\theta_\diamond$ to $\theta_\times$, so only angles between θ_{2_0} and $\theta_\diamond$ can possibly rotate into orientation θ_{2_0}. The density is

$$f_{\theta_2}(\theta_2) = \int_{\theta_2}^{\theta_\diamond} f(\theta_2, \theta_1)\, d\theta_1 \qquad \text{(C.2)}$$

$$= \int_{\theta_2}^{\theta_\diamond} f(\theta_2|\theta_1)\, f_{\theta_1}(\theta_1)\, d\theta_1 \tag{C.3}$$

If we assume a uniform distribution for θ_1 as in Section 3.9, this becomes

$$f_{\theta_2}(\theta_2) = \int_{\theta_2}^{\theta_\diamond} \frac{1}{\theta_1 - \theta_\times} \frac{1}{2\pi}\, d\theta_1.$$

Integrating,

$$f_{\theta_2}(\theta_2) = \frac{1}{2\pi} ln \left(\frac{\theta_\diamond - \theta_\times}{\theta_2 - \theta_\times} \right). \tag{C.4}$$

Under this model, $f_{\theta_2}(\theta_\diamond) = 0$ and $f_{\theta_2}(\theta_\times) \to \infty$. The pdf for θ_2 in this interval is thus a normalized logarithmic function sloping up toward the nearest $\times$. This implies that the part is likely to rotate toward a stable orientation during the pushing phase, which is consistent with our intuition. However, this model may tend to underestimate the pdf near a $\diamond$ mark.

Recall that for a given θ_2, the squeeze-stability diagram tells if the outcome will be stable or not. Let θ_a and θ_b be bounds on the strong pre-image in the squeeze-stability diagram for a particular stable outcome, θ_s. By integrating the expression in Equation (C.4) over this interval, we can find the probability that the final orientation will be θ_s.

$$P(\theta_s) \geq \int_{\Pi(\theta_s)} f_{\theta_2}(\theta_2)\, d\theta_2 \tag{C.5}$$

$$= \int_{\theta_a}^{\theta_b} f_{\theta_2}(\theta_2)\, d\theta_2. \tag{C.6}$$

Integrating,

$$P(\theta_s) \geq \frac{1}{2\pi} \left[\theta_{ba}(1 + \ln \theta_{\diamond\times}) + \theta_{a\times} \ln \theta_{a\times} - \theta_{b\times} \ln \theta_{b\times} \right], \tag{C.7}$$

where $\theta_{UV} = \theta_U - \theta_V$.

If we assume that either jaw is equally likely to make contact first, then by normalizing the pdf for each jaw so that it integrates to $1/2$, the summed probabilities from Equation (C.7) give a tighter lower bound for $P(S)$.

To compare this lower bound to the one derived in Chapter 3, consider the square shape where all the stable edges for pushing are also stable edges for squeezing: $\theta_a = \theta_\times$. The lower bound from Equation (3.13) is

$$P(\theta_s) \geq \frac{\theta_{b\times}}{2\pi}. \tag{C.8}$$

while Equation (C.7) gives

$$P(\theta_s) \geq \frac{\theta_{b\times}}{2\pi}\left(1 + \ln\frac{\theta_{\diamond\times}}{\theta_{b\times}}\right). \tag{C.9}$$

Since $\theta_{\diamond\times} \geq \theta_{b\times}$, the lower bound from Equation (C.9) is tighter than the lower bound from Equation (C.8). The difference arises from the assumption that favorable rotation is likely to occur during the pushing phase. The two lower bounds are identical only when $\theta_b = \theta_\diamond$, that is, when friction is so high that almost every orientation gets wedged.

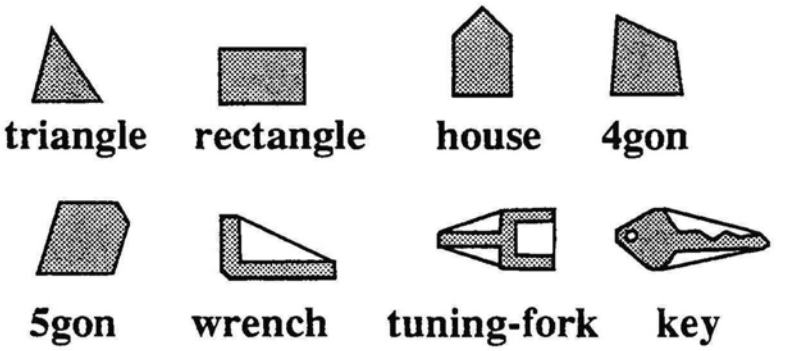
C.1 Examples

We implemented equation C.7 in Common Lisp to derive a lower bound on $P(S)$ based on the uniformity assumption. The input is a list of vertices representing a polygonal part, the part's center of mass, and the coefficient of friction. As before, part vertices are represented with rational numbers so that the angles used in the analysis can be represented exactly using rational complex numbers. The output is a tighter lower bound than the one found in the text. Compare Table C.1 with Table 3.1.

Observe that a stable grasp becomes more likely as friction is reduced. This is because as friction between the gripper and part is reduced it is increasingly difficult to achieve a wedged orientation.

In this appendix we derived a lower bound for $P(S)$ under the assumption that the amount of rotation during the pushing phase is uniformly distributed. That is, for each initial orientation there is a set of possible orientations that can be produced by pushing. With no basis for preferring one orientation over another, we assume that every orientation in this set is equally likely. This assumption leads to an exponential distribution for the part's orientation prior to squeezing that is mildly consistent with intuition.

Although this assumption serves to illustrate how the analysis can be extended to incorporate a stochastic model of pushing, it tends to underestimate the probability that part orientation will be near an unstable equilibrium. This is because the uniformity assumption does not take into account the fact that rotation rate is slower when the object is near an unstable equilibrium. In the absence of a better analytic model, an alternative is to *empirically* estimate the distribution based on physical experiments. In Chapter 6, we use experimental data to derive a probabilistic model of system behavior.



part shape	sides	$\mu = .25$	$\mu = .10$	$\mu = .05$
triangle	3	0.59	0.85	0.93
rectangle	3	0.94	0.99	0.99
house	5	0.72	0.90	0.95
4gon	4	0.59	0.75	0.87
5gon	5	0.85	0.95	0.97
wrench	6	0.66	0.82	0.91
tuning-fork	6	0.87	0.96	0.98
key	11	0.66	0.73	0.84

Table C.1: Lower bound for $P(S)$, varying shape and coefficient of friction. All parts have uniform mass density.

Appendix D

The Diameter Function

In Chapter 4 we showed that a diameter function could be used to analyze the mechanics of squeeze-grasp actions. In this appendix we discuss the diameter function in more detail and give an $O(n \log n)$ algorithm for computing the diameter function for an n-sided object.

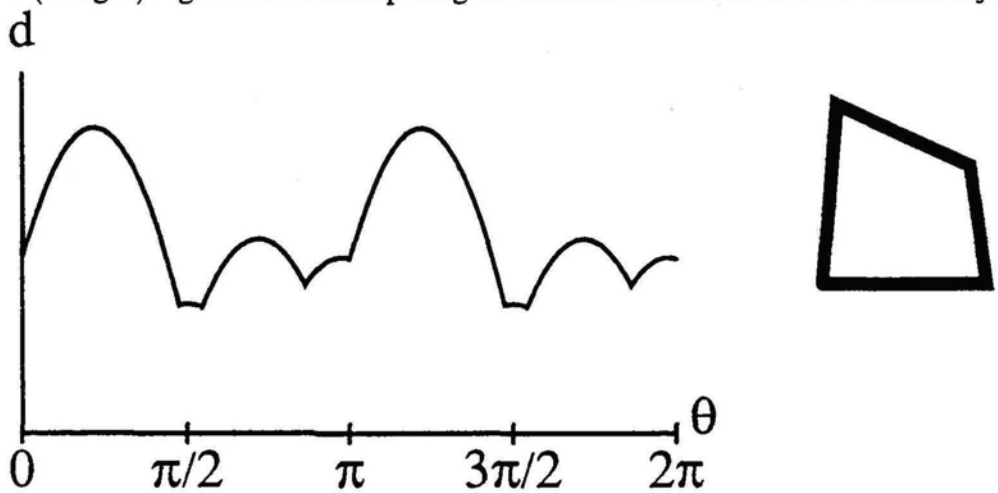


Figure D.1: The diameter function for the four-sided part shown at the right in its zero orientation. During a squeeze, the part rotates so as to reduce the diameter, terminating when the diameter reaches a local minimum.

Let a two-dimensional object be described with a continuous curve in the plane, C. The distance between two parallel tangents varies with the orientation of the lines. We define the diameter function, $d(\theta)$, to be the distance between parallel lines of support at angle θ. Jameson [1985] defined the same function and used it to show that any two-dimensional convex body must have at least two stable equilibria where it can be grasped between parallel jaws. The maximum of the diameter function is known as the *diameter* of the set of points contained in C

[Preparata and Shamos, 1985]. The maximum is useful for clustering point sets.

- The diameter function is continuous: $\Delta d \to 0$ as $\Delta\theta \to 0$.
- The diameter function for C is equal to the diameter function for the convex hull of C.
- The diameter function has period π.

Given a list of n vertices describing the polygon, the diameter function is defined by a list of sinusoidal functions (phase and amplitude) and the associated transition angles. Transitions between sinusoids can only occur when an edge is aligned with the gripper, so there are at most $2n$ sinusoidal pieces. Since each sinusoid arises from contact between two opposing vertices in the object, we can trivially compute the diameter function by enumerating all n^2 pairs of vertices. The running time of the trivial algorithm is thus $O(n^2)$. A faster algorithm exists.

Preparata and Shamos [1985] describe a linear-time algorithm for finding the maximum diameter of a convex polygon with n sides. It proceeds by enumerating the set of all pairs of vertices that admit parallel tangents. There are at most $3n/2$ such pairs. Each pair defines a *chord* of length l_i and angle θ_i. The longest chord gives the diameter of the polygon.

To find the diameter *function*, we sort this list of chords by increasing angle θ_i. If two chords have the same angle, discard the shorter chord. Also sort the list of polygon edges by angle and discard duplicates (corresponding to parallel edges). Every adjacent pair of edge angles in this sorted list corresponds to a sinusoid in the diameter function, $d_i(\theta) = l_i|\sin(\theta_i - \theta)|$, where l_i and θ_i are taken from the longest chord in the interval orthogonal to the interval between edges. Finding the longest chord in each interval requires a single sweep through the sorted list of chords.

Sorting dominates the running time, so we can compute the diameter function in time $O(n \log n)$. This is also the complexity of finding the convex hull for a set of two-dimensional points. A simple reduction from SET DISJOINTNESS can be used to show that this running time is optimal.

D.1 Diameter as a Potential Function

The diameter function can be viewed as a potential energy function in a conservative system. We define a potential function for a spring-loaded parallel-jaw gripper as follows, assuming zero friction between the part and the jaws to insure that the system is conservative. We assume that all motion and forces occur in the plane. Let d be the jaw separation ($d = 0$ when the jaws

are fully closed). Assume that the jaws tend to close due to the action of springs in compression, so that the closing force is $F(d) = kd$. Using the standard definition of potential energy,

$$U(d) = -\int_0^d F(d)dd \tag{D.1}$$

$$= -\frac{1}{2}kd^2. \tag{D.2}$$

Now consider d to be a function of part orientation, θ; so that

$$U(\theta) = -\frac{1}{2}kd(\theta)^2. \tag{D.3}$$

It is easy to show that a two-dimensional grasp has rotational stability if and only if it corresponds to a local minimum in $U(\cdot)$.

This definition of stability applies to curved or polyhedral parts. For a polygon, there can be up to n stable grasps, where n is the number of edges in the polygon.

D.2 A Cost Metric Related to The Diameter Function

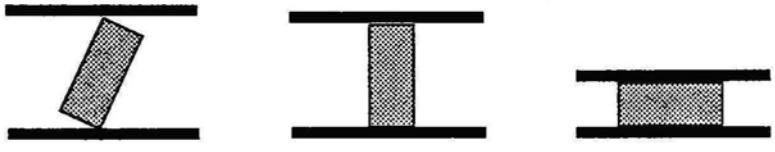


Figure D.2: Three parallel-jaw grasp configurations.

Consider the three grasp configurations shown in figure D.2. Which seems to be the most stable? Intuition suggests that stability increases from left to right.

Several metrics related to stability have been proposed for multi-fingered hands. We can rank grasps using the potential function. Let us define the stability of a grasp to be the smallest perturbation in angle before the part switches to a neighboring minimum [Whitney, 1990]. That is, the stability of a minimum is the distance to the nearest maximum. Using this definition we can rank the minima and find one with maximal stability. That is, for a given part geometry let the *stability measure* be

$$\Psi(\theta_s) = \min[|\theta_s - \theta_L(\theta_s)|, \ |\theta_s - \theta_R(\theta_s)|], \tag{D.4}$$

where $\theta_L(\theta_s), \theta_R(\theta_s)$ are the left- and right-hand maxima enclosing θ_s. For any part, a most stable grasp is one with maximal stability:

$$\theta_s^* = \arg\max_{\theta_s} \Psi(\theta_s). \tag{D.5}$$

Appendix E

Push-Grasping

In this appendix we consider a way to relax assumption 6 of section 4.3, where we assumed that both jaws make contact simultaneously. Recall that this assumption is not always satisfied in practice, since there may be a period of time when one jaw *pushes* the part before the second jaw makes contact. The part can rotate during this period so that its final orientation is not as predicted by the pure squeezing analysis.

We consider the class of actions where one jaw intentionally pushes the part prior to grasping. Brost [1988] refers to such actions as *push-grasp* actions. Given sufficient pushing distance [Peshkin, 1986], the part rotates so that one of its edges is aligned with the pushing jaw with probability one. We assume here that pushing distance is sufficient to align the part. The part's motion after the second jaw makes contact can be predicted using the squeeze function as before. By pushing with one jaw, we can eliminate the 180° ambiguity in the part's final orientation that is inherent in squeeze-grasping.

As with squeeze-grasping, the transfer function for push-grasping is a monotonically non-decreasing step function on the space of planar rotations. Since this is the only assumption required by the planning algorithm of chapter 5, we can use that algorithm to plan stochastically optimal push-grasping strategies.

To analyze the mechanics of push-grasping we need one additional piece of information, the part's center-of-mass. A result due to Mason says that the part's rotation is determined by a vector from the point where the jaw makes contact with the part to a point at the part's center-of-mass. The mechanics of pushing can be captured with an analog to the diameter function: the *radius function* [Mason, 1982].

E.1 The Transfer Function

E.1.1 The Radius Function

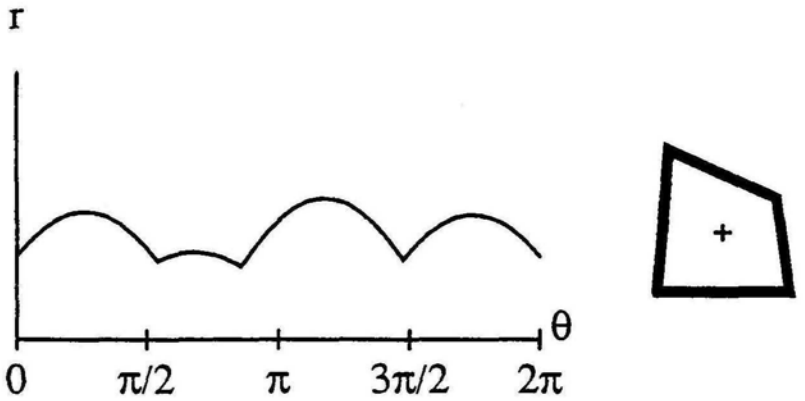


Figure E.1: The radius function for the four-sided part shown at the right in its zero orientation. During a squeeze, the part rotates so as to reduce the diameter, terminating when the diameter reaches a local minimum.

As with the diameter function, let a two-dimensional part be described with a continuous curve, C, in the plane. Also, define some point on the part as the part's *centroid.* The perpendicular distance from a tangent line to the centroid varies with the orientation of the line. We define the radius function, $r(\theta)$, to be the distance from a tangent line at angle θ to the centroid. See figure E.1. The radius function for an n-sided part can be computed in time $O(n)$.

E.1.2 The Push Function

The *push function* maps an initial orientation of the part to a final orientation after the pushing action. It is a step function derived from the radius function in the same way that the squeeze function is derived from the diameter function. That is, discontinuities between steps occur at local maxima in the radius function. The height of a step corresponds to the enclosed local minimum.

E.1.3 The Push-Grasp Function

To analyze the mechanics of pushing followed by grasping, we must compose the push function with the squeeze function to get a transfer function that we call the *push-grasp function.* It is illustrated in figure E.3.

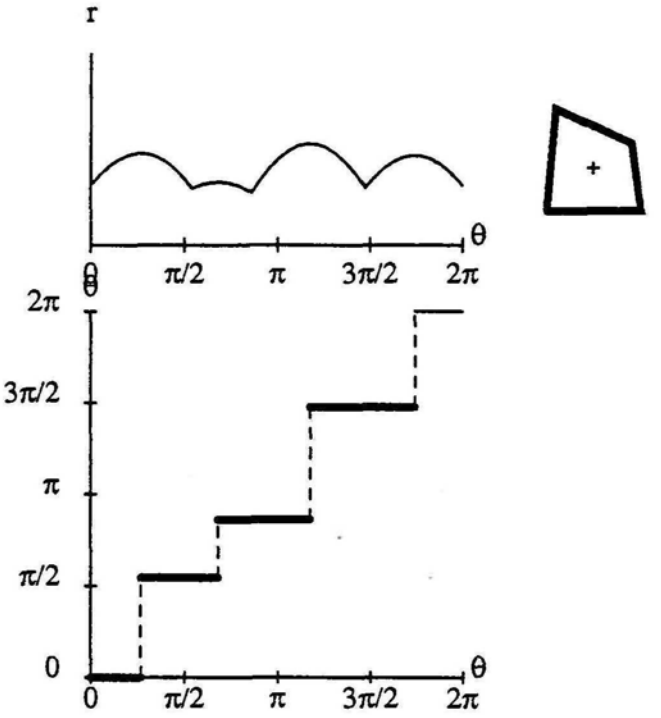


Figure E.2: The radius and push functions for the four-sided part.

E.2 Planning

The planning algorithm described in chapter 5 can generate plans for the class of push-grasp actions since the transfer function meets the assumptions of section 5.3. The preimages found by the algorithm are shown in Figure E.4.

Figure E.5 shows a plan for the 4-sided part with the evolution of the probability distribution for the part as it goes through the plan. (Note that we can orient the part uniquely with a sequence of push-grasp actions since the transfer function for push-grasping does not have period π.) If we assume that each action requires one time unit as does the filter step, the 1-step plan has minimal expected cost, 3.7 time units. Although the 4-step plan does not require a filter step to detect failure (since it orients the part with probability one), it still requires one time unit for transferring the part to the conveyor after the plan is completed.

Figure E.8 shows a plan for the house-shaped part with the evolution of the probability distribution for the house-shaped part as it goes through the plan. If we assume that each action requires one time unit as does the filter step, the 3-step plan has minimal expected cost of 4 time units.

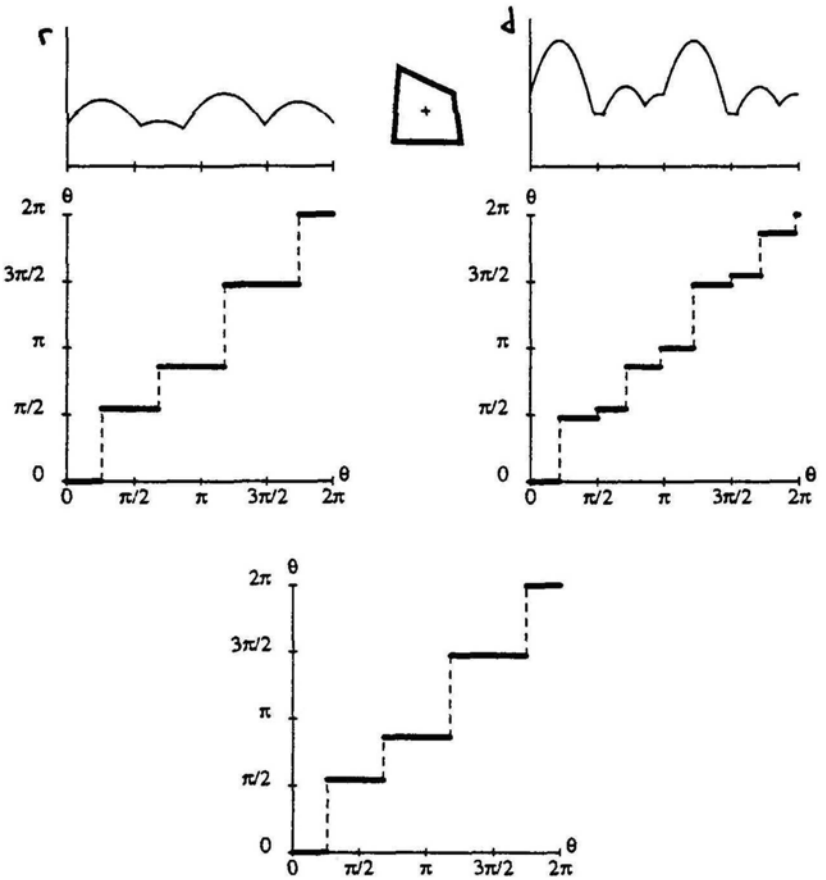


Figure E.3: Push-grasp analysis for the four-sided part shown in the center. In the upper left is the radius function. Directly below it is the push function. In the upper right is the diameter function. Directly below it is the squeeze function. The push-grasp function is shown at the bottom.

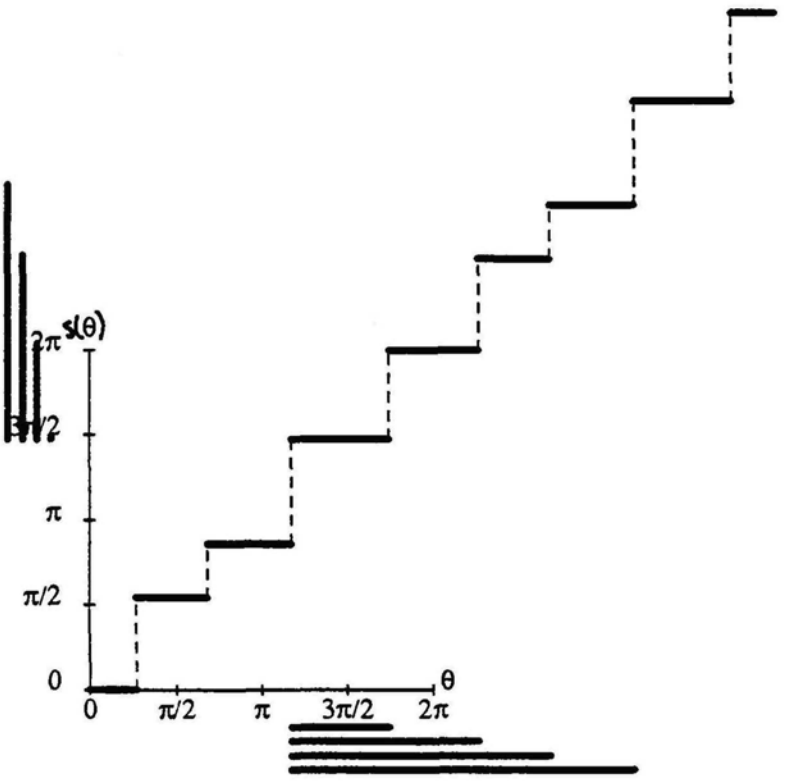
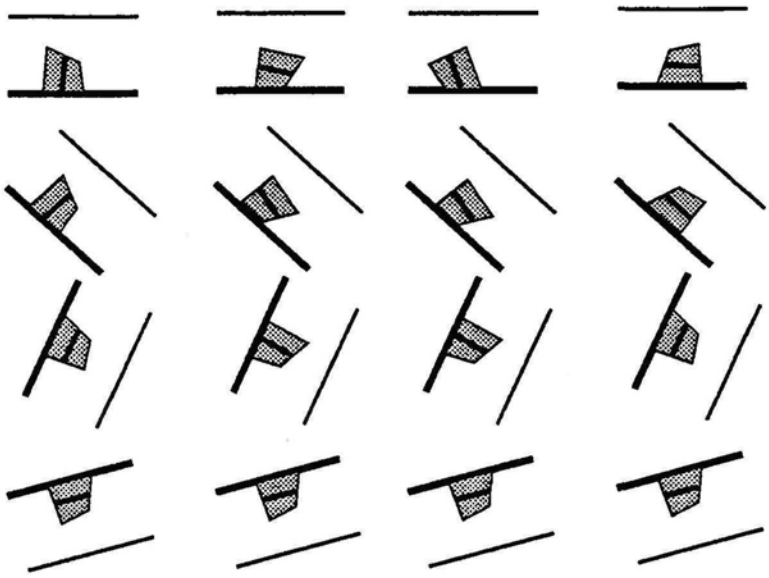


Figure E.4: Two periods of the push-grasp function (shown from 0 to 4π) for the 4-sided part. The planning algorithm finds preimages corresponding to the horizontal bars at the bottom.



Random Push-Grasp:	[.26 .21 .25 .28]
Push-Grasp at 25°:	[.26 .46 .00 .28]
Push-Grasp at 79°:	[.46 .00 .00 .54]
Push-Grasp at −23°:	[.00 .00 .00 1.0]

Figure E.5: Above: Four traces of push-grasp plan: push-grasp at 0°, 25° 79°, and −23°. Darkened line indicates pushing jaw. Below: Evolution of probability distribution.

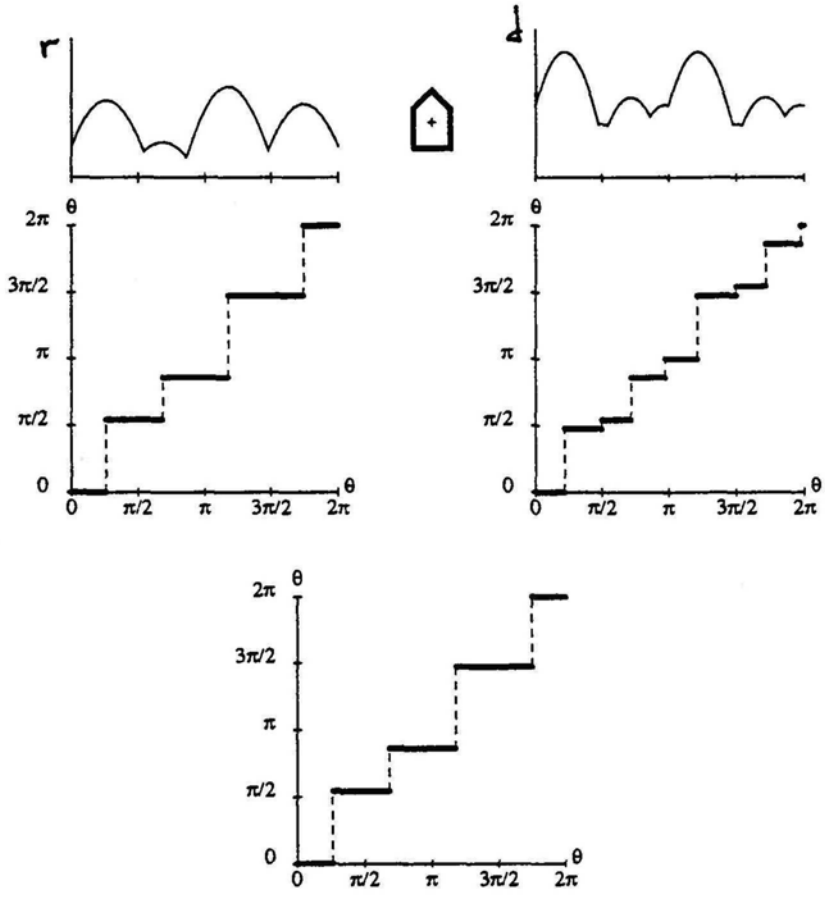


Figure E.6: Push-grasp analysis for the house-shaped part shown in the center. In the upper left is the radius function. Directly below it is the push function. In the upper right is the diameter function. Directly below it is the squeeze function. The push-grasp function is shown at the bottom.

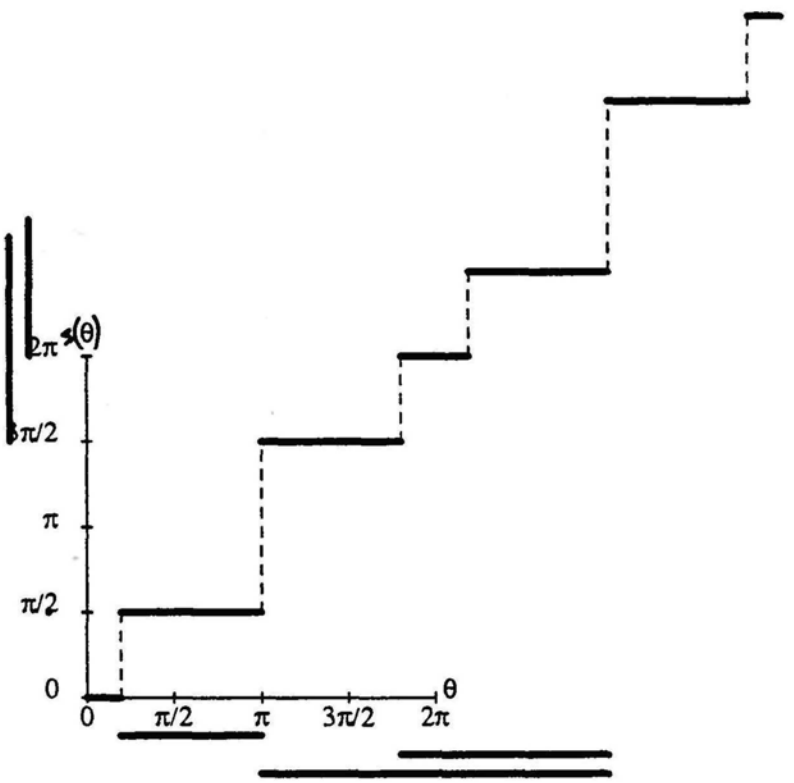
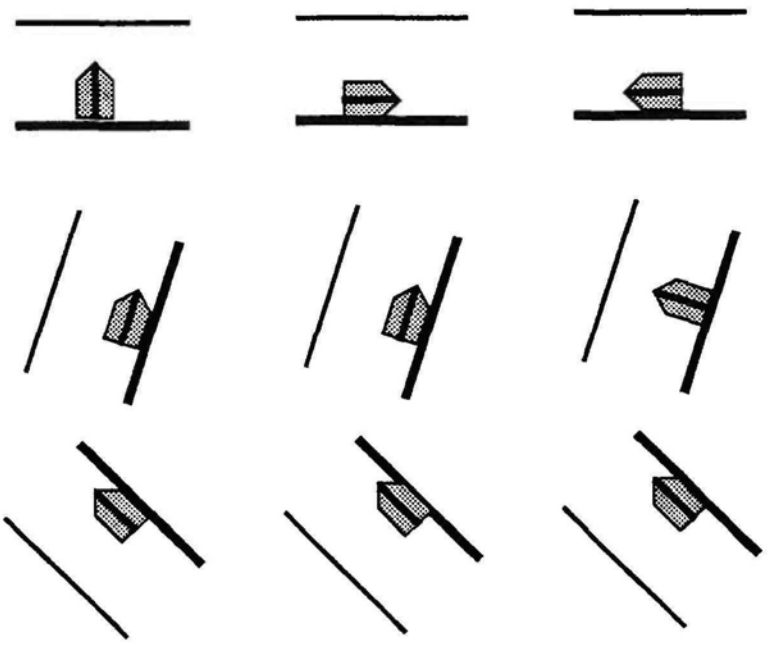


Figure E.7: Two periods of the push-grasp function (shown from 0 to 4π) for the house-shaped part. The planning algorithm finds preimages corresponding to the horizontal bars at the bottom.



Random Push-Grasp:	[.20 .40 .40]
Push-Grasp at 72°:	[.40 .60 .00]
Push-Grasp at 135°:	[.00 1.0 .00]

Figure E.8: Above: Three traces of push-grasp plan to orient the house-shaped part. Darkened line indicates pushing jaw. Below: Evolution of probability distribution.

Appendix F

Planning with Non-Uniform Priors

In Chapter 5 we showed that the backchaining planner generates stochastically optimal plans under the assumption that the prior distribution of part angles is uniform. In this appendix we show how the planner can be extended to non-uniform priors.

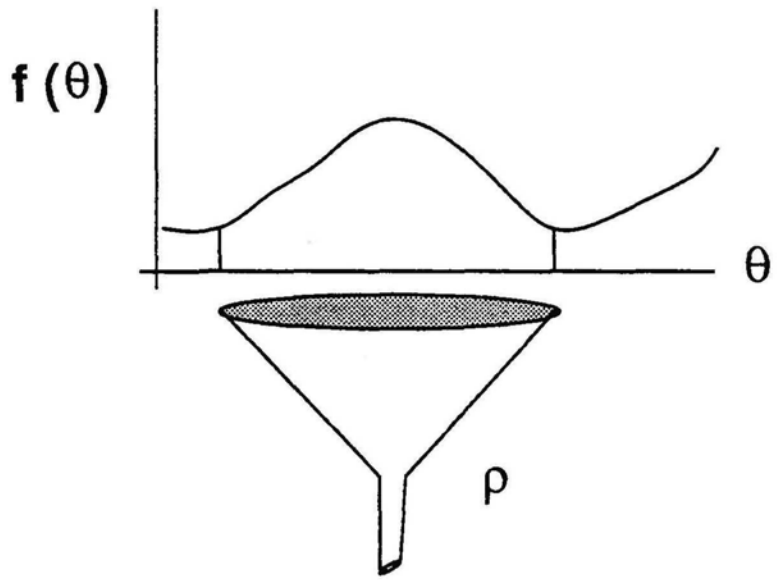


Figure F.1: The figure shows a prior probability density function. Plan ρ is represented with a funnel. The funnel is positioned under the prior density function so as to capture the maximal amount of probability mass.

Recall that each i-step plan generated by Phase I of the algorithm is geometrically optimal – it acts like a funnel with the largest possible "mouth". The position of the mouth specifies the

initial gripper angle for plan ρ. The included probability mass specifies the probability that the plan will succeed in one iteration. When the prior probability density is uniform, all positions for the mouth are equivalent. For cases where the prior probability density is non-uniform, we must position the mouth so as to capture as much probability mass as possible. See Figure F.1.

The optimal funnel position is related to the statistical concept of a $100(1-\alpha)\%$ *credible set* – a subset of the state space such that the true state is included with probability at least $1-\alpha$. Generally one is given α and asked to find the smallest credible set, this is known as the $100(1-\alpha)\%$ *highest posterior density (HPD) credible set.* Berger [1985] gives a procedure for finding the HPD credible set. In our application we want to find the optimal funnel position. This is equivalent finding a connected credible set of given size that minimizes α.

If the prior probability distribution is discrete over n points, then we can find an optimum position for the mouth in time $O(n)$ by aligning the mouth with each point in turn and computing the enclosed probability.

When the prior probability distribution is continuous, let $f(\theta)$ be the initial probability density function for the part on the set of angles S^1 (all values are modulo 2π). Assume $f(\theta)$ is differentiable. Let $F(\theta)$ be the cumulative distribution function,

$$F(\theta) = \int_0^\theta f(\phi)d\phi. \tag{F.1}$$

Let $h = \Theta$ be the width of the funnel mouth corresponding to plan ρ. To specify a position for the mouth, let θ be the position of the left-most edge of the mouth. Let $p(\theta)$ be the probability that the initial orientation is included in the mouth when the mouth is aligned with θ:

$$p(\theta) = F(\theta + h) - F(\theta). \tag{F.2}$$

That is, $p(\theta)$ is the probability that the plan succeeds when the mouth corresponding to the plan is aligned with θ. We want to maximize $p(\theta)$. Let p^* be the maximum probability

$$p^* = \max_\theta p(\theta). \tag{F.3}$$

Let θ^* be the position that attains this maximum. Technical note: since S^1 is compact, if $p(\theta)$ is continuous it will attain its maximum.

If there is a unique solution, we can find the optimal position for the funnel by differentiating F.2 and setting the result equal to zero,

$$\frac{dp}{d\theta} = f(\theta) - f(\theta + h) = 0, \tag{F.4}$$

to find the θ^* that satisfies the equation

$$f(\theta^*) = f(\theta^* + h). \tag{F.5}$$

That is we align the mouth of the funnel such that both edges have the same probability density. The intuition is that is that if one edge had higher density, then we could increase the included probability mass by shifting the mouth in the direction of that edge. Note that we must check that this is a local maximum by taking the second derivative.

When there is more than one solution to F.2, numerical methods for global optimization must be used, *i.e.* discretizing the set Θ and testing each possible position.

For non-uniform density functions, we must modify step 2 of Phase II to compute θ^* and p^* for each subplan. Given the prior probability density f, let $a(f)$ denote the time complexity for computing θ^* and p^* for a given h. The complexity of the planning algorithm is then $O(an + n^2)$. When the prior is uniform, then $a = 1$, since it doesn't matter where we position the funnel. When the prior is discrete over m points, then $a = m$. If the prior is a normal distribution, then $a(f) = 1$, since the best position is centered around the mean and the included probability is related to the variance.

We can prove that the modified algorithm finds stochastically optimal i-step plans by showing that Lemma 5.4 holds when the prior is not uniform.

Lemma F.1 *If an i-step plan is geometrically optimal then it is also stochastically optimal.*

Proof: Let $F(\cdot)$ be the prior cumulative probability distribution on the space of planar rotations. Let ρ_i be a geometrically optimal i-step plan with pre-image of length h. Let $p(\theta) = F(\theta + h) - F(\theta)$. We show that no other i-step plan can collapse a larger probability mass than ρ_i.

Let ρ_i' be any other i-step plan. Denote the length of its pre-image by h'. Let $p'(\theta) = F(\theta + h') - F(\theta)$. For all θ, $p'(\theta) \leq p(\theta)$ since $h' \leq h$ by assumption. Thus no matter where we align the funnel for plan ρ_i', it cannot have higher probability of success than plan ρ_i. Thus no i-step plan can have higher probability of success than ρ_i. ■

Appendix G

The Dirichlet Distribution

In Chapter 6 we used the Dirichlet distribution to model the prior distribution of elements in a transition matrix. In this appendix we give some intuition for choosing a prior Dirichlet by showing how a one-dimensional Dirichlet distribution (the Beta) varies with its parameters. Recall that a random vector $\mathbf{p} = (p_1, ..., p_k)$ has a Dirichlet distribution with parametric vector $\alpha = (\alpha_1, ..., \alpha_k)$, where $\alpha_i > 0$. For any point in Ω,

$$f(\mathbf{p}|\alpha) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)...\Gamma(\alpha_k)} p_1^{\alpha_1 - 1} ... p_k^{\alpha_k - 1}. \tag{G.1}$$

where $\alpha_0 = \sum \alpha_i$, and the Gamma function is the continuous counterpart to the factorial.

Note that the expected value along any dimension of the random vector is $E(p_i) = \alpha_i/\alpha_0$. Similarly, $Var(p_i) = \alpha_i(\alpha_0 - \alpha_i)/\alpha_0^2(\alpha_0 + 1)$ and $Cov(p_i, p_j) = -\alpha_i\alpha_j/\alpha_0^2(\alpha_0 + 1)$.

The Dirichlet is a continuous distribution over a $(k-1)$-dimensional simplex, so it can be hard to visualize. In this appendix we develop some intuition for the Dirichlet distribution by plotting the univariate Dirichlet distribution (the beta distribution) as we vary the parameters α_1, α_2.

Consider a binary state space with two states, x_1 and x_2. Let Prob$(x_i) = p_i$. We represent a probability distribution over the binary state space with the two-element vector $\mathbf{p}$. But since $p_2 = 1 - p_1$, we can plot the probability distribution for $\mathbf{p}$ as a function of one variable. In the following figures we plot $f(p_2)$, the probability density function for p_2 as we vary α_1, α_2.

If $\alpha_1 = \alpha_2 = 1.0$, then the distribution is uniform over the interval $(0, 1)$ (Figure G.1).

If $\alpha_1 = \alpha_2 = .1$, then the distribution is skewed toward both zero and one (Figure G.2).

If $\alpha_1 = \alpha_2 = 8.0$, then the distribution is skewed toward .5 and is similar to a Gaussian (Normal) distribution (Figure G.3).

If $\alpha_1 = 4.0$ while $\alpha_2 = 1.0$, then the distribution is skewed toward 0.0 (Figure G.4).

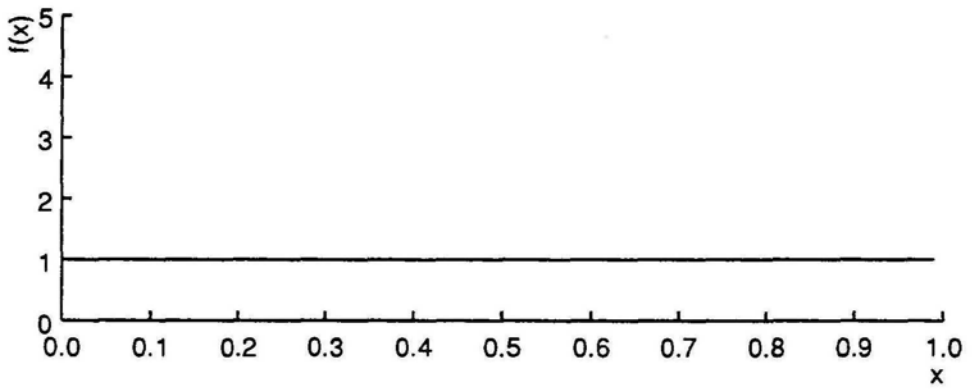


Figure G.1: Beta(x) for $\alpha_1 = \alpha_2 = 1.0$. Mean = 0.5, Var = 0.083.

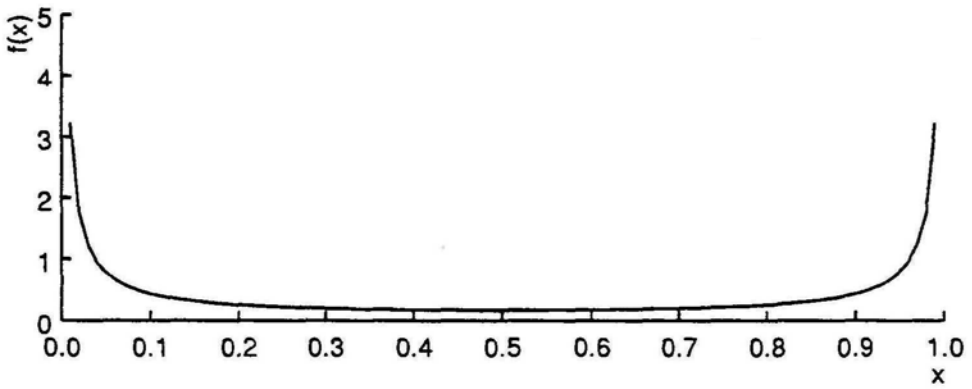


Figure G.2: Beta(x) for $\alpha_1 = \alpha_2 = .1$. Mean = 0.5, Var = 0.208.

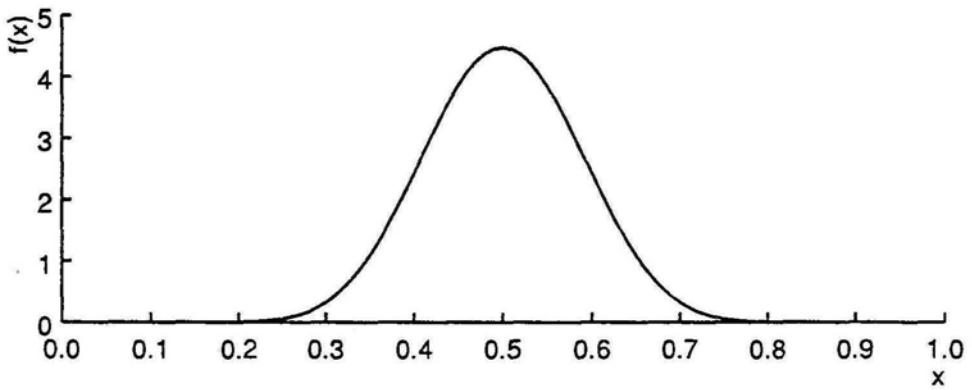


Figure G.3: Beta(x) for $\alpha_1 = \alpha_2 = 16.0$. Mean = 0.5, Var = 0.008.

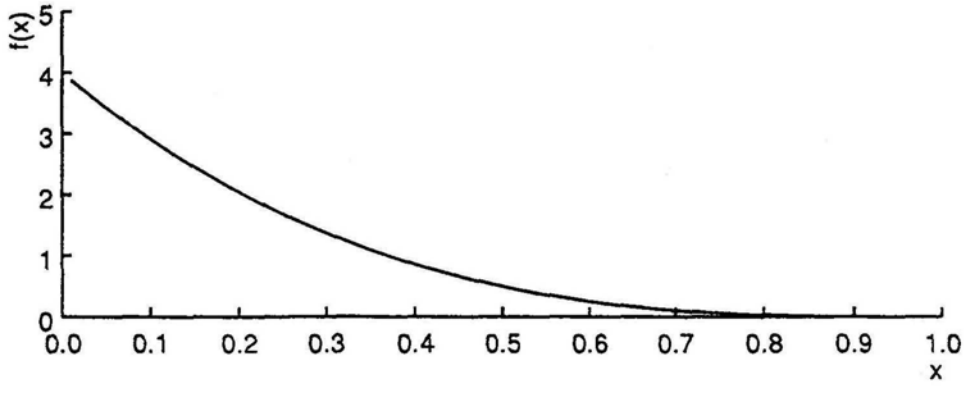


Figure G.4: Beta(x) for $\alpha = 4.0, \alpha_2 = 1.0$. Mean = 0.20, Var = 0.027.

Appendix H

Notation

α ... friction angle $\alpha = \arctan \mu$, Dirichlet parameters
$\mathcal{A}$ ... action space
a .... action (command, input)
b .... number of bits
C ... cost function
c .... ratio of filter cost to action cost
$\bar{C}$ ... expected cost
$\mathcal{C}$ ... continuous curve in the plane
$C\Theta$. closure of set Θ
δ .... decision rule
Δ ... space of decision rules
$d(\theta)$. diameter of part at orientation θ
$\diamondsuit$ ... indicates unstable equilibria
ϵ .... tolerance margin for grasp angle
$E(A)$ expected value (mean) of random variable A
f ... probability density function, or generic function
F ... cumulative distribution function
G_p .. mapping for pushing
G_q .. mapping for squeezing
G ... mapping for grasping
Γ ... Gamma function (continuous factorial)
h ... length of an interval of state space
H ... Shannon entropy

I ... interval of state space
k ... number of actions
L ... Loss function
λ ... probability vector
Λ ... set of all probability vectors
μ ... coefficient of friction
$m(x)$ marginal probability of x
n ... number of edges in a polygon
O ... Order notation, $f(n) = O(g(n))$ iff $\exists c, n_0$ such that $|f(n)| \leq c|g(n)|$ for all $n \geq n_0$.
Ω ... parameter space for multinomial distribution
$P(A)$ probability of event A
p .... probability
$\mathbf{p}$ ... parameter vector
$\mathbf{P}$ ... stochastic transition matrix
π ... 3.14159....
Ψ ... stability measure for grasping
$r(\theta)$. radius of part at angle θ
R ... risk
$\Re$ ... set of real numbers
ρ .... plan, a sequence of actions, also Bayesian expected loss
S ... event corresponding to a stable grasp
S^1 .. set of rotation angles $[0, 2\pi)$
$s(\theta)$. deterministic transfer function
σ ... orientation of gripper in world frame
T ... smallest period in a transfer function
θ .... system state (part orientation)
Θ ... state space
Θ_s .. set of stable orientations
x ... sensor datum
X ... sensor data space or matrix of observations
$\times$ ... indicates stable angles
W .. event corresponding to a wedged grasp
${}^w x$ .. indicates that x is measured in the world frame

Bibliography

[Abel *et al.*, 1985] J. M. Abel, Will Holtzmann, and John M. McCarthy. On grasping planar objects with two articulated fingers. In *International Conference on Robotics and Automation*. IEEE, 1985.

[Allen, 1987] Peter K. Allen. *Robotic Object Recognition Using Vision and Touch*. Kluwer Academic Publishers, 1987.

[Astrom, 1987] Karl Johan Astrom. Adaptive feedback control. *Proceedings of the IEEE*, 75(2), 1987.

[Ayache and Faugeras, 1987] N. Ayache and O. D. Faugeras. Maintaining representations of the environment of a mobile robot. *ISRR*, 1987.

[Bajcsy *et al.*, 1989] Ruzena Bajcsy, Susan Lederman, and Roberta Klatzky. Machine systems for exploration and manipulation: A conceptual framework and method of evaluation. GRASP Lab Tech Report MS-CIS-89-03, University of Pennsylvania, 1989.

[Baker, Fortune, and Gross , 1985] B. S. Baker, S. Fortune, and E. Grosse. Stable prehension with a multi-fingered hand. In *IEEE International Conference on Robotics and Automation*, 1985.

[Barber *et al.*, 1986] J. Barber, R. A. Volz, R. Desai, R. Rubinfeld, B. Schipper, and J. Wolter. Automatic two-fingered grip selection. In *IEEE International Conference on Robotics and Automation*, 1986.

[Barraquand and Latombe, 1990] Jerome Barraquand and Jean-Claude Latombe. A monte-carlo algorithm for path planning with many degrees of freedom. In *IEEE International Conference on Robotics and Automation*, 1990.

[Berger, 1985] James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

[Bernardo, 1979] J. M. Bernardo. Expected information as expected utility. *Annals of Statistics*, 7(3), 1979.

[Bertsekas, 1987] Dimitri P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, 1987.

[Bolle and Cooper, 1986] R. M. Bolle and D. B. Cooper. On optimally combining pieces of information, with application to estimating 3d complex object position from range data. *IEEE*, PAMI-8:5:619–638, 1986.

[Boothroyd, Poli, and Murch , 1982] Geoffrey Boothroyd, Corrado Poli, and Laurence E. Murch. *Automatic Assembly*. Marcel Dekker, Inc., 1982.

[Brady, 1989] Michael Brady. Problems in robotics. In *The Robotics Review I*, pages 1–25. MIT Press, 1989. edited by O. Khatib, J. J. Craig, and T. Lozano-Perez.

[Brassard and Bratley, 1988] G. Brassard and P. Bratley. *Algorithmics, Theory and Practice*. Prentice Hall, 1988.

[Brooks, 1982] Rod A. Brooks. Symbolic error analysis and robot planning. *International Journal of Robotics Research*, 1(4):29–68, Winter 1982.

[Brost and Mason, 1989] R. C. Brost and M. T. Mason. Graphical analysis of planar rigid-body dynamics with multiple frictional contacts. (submitted to the 5th ISRR), 1989.

[Brost, 1988] Randy C. Brost. Automatic grasp planning in the presence of uncertainty. *The International Journal of Robotics Research*, December 1988. Also appeared in Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, April, 1986.

[Brost, 1990] Randy C. Brost. *A State/Action Space Approach to Planning Robot Actions*. PhD thesis, CMU, 1990. (to appear).

[Buckley, 1986] Stephen J. Buckley. *Planning and Teaching Compliant Motion Strategies*. PhD thesis, MIT, January 1986.

[Buckley, 1988] Stephen J. Buckley. On sensory planning. Research Report RC 13609, IBM Yorktown, 1988.

[Cai and Roth, 1987] C. Cai and B. Roth. On the spatial motion of rigid bodies with point contact. In *International Conference on Robotics and Automation*. IEEE, 1987.

[Cameron and Durrant-Whyte, 1989] Alec Cameron and Hugh Durrant-Whyte. A bayesian approach to optimal sensor placement. Robotics Research Report OUEL 1759/89, University of Oxford, U.K., 1989.

[Canny, 1987] John Francis Canny. *The Complexity of Robot Motion Planning*. PhD thesis, MIT, May 1987.

[Cheeseman, 1985] Peter Cheeseman. In defense of probability. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Los Angeles, CA, August 1985. IJCAI.

[Chen, 1982] Fan Yu Chen. Gripping mechanisms for industrial robots. *Mechanism and Machine Theory*, 17(5), 1982.

[Chiu, 1985] S. L. Chiu. Generating compliant motion of objects with an articulated hand. Master's thesis, MIT Dept. of EEC, 1985.

[Christiansen and Goldberg, 1990] Alan Christiansen and Kenneth Y. Goldberg. Robotic manipulation planning with stochastic actions. In *DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control*, November 1990. San Diego, California.

[Christiansen *et al.*, 1990] Alan D. Christiansen, Matthew T. Mason, and Tom M. Mitchell. Learning reliable manipulation strategies without initial physical models. In *International Conference on Robotics and Automation*. IEEE, May 1990. To appear.

[Christiansen, 1991] Alan D. Christiansen. *Automatic Acquisition of Task Theories for Robotic Manipulation*. PhD thesis, CMU School of Computer Science, 1991. To appear.

[Cutkosky, 1985] Marc R. Cutkosky. *Grasping and Fine Manipulation for automated manufacturing*. PhD thesis, Department of Mechanical Engineering, CMU, 1985.

[Davenport, 1970] Wilbur B. Davenport. *Probability and Random Processes*. McGraw-Hill, 1970.

[DeGroot, 1970] Morris H. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill, 1970.

[DeGroot, 1975] Morris H. DeGroot. *Probability and Statistics*. Addison Wesley, 1975.

[Demmel and Lafferriere, 1989] Jim Demmel and Gerardo Lafferriere. Optimal three finger grasps. Robotics Report 170, NYU Courant Institute, February 1989.

[Dickey *et al.*, 1987] James M. Dickey, Jhy-Ming Jiang, and Joseph B. Kadane. Bayesian methods for censored categorical data. *Journal of the American Statistical Association*, 82(399), 1987.

[Donald, 1987] Bruce R. Donald. *Error Detection and Recovery in Robotics*. Springer-Verlag, 1987.

[Donald, 1990] Bruce R. Donald. Planning and executing robot assembly strategies in the presence of uncertainty. In *96th Annual Meeting of the American Mathematical Society*, 1990.

[Drummond and Bresina, 1990] Mark Drummond and John Bresina. Anytime synthetic projection: Maximizing the probability of goal satisfaction. In *AAAI-90*, 1990.

[Durrant-Whyte, 1988] Hugh Durrant-Whyte. *Integration, Coordination, and Control of Multi-Sensor Systems*. Kluwer, 1988.

[Dynkin and Yushkevich, 1979] E. B. Dynkin and A. A. Yushkevich. *Controlled Markov Processes*. Springer-Verlag, 1979.

[Eppstein, 1990] David Eppstein. Reset sequences for monotonic automata. *SIAM Journal of Computing*, 19(5), 1990.

[Erdmann and Mason, 1986] M. A. Erdmann and M. T. Mason. An exploration of sensorless manipulation. In *IEEE International Conference on Robotics and Automation*, 1986.

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

[Erdmann, 1989] Mike A. Erdmann. *On Probabilistic Strategies for Robot Tasks*. PhD thesis, MIT, 1989.

[Etzioni, 1989] Oren Etzioni. Tractable decision-analytic control. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann, 1989. An expanded version is available as technical report CMU-CS-89-119.

[Fearing, 1983] Ron S. Fearing. Touch processing for determining a stable grasp. Master's thesis, MIT, September 1983.

[Feldman and Sproull, 1977] J. A. Feldman and R. F. Sproull. Decision theory and artificial intelligence ii: The hungry monkey. *Cognitive Science*, 1:158–192, 1977.

[Fisher, 1912] R. A. Fisher. On an absolute criterion for fitting frequency curves. *Messenger of Math*, 41:155–, 1912.

[Fisher, 1942] R. A. Fisher. *The Design of Experiments*. Oliver and Boyd, Edinburgh, 1942.

[Gauss, 1809] K. F. Gauss. *Theoria Motus corporum coelestium in sectionibus conicis solem ambientium*. Perthes and Besser: Hamburg, 1809.

[Gelb, 1986] Arthur Gelb. *Applied Optimal Estimation*. MIT Press, 1986.

[Genesereth and Nilsson, 1987] M. Genesereth and N. Nilsson. *Logical Foundations of AI*. Morgan-Kaufmann, 1987.

[Goldberg and Bajcsy, 1984] Kenneth Y. Goldberg and Ruzena Bajcsy. Active touch and robot perception. *Cognition and Brain Theory*, 7(2):199–214, 1984.

[Goldberg and Mason, 1990] K. Y. Goldberg and M. T. Mason. Bayesian grasping. In *International Conference on Robotics and Automation*. IEEE, May 1990.

[Goldberg and Pearlmutter, 1988] Kenneth Yigael Goldberg and Barak Pearlmutter. Using a neural network to learn the dynamics of the cmu direct-drive arm ii. Technical Report CMU-CS-88-160, Carnegie-Mellon University, August 1988.

[Good, 1969] I. J. Good. What is the use of a distribution? *Multivariate Analysis*, 2:183–203, 1969.

[Gray and Davisson, 1986] Robert M. Gray and Lee D. Davisson. *Random Processes: A Mathematical Approach for Engineers*. Prentice-Hall, 1986.

[Grupen *et al.*, 1989] R. A. Grupen, T. C. Henderson, and I. D. McCammon. A survey of general purpose manipulation. *International Journal of Robotics Research*, 8(1), 1989.

[Hager and Mintz, 1989] Gregory D. Hager and Max Mintz. Sensor modeling and robust sensor data fusion. *Fifth International Symposium on Robotics Research*, 1989.

[Hager, 1988] Gregory D. Hager. *Active Reduction of Uncertainty in Multi-Sensor Systems*. PhD thesis, University of Pennsylvania CIS, 1988.

[Hanafusa and Asada, 1977] H. Hanafusa and H. Asada. Stable prehension by a robot hand with elastic fingers. *Trans. of Soc. of Inst. and Control Engineers*, 13(4), 1977.

[Hansson *et al.*, 1990] Othar Hansson, Andrew Mayer, and Stuart Russel. Decision-theoretic planning in bps. In *AAAI Symposium on Planning*, 1990.

[Hitakawa, 1988] Hajime Hitakawa. Advanced parts orientation system has wide application. *Assembly Automation*, 8(3), 1988.

[Hogan, 1984] N. Hogan. Impedance control of industrial robots. *Robotics and Computer Integrated Manufacturing*, 1984.

[Hong, Ikeuchi, and Gremban , 1990] Ki Sang Hong, Katsushi Ikeuchi, and Keith Gremban. Minimum cost aspect classification: A module of a vision algorithm compiler. CS Tech Report CMU-CS-90-124, Carnegie Mellon School of Computer Science, 1990.

[Howard, 1971] Ronald A. Howard. *Dynamic Probabilistic Systems (2 volumes).* John Wiley, 1971.

[Hutchinson and Kak, 1990] Seth A. Hutchinson and Avinash C. Kak. Extending the classical ai planning paradigm to robotic assembly planning. In *IEEE International Conference on Robotics and Automation*, 1990.

[Inoue, 1974] H. Inoue. Force feedback in precise assembly tasks. Memo 308, MIT Artificial Intelligence Laboratory, August 1974.

[Jameson, 1985] J. Jameson. *Analytic Techniques for Automated Grasp.* PhD thesis, Department of Mechanical Engineering, Stanford University, June 1985.

[Jeffrey, 1965] Richard C. Jeffrey. *The Logic of Decision.* Univ. of Chicago Press, 1965.

[Kanazawa and Dean, 1989] Keiji Kanazawa and Thomas Dean. A model for projection and action. In *1989 IJCAI*, 1989.

[Karp, 1986] R. M. Karp. 1985 turing award lecture: Combinatorics, complexity, and randomness. *CACM*, pages 97–117, February 1986.

[Kato, 1982] Ichiro Kato. *Mechanical Hands Illustrated.* Hemisphere Publishing Corp, NYC, 1982.

[Kerr and Roth, 1986] J. Kerr and B. Roth. Analysis of multifingered hands. *IJRR*, 4(4), 1986.

[Kerr, 1984] J. R. Kerr. *An Analysis of Multifingered Hands.* PhD thesis, Stanford Dept. of MechE, 1984.

[Knuth, 1973] Donald E. Knuth. *The Art of Computer Programming.* Addison-Wesley, 1973.

[Koutsou, 1988] A. Koutsou. Object exploration using a parallel-jaw gripper. GRASP Lab Tech Report MS-CIS-88-48, University of Pennsylvania, 1988.

[Kozen, 1977] Dexter Kozen. Lower bounds for natural proof systems. In *18th Annual Symposium on Foundations of Computer Science.* IEEE, 1977.

[Lange, 1984] J. C. Lange. *Design Dimensioning with Computer Graphics Applications.* Marcel Dekker, Inc., 1984.

[Latombe, 1989] Jean-Claude Latombe. Motion planning with uncertainty: On the preimage backchaining approach. In *The Robotics Review I*, pages 53–70. MIT Press, 1989. edited by O. Khatib, J. J. Craig, and T. Lozano-Perez.

[Li and Sastry, 1988] Zexiang Li and S. Shankar Sastry. Task-oriented optimal grasping by multifingered robot hands. *IEEE J. of Robotics and Automation*, 4(1):32–43, 1988.

[Lindley, 1956] D. V. Lindley. On the measure of the information provided by an experiment. *Annals of Mathematical Statistics*, 27, 1956.

[Lozano-Perez, Mason, and Taylor , 1984] T. Lozano-Perez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research*, 3(1):3–24, Spring 1984.

[Luce and Raiffa, 1957] R. Duncan Luce and Howard Raiffa. *Games and Decisions*. John Wiley and Sons, 1957.

[Mani and Wilson, 1985] M. Mani and R. D. W. Wilson. A programmable orienting system for flat parts. In *Proc: North American Mfg. Research Inst. Conf XIII*, 1985.

[Markenscoff and Papadimitriou, 1989] Xanthippi Markenscoff and Christos H. Papadimitriou. Optimum grip of a polygon. *IJRR*, 8(2), April 1989.

[Mason, Goldberg, and Taylor , 1988] Matthew T. Mason, Kenneth Y. Goldberg, and Russell H. Taylor. Planning sequences of squeeze-grasps to orient and grasp polygonal objects. Technical Report CMU-CS-88-127, Carnegie Mellon University, Computer Science Dept., Schenley Park, Pittsburgh, PA 15213, April 1988.

[Mason, 1978] M. T. Mason. Compliance and force control for computer controlled manipulators. Master's thesis, MIT, May 1978.

[Mason, 1982] Matthew T. Mason. *Manipulator Grasping and Pushing Operations*. PhD thesis, MIT, June 1982. published in *Robot Hands and the Mechanics of Manipulation*, MIT Press, 1985.

[Mason, 1984] Matthew T. Mason. Automatic planning of fine motions: Correctness and completeness. In *proceedings, International Conference on Robotics*, Atlanta, GA, March 1984. IEEE Computer Society.

[Mason, 1986] Matthew T. Mason. On the scope of quasi-static pushing. In O. Faugeras and G. Giralt, editors, *The Third International Symposium on Robotics Research*. MIT Press, 1986.

[Matthies and Shafer, 1987] Larry H. Matthies and Steve A. Shafer. Error modelling in stereo navigation. *IEEE Journal of Robotics and Automation*, 3(3):239–248, 1987.

[Matthies, Szeliski, and Kanade , 1987] Larry Matthies, Richard Szeliski, and Takeo Kanade. Kalman filter-based algorithms for estimating depth from image sequences. Technical Report CMU-CS-87-185, CMU Computer Science, 1987.

[Mishra, Schwartz, and Sharir , 1987] Bud Mishra, J. T. Schwartz, and M. Sharir. On the existence and synthesis of multifinger positive grips. *Algorithmica*, 2(4):641–558, 1987.

[Montana, 1988] David J. Montana. The kinematics of contact and grasp. *IJRR*, 7(3), 1988.

[Moravec, 1988] H. P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2):61–74, 1988.

[Narendra and Thathachar, 1989] Kumpati S. Narendra and Mandayam A. L. Thathachar. *Learning Automata: An Introduction*. Prentice Hall, 1989.

[Natarajan, 1986] Balas K. Natarajan. *On Moving and Orienting Objects*. PhD thesis, Cornell Computer Science, 1986.

[Natarajan, 1989] Balas K. Natarajan. Some paradigms for the automated design of parts feeders. *International Journal of Robotics Research*, 8(6):98–109, December 1989. Also appeared in Proceedings of the 27th Annual Symposium on Foundations of Computer Science, 1986.

[Nguyen, 1988] Van-Duc Nguyen. Constructing force-closure grasps. *International Journal of Robotics Research*, 7(3), 1988.

[Nguyen, 1989] Van-Duc Nguyen. Constructing stable grasps. *International Journal of Robotics Research*, 8(1), 1989.

[Nilsson, 1980] Nils J. Nilsson. *Principles of Artificial Intelligence*. Tioga, 1980.

[Onions and Archard, 1973] R. A. Onions and J. F. Archard. The contact of surfaces having a random structure. *J. Physics D.: Applied Physics*, 6:289–304, 1973.

[Papadimitriou and Tsitsiklis, 1987] Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3), August 1987.

[Pearl, 1984] Judea Pearl. *Heuristics*. Addison-Wesley, 1984.

[Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan-Kaufmann, 1988.

[Pertin-Troccaz, 1989] Jocelyn Pertin-Troccaz. Grasping: A state of the art. In *The Robotics Review I*, pages 71–98. MIT Press, 1989. edited by O. Khatib, J. J. Craig, and T. Lozano-Perez.

[Peshkin and Sanderson, 1988] Michael A. Peshkin and Art C. Sanderson. Planning robotic manipulation strategies for workpieces that slide. *IEEE Journal of Robotics and Automation*, 4(5), October 1988.

[Peshkin, 1986] Michael A. Peshkin. *Planning Robotic Manipulation Strategies for Sliding Objects*. PhD thesis, CMU Dept. of Physics, Nov 1986.

[Pingle *et al.*, 1974] K. Pingle, R. Paul, and R. Bolles. *Programmable Assembly: Three Short Examples*. Stanford AI Lab Film, 1974.

[Preparata and Shamos, 1985] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

[Rabin, 1980] Michael O. Rabin. A probabilistic algorithm for testing primality. *Journal of Number Theory*, 12:128–138, 1980.

[Rajan, Burridge, and Schwartz , 1987] V. T. Rajan, R. Burridge, and J. T. Schwartz. Dynamics of a rigid body in frictional contact with rigid walls. In *IEEE Conference on Robotics and Automation*, 1987.

[Requicha, 1983] A. A. G. Requicha. Toward a theory of geometric tolerancing. *IJRR*, 2(4), 1983.

[Russell and Wefald, 1988] Stuart Russell and Eric Wefald. Decision-theoretic control of reasoning: General theory and an application to game playing. Technical Report UCB/CSD 88/435, UC Berkeley, October 1988.

[Salisbury, 1982] J. Kenneth Salisbury. *Kinematic and Force Analysis of Articulated Hands*. PhD thesis, Department of Mechanical Engineering, Stanford University, May 1982. published in it Robot Hands and the Mechanics of Manipulation, MIT Press, 1985.

[Sanderson, 1984] A. C. Sanderson. Parts entropy methods for robotic assembly system design. In *International Conference on Robotics and Automation*. IEEE, 1984.

[Schwartz *et al.*, 1987] J. Schwartz, J. Hopcroft, and M. Sharir. *Planning, Geometry, and Complexity of Robot Motion*. Ablex, 1987.

[Shafer, 1990] Steve Shafer. Talk at the CMU School of Computer Science 25th Anniversary, 1990.

[Shannon, 1948] Claude E. Shannon. A mathematical theory of communcation. *Bell System Technical Journal*, 27:379–423 and 623–656, 1948.

[Shefrin, 1981] H. M. Shefrin. On the combinatorial structure of bayesian learning with imperfect information. *Discrete Mathematics*, 37, 1981.

[Simon, 1955] Herb A. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 1955. Reprinted in Models of Thought, Yale University Press, 1979.

[Singer and Seering, 1987] Neil C. Singer and Warren Seering. Utilizing dynamic stability to orient parts. *Journal of Applied Mechanics*, 54, 1987. See also Singer's 1985 MIT Masters Thesis.

[Singer, 1985] Neil C. Singer. Utilizing dynamic and static stability to orient parts. Master's thesis, MIT, 1985.

[Smith and Cheeseman, 1986] Randall C. Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *IJRR*, 5(4):56–68, 1986.

[Stansfield, 1987] Sharon A. Stansfield. *Visually Guided Haptic Object Recognition.* PhD thesis, Univ. Of Pennsylvania Dept. of CIS, 1987.

[Stengel, 1986] Robert T. Stengel. *Stochastic Optimal Control.* John Wiley, 1986.

[Suzuki and Kohno, 1981] T. Suzuki and M. Kohno. The flexible parts feeder which helps a robot assemble automatically. *Assembly Automation*, 1981.

[Szeliski, 1988] Richard S. Szeliski. *Bayesian Modeling of Uncertainty in Low-Level Vision.* PhD thesis, CMU School of Computer Science, August 1988.

[Tan, 1990] Ming Tan. Csl: A cost-sensitive learning system for sensing and grasping objects. In *IEEE International Conference on Robotics and Automation*, 1990.

[Taylor, Mason, and Goldberg , 1987] Russ H. Taylor, Matthew M. Mason, and Kenneth Y. Goldberg. Sensor-based manipulation planning as a game with nature. In *Fourth International Symposium on Robotics Research*, August 1987.

[Taylor, 1976] Russ H. Taylor. *A Synthesis of Manipulator Control Programs from Task-Level Specification.* PhD thesis, Stanford AIM-282, july 1976.

[Tella *et al.*, 1982] Richard Tella, John Birk, and Robert Kelley. General purpose hands for bin-picking robots. *IEEE Trans. on Systems, Man, and Cybernetics*, 12(6), Nov/Dec 1982.

[Trinkle and Paul, 1988] J. C. Trinkle and R. P. Paul. Planning for dextrous manipulation with sliding contacts. *International Journal of Robotics Research*, pages 468–485, June 1988.

[Wald, 1947] Abraham Wald. *Sequential Analysis.* Dover Publications, 1947.

[Whitney and Junkel, 1982] Dan E. Whitney and Eric F. Junkel. Applying stochastic control theory to robot sensing, teaching, and long term control. In *Proceedings of the American Control Conference*, 1982.

[Whitney, 1982] Dan E. Whitney. Quasi-static assembly of compliantly supported rigid parts. *Journal of Dynamic Systems, Measurement, and Control*, 104:65–77, March 1982.

[Whitney, 1990] Dan Whitney. Personal Communication, 1990.