Proceedings of the 1991 IEEE
International Conference on Robotics and Automation
Sacramento, California - April 1991

Generating Stochastic Plans for a Programmable Parts Feeder

Kenneth Y. Goldberg

Dept. of Computer Science
University of Southern California
Los Angeles, CA 90089

Matthew T. Mason
Michael A. Erdmann
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

The problem of *parts feeding* – orienting parts that are initially jumbled – is common in industrial automation. In this paper we consider a *programmable* parts feeder: a mechanism that can be reprogrammed to handle differently shaped parts. We present a planning algorithm that accepts an n-sided polygonal part as input and, in time $O(n^2)$, generates a program (plan) for the feeder that maximizes *expected feedrate*. We have implemented the planner and verified some of the resulting plans in our laboratory. This work illustrates a stochastic framework for manipulation planning described in [5].

1 Introduction

Manufacturing processes such as injection molding and stamping often produce a stream of unoriented parts that must be reoriented before assembly. A parts feeder is a machine that orients parts (Figure 1). Traditional parts feeders are often inflexible. Although there are a vast number of different techniques for feeding parts, most are hand-crafted mechanisms that depend critically on the shape of the part. When part geometry changes, the feeder must be mechanically redesigned with a trial-and-error process that can require several months [14].

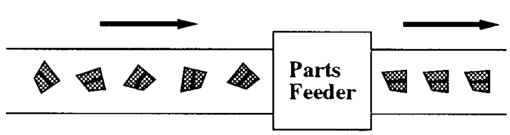

Figure 1: A parts feeder orients parts as they arrive on the left-hand conveyor belt.

In contrast, a programmable parts feeder can be reprogrammed rather than physically modified when part geometry changes. Flexibility is enhanced by automatically generating the appropriate program (or *plan*). Our programmable parts feeder has two components:

- **Mechanism.** A device that can orient parts under software control. We use a "frictionless" parallel-jaw gripper [6]. This mechanism requires one controlled degree of freedom to orient the gripper, and one uncontrolled degree of freedom (such as a pneumatic actuator) to open and close the jaws (Figure 2).
- **Planning Algorithm.** A method for transforming a geometrical part description into a program for the mechanism. We present a planning algorithm that accepts an n-sided polygonal part description as input and outputs a stochastically optimal plan for orienting the part.

Our design is based on a stochastic approach to grasping described in [6]. In that paper we used a brute-force search to find stochastically optimal plans. Here we present an algorithm that runs in polynomial time and prove that the planner is complete for all polygonal parts. This paper does not address three additional components of a programmable parts feeder: a means for separating jumbled parts into a stream of isolated parts, a conveyor belt for transporting parts in and out of the feeder, and a binary filter that can distinguish between symmetric part orientations (*e.g.*, a silhouette trap).

1.0.1 Example

A parts feeding plan is a sequence of open loop squeezing actions specified by the orientation of the gripper. Consider a rectangular part whose initial orientation is unknown. Using a frictionless gripper, 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 3.

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 only 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 the one-step plan maximizes expected feedrate.

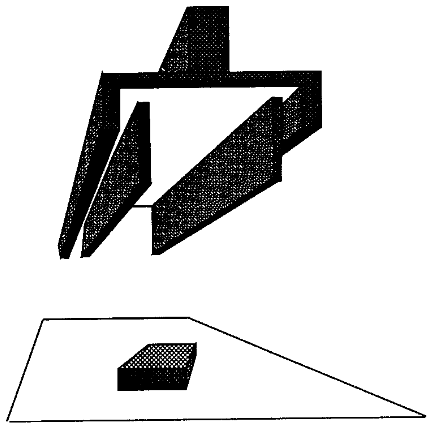

Figure 2: Schematic of the programmable parts feeder poised above a rectangular part.

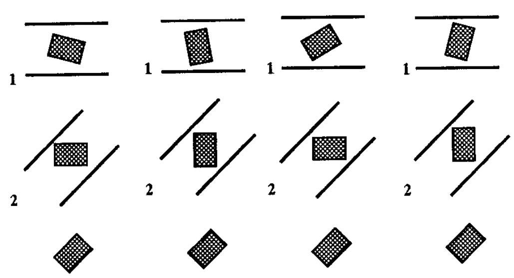

Figure 3: Top view of a plan for feeding rectangular parts. Gripper orientation is shown with two parallel lines. Four traces of a two-step plan for orienting the part. Each trace runs from top to bottom. The plan is open-loop: commanded actions do not depend on sensor data. Although the part's initial orientation is different in each trace, its final orientation is the same.

2 Related Work

An excellent introduction to mechanical parts feeders can be found in [1]. [14] 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 [15].

[8] developed plans to orient 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. [13] 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. [4] developed multi-step plans to orient parts with a sequence of tray-tilting actions. [16] and [10] considered multi-step plans to orient parts using a sequence of grasps with a parallel-jaw gripper. Although each of the planning algorithms described in this paragraph use realistic models of mechanics, none are guaranteed to find a plan in polynomial time.

Natarajan [11] ignored the mechanics of parts feeders and focussed on the abstract problem of planning with a given set of transfer functions. Citing a result by [7] as evidence that solving the problem for an arbitrary set of transfer functions is PSPACE-Complete, Natarajan developed a polynomial-time planning algorithm for the class of *monotonic* transfer functions. A function f is monotonic if the states have a cyclic ordering: $\theta_1 \preceq \theta_2 \preceq \ldots \preceq \theta_n \preceq \theta_1$, as is the case for the set of planar orientations, and the sequence $f(\theta_1), f(\theta_2), \ldots$ also has cyclic order. Recently, [3] reported an $O(kn^2)$ algorithm for the restricted problem where all functions are monotonic, where n is the number of states and k is the number of available functions.

Here we identify a class of monotonic transfer functions related to the mechanics of a parallel-jaw gripper. For an n-sided polygonal part, there are $O(n^2)$ unique transfer functions. Thus Eppstein's algorithm could be used to find a guaranteed plan in time $O(n^4)$. We present a geometric algorithm that finds a stochastically optimal plan in time $O(n^2)$.

3 Mechanical Analysis

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 [9] and [12].
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 the 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 [2], [16], and [10]. 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 ([6] describe an implementation of such a mechanism). Assuming simultaneous contact (assumption 6) greatly simplifies the mechanical analysis. In the last section we discuss how assumption 6 can be relaxed.

A *squeeze action* is the combination of orienting the gripper, closing the jaws as far as possible, and then opening the jaws. Note that no sensing is required. When a part is grasped with the frictionless gripper, it assumes one of a finite number of stable orientations corresponding to local minima in a *diameter function*. 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. Let S^1 be the set of planar orientations. The diameter function, $d : S^1 \to \Re$, is the distance between parallel tangents at angle θ. For polygonal parts, the diameter function is piecewise sinusoidal as shown in top of Figure 4.

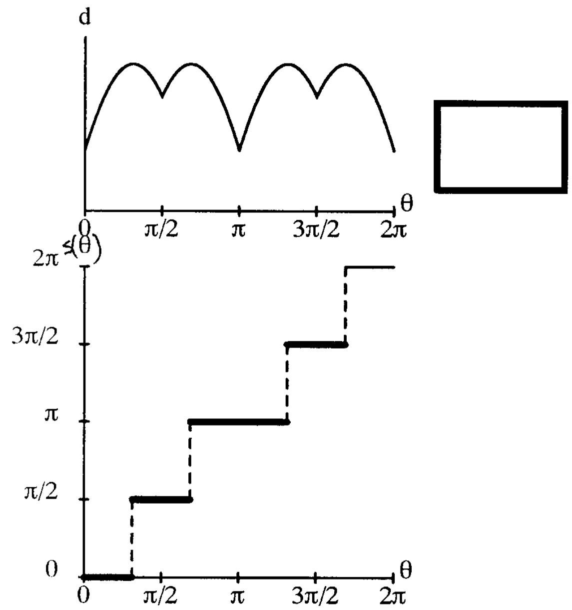

Figure 4: The diameter function (top) and squeeze function (bottom) for the rectangular part.

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 [5].

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*, $s : S^1 \to S^1$.

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 final orientation of the part with respect to the gripper. The squeeze function can be derived from the diameter function as follows. All orientations that lie between a 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 squeeze function to be single-valued, we assume that all steps are closed on the left. See bottom of figure 4.

Note that the squeeze function has period π due to rotational symmetry in the gripper. Rotational symmetry in the part also introduces periodicity into the squeeze function. In general the squeeze function has period T such that

$$s(\theta + T) = s(\theta) + T. \quad (1)$$

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 no sequence of squeeze-grasp actions can map orientations θ and $\theta + T$ into a single final orientation. Thus a part can only be oriented *up to symmetry* in its squeeze function.

3.1 Prior Probability Distribution

After the first squeeze action, the part's orientation relative to the gripper will be one of the stable orientations corresponding to local minima in the diameter function.

If the first gripper angle (action) in the plan is chosen randomly, the part's initial orientation can be described with a uniform random variable on the set of planar orientations [5]. 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 prior probability distribution by integrating the probability density between discontinuities in the transfer function. The prior probability distribution is related to the width of the steps in the squeeze 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.

3.2 Cost Metric

We relate a cost function to the time required to orient a part. Let c_a be the time cost for each action and c_f be the time cost for the binary filter (applied once per iteration). Consider an 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. For a given probability distribution we can identify a most-likely state (breaking ties arbitrarily), θ^*. After executing the plan, the gripper aligns the part over the filter so that only parts in orientation θ^* are accepted. All others are recycled for later iterations of the plan.

On average, a plan will go through $1/P(\theta^*)$ iterations until θ^* is achieved, where $P(\theta^*)$ is the probability of achieving state θ^* in one iteration. Multiplying the cost per iteration by the expected number of iterations, the expected cost for an i-step iterative plan, ρ_i, is

$$\overline{C}(\rho_i) = \frac{ic_a + c_f}{P(\theta^*)}. \quad (2)$$

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

$$\rho_i^* = \arg\min_{\rho_i} \overline{C}(\rho_i). \quad (3)$$

The globally optimal plan is the best plan over all values of i; its cost is

$$\overline{C}^* = \min_i \overline{C}(\rho_i^*) \quad (4)$$

We define a **stochastically optimal parts-feeding plan,** ρ^* to be a plan such that $\overline{C}(\rho^*) = \overline{C}^*$.

4 The Algorithm

We define the stochastic parts feeding problem as follows:

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

The algorithm for solving this problem proceeds 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.

4.1 Phase I

Phase I works backward from a single final state, finding an action that collapses a *set* of states into this state. This set then becomes the target for another action. A sequence of plans for orienting the part can be derived from the sequence of state sets. Phase I is specified below.

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 $(\Theta_0, \Theta_1, ..., \Theta_k)$.

Figures 4 through 6 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. Θ_0 is the *image* of Θ_1, and Θ_1 is the *preimage* of Θ_0. Step 3 searches for the widest interval whose image is smaller than Θ_{i-1}. This interval becomes Θ_i. 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 6.

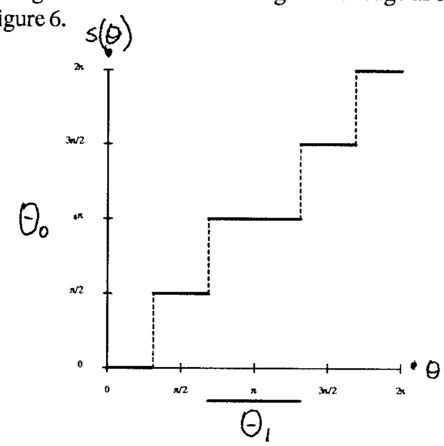

Figure 5: In step 2, the widest single step in the transfer function is identified. All the orientations in Θ_1 (horizontal bar at bottom), map into the single final orientation: Θ_0 (dot at π).

Continuing, wider and wider intervals are found until the condition in Step 4 of Phase I is satisfied. This condition is used to insure that the algorithm runs in time $O(n^2)$. Phase I returns k sets corresponding to an optimal 1-step plan, an optimal 2-step plan, ..., an optimal k-step plan.

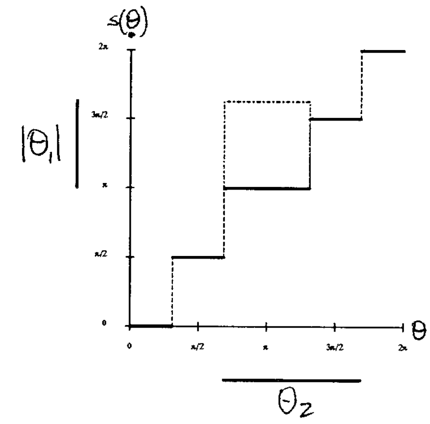

Figure 6: In step 3, we identify the largest interval whose image is smaller than $h_1 = |\Theta_1|$. This can be visualized by left-aligning a box of dimension h_1 with each step. If the squeeze function emerges from the right edge of the box, then the corresponding image is smaller than h_1. The largest such interval in this case is Θ_2 (shown at bottom).

4.2 Phase II

To find the globally optimal plan, Phase II evaluates the expected cost of each locally optimal plan found by Phase I and chooses the plan with minimal expected cost. Recall from section 3.2 that the cost of a plan is related the number of steps and the probability that it succeeds in one iteration. Each iteration of plan ρ_i^* has i steps plus one step for the filter. The probability that it succeeds in one iteration is related to the size of its preimage. With a uniform probability distribution as insured by a random twist of the gripper on the first step of each iteration, the probability that plan ρ_i^* succeeds is $2\pi/h_i$, where h_i is the length of its preimage. Thus the expected cost for plan ρ_i^* is $(ic_a + c_f)2\pi/h_i$.

4.3 Recovering the Forward Plan

To recover the forward plan we must work backward from the final interval, Θ_0 (a point). Let θ_i be the leftmost point in interval 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 must 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 interval Θ_2 to orientations in interval Θ_1. We proceed backwards until we reach Θ_k. Then by reversing and negating the sequence: $[-\sigma_k, -\sigma_{k-1}, ..., -\sigma_1]$ we have a set of gripper angles that defines an open-loop plan for converting all orientations in Θ_k to the single orientation θ_0.

Note that at each step we align the intervals 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 $\epsilon_i = (|\Theta_i| - |\Theta_{i-1}|)/2$.

The forward squeezing plan is

$$\rho_k = [-\sigma_k + \epsilon_k, -\sigma_{k-1} + \epsilon_{k-1}, ..., -\sigma_1 + \epsilon_1]. \quad (5)$$

5 Correctness

We prove the algorithm finds stochastically optimal plans by proving that each i-step plan collapses the largest possible preimage and then proving that the size of the preimage is directly related to the expected cost. For details and generalization to other cost and probability models see [5].

6 Complexity

We define the complexity of the algorithm as a function of the number of edges of the polygon, n. We neglect the numerical complexity of representing vertices and angles as rational numbers.

The computational complexity of the algorithm is $O(n^2)$. Step 1 of Phase I, computing the squeeze function, can be performed in time $O(n \log n)$ (see [6]). 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. In [5] we prove that a stochastically optimal plan requires no more than $N = \frac{2\pi}{h_1}(c+1)$ steps. Since $h_1 \geq \pi/n$, Phase I requires only $O(n)$ iterations of Step 3.

The computational complexity of Phase II is $O(n)$, since there will be at most $O(n)$ plans and it takes time $O(1)$ to compute the expected cost for each plan and $O(n)$ to choose an optimal plan.

Theorem 1 *The algorithm runs in time $O(n^2)$ and finds plans of length $O(n)$.*

7 Completeness

Theorem 2 *For any polygonal part, we can always find a stochastically optimal plan.*

Proof: As described earlier, any polygonal part will generate a squeeze function $s(\cdot)$ 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). We show that for any squeeze function and any h, Either we can find a larger preimage:

$$\exists \theta, s(\theta + h) - s(\theta) < h, \quad (6)$$

Or h is a period of symmetry in $s(\cdot)$:

$$\forall \theta, s(\theta + h) = s(\theta) + h, \quad (7)$$

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

To understand formula 6, consider that we've reached a point in the algorithm where the current set is $\Theta = [\theta, \theta + h)$. Formula 6 says that there is some set, $[\theta, \theta + h]$, larger than Θ, whose image, $[\ s(\theta), s(\theta + h)\ ]$, is smaller than Θ. We can also interpret this with reference to figure 6. Formula 6 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 $s(\cdot)$ and any h, either formula 6 or formula 7 must hold, consider the integral of the function $s(\theta + h) - s(\theta) - h$ over the domain $[0, T)$.

$$\begin{aligned}
& \int_0^T [s(\theta + h) - s(\theta) - h] d\theta \\
= & \int_h^{T+h} s(\theta) d\theta - \int_0^T s(\theta) d\theta - hT & (8)\\
= & -\int_0^h s(\theta) d\theta + \int_T^{T+h} s(\theta) d\theta - hT & (9)\\
= & -\int_0^h s(\theta) d\theta + \int_0^h [s(\theta) + T] d\theta - hT & (10)\\
= & -\int_0^h s(\theta) d\theta + \int_0^h s(\theta) d\theta + hT - hT & (11)\\
= & \ 0. & (12)
\end{aligned}$$

Since this integral is zero, then by the mean value theorem, either the function is uniformly zero (formula 7, i.e. $h = T$) or there is some point in the domain where the function is less than zero (formula 6).

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. ■

8 Implementation

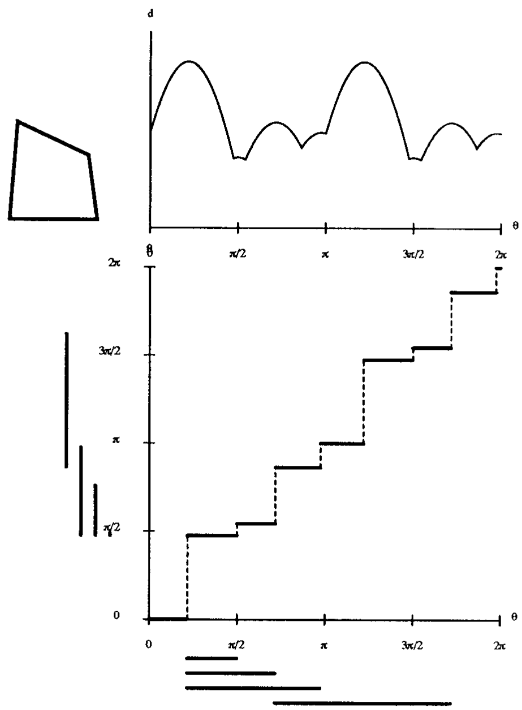

Figure 7: Geometric analysis of 4-sided part (4gon), showing diameter and squeeze functions. The horizontal bars show the preimage at each stage.

We implemented the algorithm in Common Lisp using exact (rational) arithmetic to express vertices and angles. For 1000 random parts, we compared the plans generated by the algorithm to the plans found by brute-force search. In all cases where the forward search was able to run to completion, the plans found by the two planners were equivalent. Another example is shown in figures 7 and 8.

We implemented the programmable parts feeder with a linear bearing and a Lord parallel-jaw gripper attached to a PUMA arm. For the two parts described in this report, the plans usually work as expected. Exceptions occur when Assumption 6 is violated; *i.e.* the jaws do not make contact simultaneously.

9 Discussion

We presented an algorithm to find stochastically optimal parts-feeding plans. The analysis depends on the fact that all actions can be described by cyclic shifts of a transfer function, s :

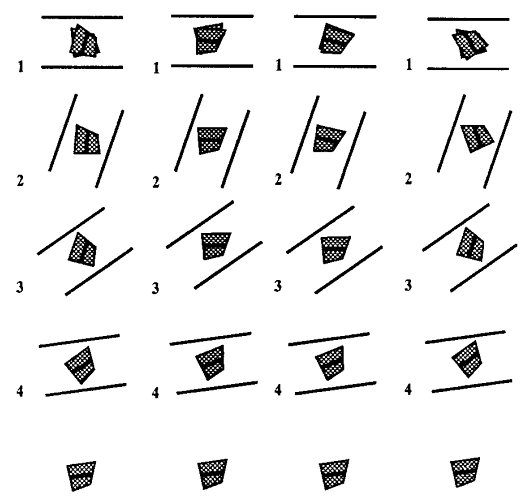

Figure 8: Plan for 4gon.

$S^1 \times S^1$, that is piecewise constant and monotonically non-decreasing on the half-open interval $[0, 2\pi)$. The class of *squeeze* actions meet these criteria, as does the class of *push-grasp* actions, where the part is pushed by one jaw prior to grasping [2]. Push-grasp actions do not require Assumption 6. The planning algorithm works for either class of actions [5]. When we physically tested plans that use push-grasp actions, we observed no failures.

Perhaps we can find other classes of actions that meet the criteria above. It would also be useful to extend this algorithm to actions that are not deterministic, such as the class of fence-push actions considered by [13].

Acknowledgements

We thank Randy Brost, John Canny, Merrick Furst, Gary Miller, and Barak Pearlmutter for technical discussions. We also thank members of the manipulation lab: Srinivas Akella, Alan Christiansen, Kevin Lynch, and Yu Wang. This work was supported by NSF grant DMC-8520475 and NASA-Ames grant NCC 2-463. The first author can be reached at `goldberg@iris.usc.edu` or (213) 740-9080.

References

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

[2] 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.

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

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

[5] Kenneth Y. Goldberg. *Stochastic Plans for Robotic Manipulation.* PhD thesis, CMU School of Computer Science, August 1990. Copies can be obtained from Catherine Copetas (copetas@cs.cmu.edu, 412-268-8525).

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

[7] Dexter Kozen. Lower bounds for natural proof systems. In *18th Annual Symposium on Foundations of Computer Science.* IEEE, 1977.

[8] M. Mani and R. D. W. Wilson. A programmable orienting system for flat parts. In *Proc: North American Mfg. Research Inst. Conf XIII*, 1985.

[9] 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.

[10] 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.

[11] 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.

[12] Michael A. Peshkin. *Planning Robotic Manipulation Strategies for Sliding Objects.* PhD thesis, CMU Dept. of Physics, Nov 1986.

[13] 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.

[14] Neil C. Singer. Utilizing dynamic and static stability to orient parts. Master's thesis, MIT, 1985.

[15] 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.

[16] 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.