The International Journal of Robotics Research

http://ijr.sagepub.com

Using Partial Sensor Information to Orient Parts Srinivas Akella and Matthew T. Mason

Srinivas Akella and Matthew T. Mason
The International Journal of Robotics Research 1999; 18; 963
DOI: 10.1177/02783649922067663

The online version of this article can be found at: http://ijr.sagepub.com/cgi/content/abstract/18/10/963

Published by: \$SAGE Publications

http://www.sagepublications.com

On behalf of:

Multimedia Archives

Additional services and information for The International Journal of Robotics Research can be found at:

Email Alerts: http://ijr.sagepub.com/cgi/alerts

Subscriptions: http://ijr.sagepub.com/subscriptions

Reprints: http://www.sagepub.com/journalsReprints.nav

Permissions: http://www.sagepub.com/journalsPermissions.nav

Citations (this article cites 21 articles hosted on the SAGE Journals Online and HighWire Press platforms): http://ijr.sagepub.com/cgi/content/refs/18/10/963

Srinivas Akella

Beckman Institute for Advanced Science and Technology University of Illinois at Urbana-Champaign Urbana, IL 61801, USA

Matthew T. Mason

Robotics Institute & Computer Science Department Carnegie Mellon University Pittsburgh, PA 15213, USA

Using Partial Sensor Information to Orient Parts

Abstract

Parts orienting, the process of bringing parts in initially unknown orientations to a goal orientation, is an important aspect of automated assembly. The most common industrial orienting systems are vibratory bowl feeders, which use the shape and mass properties of parts to orient them. Bowl feeders rely on a sequence of mechanical operations and typically do not use sensors. In this paper, we describe the use of partial information sensors along with a sequence of pushing operations to eliminate uncertainty in the orientations of parts. We characterize the shorter execution lengths of sensor-based plans and show that sensor-based plans are more powerful than sensorless plans in that they can bring a larger class of parts to distinct orientations.

We characterize the relation among part shape, orientability, and recognizability to identify conditions under which a single plan can orient and recognize multiple part shapes. Although part shape determines the results of the actions and the sensed information, we establish that differences in part shape do not always lead to differences in part behavior. We show that for any convex polygon, there exists an infinite set of nonsimilar convex polygons that behave identically under linear normal pushes. Furthermore, there exists an infinite set of nonsimilar convex polygons whose behavior cannot be distinguished even with diameter sensing after each push. We have implemented several planners and demonstrated generated plans in experiments.

1. Introduction

A parts orienting system has to bring randomly oriented parts to desired orientations for subsequent assembly. Vibratory bowl feeders are the most common industrial systems for orienting parts (Boothroyd, Poli, and Murch 1982; Riley 1983). Although very effective, bowl feeders are not reconfigurable and have to be designed by skilled craftsmen, a process that

The International Journal of Robotics Research Vol. 18, No. 10, October 1999, pp. 963-997, ©1999 Sage Publications, Inc.

can take a couple of months to handle a new part or to discover that the part cannot be oriented. So there is a need for flexible parts orienting systems whose capabilities can be characterized in advance. Our approach is to use knowledge of the part geometry and mechanics of the manipulation operations along with partial sensor information to flexibly generate orienting strategies with simple hardware elements. This can reduce changeover costs and time to market for new products.

Most orienters and feeders use a variety of mechanical operations to change the state of the part (Boothroyd, Poli, and Murch 1982; Shirai and Saito 1989; Peshkin and Sanderson 1988a; Mirtich et al. 1996; Goldberg 1993). These include shaking the parts, capturing parts in nests, using fences and cutouts to align or reject parts, aligning parts on a conveyor against stationary or moving fences, and parallel-jaw grasping. The mechanical operations serve to sequentially reduce the uncertainty in the part orientation. Inexpensive and robust sensors, such as LED sensors currently used to verify part presence, can aid this process by reducing the number of orienting steps required. This translates into systems with faster cycle times and fewer orienting stages.

We use sensors that provide partial information on part orientation in combination with mechanical operations to reduce uncertainty in the part orientation. We focus on a particular implementation where a polygonal part on a moving conveyor belt contacts a stationary fence and rotates into alignment with it. The part is made to repeatedly align itself against the fence by being picked up, rotated, and dropped off upstream by a robot with a suction device. An LED sensor then measures the aligned part diameter at the fence. A sequence of such *push-align operations* is executed until the part's orientation is determined (Figure 1). We analyze these operations both with and without sensing of the resting diameter of the part. The principal results of this paper are the following:

 Sensor-based and sensorless plans using a sequence of push-align operations always exist to orient any polygonal part up to symmetry in its push-diameter and push functions (introduced in Section 3).

- 2. We prove bounds on the worst-case length of sensor-based orienting plans and present implemented planners to generate minimum length plans.
- 3. We show that even with sensing, differences in part shape do not guarantee parts can be distinguished during orienting. We identify conditions under which multiple part shapes can be oriented and recognized with a single sensor-based plan.

In our system, the orienting actions bring the part to a finite set of states and the sensors provide only partial information on part orientation. Our results have a direct bearing on a broader class of orienting systems that includes the passive fence-orienting system of Wiegley et al. (1996), the sensorless 1JOC single-degree-of-freedom fence of Akella et al. (1997), and the parallel-jaw grippers of Rao and Goldberg (1994).

1.1. Outline of the Paper

This paper focuses on the use of partial sensor information with manipulation operations to orient and recognize polygonal parts. We show that with inexpensive LED sensors, parts can be oriented faster, the orientable class of parts is larger, and a single plan can orient different part shapes. The paper is organized as follows. After a discussion of related work in Section 2, the first half of the paper concentrates on the problem of orienting a single part of known shape. Section 3 describes the assumptions made and the class of actions we use. The radius function, push function, and push-diameter function used to predict the results of push-align operations are described, and a procedure to generate actions for a set of states is presented. After a brief description of sensorless orienting in Section 4, our sensor model is discussed in Section 5. Section 6 concentrates on sensor-based orienting of a single part. It describes an AND/OR search algorithm to find plans with minimum worst-case length, and proves bounds on the lengths of orienting plans and the completeness of the search-based planner. The section also presents the alternative bottom-to-top algorithm to find minimum length plans and bounds its running time. Section 7 shows that sensorbased strategies are more powerful than sensorless strategies in that they can orient a larger class of parts.

Section 8, which forms the bulk of the latter part of the paper, concentrates on the orienting and recognizing of multiple part shapes. We first elucidate the relation between part shape and the radius, push, and push-diameter functions. Although the mapping from the set of part shapes to the set of radius functions is one-to-one, the mappings from the set of part shapes to the set of push-diameter functions are many-to-one. Since an infinite set of nonsimilar polygonal parts can share the same push-diameter function,

necessary and sufficient conditions for a single sensor-based plan to orient and recognize parts from a known set of parts are identified. Section 9 briefly discusses the implemented planners and experiments conducted to verify generated plans. Section 10 concludes with a summary of our results and a discussion of future work. Portions of the work reported in this paper appeared earlier in Akella 1996 and Akella and Mason 1998a.

2. Related Work

2.1. Pushing and Grasping

Pushing is a commonly used manipulation operation. The motion of a pushed object depends on its geometry, the pressure distribution between the object and the support surface, the nature of contact between the pusher and the object, and the motion of the pusher. Mason (1986) analyzed the mechanics of robotic pushing operations. He developed a procedure to determine the instantaneous motion of a pushed object with a known support pressure distribution, and he derived rules to predict the rotation direction of a pushed object with an unknown pressure distribution. Mani and Wilson (1985) used the pushing rules of Mason to derive an Edge Stability Map for straight-line pushes and developed a system to orient parts with a sequence of fence pushes at different angles. Peshkin and Sanderson (1988b) found the locus of centers of rotation of a pushed object for all possible pressure distributions over an enclosing circle centered at the object center of mass. These centers of rotation provide bounds on the rate of rotation of an object being pushed. From the center of rotation corresponding to the slowest rotation, they calculated the minimum push distance guaranteed to align the object with the fence. Balorda and Bajd (1994) used a two-finger tool to reduce the positional uncertainty of an object by pushing it. They discuss the effect of finger configurations on accurate positioning of the object. Brost (1992) presents a numerical integration procedure that returns the initial configurations that guarantee the linear pushing motion of a polygon will bring a contacting polygon to a goal equilibrium configuration.

Grasping, typically with a parallel-jaw gripper, is probably the most common mode by which robots manipulate objects. Brost (1988) developed an algorithm to plan grasps with a parallel-jaw gripper that are robust to bounded uncertainties in object orientation. As an intermediate result, using the pushing rules of Mason, he developed the push stability diagram to describe the possible motions of an object being pushed by a fence. Goldberg (1993) simplified Brost's model for parallel-jaw grasping by using only pushes normal to the face of the gripper and extended Brost's results by analyzing multistep strategies to orient an object by grasping. He developed an algorithm to generate sensorless multistep orienting plans for polygonal objects using a frictionless gripper. This algorithm is provably complete, that is, it is

guaranteed to generate a plan to orient any polygonal object (up to symmetry). Furthermore, it is guaranteed to return the shortest plan. Chen and Ierardi (1995) showed these plans to reorient an object by grasping are O(n) in length, where n is the number of edges of the object. Using Brost's push stability diagram, Deacon, Low, and Malcolm (1993) perform search to find the minimum number of linear pushes to orient a part without sensors, for a given coefficient of friction between the fence and part. Rao and Goldberg (1995) present algorithms for orienting curved planar parts by grasping. In recent work, van der Stappen, Goldberg, and Overmars (1996) characterize the number of sensorless orienting actions for a part in terms of its eccentricity rather than the number of equilibrium orientations. Their analysis, which includes nonpolygonal parts, shows that with increasing eccentricity, the number of orienting steps decreases.

2.2. Parts Feeding

Our long-term goal is to develop parts feeders for use in industry. Riley (1983) provides a good introduction and survey of the area of industrial automatic assembly. Vibratory bowl feeders have been in widespread industrial use since their introduction in the 1940s. Boothroyd, Poli, and Murch (1982) describe parts feeding and orienting devices for automated assembly, including bowl feeders. The SONY APOS system (Shirai and Saito 1989) relies on vibration and shape to orient parts. Parts are made to flow over a vibrating pallet with nests designed to trap only parts in the correct orientation. Brost (1991, 1992) demonstrates the use of shape constraints to orient a part by designing a nest to "catch" the part. Caine (1994) considers the design of interacting part shapes to constrain motion and applies it to a vibratory bowl feeder track. Berkowitz and Canny (1996) use a dynamic simulator to perform a series of simulated experiments to select bowl feeder gate parameters. Christiansen, Edwards, and Coello (1996) use genetic algorithms to identify sequences of bowl feeder gates that improve feeding efficiency. Krishnasamy, Jakiela, and Whitney (1996) analyze the effect of shape and vibration parameters on the energy of parts and hence their efficiency of entrapment in an APOS-like vibration system. Zumel and Erdmann (1997) use a frictionless two-palm manipulator to bring polygons in an unknown initial orientation to a goal orientation. Each palm has a single degree of freedom, and the manipulator relies on gravity to manipulate parts without grasping. Erdmann (1996) developed a two-palm manipulation system that uses a sequence of nonprehensile operations such as sliding to rotate an object.

In early work on orienting parts using the task mechanics, Boothroyd et al. (1972) analyzed the statistical distribution of the natural rest states of objects with and without bouncing and rolling after impact. Grossman and Blasgen (1975) developed a system that brought objects to a finite number of orientations in a tilted tray, where their orientation was determined by a

tactile probe. Erdmann and Mason (1988) explored the use of sensorless manipulation strategies to eliminate uncertainty in the configuration of a part in a tray by repeated tilts of the tray. They implemented a planner that used knowledge of the mechanics of sliding to predict the results of actions. Natarajan (1989) focused on the computational issues related to the automated design of sensorless parts feeders. By assuming deterministic transitions of an orienting device and monotonicity conditions, he was able to derive polynomial-time algorithms for certain classes of object orienting problems. Eppstein (1990) extended Natarajan's results by presenting a more efficient algorithm that is guaranteed to find the shortest reset sequences for monotonic automata. He also extended his technique to certain classes of nonmonotonic automata. Erdmann, Mason, and Vanecek (1993) described a polynomialtime planner for the problem of orienting a polyhedral part by tilting a tray with infinite friction surfaces.

Singer and Seering (1987) analyzed the effects of impacts and oscillatory motions on a part and determined conditions for its reorientation. Their goal was to identify impacts and oscillatory motions that can change the initially unknown state of the part to a desired goal state. Böhringer, Bhatt, and Goldberg (1995) used a vibrating plate to position and orient parts sensorlessly. The vibrations generate a force field, and parts move to nodes of the vibratory force field. Swanson, Burridge, and Koditschek (1995) analyzed the motion of a part subjected to a vibratory juggling motion and indicated conditions under which all initial orientations of the part acquired a unique stable motion.

Peshkin and Sanderson (1988a) described the orienting effect of a fence on a pushed object in terms of its configuration map, which maps all initial orientations of the object to all possible resulting orientations. They used these configuration maps to find a sequence of fences to automatically orient a sliding part. Brokowski, Peshkin, and Goldberg (1993) designed curved fence sections to eliminate uncertainty in the orientations of parts being oriented by the fences. Wiegley et al. (1996) developed a complete algorithm to find the shortest sequence of frictionless curved fences to orient a polygonal part. Akella et al. (1997) developed an actuated sensorless one-joint fence system to orient parts on a conveyor by combining drift motions of a part on the conveyor with fence pushing motions.

2.3. Sensor-Based Manipulation

Grossman and Blasgen (1975) describe a system that orients parts by a combination of tilt, vibration, and probing operations. A part is dropped into a tilted tray that is then vibrated. The part comes to rest in one of a finite set of orientations, which are discriminated using mechanical probing operations at points specified by an operator. Sanderson (1984) used positional entropy measures to determine the relative efficiency of mechanical operations and sensor operations for mechan-

ical assembly tasks. Taylor, Mason, and Goldberg (1988) approached automatic planning of sensor-based manipulation programs as a game-theory problem where the goal of the robot is to achieve a specified task state. The robot chooses actions, nature chooses sensor readings, and planning is performed by searching the game tree. Uncertainty in the robot's world model, control errors, and noisy sensors are explicitly modeled. Using an AND/OR search tree, they applied this framework to the tasks of orienting an object by tray tilting and orienting an object by parallel-jaw grasping. Rao and Goldberg (1994) described optimal and suboptimal strategies to orient and recognize a part from a known set of parts by grasping it and sensing the jaw diameter. They also showed that for the class of planar polygonal parts, the shape of a part cannot be uniquely recovered from its diameter function since there is an infinite set of parts with the same diameter function. Govindan and Rao (1994) presented an algorithm for determining the minimum length plan to recognize and orient a part by parallel-jaw grasping where the jaw diameter is sensed after each grasp. They conjectured that finding the minimum length grasp plan is NP-hard. Arkin et al. (1998) recently showed that computing the minimum length grasp plan to recognize parts is NP-hard, and gave a polynomial time approximation algorithm that generates plans within a log factor from optimal.

Erdmann (1995) explored the advantages of using sensors tailored to the task and actions so they can be used in feedback loops. Canny and Goldberg (1994) advocated the use of simple, inexpensive sensors and actuators for industrial automation tasks. Donald, Jennings, and Rus (1995) analyzed the information requirements of manipulation tasks performed by cooperating mobile robots in terms of information invariants. A key question they explored is determining how much information is provided by the task mechanics, sensors, state information, communication, and computation.

2.4. Pose Determination and Object Localization

Much research has been devoted to the problem of recognizing an object from a known set of objects and determining its pose. We restrict our discussion to cases where the sensor data are incomplete and multiple sensory operations are required. Gaston and Lozano-Perez (1984) introduced the Interpretation Tree as a structure to recognize and localize polyhedral objects in the plane using tactile sensors. Grimson and Lozano-Perez (1984) used local measurements of three-dimensional positions and surface normals to recognize and locate objects with 6 degrees of freedom. The local constraints used, such as distance between points and angles between the surface normal vectors, are very effective in pruning invalid interpretations. Ellis (1992) found paths for a tactile recognition probe to uniquely determine the pose of a 2D object in the presence of sensed data error given initial tactile data.

There is also related theoretical work on determining the shapes and poses of polygons by geometric probing. Cole and Yap (1987) discussed the number of probes necessary to determine the shape and position of a polygon. Skiena (1989) described a variety of probing models and provided a list of open problems in probing. Alevizos, Boissonnat, and Yvinec (1989) presented results on determining the shape of a simple polygon by a minimal number of probes. Wallack, Canny, and Manocha (1993) discussed the use of cross-beam sensors to determine the orientation of an object by measuring its diameter at three angles. Jia and Erdmann (1995) examined the issue of finding the minimum number of sensing points required to distinguish between a finite set of poses of a known set of polygonal shapes.

2.5. Completeness

For a given task, we select a set of actions and determine what combination of the actions, if any, can solve the task. An interesting aspect of the orienting task described here is that a solution always exists for the selected class of actions, and furthermore, we can always find it. Such tasks, which always have a solution for any instance of the problem, have been termed solution-complete by Goldberg (1995). Other examples of solution-complete problems include sensorless orienting of parts by parallel-jaw grasping (Goldberg 1993), controllability of a sphere in rolling contact with another sphere or a plane (Li and Canny 1990), and controllability and motion planning for nonholonomic mobile robots in the obstacle-free plane (Barraquand and Latombe 1993). Goldberg describes a modular fixturing problem for polygonal parts (Zhuang, Wong, and Goldberg 1994) that is not solution-complete, but for which a complete algorithm, an algorithm guaranteed to return a solution if it exists and report failure otherwise, exists (Brost and Goldberg 1996). Characterizing tasks in this manner and developing complete algorithms to solve them enable us to identify and guarantee capabilities of robot systems.

3. Action Model

We assume that a singulated part in an unknown initial orientation drifts on the conveyor belt until it contacts a fence placed perpendicular to the direction of motion (Figure 1). On contacting the fence, the part rotates into alignment with one of its stable edges against the fence. An LED sensor measures the resting diameter, or width, of the part perpendicular to the fence, which provides only partial information on the orientation of the part. The robot has to find and execute a sequence of push-align operations to orient the part. Each operation consists of an action followed by the sensor measurement of the diameter of the aligned part. The action consists of the robot picking up an aligned part at the fence using a suction cup, translating upstream from the fence, rotating the part through a chosen angle, and releasing the part

back on the conveyor for it to be aligned at the fence. Orienting a part amounts to identifying the part edge that is aligned against the fence (Figure 2).

The result of a push depends on the part geometry, support pressure distribution, contact friction, and length and direction of the push (Mason 1986; Peshkin and Sanderson 1988b). Although the support pressure distribution is usually unknown, for a uniform coefficient of support friction the rotation resulting from a push can be determined from the center of mass location. In analyzing our system, we make the following assumptions:

- All parts are polygons. Since the parts contact a flat fence, nonconvex polygons are treated by considering their convex hulls.
- 2. The part shape is known, and the center of mass of the part is at a known position in its interior.
- 3. All motions are in the plane and are quasi-static.
- 4. The fence is normal to the conveyor motion direction.
- 5. All frictional interactions are described by Coulomb friction.
- 6. The coefficient of friction between the part and the conveyor surface is uniform.
- 7. All bodies in contact are perfectly rigid.

When a part on the moving conveyor contacts the stationary fence, it is being pushed normal to the fence face. Viewed from a frame fixed in the conveyor, this is exactly a *linear normal push*, where a moving fence pushes a part in a direction normal to the fence face. The part rotation due to such a push can be predicted using the radius function (Goldberg 1993). The part comes to rest on the fence with only stable edges aligned with the fence. An edge is *stable* if the orthogonal projection of the center of mass onto the edge lies in the interior of the edge. An edge is *unstable* if the projection of the center of mass onto the edge, and is *metastable* if the center of mass projects onto a vertex of the edge.

When the part contacts the fence, it takes a certain amount of time to rotate into alignment with the fence. For a known constant conveyor velocity, we can find an upper bound on the time required for the rotation using the results of Peshkin and Sanderson (1988b). During the initial alignment, if the part center of mass lies exactly on the fence normal during the push, the alignment time is unbounded. A practical solution is to empirically determine a time bound within which the part is aligned with high probability. When the part may come to rest on a metastable edge, we can perform additional small rotations of the part to perturb its orientation and cause alignment on a neighboring stable edge.

3.1. Radius Function, Push Function, and Push-Diameter Function

The rotational behavior of a part being pushed can be predicted from its radius function, push function, and push-diameter function, which we now describe. To determine the result of a push-align operation, we follow Goldberg (1993) in using the *radius function*. The radius r of a polygon at an orientation ϕ is the perpendicular distance from a reference point in the polygon to a supporting line of the polygon with orientation ϕ . The radius function describes the variation in the radius as the supporting line is rotated (Figure 3). Without loss of generality, we assume the supporting line orientation is measured relative to a stable edge.

DEFINITION 1. The **radius function** of a polygon is a mapping $r: \mathcal{S}^1 \to \mathcal{R}$ from the orientation ϕ of a supporting line of the polygon to the perpendicular distance r from a reference point in the polygon to the supporting line. We specify this reference point to be the center of mass of the polygon.

When the center of mass is the reference point and the fence is the supporting line, the local minima of the radius function correspond to stable edges of the part. A *kink* in the radius function is a point with a discontinuity in slope that is not a local minimum of the radius function. A kink corresponds to an unstable edge or metastable edge of the part. A part being pushed against a fence rotates to achieve a minimum radius. Each local minimum determines a convergent orientation, and each local maximum determines a divergent orientation. A push has the net effect of mapping the entire interval between two adjacent divergent orientations to the enclosed convergent orientation. The radius function has a period of 360°. Symmetry in part shape leads to *periodicity* in the radius function with a period less than 360°.

The *push function* (Goldberg 1993) describes a normal push (Figure 4). It is a monotonic step function with a period of 360°. We use the push functions of parts to find sensorless orienting plans.

DEFINITION 2. The **push function** is a mapping $p: \mathcal{S}^1 \to \mathcal{S}^1$ from the initial orientation of the fence relative to the part to the final orientation of the fence relative to the part after a push, assuming the part orientation is held constant.

DEFINITION 3. A **symmetric** push function has a period T, 0 < T < 360, such that $p(\phi + T) = (p(\phi) + T) \mod 360$.

DEFINITION 4. The **diameter function** of a polygon is a mapping $w: \mathcal{S}^1 \to \mathcal{R}$ from the orientation ϕ of a pair of parallel supporting lines of the polygon to the perpendicular diameter w of the part between the supporting lines.

The *push-diameter function* (Figure 5) describes the part diameter resulting from a push. The diameter is the maximal width of an aligned part measured normal to the fence. The push-diameter function is a step function with a period of

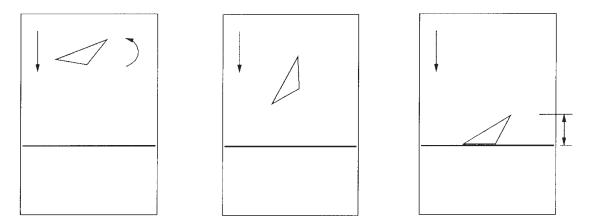


Fig. 1. Schematic overhead view of a part on the conveyor during a push-align operation, shown from left to right. The conveyor motion is "downward." The robot first picks up a part and rotates it through a chosen angle before placing it on the conveyor. The part then drifts on the conveyor into contact with the fence. The part rotates into alignment with the fence and its diameter is then sensed.

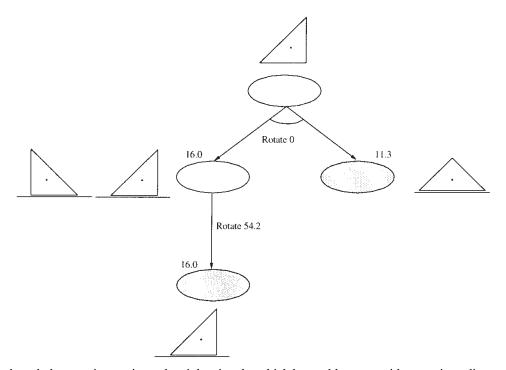


Fig. 2. A sensor-based plan to orient an isosceles right triangle, which has stable states with nonunique diameter values. Each node contains a set of indistinguishable states. Goal nodes, shown shaded, have a single state. The sensor value shown at a node indicates the diameter value for the set of states at that node.

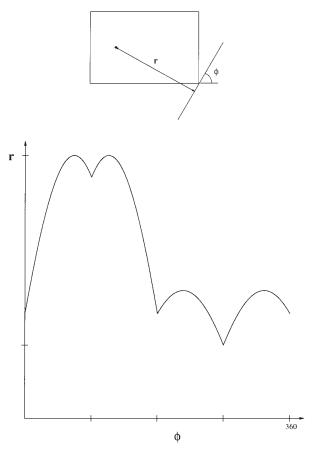


Fig. 3. The radius function for the rectangle with its center of mass indicated by the black dot. The radius function is the plot of the part radius r as the fence orientation ϕ is varied, and wraps around at 360° . The four minima of the radius function correspond to the four stable edges of the rectangle. Based on Goldberg (1993).

 360° and is used to generate sensor-based orienting plans that measure part diameter after every push.

DEFINITION 5. The **push-diameter function** is a mapping $d: S^1 \to \mathcal{R}$ from the initial fence orientation ϕ to the perpendicular part diameter d resulting from a push, for a constant part orientation. That is, $d(\phi) = w(p(\phi))$.

Let *S* be the set of stable orientations, where each stable orientation corresponds to a local minimum of the radius function. By construction, $S = \{\phi_s | \phi_s = p(\phi), 0 \le \phi < 360\}$.

DEFINITION 6. A **periodic** push-diameter function has a period T, 0 < T < 360, such that $d(\phi) = d(\phi + T)$. Else the push-diameter function is considered **aperiodic**.

DEFINITION 7. A part has a **quasi-symmetric** push-diameter function d if there exists a period T, 0 < T < 360, such that $d(\phi) = d(\phi + T)$, and there exists at least one stable orientation $\phi_s \in S$ such that $p(\phi_s + T) \neq (\phi_s + T) \mod 360$.

DEFINITION 8. A part has a **symmetric** push-diameter function d if there exists a period T, 0 < T < 360, such that

 $d(\phi) = d(\phi + T)$, and for every $\phi_s \in S$, $p(\phi_s + T) = (\phi_s + T) \mod 360$.

These characterizations of the push and push-diameter functions will be used in Section 6.2. Note that symmetry of the push-diameter function does not imply symmetry of the push function, and symmetry of the push function does not imply symmetry of the push-diameter function.

3.2. Finding Representative Actions

Using knowledge of a part's rotational behavior, we identify a set of orienting actions. To compute actions, it is convenient to consider the polygon orientation relative to a fixed line, the fence.

3.2.1. Resting and Action Ranges

The *resting range* of a stable orientation of a part is the set of initial orientations for which the part comes to rest in that stable orientation (Figure 6). The stable orientations correspond

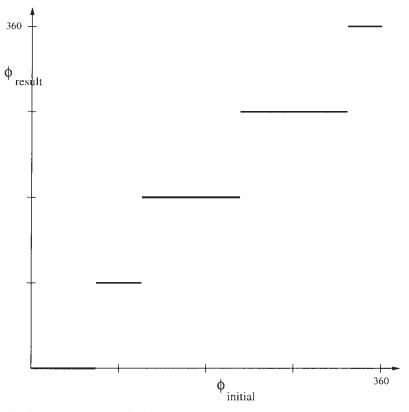


Fig. 4. The push function for the rectangle of Figure 3. The horizontal and vertical axes represent the initial and resulting relative orientations respectively. Based on Goldberg 1993.

to the minima of the radius function, and the resting ranges are obtained from the enclosing left and right maxima of the stable orientations on the radius function.

A push-align action corresponds to an angle, in the interval [0, 360), that the part is rotated through before being pushed. The task mechanics dictate that actions can be grouped into equivalence classes where all member actions of a class have the same result.

DEFINITION 9. The **action range** of a stable edge e_i with respect to a destination edge e_j is the interval of rotations for which the part transitions from e_i to e_j . That is, every action in an action range results in the same final state when applied to a given initial state.

Each edge of a part with n stable edges has n action ranges. Consider the range of actions to transition from start state s_i to destination state s_j , where state s_i corresponds to edge e_i (Figure 7). Let the stable orientation of state s_i be ψ_i , and the right and left limits of the resting range of state s_j be ρ_j and λ_j , respectively (see Figure 8). Each push-align action is

identified by the angle by which the robot rotates the part. The entire range of counterclockwise rotations from the left limit at λ_j to the right limit at ρ_j enables the transition from state s_i to state s_j . So the action range specified by the open interval $(\lambda_j - \psi_i, \rho_j - \psi_i)$ is an equivalence class for the transition from state s_i to state s_j . That is, any counterclockwise rotation $\theta \in (\alpha^{\min}, \alpha^{\max})$, where $\alpha^{\min} = \lambda_j - \psi_i$ and $\alpha^{\max} = \rho_j - \psi_i$, causes a deterministic transition from edge e_i to edge e_j . We compute these action ranges for the transitions from s_i to every other state. Proceeding in this manner and taking care to handle angle wraparound at 360°, we determine the action ranges for the different states (Figure 9).

3.2.2. Overlap Ranges for a Set of States

From the action ranges of the individual states, we can determine the action ranges for a given set of states. We overlap the set of intervals corresponding to the action ranges from a set of individual states to obtain another set of intervals, the *overlap ranges* (Figure 9). A set of *k* stable states of a part

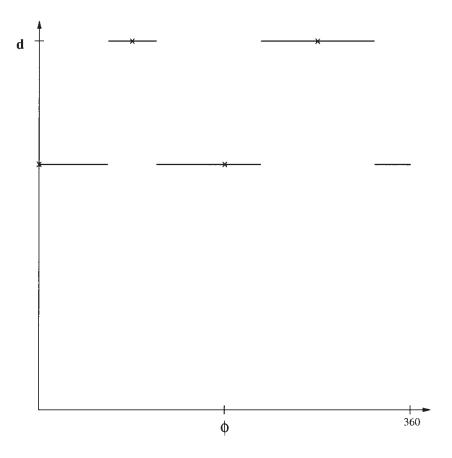


Fig. 5. The push-diameter function for the rectangle of Figure 3. The function plots the stable diameter value resulting from a push-align operation for each initial relative orientation of the part. The xs indicate stable orientations of the part.

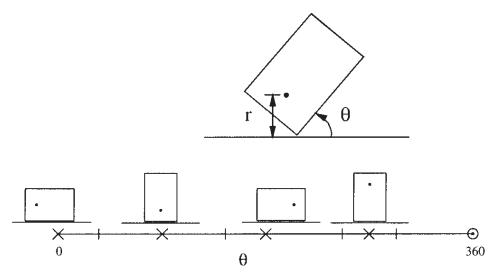


Fig. 6. Resting ranges of the rectangle of Figure 3. The xs indicate stable orientations of the rectangle for each resting range, and the vertical bars indicate the limits of each resting range. This diagram corresponds to a slice of the push stability diagram of Brost (1988) along the 90° pushing direction.

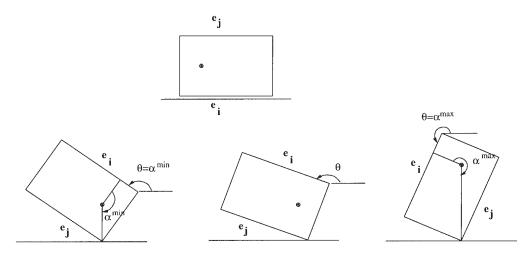


Fig. 7. The rectangle transitions from edge e_i to edge e_j for any $\theta \in (\alpha^{\min}, \alpha^{\max})$.

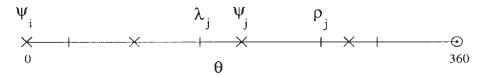


Fig. 8. The action range for the transition from state s_i to state s_j is computed from the resting ranges. The action range is $(\lambda_j - \psi_i, \rho_j - \psi_i)$, where ψ_i is the stable orientation of state s_i , and λ_j and ρ_j are the left and right limits of the resting range of state s_j .

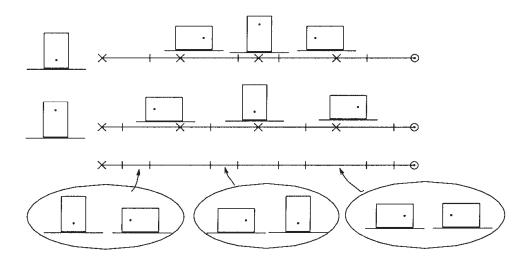


Fig. 9. Action ranges for an indistinguishable set of states of the rectangle, with the resulting stable states shown. The corresponding overlap ranges are shown at the bottom along with sets of resulting stable states for three selected ranges.

with n stable states has kn overlap ranges. The overlap ranges are also equivalence classes. For the given set of initial orientations, any action chosen from an overlap range gives the same resulting set of orientations. For each overlap range, we select the middle of the interval to be a representative action, and thus have kn representative actions to choose from.

There is always an action to get from any state to any other state. The set of representative actions covers the interval [0, 360) except for the finite number of overlap range endpoints. As long as each overlap range has nonzero extent, a deterministic action can be found to the left or right of the endpoint, and this does not affect the covering of the action space. To guarantee robustness of the actions to uncertainty, we can specify a minimum size for the overlap ranges. When there are overlap ranges smaller than the minimum size, we cannot guarantee that there is always a representative action to get from any state to any other state, or that a plan exists.

We now compute the time taken to generate the representative actions for a set of k states. For each of the k states, we can determine the action ranges to each of the n stable states in O(n) time. The action ranges can be viewed as a list of ordered range limits. The kn overlap ranges can be computed by merging the sorted action range lists for the k states into one sorted list, which can be done in $O(kn \log k)$ time. So the time to find the representative actions for a set of k states, $T_a(k, n)$, is $O(kn \log k)$.

3.3. State Transition Graph

We can represent the orienting problem for a given part by a directed graph, the *state transition graph* (Figure 10). The vertices of the graph are the stable orientations of the part. Each directed arc contains the action range for the state transition from the state at its tail to the state at its head. The sensed value at each vertex depends on the sensor resolution and sensor noise. The planning problem consists of searching this graph to find a sequence of actions to determine the orientation of the part. For sensor-based orienting, we have to determine the current state of the part based on the executed actions and measured sensor values. For sensorless orienting, we have to find a sequence of actions to bring any initial part state to the same final part state.

An alternative representation of the problem is as a finite state machine where the machine outputs a signal based on the state to which it transitions. In fact, the problem of finding the minimal length orienting plan is the same as finding the minimal length *adaptive homing sequence* of a finite state machine. See the textbook by Kohavi (1978) for an introduction to finite state machines.

4. Sensorless Orienting

To orient a part without sensors, we must find a sequence of push-align actions that brings all possible initial orientations of the part to the same final orientation. An action consists of rotating the part by a chosen angle before it is made to contact the fence. We have implemented a planner that performs breadth-first search to find minimum-length sensorless plans (Figure 11). (See Rich and Knight (1991) for an introduction to search techniques.)

In fact, there is a polynomial-time algorithm to find the shortest sequence of push-align actions to bring all initial orientations to the same final orientation. Goldberg (1993) developed a polynomial-time algorithm to find the shortest sequence for sensorless orienting of a polygon by frictionless parallel-jaw grasping. This can be applied to the push function to obtain a minimum length sensorless plan for orienting the part by pushing. Chen and Ierardi (1995) showed that for a part with n stable edges, the algorithm is guaranteed to find a solution of O(n) steps in $O(n^2)$ time. When the push function has a period of symmetry less than 360° , the part can be oriented only up to symmetry.

5. Sensor Model and State Distinguishability

To generate sensor-based plans, we have to consider the ability of the sensor to distinguish the stable states of the part. Our sensor measures the diameter of the aligned part, and this provides only partial information on the part orientation. The sensor consists of a linear array of LEDs arranged perpendicular to the fence at the side of the conveyor, with a corresponding parallel array of phototransistors on the opposite side of the conveyor. The resolution of the sensor is determined by the spacing of the LEDs. When a part is aligned with the fence, we can determine its diameter up to the resolution of the sensor by identifying which LED—phototransistor pairs are blocked by the part. Two stable orientations of the part that have diameters that are beneath the resolution of the sensor are indistinguishable.

The discriminability of the sensors depends on their resolution and noise characteristics. By determining the range of sensor values consistent with each state, we can capture the effect of sensor resolution and sensor noise. Two states with overlapping sensor ranges are considered *indistinguishable*, and two states whose sensor ranges do not overlap are *distinguishable*. Using this model for the sensor, we can group states with overlapping sensor ranges.

DEFINITION 10. An **indistinguishable set** is a set of states where each state is indistinguishable from at least one other state in the set. We refer to a set of indistinguishable states with k elements as a k-indistinguishable set.

Note that some of the states in a k-indistinguishable set may be pairwise distinguishable. This is because the indistinguishability relation is not transitive due to sensor noise. Consider three states a, b, and c with increasing diameter values such that a and b are indistinguishable, and b and c are

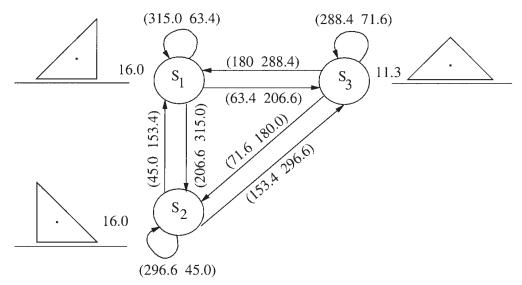


Fig. 10. State transition graph for an isosceles right triangle. The state transition graph is a directed graph whose edges correspond to action ranges and vertices correspond to stable orientations of the part.

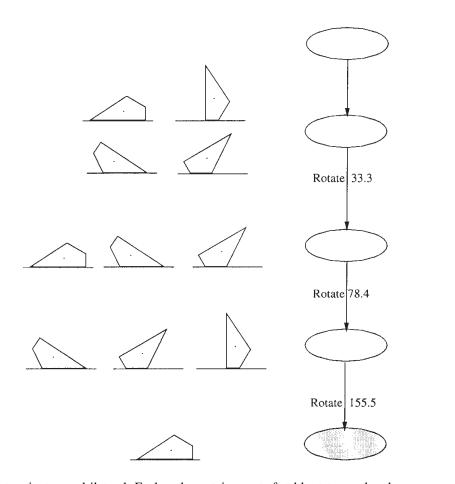


Fig. 11. Sensorless plan to orient a quadrilateral. Each node contains a set of stable states, and each arrow represents an action. The goal node, shown shaded, contains a single state.

indistinguishable. These states form a three-indistinguishable set. However a and c will be distinguishable if their diameter values differ sufficiently.

6. Sensor-Based Orienting

We now consider generating an orienting plan when the diameter of the part is measured after every push. When the part has a unique diameter value for every edge, a single push-align operation can determine the part orientation (Figure 12). When the part has several orientations with the same sensed diameter value, we need a plan consisting of a conditional sequence of push-align operations to orient the part. Branching during plan execution is determined by the sensed diameter value. The sensor data reduce the set of possible states to those consistent with the sensed value and the executed sequence of actions. For each k-indistinguishable set of states to be distinguished, we find the representative actions as described in Section 3.2. The number of states the robot eliminates after each action depends on the actual part orientation, the number of states with the same diameter value, and the applied action. Whether the part can be oriented uniquely depends on the characteristics of its push-diameter function. If the part has at least one diameter with a unique value or its pushdiameter function is aperiodic, it can be oriented uniquely. We will show that the length of the plan is never greater than 2m-1, where m is the maximum number of indistinguishable states.

We present two algorithms to find orienting plans that are *optimal* in the sense that they minimize the worst-case, or maximum, number of operations to orient a part. The first uses breadth-first AND/OR search, and the second is the bottom-to-top algorithm. Both are exponential-time algorithms that find optimal sensor-based orienting plans. We will show that sensor-based plans are usually shorter and never longer than sensorless plans. Note that other optimality criteria such as minimizing the expected number of orienting operations can be useful, particularly when the expected number of operations is significantly smaller than the worst-case number of operations. The expected number of operations depends on the initial distribution of part orientations and the shape of the part.

6.1. AND/OR Search

Our first method to find an orienting plan is to perform breadth-first AND/OR search. (See Rich and Knight (1991) for a description of AND/OR graphs and the similar AO* algorithm.) The root node corresponds to the set of all possible orientations of the part. A node in the search graph contains the set of orientations consistent with the push-align operations along the path from the root to the node. Each alternative (OR) push-align operation corresponds to a link. When a push-align operation is applied at a node, all stable orienta-

tions that can result are generated and classified into sets of indistinguishable states. The time to group k resulting states into indistinguishable sets, $T_g(k)$, is $O(k \log k)$ since it takes $O(k \log k)$ time to sort the states by diameter values and O(k) time to traverse the sorted list of states. The AND link from a node for a given operation points to a set of child nodes where each child node contains a set of indistinguishable states.

Search begins at the root node, which contains the set of all possible initial part orientations. The first push-align operation results in a set of nodes that are the indistinguishable sets of stable states. The search explores the nodes in a breadth-first manner by applying the representative actions for their constituent states to generate their child nodes. A node is solved if it is a goal node with a single state or when all its child nodes from an operation are solved. When a node is solved, its parent nodes are updated. Search terminates when the root node is solved or all nodes have been explored. The generated plan consists of a conditional sequence of operations to determine part orientation (Figure 13). The search process is exhaustive, but it need be performed off-line only once for each part.

6.2. Plan Length

The time taken to orient a part depends on the plan length, which is the worst-case number of operations required to orient the part. This depends on the part geometry, and in particular, the uniqueness of the diameter values of the stable states and the degree of symmetry of the part. We will use the term $diameter\ value$ to refer to the part diameter when the part is in a stable orientation on the fence and use it interchangeably with sensor value. We now prove bounds on the worst-case lengths of plans in terms of m, the maximum number of stable states with an indistinguishable diameter value, and n, the number of stable edges of the part, for the different polygon classes shown in Figure 14. The polygon classes are characterized by the following features:

- 1. All states having unique diameter values: Any state the part comes to rest in can be identified uniquely from its sensed diameter value. Therefore, a single step is sufficient to orient such a part.
- 2. Some states with unique diameter values: Group the states into indistinguishable sets. Assume that after the initial alignment, the part is in a state that belongs to an m-indistinguishable set. Consider performing an action of θ on all states in the indistinguishable set (Figure 15). Since the part has at least one state with a unique diameter value, there is some value of θ and at least one initial state for which the resulting state is a unique diameter valued state. This action reduces the number of possible states by at least one. In the worst case, we have remaining a set of m-1 states to be distinguished. We recursively continue this process on the resulting set of

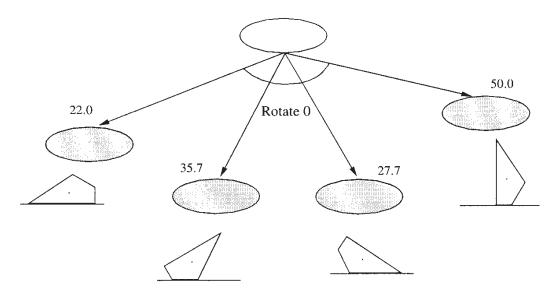


Fig. 12. A part with unique diameter values at its stable orientations can be oriented in a single step. The arrows linked by an arc represent a push-align operation. The diameter value of each stable edge is shown at its corresponding node. Compare with the four-step sensorless plan of Figure 11.

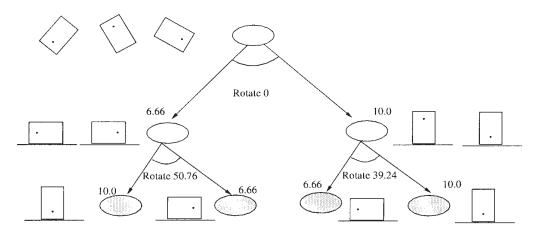
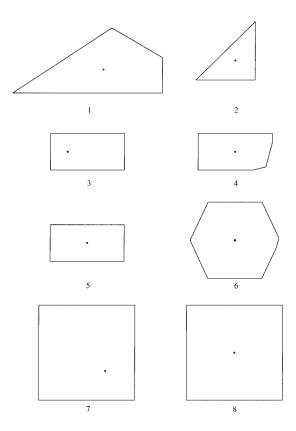


Fig. 13. A plan generated by the AND/OR search planner to orient the rectangle of Figure 3.



	Polygon class	Worst-case
		plan length
1.	All unique diameter values	1
2.	Some unique diameter values	m
3.	Multiple diameter values, none unique, aperiodic push-diameter function	m
4.	Multiple diameter values, none unique, quasisymmetric push-diameter function	2m-1
5.	Multiple diameter values, symmetric push-diameter function, symmetric push function	2m/p - 1
6.	Multiple diameter values, symmetric push-diameter function, asymmetric push function	2m-1
7.	Single diameter value, asymmetric push function (sensorless case)	2n-1
8.	Single diameter value, symmetric push function (sensorless case)	2n/p - 1

Fig. 14. Example polygons belonging to classes with different worst-case plan length characteristics. m is the maximum number of indistinguishable states, n is the number of stable edges of the polygon, and p is the number of periods of the push-diameter (or push) function.

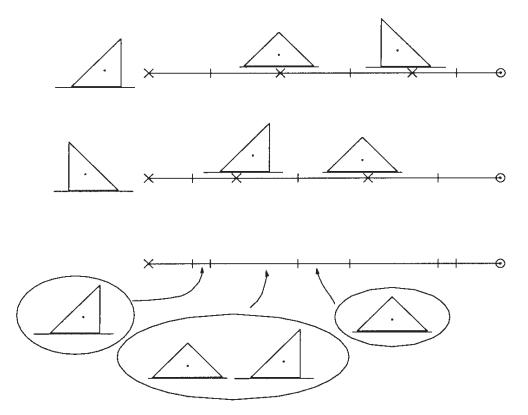


Fig. 15. Action and overlap ranges for indistinguishable states of the isosceles right triangle.

states, grouping them by diameter value, and for each indistinguishable set, eliminating at least one state with each action, until we are left with a single state. Since each action reduces the number of states by at least one, the maximum number of actions to orient the initial set of states is no greater than m. Since every other initial set of states is of size m or smaller, the worst-case number of steps to orient a part is m.

The number of steps to orient a part at execution time can range from 1 up to m, depending on the initial state. The actual maximum length of a plan depends on the part shape and can be less than m.

3. Multiple nonunique diameter values and an aperiodic push-diameter function: Assume that the initial state belongs to the largest indistinguishable set of size m. The overlap ranges of states with the same diameter value (Figure 16) always contain some action with distinguishable outcomes since the push-diameter function is aperiodic and has multiple diameter values. We can therefore reduce the size of the largest resulting indistinguishable set by at least one. This property, applied recursively to the resulting set of states, shows that the maximum number of steps to orient a part in this case is no greater than m.

The minimum number of steps to orient a part is always 2 or greater since there is no state with a unique diameter value.

4. Multiple nonunique diameter values and a quasisymmetric push-diameter function: The push-diameter function is quasi-symmetric, which means the stable orientations of some of the indistinguishable sets of states are not periodically spaced. First consider the case when no stable orientation is spaced an integer number of periods from another stable orientation. This implies there is always an action to distinguish any two indistinguishable states. Such a part can be oriented in a maximum of m steps.

The worst case occurs under the following conditions. Assume there is only one state that differs from its indistinguishable states in having an aperiodic stable orientation, and that this state belongs to an mindistinguishable set. Also assume the part is initially

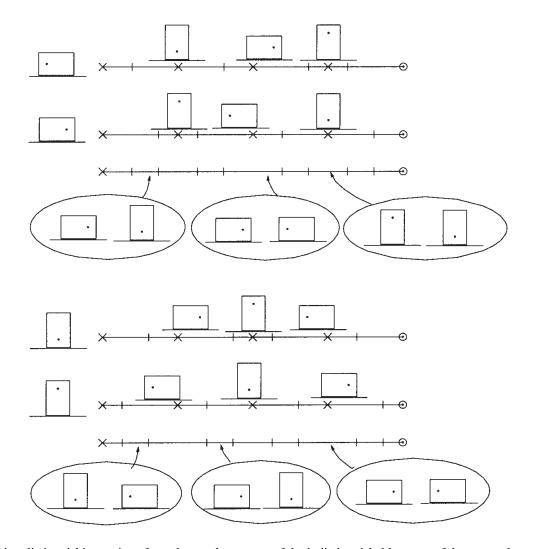


Fig. 16. Selecting distinguishing actions from the overlap ranges of the indistinguishable states of the rectangle.

in an m-indistinguishable set that does not include the aperiodic state. Execute the smallest action that causes a transition to the m-indistinguishable set that contains the aperiodic state. Since the action ranges of the aperiodic state differ from those of the other states, execute the smallest action that can distinguish it. If the part was in the aperiodic state, its orientation is now determined. Otherwise in the worst case, we have (m-1) remaining indistinguishable states. We recursively continue this process of eliminating each state from consideration by a two-step process. For m initial states, a maximum of 1 + 2(m-1), or 2m-1 steps can orient the part.

The minimum number of steps to orient a part is always 2 or greater since there is no state with a unique diameter value.

- 5. Multiple nonunique diameter values, a symmetric push-diameter function, and a symmetric push function: Let p be the number of periods of the push-diameter function. Since we can orient the part only up to symmetry, for each indistinguishable set of size k we effectively have k/p states to consider. We can transform our problem to the case of an asymmetric push-diameter function of period 360/p with n/p states. Such a push-diameter function may have all unique valued states, or some states with unique diameters, or multiple nonunique diameter valued states and be aperiodic or quasi-symmetric. It follows that the maximum number of steps to orient the part up to symmetry is 2m/p 1, and the minimum number of steps can be 1
- 6. Multiple diameter values, a symmetric push-diameter function, and an asymmetric push function: Since the push-diameter function is symmetric and the push function is asymmetric, there is at least one pair of neighboring indistinguishable states whose action ranges differ from a corresponding pair of states an integer number of periods away. Call this the asymmetric pair and this set of indistinguishable states the asymmetric set. Proving the length bound requires consideration of several cases. The worst case occurs when the part is initially in the asymmetric set, and it is an m-indistinguishable set. The strategy is to reach the asymmetric pair of states since for some action their transitions differ from the other pairs in the asymmetric set and can be used to distinguish the states. So the smallest action to get to one of the asymmetric states is executed, and then the smallest action that can distinguish that state from the corresponding states is performed. In the worst case, we will have (m-1) states remaining to be distinguished. This process is continued until all states are distinguished. Therefore, the maximum number of actions to determine the part orientation is 1 + 2(m-1), or 2m - 1.

The minimum number of steps to orient a part is always 3 or greater since there is no state with a unique diameter value and we need multiple actions to distinguish any state.

- 7. An asymmetric push function and all states with the same diameter value: When all states have the same diameter value, this is identical to the sensorless orienting problem since the sensor provides no useful information. Using a backchaining algorithm (Goldberg 1993), we can show that a plan always exists for any part with an asymmetric push function. The maximum length of the plan is 2n-1 steps (Chen and Ierardi 1995). Here the minimum number of steps and the worst-case number of steps to orient the part are identical.
- 8. A symmetric push function and all states with the same diameter value: This case again corresponds to the sensorless orienting case. Since there is symmetry in the push function, the part can be oriented only up to symmetry. If p is the number of periods of the push function, we can transform the problem to one with a push function of period 360/p and n/p states. The maximum number of steps to orient the part up to symmetry is then 2n/p 1. Here the minimum number of steps and the worst-case number of steps to orient the part are identical.

6.3. Completeness

The search-based planner is complete. For each indistinguishable set explored during search, the procedure to generate representative actions (Section 3.2) returns a set of actions that cover the action space. The search procedure of Section 6.1 guarantees that all representative actions of an explored indistinguishable set are applied to it. The planner performs breadth-first search, and the termination conditions ensure that the planner either finds a solution to orient the part uniquely when one exists or finds a solution to orient the part up to symmetry otherwise. The maximum depth of the search tree is 2m-1, where m is the size of the largest indistinguishable set. These conditions guarantee completeness of the planner. The search procedure is exponential in complexity.

6.4. Bottom-to-Top Algorithm

We now present a dynamic programming style algorithm to find the minimum length plan to orient a part, and analyze its worst-case time complexity. We find the shortest plan by first finding the best action for each possible indistinguishable set, and then identifying the shortest sequence of actions for each of the initial indistinguishable sets. The best action for an indistinguishable set minimizes the maximum action length of the resulting child indistinguishable sets. The action length of an indistinguishable set is the smallest number of actions guaranteed to distinguish it and is one plus the maximum action length of its child indistinguishable sets from its best action. The n_i states that share the *i*th sensed diameter value have 2^{n_i} corresponding indistinguishable sets. The algorithm begins by finding the best actions to distinguish the smallest indistinguishable sets, with 2 elements. (In case of a tie, select the action with a smaller rotation.) We find the best actions for all 2-indistinguishable sets over all sensor values. We then find the best actions for all 3-indistinguishable sets. We continue to sequentially work on larger indistinguishable sets until we find the best actions for all the indistinguishable sets. Since the child indistinguishable sets are smaller or of the same size as their parent indistinguishable set, by proceeding from the smallest to the largest indistinguishable sets, at each level we know the best actions for all the child indistinguishable sets. Once all the initial indistinguishable sets have been processed, we work backward to create a conditional plan using the best actions.

We assumed above that every indistinguishable set has some action that results in smaller sets to be distinguished. For some polygons and for some k, every k-indistinguishable set may not have an action that leads to smaller child indistinguishable sets. In such cases, we perform a second pass to select the best actions for the unresolved indistinguishable sets. We perform the second pass only when at least one kindistinguishable set has an action that leads to smaller indistinguishable sets. If no k-indistinguishable set has an action that leads to an indistinguishable set of smaller size, the part is orientable only up to symmetry.

Consider the n_i states corresponding to the *i*th sensed diameter value. A subset of k of these states forms a kindistinguishable set, with kn representative actions. The time taken to find the representative actions, $T_a(k, n)$, is $O(kn \log k)$. The result of an action for a given orientation of the part is determined by computing its orientation after the specified rotation, identifying the resting range to which it belongs, and determining the resulting orientation. This is an $O(\log n)$ operation for each action and each state. So the time taken to find the result of an action on k states, $T_r(k, n)$, is $O(k \log n)$. For each of the kn representative actions, compute the result in time $T_r(k, n)$ and group the resulting states in time $T_g(k)$. The comparison time to select the best action is O(kn). The time T(k) to find the best action for a *k*-indistinguishable set is $T_a(k, n) + (kn)(T_r(k, n))$ $+T_{\varrho}(k)$) + O(kn). The time to find the best actions for all k-indistinguishable sets with the ith sensed diameter value is $\{T_a(k,n) + (kn)[T_r(k,n) + T_g(k)] + O(kn)\}C_k^{n_i}$. When a second pass is required, the increase in the running time is bounded by a constant factor.

The time to compute the best actions over all subsets and all sensor values is

$$\sum_{i} \sum_{k=2}^{n_i} \{ T_a(k,n) + (kn)[T_r(k,n) + T_g(k)] + O(kn) \} C_k^{n_i}$$

$$= \sum_{i} \sum_{k=2}^{n_{i}} \{O(kn \log k) + kn(O(k \log n) + O(k \log k)) + O(kn)\} C^{n_{i}}$$

$$+ O(kn) \} C_k^{n_i}$$

$$= \sum_{i} O(nn_i^2 2^{n_i} \log n).$$

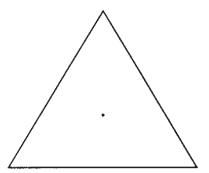
Since m is the maximum number of states with the same diameter value and the maximum number of indistinguishable sets of size k is n/k, the worst-case time complexity of the algorithm is $O(n^2m^2m\log n)$. See the example plan in Figure 2 generated by the implemented algorithm.

7. Are Sensorless and Sensor-Based Strategies **Equivalent?**

We have seen that sensor-based plans are usually shorter and never longer than sensorless plans. Are sensorless and sensorbased plans equivalent in their ability to orient any part? Can a sensorless system orient any part that a sensor-based system can orient? It turns out that a sensorless system may not be able to orient a part that a sensor-based system can orientadding a partial information sensor increases the power of the system. We illustrate with an example (Figure 17). Consider an equilateral triangle. It can be oriented only up to symmetry in its shape by both sensorless and sensor-based plans. Using a construction described in Theorem 4 (following in Section 8.2), we can generate another polygon with the same push function as the equilateral triangle but whose push-diameter function is not symmetric. A sensorless orienting plan will be able to orient the part only up to symmetry in its push function, whereas a sensor-based plan can bring the part to a distinct orientation. Thus sensor-based plans can not only reduce plan length but also orient certain parts that sensorless plans cannot.

8. Orienting and Recognizing Multiple Part **Shapes**

We now consider the problem of orienting any part from a known set of parts using the same plan. During product changeovers or when different parts are processed on the same line, parts with different shapes may have to be oriented by the same feeder (Figure 18). Consequently, we ask the following



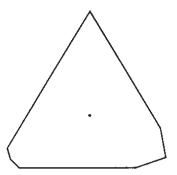


Fig. 17. The equilateral triangle on the left can be oriented only up to symmetry by both sensorless and sensor-based plans. The heptagon on the right has the same push function as the equilateral triangle, but its push-diameter function is not symmetric. Therefore, it can be brought to a distinct orientation using the sensor but oriented only up to symmetry in its push function without the sensor.

question: Can our parts-orienting system orient multiple part shapes with a single plan?

There are two aspects to this problem. The first is orienting the input part, which means bringing each part shape to a selected orientation. The second is recognizing the input part, which is identifying the shape that has been oriented. To orient and recognize multiple part shapes, we need to elucidate the relation between part shape and the radius, push, and pushdiameter functions.

8.1. Part Shape and the Radius Function

The rotational behavior of a pushed part depends on its shape and can be determined from its radius function. We therefore determine the properties of the radius function in relation to part shape. We prove that a radius function uniquely defines a convex polygon, where the polygon's shape is specified by the relative locations of its vertices and center of mass. Note that all nonconvex parts with the same convex hull have the same radius function.

A *valid* radius function is the radius function of some polygon, and is a piecewise sinusoidal function of period 360 degrees. Consider the variation in radius as the supporting line is rotated from an edge e_i to its counterclockwise neighboring edge e_{i+1} (Figure 19). Let the distance from the center of mass to their common vertex v_{i+1} be d_{i+1} and the orientation of the line joining the center of mass and the common vertex be δ_{i+1} . The radius r at an orientation ϕ is then given by $r = d_{i+1} \sin(\phi - \delta_{i+1}), \phi_i \le \phi \le \phi_{i+1}$. Changes in the sinusoids occur only at minima and kinks of the radius function. Minima correspond to stable edges, and kinks correspond to unstable and metastable edges of the part. This suggests that

we can recover the shape of a part from the minima and kinks of its radius function. A valid set of minima and kinks is a set of minima and kinks associated with a valid radius function.

LEMMA 1. Two valid radius functions are identical if they have the same set of minima and kinks.

Proof. It is sufficient to show that the orientations and radius values of the minima and kinks determine the value of the radius function everywhere. Each minimum or kink is specified by its orientation ϕ_i and radius r_i . Order the minima and kinks by their orientations and consider successive pairs. Each such minimum-minimum, minimum-kink, or kink-minimum pair specifies the endpoints of a sinusoidal segment of the radius function. Consider a pair $[(r_i, \phi_i), (r_{i+1}, \phi_{i+1})]$ corresponding to the endpoints of a sinusoidal segment. The radius r of the sinusoidal segment at an orientation ϕ is given by $r = d_{i+1} \sin(\phi - \delta_{i+1}), \phi_i \le \phi \le \phi_{i+1}$. So

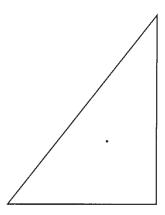
$$r_i = d_{i+1} \sin(\phi_i - \delta_{i+1})$$
 and $r_{i+1} = d_{i+1} \sin(\phi_{i+1} - \delta_{i+1})$.

We solve for d_{i+1} and δ_{i+1} to obtain

$$\delta_{i+1} = \tan^{-1} \left(\frac{r_i \sin \phi_{i+1} - r_{i+1} \sin \phi_i}{r_i \cos \phi_{i+1} - r_{i+1} \cos \phi_i} \right)$$

$$d_{i+1} = \frac{r_i}{\sin(\phi_i - \delta_{i+1})}.$$

From the polygon geometry, $|\phi_i - \phi_{i+1}| \le 180$, and this implies there is a unique solution for δ_{i+1} and d_{i+1} . Therefore, the pair $[(r_i, \phi_i), (r_{i+1}, \phi_{i+1})]$ uniquely defines a sinusoidal segment. We can perform this computation for each pair of successive minima and kinks to uniquely determine the radius function.



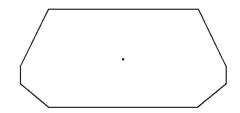


Fig. 18. Can these two parts be oriented using the same plan?

THEOREM 2. Two convex polygons are identical if and only if they have identical radius functions.

Proof. We show that a valid radius function is consistent with a single polygon. We present a proof in the form of a procedure to construct a convex polygon from a radius function. The edges of the polygon define the minima and kinks of the radius function, with every edge corresponding to a minimum or a kink. The polygon is defined by the relative locations of its vertices and center of mass. Choose a point as the center of mass (see Figure 20). The minima of the radius function give the orientations of the lines along which the stable edges lie and their distances from the center of mass. The kinks give the orientations and distances of the lines along which unstable and metastable edges of the part lie. Draw these lines, each of which contains a polygon edge. Since the edge orientations provide the ordered sequence of edges, compute the intersections of lines containing adjacent edges to find the vertices of the polygon. Each pair of adjacent lines intersects at a single point, and by construction the set of minima and kinks yields a single convex polygon. So there is a 1-to-1 mapping between a convex polygon and the set of minima and kinks of its radius function.

If the polygons have identical radius functions, they have identical sets of minima and kinks, and the polygons are identical by the above construction. To see the "only if" part, assume two different radius functions map to two identical polygons. Lemma 1 implies the two radius functions have different sets of minima and kinks. These sets of minima and kinks map to two polygons that are not identical, leading to a contradiction. Therefore, the two polygons are identical only if they have identical radius functions.

We now generalize Theorem 2 to proving that two convex polygons have the same radius function if and only if they have the same shape. That is, we wish to treat all polygons generated by rotating a given polygon as the same shape. Two parts P_A and P_B with radius functions $r_A(\phi)$ and $r_B(\phi)$ are said to have the same radius function if there exists an angle α such that $r_A(\phi) = r_B(\phi + \alpha)$. If $r_A(\phi) = r_B(\phi + \alpha)$, let the part P_A rotated counterclockwise by α degrees have the radius function $r_A'(\phi)$. Then $r_A'(\phi) = r_B(\phi)$. So without loss of generality, we assume that α is zero.

THEOREM 3. Two convex polygons have the same radius function if and only if they have the same shape.

Proof. The "if" part of the theorem is obvious. That is, two convex polygons have the same radius function if they have the same shape.

We prove the "only if" part using proof by contradiction. Assume there are two convex polygons P_A and P_B with radius functions $r_A(\phi)$ and $r_B(\phi)$, respectively, such that P_A and P_B have different shapes and $r_A(\phi)$ and $r_B(\phi)$ are identical radius functions. Using the radius function $r_A(\phi)$, construct P_A as described above. Similarly, construct P_B from $r_B(\phi)$. Since $r_A(\phi)$ and $r_B(\phi)$ are the same, the sets of minima and kinks of the radius functions for the two parts are the same, and the two constructions yield the same shape, leading to a contradiction. Therefore, two convex polygons with different shapes cannot have the same radius function.

Note that the radius function is a support function of a polygon, and prior results (Santaló (1976), for example) show that there is a 1-to-1 mapping between a smooth convex shape and its support function.

8.2. Part Shape and the Push Function

Since the push function is used to generate sensorless orienting plans, the relation between part shape and the push function determines if multiple part shapes can be oriented by the same plan. The push function of a convex polygon, in contrast to the radius function, does not map to a unique part shape. Different part shapes can have the same push function.

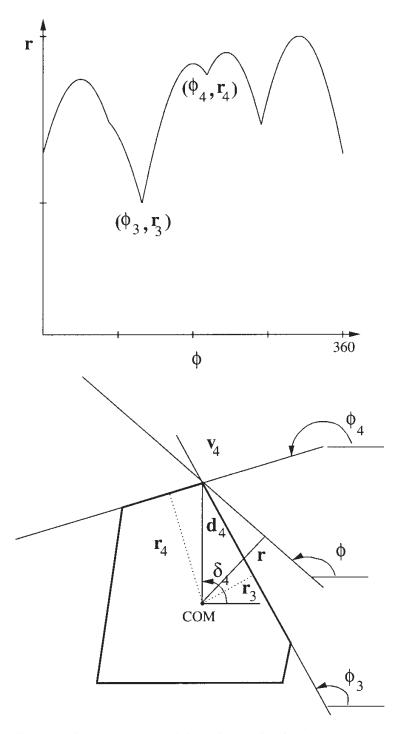


Fig. 19. Variation in the radius as the fence is rotated. Minima of the radius function correspond to stable edges, and kinks correspond to unstable and metastable edges. The radius function is uniquely determined by and can be reconstructed from the set of minima and kinks.

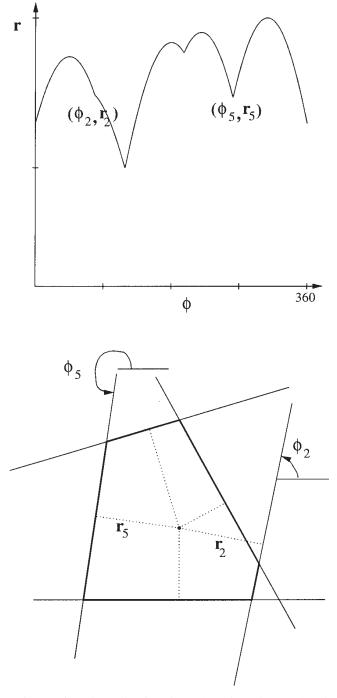


Fig. 20. Reconstructing the shape of a part from its radius function. The orientations and radius values of edges e_2 and e_5 are shown in the figure.

THEOREM 4. For every convex polygon, there exists an infinite set of nonsimilar convex polygons having the same push function

Proof. We present a constructive proof that, given any convex polygon, we can always generate another nonsimilar convex polygon having the same push function. The proof relies on showing that we can always generate a new transition vertex from any transition vertex of the given polygon without altering the push function. A *transition vertex* is a polygon vertex at which a local maximum of the radius function occurs; every polygon has at least one transition vertex. We show that we can generate an infinite set of new transition vertices for each original transition vertex, and can therefore generate an infinite set of convex polygons with the same push function.

Consider a transition vertex V of a polygon with incident edges e_i and e_j (Figure 21). Let the edges be defined by the line segments VE_i and VE_j , respectively. Draw the line segment VC that connects V to the polygon center of mass C. Draw the perpendicular lines to the edges e_i and e_j from C to obtain points P_i and P_j , respectively. If P_i is in the interior of the segment VE_i , then e_i is a stable edge. When the perpendicular does not intersect the edge in its interior, the edge is unstable, as in the case of P_j and e_j , or metastable. A local maximum of the radius function occurs at vertex V when the supporting line is perpendicular to VC. Call this line l_{\perp} .

Our construction splices in two new unstable edges to the convex polygon that meet at a new transition vertex such that the modified convex polygon has the same push function as the given polygon. Pick a point Q_i in the interior of both VP_i and VE_i . Draw a line perpendicular to CQ_i through Q_i to intersect VC at the point R_i . R_i lies in the interior of VC since P_i and E_i lie on the same side of the line I_{\perp} . Any line segment with one endpoint in the interior of the segment VR_i and with Q_i as the other endpoint will form an edge whose included angle with CQ_i is greater than 90°. Such an edge is unstable since the perpendicular projection of the center of mass C cannot lie in its interior. A similar construction gives the points Q_j and R_j for the edge e_j , and another set of line segments that can form unstable edges.

We now pick a point T in the interior of the segment formed by the intersection of VR_i and VR_j to be the new transition vertex. The line segments TQ_i and TQ_j identify the new unstable edges u_i and u_j , respectively. Adding these unstable edges to the polygon does not change the stable orientations of the polygon and the push function.

We must also prove that there is no change in the divergent orientations of the push function, which correspond to local maxima of the radius function. When the supporting line is in contact with T, the maximum radius value occurs when the line is perpendicular to CT. This line l_{\perp}' is parallel to l_{\perp} , and so this maximum occurs at the same orientation as the original divergent orientation corresponding to vertex V. To establish

that T is a transition vertex with this divergent orientation, we prove that this maximum radius value, length(CT), is the locally maximal radius of the modified polygon. We show that length(CT) is greater than the radius values of the edges e_i , e_j , u_i , and u_j , and that the radius values of edges u_i and u_j are greater than those of edges e_i and e_j , respectively.

We prove the radius is locally maximal at vertex T with respect to edges e_i and u_i (see Figure 22). The radius value of edge e_i , r_i , is equal to length(CP_i). Let the radius of edge u_i be r_{u_i} . The radius of segment Q_iR_i is length(CQ_i). Consider a line rotating about the point Q_i , with its initial orientation along Q_iR_i , the perpendicular to the line CQ_i . The radius of the line is proportional to the cosine of the angle it is rotated by, with the maximum at its initial orientation. Therefore, length(CQ_i) > r_{u_i} > r_i . From ΔCQ_iT , length(CT) > length(CQ_i), which implies length(CT) > r_{u_i} > r_i . Similarly, we can show length(CT) > r_{u_j} > r_j . Therefore, the vertex T corresponds to a local maximum of the radius function.

The new vertex T is a transition vertex while the vertices Q_i and Q_j are not. Therefore, the new polygon with vertices Q_i , T, and Q_j replacing the vertex V is a convex polygon with the same push function as the given convex polygon. Since there is an infinite set of choices for Q_i , Q_j , and T, we can generate an infinite set of polygons having the same push function as the original polygon.

See Figure 23 for an example of nonsimilar parts with the same push function. The part at the bottom was constructed from the part at the top using the above construction. The following corollary is a direct consequence of Theorem 4.

COROLLARY 5. A sensorless plan to orient a given polygon can orient the infinite set of polygons sharing the same push function.

8.3. Part Shape and the Push-Diameter Function

When orienting multiple part shapes, we would like to know if differences in part shape always lead to recognizable differences in part behavior. We show that different part shapes can have the same push-diameter function.

Two push-diameter functions d_1 and d_2 are considered the *same* if all the following conditions are satisfied, and they are considered *different* otherwise:

- 1. $\exists \alpha$ such that $d_1(\phi) = d_2(\phi + \alpha), 0 \le \alpha < 360$.
- 2. $\forall \phi_{s1} \in S_1, \exists \phi_{s2} \in S_2$ such that $\phi_{s2} = (\phi_{s1} + \alpha) \mod 360$, where S_1 and S_2 are the sets of stable orientations of d_1 and d_2 , respectively.
- 3. $\forall \phi_{s1} \in S_1, \exists \phi_{s2} \in S_2$ such that $\kappa(\phi_{s1}) \phi_{s1} = \kappa(\phi_{s2}) \phi_{s2}$ and $\phi_{s1} \omega(\phi_{s1}) = \phi_{s2} \omega(\phi_{s2})$, where $\kappa(\phi_{si})$ and $\omega(\phi_{si})$ are the counterclockwise and clockwise divergent orientations corresponding to ϕ_{si} .

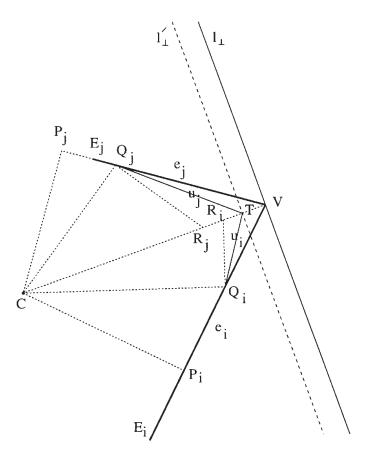


Fig. 21. Generating a new polygon with the same push function as the given polygon. The edges e_i and e_j of the original polygon, drawn bold, meet at the transition vertex V. The construction splices in unstable edges u_i and u_j so the new vertices Q_i and Q_j and transition vertex T replace vertex V.

These conditions state that two parts are considered to have the same push-diameter function if their push-diameter functions are phase-shifted copies of each other, with the stable states of the two parts having the same phase-shifted orientations and their phase-shifted divergent orientations specifying identical resting ranges for the stable states. For the remainder of the paper, we assume without loss of generality that the phase shift α is zero.

THEOREM 6. For every convex polygon, there exists an infinite set of convex polygons having the same push-diameter function.

Proof. For two parts to have the same push-diameter function, the orientations of their stable edges and the divergent orientations specified by their transition vertices must be identical. The identical diameter values of the stable edges also impose constraints on the vertices defining the diameters. Our proof is constructive and revolves around modifying one or more transition vertices or diameter-determining vertices of the given polygon. A *diameter-determining vertex* of a sta-

ble edge is the vertex with the largest perpendicular distance to the edge. Each stable edge of a convex polygon has one or two diameter-determining vertices. Every polygon has at least one transition vertex. We classify convex polygons as follows.

- 1. Polygons with at least one transition vertex that is not also a diameter determining vertex. We can use the construction of Theorem 4 to modify this vertex and generate an infinite set of polygons with the same push function as the original polygon. Since the modified transition vertex is not diameter determining, no stable edge diameters change and these polygons have the same push-diameter function as the original polygon.
- Polygons whose transition vertices are all diameterdetermining vertices. We classify these polygons as follows.
 - Polygons with every stable edge having a single diameter-determining vertex. If the polygon is

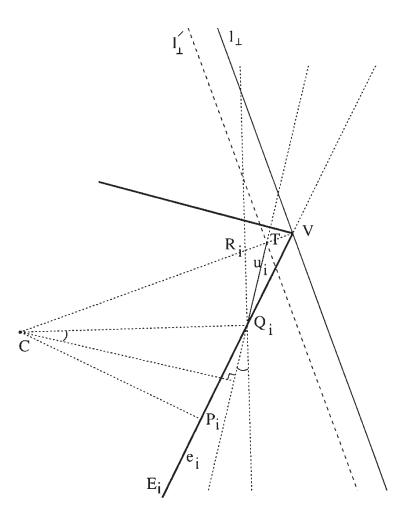


Fig. 22. Establishing that a local maximum of the radius function occurs at the new transition vertex T.

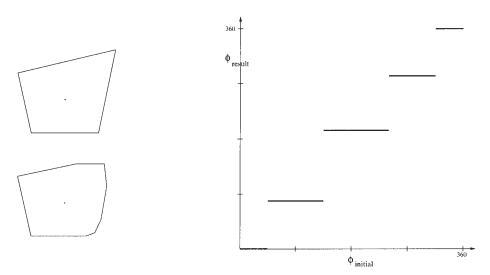


Fig. 23. Two parts with the same push function. The push-diameter functions of the two parts differ.

a regular polygon with its center of mass at its geometric center, use the construction of Theorem 4 to modify every vertex identically to generate another polygon with an identical push function and a push-diameter function that is a shrunken copy of the original push-diameter function. Now uniformly expand the modified polygon by the required amount to obtain a polygon with the same push-diameter function as the original polygon. See example in Figure 24.

In the general case, not all polygon edges are stable and not all diameter-determining vertices are transition vertices. Given a nontransition vertex, we can splice in new unstable edges to create a new nontransition vertex without modifying the push function, using a similar construction to that in Theorem 4. We use the constructions for transition and nontransition vertices to modify every diameter-determining vertex so that the diameter of each stable edge is reduced by the same percentage. Now expand the truncated polygon by the desired percentage so that the new polygon has the same push-diameter function as the original polygon. We can thus generate an infinite set of polygons with the same push-diameter function.

• Polygons with at least one stable edge whose diameter is determined by two vertices. Since the diameter-determining vertices of the chosen stable edge are equidistant from the stable edge, they are the vertices of a parallel diameter-determining edge. This diameter-determining edge may be stable, unstable, or metastable. When either of the diameter-determining vertices determines the diameter of only the chosen stable edge, it can be modified using the appropriate construction for a transition or nontransition vertex. Since a vertex may be diameter determining for more than one edge, we have to consider all possible cases.

The chosen stable edge and its parallel diameter-determining edge will be called the diameter-determining pair. The clockwise (CW) neighboring edge of the stable edge and the CW neighbor of the diameter-determining edge form a pair (CW neighboring pair). Similarly, the counterclockwise (CCW) neighboring edges of the stable edge and the diameter-determining edge form a CCW neighboring pair. For a given neighboring edge pair, let the interior angle between the chosen stable edge and its neighboring edge from the pair be σ and the interior angle between the diameter-determining edge and its neighboring edge from the pair be δ .

- (a) The neighboring edge pairs of the diameterdetermining pair consist of parallel edges. The diameter-determining pair may have one or two pairs of neighboring parallel edges. Select a diameter-determining vertex belonging to one of the pairs of neighboring parallel edges. Using the appropriate construction for a transition or nontransition vertex, this vertex can be modified without altering the push function. This vertex is diameter determining for the chosen stable edge, and its neighboring edge from the selected pair if it is stable. Since both these edges have two diameter-determining vertices, the pushdiameter function does not change when the vertex is modified. See Figure 25 for two example parts in this class that share the same push-diameter function.
- (b) The diameter-determining pair has at least one neighboring edge pair that consists of nonparallel edges. Consider the neighboring edge pair that consists of nonparallel edges. If both pairs have nonparallel edges, compute σ and δ for each and select the pair with the minimum value of $(\sigma \delta)$.

If $\sigma < \delta$, select the diameter-determining vertex that belongs to the diameter-determining edge and its neighboring edge from the selected pair. From the geometry of the convex polygon, this vertex can be diameter determining for only the stable edge and can be modified without affecting the push and push-diameter functions. If σ > δ , select the vertex belonging to the stable edge and its neighbor from the selected pair. From the polygon geometry, it is either not a diameter-determining vertex or is a diameterdetermining vertex only for stable instances of the diameter-determining edge. In either case, we can modify it without changing the push-diameter function.

For each case, we can generate an infinite set of polygons with the same push-diameter function by modifying the appropriate vertex.

Hence there is an infinite set of polygons with the same push-diameter function as the original polygon. \Box

Since different part shapes can share the same pushdiameter function, given a valid push-diameter function we cannot uniquely reconstruct the part from which it was generated. The following corollary is a direct consequence of Theorem 6.

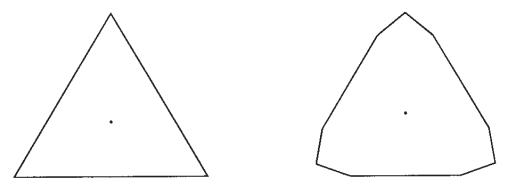


Fig. 24. A regular polygon modified to generate another polygon with the same push-diameter function.

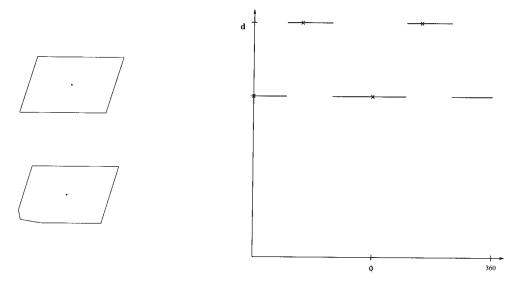


Fig. 25. Two parts with the same push-diameter function. Their push functions are also identical.

COROLLARY 7. A sensor-based plan to orient a given polygon can orient the infinite set of polygons sharing the same push-diameter function.

Let S_{pf}^P be the set of parts with the same push function as a part P and let S_{pdf}^P be the set of parts with the same push-diameter function as the part P. Since two parts with the same push-diameter function share the same push function, $S_{\mathrm{pdf}}^P \subseteq S_{\mathrm{pf}}^P$. Any sensorless plan that can orient part P can orient any part $Q \in S_{\mathrm{pf}}^P$, and any sensor-based plan that can orient part P can orient any part P can orient a

8.4. Orientability and Recognizability of Multiple Parts

We wish to orient and recognize any part from a known set of parts using a single plan. Theorems 4 and 6 show that parts with different radius functions can share the same push function and can also share the same push-diameter function. Therefore, when orienting multiple part shapes, differences in part shape do not guarantee that the parts can be identified. When can we guarantee that any part from a known set of parts can be oriented and recognized using the same plan? Here we identify conditions for a set of parts to be orientable (up to symmetry in their push-diameter functions) and recognizable. We begin by considering pairs of parts and then extend our results to a set of parts.

DEFINITION 11. Two parts are said to be *distinguishable* if there exists an action sequence such that the output strings of stable diameter values for the two parts differ.

LEMMA 8. Two parts that have the same push-diameter function are orientable but are not distinguishable.

Proof. Since the two parts have the same push-diameter function, their action ranges are identical and hence their mechanical behavior is identical. The diameter values of corresponding states are also identical. Therefore, there is no action sequence that can distinguish the parts. However, the same orienting plan can be used for both parts.

When can two parts be oriented and distinguished? From Lemma 8, a necessary condition is that their push-diameter functions be different. When two parts with different push-diameter functions have the same diameter value for all their stable states, the sensor provides zero information and this is equivalent to the sensorless case. Even if we can orient the parts, we cannot tell them apart. A simple example is two squares of the same size with their centers of mass at different points. Therefore, a second necessary condition is that either the parts have different diameter values or at least one of the parts has multiple distinct diameter values. A sufficient condition for two parts to be orientable and distinguishable is that each part have at least one unique diameter value.

THEOREM 9. Two parts with different push-diameter functions are always orientable and distinguishable if |D| > 1, where D is the set of distinct diameter values of all stable orientations of the two parts.

Proof. Since |D| > 1, there are multiple distinct diameter values. When one or both parts have some diameter values unique to them, we can distinguish the parts by bringing them to the corresponding stable orientations.

Consider the other case when both parts have the same set of multiple diameter values. Since the push-diameter functions of the two parts differ, there is at least one pair of states whose action ranges differ. The most difficult distinguishability case occurs when the two parts have almost identical push-diameter functions. We will treat the case when each part has the same two diameter values (see Figure 26); the argument can be extended to parts with more than two identical diameter values. It is hard to tell the parts apart because every state in one part is paired with a corresponding state in the other part. Since the action ranges of at least one pair of states differ, there is some action for which the parts transition from this pair of states to two states that are not paired. This causes the parts to go "out of phase." Once the two parts are in states that are not paired, there is an action sequence guaranteed to bring the parts to distinguishable states.

LEMMA 10. Distinguishability of parts is not transitive.

Proof. We prove this with an example. Consider three parts P_A , P_B , and P_C . Let the push-diameter functions of parts P_A and P_B differ such that the parts are orientable and distinguishable. Similarly, let parts P_B and P_C be orientable and distinguishable. However, parts P_A and P_C can have the same push-diameter function, in which case they are not distinguishable.

DEFINITION 12. A set of known parts *P* is said to be *recognizable* if there exists an action sequence to distinguish every member of the set from every other member of the set.

Lemma 10 implies that a necessary condition for any set of parts to be recognizable is that every pair of parts be distinguishable.

THEOREM 11. For a given set of parts P, if every pair of parts is orientable and distinguishable, then any subset of parts $S \subseteq P$ is orientable and recognizable.

Proof. Since every pair of parts can be oriented and distinguished, no two parts have the same push-diameter function. Therefore, each part has a state whose action range differs from at least one state in every other part. As in Theorem 9, we can show that an action sequence always exists to rotate parts "out of phase" with the others one at time and use this to recognize them.

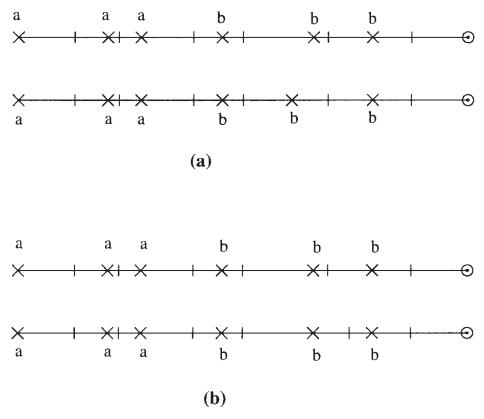


Fig. 26. Resting ranges of two parts with almost identical push-diameter functions. All states with the same character are indistinguishable states. In both cases, the parts can be distinguished. (a) Resting ranges of the two parts are identical, but stable orientations differ. The two parts have one pair of states with different action ranges. (b) Resting ranges of the two parts differ, but stable orientations are identical. The action ranges of every corresponding pair of states differ.

8.5. Sensor-Based Orienting and Recognition of Multiple Parts

We have extended the bottom-to-top planner of Section 6.4 to generate orienting plans for multiple parts by making two modifications. First, a state is now encoded by the part it belongs to in addition to its stable orientation and diameter value. Second, indistinguishable sets now consist of states with the same sensed diameter value over all parts. The representative actions are found from overlap ranges generated using the action ranges of states belonging to these indistinguishable sets. An example plan is shown in Figure 27.

8.6. Sensorless Orienting of Multiple Parts

A natural question is, can multiple parts be oriented by sensorless strategies? The answer is yes for certain cases. When a set of parts all have the same push function, the same plan can orient all the parts. By Theorem 4, for every part there exists an infinite set of parts with the same push function, and they can all therefore be oriented by the same plan. See Fig-

ure 23 for two example parts. We cannot recognize the parts, however.

For parts with different push functions, a breadth-first search planner can be used to find a sensorless plan when one exists. An alternative approach is to generate individual plans for each part in the input set and to execute them sequentially. As long as all actions are chosen to be deterministic for all the parts, at the end of the executed sequence each part shape will be in a known orientation. However, such plans will be long, and we need subsequent sensing operations to identify the oriented part.

9. Implementation

We have implemented planners to generate orienting plans for convex polygonal parts using AND/OR search and the bottom-to-top algorithm in Common Lisp. Given a part shape and sensor noise, the planners return a plan to orient the part uniquely when possible, and up to symmetry otherwise. These planners have been tested on several parts including

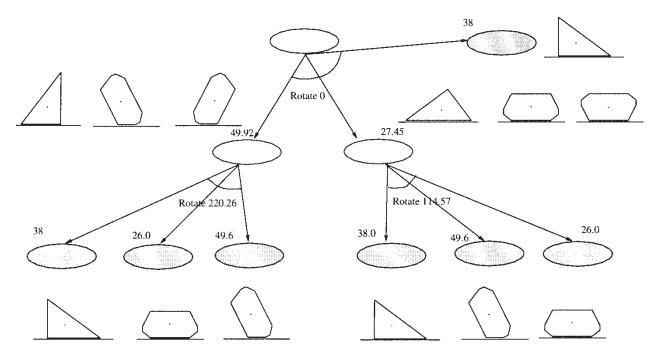


Fig. 27. A sensor-based plan to orient two part shapes. It was generated by a modified version of the bottom-to-top planner. For clarity, we indicate the range of sensor values corresponding to each set of states by the average of the sensor values.

those shown in Figure 28. Example plans generated by the AND/OR search planner and bottom-to-top planner are shown in Figure 13 and Figure 2, respectively.

We timed the AND/OR search planner and bottom-to-top planner on a Sparc ELC. Traversing top to bottom and left to right in Figure 28, the AND/OR search planner took an average of 0.064 secs, 0.116 secs, 0.096 secs, 0.042 secs, 0.084 secs, and 0.142 secs, respectively, to generate a sensorbased plan. The bottom-to-top planner took an average of 0.060 secs, 0.112 secs, 0.078 secs, 0.048 secs, 0.096 secs, and 0.210 secs for the same parts.

We have implemented and demonstrated orienting of singulated parts using an Adept 550 robot and a conveyor belt. To pick up a part, the robot uses a suction cup. We implemented both sensor-based and sensorless orienting plans for four parts. The parts and the fence were made of delrin. We selected the two parts of Figure 18 for further experimentation and ran 20 trials using a sensor-based plan and 10 trials using a sensorless plan. For the right triangle, 17 of the 20 trials with the sensor-based plan and 8 of the 10 trials with the sensorless plan succeeded. For the 8-gon, 19 of the 20 trials with the sensor-based plan and all 10 trials of the sensorless plan succeeded. All observed failures occurred when the suction cup made insufficient surface contact with the part to pick it up. Since the robot picks up a part at an unknown point on the part, rotating the part causes bounded but unknown changes

in the position of the part along the fence. To a lesser extent, the part also slides along the fence. These variations in part position along the fence occasionally lead to the observed failures. The frequency of failures for a part depends on the sequence of rotations in the plan. An array of suction cups can be used to eliminate such failures.

We also tested the sensor-based plan of Figure 27 to orient and recognize the parts of Figure 18. We ran 20 trials, 10 with each part. The triangle was successfully oriented and recognized all 10 times, and the 8-gon was successfully oriented and recognized 9 times. The single failure occurred due to a pickup failure with the suction cup.

10. Conclusion

In this paper, we characterized the advantages of using simple and inexpensive LED sensors with manipulation operations for parts orienting. We showed that the sensors reduce the number of orienting steps from O(n) for sensorless orienting to O(m), where n is the number of stable states and m is the maximum number of indistinguishable states. This reduces execution time and the number of stages in a pipelined orienting process. Sensor-based plans are also more powerful than sensorless plans in that they can orient a larger class of parts. Furthermore, the sensors permit a feeder to orient multiple

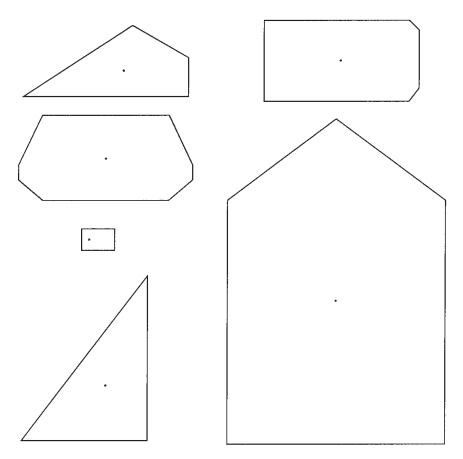


Fig. 28. Sample part shapes on which planners were tested.

part shapes with a single plan, leading to increased flexibility.

We presented implemented AND/OR search and bottomto-top algorithms to find minimum length plans. They are both exponential-time planners but need be run off-line only once for each new part. We characterized the relation between part shape, orientability, and recognizability to identify conditions under which a single plan can orient and recognize multiple part shapes. Unlike a sensorless plan, the execution time of a sensor-based plan may depend on the initial part orientation. Nevertheless, the use of sensors can significantly increase throughput since the maximum execution time for sensor-based orienting is usually shorter than that for sensorless orienting and never greater.

Our results can potentially be extended to other orienting tasks such as the fence-orienting system of Wiegley et al. (1996), the sensorless 1JOC single-degree-of-freedom fence of Akella et al. (1997), and the parallel-jaw grasping work of Rao and Goldberg (1994). For example, sensing of part diameter can substantially speed up the parts orienting process and help in the orienting of different parts.

In this paper, we presented planning algorithms to orient a single part shape, a finite set of known part shapes, and an infinite set of part shapes with identical mechanical behavior. Industrial parts are manufactured to tolerances, and orienting a toleranced part involves generating a plan for an infinite set of valid part shapes that do not have identical mechanical behavior. We have therefore extended this work to incorporate the effects of shape uncertainty and shown that parts can be oriented despite the nondeterminism introduced by bounded shape variations (Akella 1996; Akella and Mason 1998b).

Important extensions include treating more general part shapes, such as 3-D parts and parts with curved edges, and identifying techniques to singulate parts. It would be interesting to characterize the completeness and execution length properties of a broader set of orienting tasks that use nondeterministic actions. It would also be useful to characterize the optimal sensor spacing to orient a set of parts, and to develop planners that minimize the expected length of plans when probability distributions of the initial part orientations are available.

Acknowledgments

Funding was provided by NSF under grants IRI-9213993 and IRI-9318496 (with ARPA). Additional support for Srinivas Akella was provided by the Beckman Institute. This work benefited from discussions with Mike Erdmann, Ken Goldberg, Reid Simmons, and Dafna Talmor.

References

- Akella, S. 1996. Robotic manipulation for parts transfer and orienting: Mechanics, planning, and shape uncertainty.
 Ph.D. thesis, The Robotics Institute, Carnegie Mellon University. Robotics Institute Technical Report CMU-RI-TR-96-38.
- Akella, S., Huang, W. H., Lynch, K. M., and Mason, M. T. 1997. Sensorless parts orienting with a one-joint manipulator. In *IEEE International Conference on Robotics* and Automation, Albuquerque, NM, April, 2383–2390.
- Akella, S., and Mason, M. T. 1998a. Parts orienting with partial sensor information. In *IEEE International Con*ference on Robotics and Automation, Leuven, Belgium, May, 557–564.
- Akella, S., and Mason, M. T. 1998b. Parts orienting with shape uncertainty. In *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May, 565–572.
- Alevizos, P., Boissonnat, J. D., and Yvinec, M. 1989. On the order induced by a set of rays: Application to the probing of nonconvex polygons. In *IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, May, 202–207.
- Arkin, E. M., Held, M., Mitchell, J.S.B., and Skiena, S. S. 1998. Recognizing polygonal parts from width measurements. *Computational Geometry: Theory and Applications* 9(4):237–246.
- Balorda, Z., and Bajd, T. 1994. Reducing positioning uncertainty of objects by robot pushing. *IEEE Transactions on Robotics and Automation* 10(4):535–541.
- Barraquand, J., and Latombe, J.-C. 1993. Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. *Algorithmica* 10(2/3/4):121–155.
- Berkowitz, D. R., and Canny, J. F. 1996. Designing parts feeders using dynamic simulation. In *IEEE International Conference on Robotics and Automation*, Minneapolis, MN, April, 1127–1132.
- Böhringer, K. F., Bhatt, V., and Goldberg, K. Y. 1995. Sensorless manipulation using transverse vibrations of a plate. In *IEEE International Conference on Robotics and Automation*, Nagoya, Japan, May, 1989–1996.
- Boothroyd, G., Poli, C., and Murch, L. E. 1982. Automatic

- Assembly. New York: Marcel Dekker.
- Boothroyd, G., Redford, A. H., Poli, C., and Murch, L. E. 1972. Statistical distributions of natural resting aspects of parts for automatic handling. *Manufacturing Engineering Transactions, Society of Manufacturing Automation* 1:93–105.
- Brokowski, M., Peshkin, M., and Goldberg, K. Y. 1993. Curved fences for part alignment. In *IEEE International Conference on Robotics and Automation*, Atlanta, GA, May, 3:467–473.
- Brost, R. C. 1988. Automatic grasp planning in the presence of uncertainty. *International Journal of Robotics Research* 7(1):3–17.
- Brost, R. C. 1991. Analysis and planning of planar manipulation tasks. Ph.D. thesis, School of Computer Science, Carnegie Mellon University, January. Technical Report CMU-CS-91-149.
- Brost, R. C. 1992. Dynamic analysis of planar manipulation tasks. In *IEEE International Conference on Robotics and Automation*, Nice, France, May, 2247–2254.
- Brost, R. C., and Goldberg, K. Y. 1996. A complete algorithm for designing planar fixtures using modular components. *IEEE Transactions on Robotics and Automation* 12(1):31–46.
- Caine, M. 1994. The design of shape interactions using motion constraints. In *IEEE International Conference* on Robotics and Automation, San Diego, CA, May, 366– 371.
- Canny, J. F., and Goldberg, K. Y. 1994. RISC for industrial robotics: Recent results and open problems. In *IEEE International Conference on Robotics and Automation*, San Diego, CA, May, 1951–1958.
- Chen, Y.-B., and Ierardi, D. J. 1995. The complexity of oblivious plans for orienting and distinguishing polygonal parts. *Algorithmica* 14:367–397.
- Christiansen, A. D., Edwards, A. D., and Coello, C.A.C. 1996. Automated design of parts feeders using a genetic algorithm. In *IEEE International Conference on Robotics and Automation*, Minneapolis, MN, April, 846–851.
- Cole, R., and Yap, C. K. 1987. Shape from probing. *Journal of Algorithms* 8:19–38.
- Deacon, G. E., Low, P. L., and Malcolm, C. 1993. Orienting objects in a minimum number of robot sweeping motions. In *Sixth International Conference on Applications of Artificial Intelligence and Expert Systems*, Edinburgh, Scotland, June.
- Donald, B. R., Jennings, J., and Rus, D. 1995. Information invariants for distributed manipulation. In *Algorithmic Foundations of Robotics*, K. Y. Goldberg, D. Halperin, J.-C. Latombe, and R. H. Wilson, eds., 431–457. Wellesley, MA: A. K. Peters.

- Ellis, R. E. 1992. Planning tactile recognition paths in two and three dimensions. *International Journal of Robotics Research* 11(2):87–111.
- Eppstein, D. 1990. Reset sequences for monotonic automata. *SIAM Journal of Computing* 19(3):500–510.
- Erdmann, M. 1995. Understanding action and sensing by designing action-based sensors. *International Journal of Robotics Research* 14(5):483–509.
- Erdmann, M. 1996. An exploration of nonprehensile twopalm manipulation: Planning and execution. In *Robotics Research: The Seventh International Symposium*, G. Giralt and G. Hirzinger, eds., 16-27. London, UK: Springer.
- Erdmann, M., and Mason, M. T. 1988. An exploration of sensorless manipulation. *IEEE Journal of Robotics and Automation* 4(4):369–379.
- Erdmann, M., Mason, M. T., and Vanecek, G. 1993. Mechanical parts orienting: The case of a polyhedron on a table. *Algorithmica* 10(2/3/4):201–225.
- Gaston, P. C., and Lozano-Perez, T. 1984. Tactile recognition and localization using object models: The case of polyhedra on a plane. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6(3):257–266.
- Goldberg, K. Y. 1993. Orienting polygonal parts without sensors. *Algorithmica* 10(2/3/4):201–225.
- Goldberg, K. Y. 1995. Completeness in robot motion planning. In *Algorithmic Foundations of Robotics*, K. Y. Goldberg, D. Halperin, J.-C. Latombe, and R. H. Wilson, eds., 419–429. Wellesley, MA: A. K. Peters.
- Govindan, R., and Rao, A. S. 1994. Optimal strategies for recognizing polygonal parts. In *IEEE International Conference on Robotics and Automation*, San Diego, CA, May, 3564–3569.
- Grimson, W.E.L., and Lozano-Perez, T. 1984. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research* 3(3):3–35.
- Grossman, D. D., and Blasgen, M. W. 1975. Orienting mechanical parts by computer-controlled manipulator. *IEEE Transactions on Systems, Man, and Cybernetics* 5(5):561–565.
- Jia, Y.-B., and Erdmann, M. 1995. The complexity of sensing by point sampling. In *Algorithmic Foundations of Robotics*, K. Y. Goldberg, D. Halperin, J.-C. Latombe, and R. H. Wilson, eds., 283-300. Wellesley, MA: A. K. Peters.
- Kohavi, Z. 1978. *Switching and Finite Automata Theory*. 2d ed. New York: McGraw-Hill.
- Krishnasamy, J., Jakiela, M. J., and Whitney, D. E. 1996. Mechanics of vibration-assisted entrapment with application to design. In *IEEE International Conference on Robotics and Automation*, Minneapolis, MN, April, 838–845.

- Li, Z., and Canny, J. 1990. Motion of two rigid bodies with rolling constraint. *IEEE Transactions on Robotics and Automation* 6(1):62–72.
- Mani, M., and Wilson, W.R.D. 1985. A programmable orienting system for flat parts. In *North American Manufacturing Research Institute Conference XIII*, Berkeley, CA, May.
- Mason, M. T. 1986. Mechanics and planning of manipulator pushing operations. *International Journal of Robotics Research* 5(3):53–71.
- Mirtich, B., Zhuang, Y., Goldberg, K., Craig, J., Zanutta, R., Carlisle, B., and Canny, J. 1996. Estimating pose statistics for robotic part feeders. In *IEEE International Conference on Robotics and Automation*, Minneapolis, MN, April, 1140–1146.
- Natarajan, B. K. 1989. Some paradigms for the automated design of parts feeders. *International Journal of Robotics Research* 8(6):98–109.
- Peshkin, M. A., and Sanderson, A. C. 1988a. Planning robotic manipulation strategies for workpieces that slide. *IEEE Journal of Robotics and Automation* 4(5):524–531.
- Peshkin, M. A., and Sanderson, A. C. 1988b. The motion of a pushed sliding workpiece. *IEEE Journal of Robotics and Automation* 4(6):569–598.
- Rao, A. S., and Goldberg, K. Y. 1994. Shape from diameter: Recognizing polygonal parts with a parallel-jaw gripper. *International Journal of Robotics Research* 13(1):16–37.
- Rao, A. S., and Goldberg, K. Y. 1995. Manipulating algebraic parts in the plane. *IEEE Transactions on Robotics and Automation* 11(4):598–602.
- Rich, E., and Knight, K. 1991. *Artificial Intelligence* 2d ed. New York: McGraw-Hill.
- Riley, F. J. 1983. *Assembly Automation: A Management Handbook*. New York: Industrial Press.
- Sanderson, A. C. 1984. Parts entropy methods for robotics assembly system design. In *IEEE International Conference on Robotics and Automation*, Atlanta, GA, March, 600–608.
- Santaló, L. A. 1976. Integral geometry and geometric probability. In *Encyclopedia of Mathematics and Its Applications*. Vol. 1. Reading, MA: Addison-Wesley.
- Shirai, M., and Saito, A. 1989. Parts supply in Sony's general-purpose assembly system "SMART". *Jpn. J. Adv. Automation Tech.* 1:108–111.
- Singer, N. C., and Seering, W. P. 1987. Utilizing dynamic stability to orient parts. *Journal of Applied Mechanics* 54:961–966.
- Skiena, S. S. 1989. Problems in geometric probing. *Algorithmica* 4:599–605.
- Swanson, P. J., Burridge, R. R., and Koditschek, D. E. 1995. Global asymptotic stability of a passive juggler: A parts

- feeding strategy. In *IEEE International Conference on Robotics and Automation*, Nagoya, Japan, 1983–1988.
- Taylor, R. H., Mason, M. T., and Goldberg, K. Y. 1988. Sensor-based manipulation planning as a game with nature. In *Robotics Research: The Fourth International Symposium*, R. C. Bolles and B. Roth, eds., 421–429. Cambridge, MA: MIT Press.
- van der Stappen, A. F., Goldberg, K. Y., and Overmars, M. 1996. Geometric eccentricity and the complexity of manipulation plans. Technical report UU-CS-1996-49, Department of Computer Science, Utrecht University. To appear in *Algorithmica*.
- Wallack, A. S., Canny, J. F., and Manocha, D. 1993. Object localization using crossbeam sensing. In *IEEE Interna-*

- tional Conference on Robotics and Automation, Atlanta, GA, May, 1:692–699.
- Wiegley, J., Goldberg, K. Y., Peshkin, M., and Brokowski, M. 1996. A complete algorithm for designing passive fences to orient parts. In *IEEE International Conference* on Robotics and Automation, Minneapolis, MN, April, 1133–1139.
- Zhuang, Y., Wong, Y.-C., and Goldberg, K. Y. 1994. On the existence of modular fixtures. In *IEEE International Conference on Robotics and Automation*, San Diego, CA, May, 543–549.
- Zumel, N. B., and Erdmann, M. A. 1997. Nonprehensile manipulation for orienting parts in the plane. In *IEEE International Conference on Robotics and Automation*, Albuquerque, NM, April, 2433–2445.