Parts Orienting by Push-aligning

Srinivas Akella Matthew T. Mason

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Abstract

Programmable parts orienting is an important capability for flexible automation systems. Here we study how a part grasped in an unknown orientation by a force-controlled robot can be oriented by a sequence of push-align actions against a wall followed by sensor measurements of the distance from the grasp point to the wall. This paper concentrates on three issues: planning a sequence of actions to orient a part, exploring design changes that enable the part to be oriented in fewer steps, and the effect of shape uncertainty, due to manufacturing tolerances, on part orientability.

1 Introduction

Programmable automation is important in achieving flexible manufacturing and shortening product development cycles. A necessary capability is parts orienting, where parts in a bin or on a conveyor are to be oriented. We would like programmable parts orienting systems that automatically determine a sequence of actions to be executed by fence aligners and grippers, and use sensor data from beam sensors and distance sensors to orient a part. Since these tasks are repeated thousands of times, reducing the number of steps to orient a part can significantly reduce costs.

We are interested in the class of manipulation tasks where forces on a part act about a point with measurable location, and the results of a manipulation operation are determined by shape interactions. In particular, we explore the use of pushing actions coupled with distance measurements to orient polygonal parts grasped by a robot at a known point in an unknown initial orientation. When the part is grasped by a force-controlled robot with the compliance center at the grasp point, we can determine its orientation by performing a sequence of pushing operations against a wall and measuring the distance of the grasp point to the wall. We explore this *parts orienting by push-aligning* task both as a planning problem of how to select an appropriate sequence of actions, and as a design problem of where to locate the grasp point for optimal orientability. Further, we investigate the effect of shape uncertainty on part orientability.

1.1 Example tasks

Consider a parts feeding robot that uses an overhead camera to determine the orientations of parts as it picks them up from a conveyor. Parts in stable orientations with identical horizontal silhouettes cannot be oriented by the vision system (Figure 1). If the robot is force-controlled and the part has a uniform vertical cross-section, a sequence of pushing actions with the compliance center at the grasp point can determine the part orientation. These actions can be performed by the robot en route to the parts pallet.

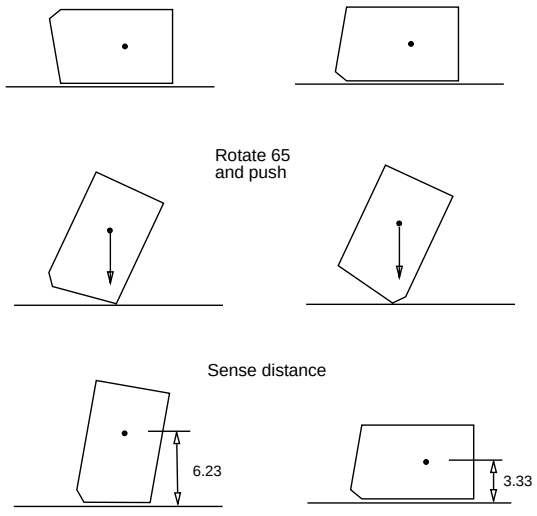

Figure 1: Two stable orientations of the part (shown in side view) cannot be distinguished by an overhead camera. A single push-align operation, consisting of a rotation and a compliant push, followed by a sensor reading, can determine the part orientation. The black dot indicates the grasp point.

A more complex orienting problem occurs when a 3-d part has a varying cross-section so different cross-sections can simultaneously contact the aligning surfaces; contact forces can no longer be treated as occur-

ring in the plane. A part grasped in a robot's fingers that is compliantly rotated by pushing it against a fixture is another example where forces act about a point and compliant motions can orient the part. The example of Figure 1 requires a plan to orient the two cross-sectional shapes that are mirror images of each other. As a step towards orienting parts with different shapes, in this paper we focus on the task of orienting a single planar object grasped at a known point in an unknown orientation.

2 Orienting by Push-aligning

We consider here the problem of orienting a planar object, grasped at a known point in an unknown orientation, by pushing it against an aligning wall. The object is defined to be oriented when a known edge is aligned with the wall. The grasp point is assumed to be determined from a feature of the part, for example, a peg that triggers a beam sensor on a conveyor. Using a force-controlled robot, we specify the compliance center to be at the grasp point; the object is held so it does not rotate relative to the gripper.

The goal is to orient the object through a sequence of *push-align operations*; a push-align operation consists of rotating the object by a chosen angle and pushing it in a direction normal to the wall to align it, followed by a measurement of the distance from the peg to the wall. The object rotates about the compliance center at the peg until an edge is aligned with the wall. The peg location determines which object edges are stable; a stable edge is an edge that becomes stably aligned with the wall when pushed against it. Since the wall is at a known location relative to the robot, the distance of the peg from the wall after a push can be determined from the gripper position. This inexpensive sensor data provides incomplete information on the object orientation.

We make the following assumptions:

- All objects are polygons, and have a feature, such as a peg or a shaft hole, at which they are grasped.
- The compliance center is located at the grasp point, referred to as the *peg* for convenience.
- All motions are in the plane and are quasi-static.
- The robot pushes the object in a direction normal to the wall.

To determine the result of a push-align operation, we follow Goldberg [8] in using the *radius function*. The radius function (Figure 2) describes the distance from a point in the polygon (the peg) to a tangent line to the polygon at a given angle (the wall). To predict the final orientation of the polygon when pushed against a wall in a given initial orientation, we note that all initial orientations between two consecutive

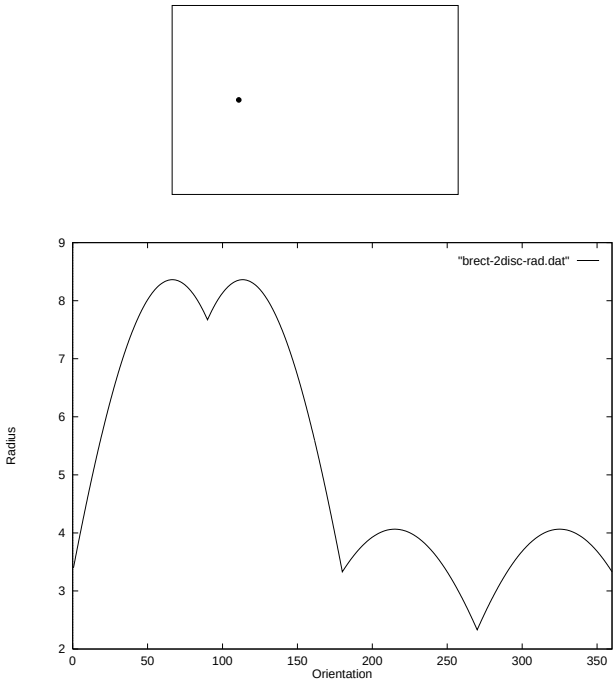

Figure 2: The radius function for the rectangle with its peg position indicated by the black circle.

local maxima of the radius function map to their enclosed local minimum. This resulting minimum radius is the perpendicular distance of the peg from the wall and the corresponding angle provides the orientation that results from the push-align operation. In the following discussion, the resulting radius and the peg-to-edge (perpendicular) distance are equivalent.

The *push-sense function* (Figure 3) is another representation of the result of a push-align operation; it describes the radius resulting from a push for a given initial orientation of the object relative to the wall. If the resulting radii are all unique and can be distinguished in the presence of sensor noise, the orientation of the object can be determined in a single step. However, if some of the resulting radii cannot be distinguished by the sensor, a sequence of steps may be necessary to orient the object.

Our goals are to find plans to orient parts, and to minimize the number of steps to orient them. This leads to the following two problems:

1. **The planning problem:** Given a part with a specified peg position, can we generate a plan to orient it? How many steps are required to orient the part?
2. **The design problem:** Given the design freedom to choose the peg location on a part, which peg locations require the least number of operations to orient the part?

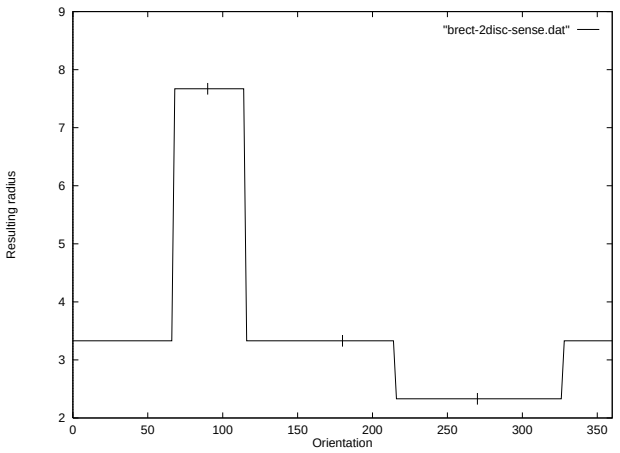

Figure 3: The push-sense function for the rectangle of Figure 2. The resultant orientation on a step is indicated by a tick mark.

2.1 Related Work

Boothroyd *et al.* [1] describe parts feeding and orienting devices for automated assembly. Whitney [20] describes the use of compliance centers in assembly tasks. Mason [12] studied pushing in manipulation operations and derived a rule to predict the rotation direction of a pushed object. Using these results, Mani and Wilson [11] developed a planner to use pushing actions to orient polygonal parts and Brost [2] planned single-step grasps of polygonal parts with a parallel-jaw gripper. Peshkin and Sanderson [14] considered part orienting by a sequence of oriented fences. Goldberg [8] developed a planner for sensorless orientation by grasping.

Grossman and Blasgen [9] used a vibratory tray to bring a part to a finite set of orientations which were then discriminated using a probe. Taylor *et al.* [19] studied planning of sensor-based manipulation strategies using AND/OR search. Rao and Goldberg [16] investigate recognition and orientation of objects by frictionless parallel-jaw grasping coupled with jaw diameter sensing. Canny and Goldberg [4] advocate the use of simple sensors and actuators in industrial tasks.

Natarajan [13] focused on computational issues in automated design of sensorless parts feeders. Erdmann [7] describes a procedure to design sensors based on the actions. Caine [3] considers the design of interacting part shapes to constrain motion and applies it to a vibratory bowl feeder track.

Requicha [17] discusses issues of tolerancing for part design and manufacture. Donald [6] treated model errors by extending the configuration space to include dimensional variations.

3 The planning problem: Generating orientation plans

We show that it is possible to orient parts by sensorless and sensor-based push-align operations, and determine the number of steps required in each case.

3.1 Sensorless orientation plans

To orient the part without sensors, we must find a sequence of push-align actions that brings all initial orientations to the same final orientation. The push function [8] is another representation of the push-align operation that describes the resulting object orientation as a function of its initial orientation. Goldberg [8] developed a backchaining algorithm for sensorless orienting of a polygon by frictionless parallel-jaw grasping. Due to the similarity between the push-align task and the grasping task, the backchaining algorithm can be applied to the push function to derive a minimum length sensorless plan for orienting the part by push-aligning. Chen and Ierardi [5] showed this algorithm is guaranteed to find a solution of $O(n)$ steps, where n is the number of stable edges of the object.

3.2 Sensor-based orientation plans

We now consider generating an orientation plan when the distance of the peg from the wall is measured after every push. When the peg has a unique distance to every edge, a single push-align operation determines the object orientation. Otherwise, we need a plan consisting of a conditional sequence of push-align operations to orient the object; branching during execution occurs based on the sensor value. We assume that there is at least one peg-to-edge distance with a unique sensor value; otherwise the part may be orientable only up to symmetry.

A push-align operation is indexed by the angle the object is rotated through before being pushed. The push-align actions can be divided into equivalence classes such that all member actions of a class result in the same orientation. The equivalence classes are found by moving a copy of the radius function past a stationary copy, and noting all angle ranges over which a minimum lies between two successive maxima. A representative action is selected for each class, taking care that it is deterministic.

Search procedure: To find an orienting plan, we perform a breadth-first search of an AND/OR tree ([18]). A node in the tree contains the set of orientations consistent with the push-align operations along the path to the node. An operation corresponds to a push-align action followed by a sensor reading. All links are AND links; the AND link from a node for a given operation points to a set of child nodes, each

of which contains a set of orientations indistinguishable from their sensor values. A node with a single orientation is a goal node.

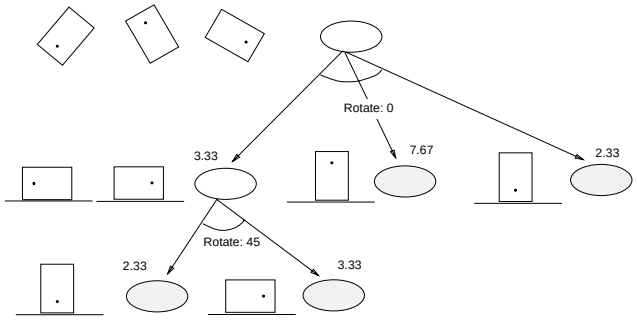

Figure 4: A plan to orient the rectangle of Figure 2. The sensor value shown at a node indicates the peg to wall distance for the possible orientations at that node. Goal nodes, shown shaded, have a single orientation.

Search begins at the root node, at which all object orientations are possible. The first push-align operation results in a set of nodes corresponding to the indistinguishable sets of stable orientations. For each non-goal node, the search proceeds in a breadth-first manner by applying each representative operation, and generating the corresponding set of child nodes. Whenever a goal node is found, this information is propagated up the search tree. A node is placed on the goal path when all its child nodes from an operation are on the goal path. Search terminates when the root node is placed on the goal path. The plan recovered from the goal path corresponds to a conditional sequence of operations to determine object orientation; the sensor values determine branching (Figure 4). The search is guaranteed to terminate if the set of operations can distinguish any two stable orientations.

Number of operations required to orient an object: The worst-case number of operations required to orient an object determines the depth of the search tree. Consider a representation of the push-sense function as a deterministic finite state machine. Each step of the push-sense function can be viewed as a state encoded by its height, angle range, and resultant orientation; the states have a cyclic arrangement, and only the step height can be sensed. Then identifying a sequence of operations to determine the object orientation is the same as identifying an adaptive homing sequence ([10]) for this finite state machine. Since the operations guarantee any state can be reached from any other state, in the worst case it takes a sequence of m operations to determine the machine state, where m is the maximum number of indistinguishable states. So the number of steps to orient a part is never greater than the maximum number of indistinguishable peg-to-edge distances. We conjecture that the problem of finding the shortest sequence of operations to orient a part is NP-complete.

4 The design problem: Where do we place the peg?

We now explore changes to task geometry that enable the object to be oriented in fewer steps. That is, where do we place the compliance center to orient the part efficiently? If every edge of the object is at a unique distance from the peg, the robot can determine the object orientation from sensor data after a single push. So, can we determine peg positions that enable single-step orientation of an object?

First consider an object whose edges are all stable. To differentiate two edges of the object in the presence of sensor noise, their peg-to-edge distances should differ by an amount greater than the sensor noise; this is the *distance constraint*. For two edges i and j with peg-to-edge perpendicular distances d_i and d_j, and a maximum sensor noise of ϵ_s, the distance constraint for the two edges is $|d_i - d_j| > \epsilon_s$. Any region in the polygon interior that satisfies the distance constraints for all pairs of stable edges is a peg placement region with unique peg-to-edge distances; a single push-align operation determines the object orientation. The union of all regions in which all n stable edges can be distinguished is the n-discriminating region. See Figure 5 for an example.

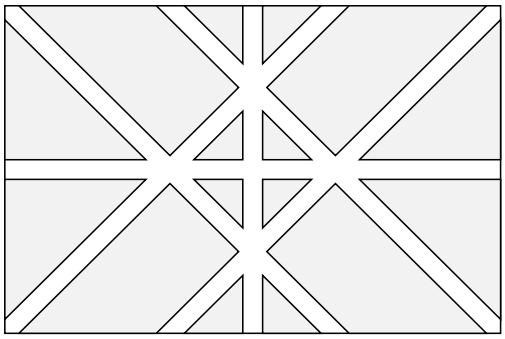

Figure 5: The n-discriminating region of the rectangle is shown shaded. ϵ_s is 2% of the longest object edge.

Now consider an object with unstable edges. Each edge is stable for a region of peg positions; this *edge stability region* consists of points in the object interior that lie between the inward normals to the edge at its vertices. The regions obtained from the planar subdivision ([15]) of the edge stability regions are the *edge-subset stability regions*; a subset of the edges are stable for each such region. Within each edge-subset stability region, we generate the distance constraints for all

pairs of member stable edges, and find the subregions that satisfy the distance constraints. The union of all these subregions is the n-discriminating region.

There is a hierarchy of discriminating regions based on the number of uniquely distinguishable stable edges. In a k-discriminating region, at least k edges can be distinguished from their peg-to-edge distances (Figure 6). Just as an n-discriminating region guarantees a single-step plan, a k-discriminating region guarantees a plan of $(n-k)$ steps or less, for $k < n$. As k decreases, the size of the region increases, but so does the worst-case plan length. For $k \geq 1$, the object can always be oriented.

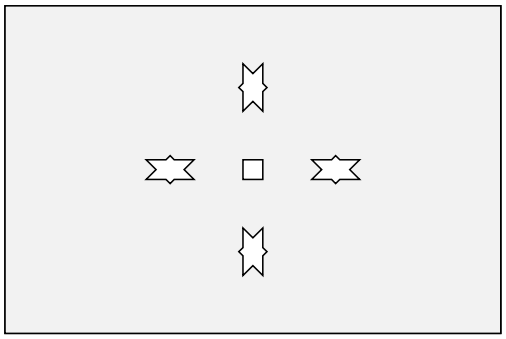

Figure 6: The 2-discriminating region of the rectangle. ϵ_s is 2% of the longest edge.

5 Shape Uncertainty

Parts manufactured to tolerances have variations in shape. We wish to find plans that can orient the range of part shapes permitted by bounded shape uncertainty; here we present some preliminary results.

5.1 Shape uncertainty model

We model shape uncertainty as follows (Figure 7):

- There is uncertainty in peg and vertex positions; we consider only convex polygons.
- The peg lies inside a circle of radius r_p centered at the nominal peg location.
- Each vertex lies inside a circle of radius r_v centered at the nominal vertex location. The vertex uncertainty circles do not intersect.
- The actual object edges are straight lines between the actual vertex positions.

5.2 Effect of shape uncertainty

Peg and vertex position uncertainties lead to uncertainties in the peg-to-edge distances, edge orientations, and transition angles between edges. These cause uncertainties in the step heights, step widths,

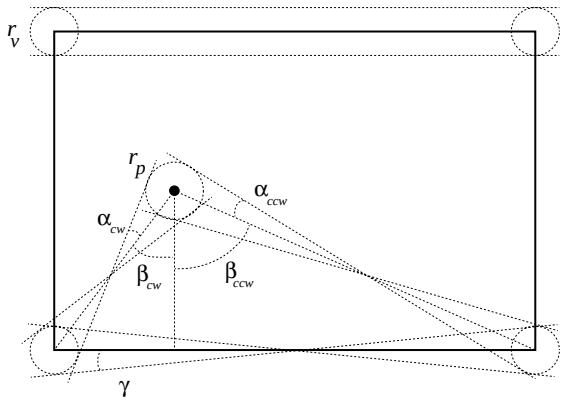

Figure 7: Shape uncertainty: The peg and vertex positions lie inside the uncertainty circles at their nominal locations. Step width ω is $(\beta_{cw} + \beta_{ccw} \pm \alpha_{cw} \pm \alpha_{ccw})$, transition angle uncertainties are $\pm\alpha_{cw}$ and $\pm\alpha_{ccw}$, edge orientation uncertainty is $\pm\gamma$, and edge position uncertainty is $\pm r_v$.

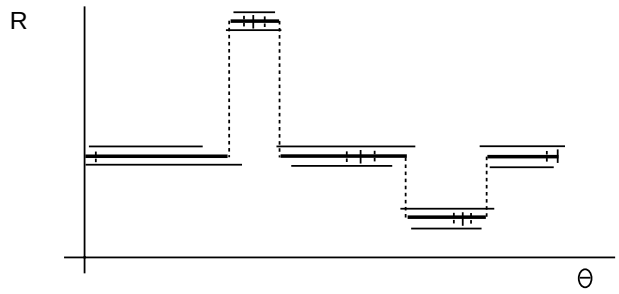

Figure 8: Push-sense function with uncertainty. Step height uncertainty is $\pm(r_p + r_v)$, transition angle uncertainties are $\pm\alpha_{cw}$ and $\pm\alpha_{ccw}$, resultant orientation uncertainty is $\pm\gamma$.

and resultant orientations in the push-sense function (Figure 8). To determine when it is possible to orient a part with shape and sensor uncertainty, we ask the following questions:

1. *Are there peg placement regions for single-step orienting with shape and sensor uncertainty?*
For a single-step plan to exist, the peg-to-edge distances of the stable edges should be distinguishable despite sensor noise and uncertainties in peg and vertex positions. So the distance constraint for edges i and j is now given by $|d_i - d_j| > (\epsilon_s + r_p + r_v)$. Each edge-subset stability region is shrunk where it intersects the uncertainty expanded edge stability regions of unstable edges to ensure such edges cannot be stable. We then determine the discriminating region of this shrunken edge-subset stability region using the distance constraints for the stable edges. The union of these discriminating regions over all edge-subset sta-

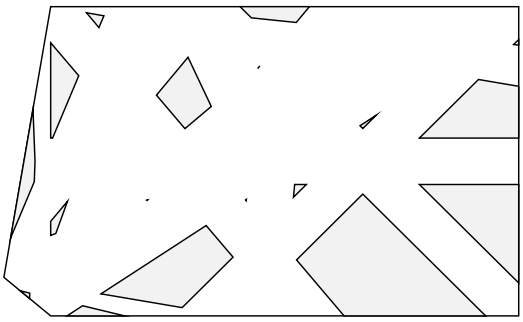

Figure 9: n-discriminating region with shape and sensor uncertainty. ϵ_s, r_p, and r_v are 2%, 1%, and 2% of the longest edge.

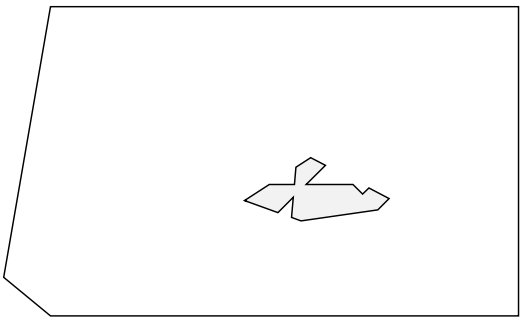

Figure 10: Peg placement region that guarantees existence of multi-step plans. ϵ_s, r_p, and r_v are 2%, 1%, and 2% of the longest edge, $k = 2$, $\beta_{min} = 18$, and $\alpha_{max} = 5$.

bility regions is the n-discriminating region (Figure 9). Peg placement in this region provides a single-step orientation plan.

2. *Can multi-step deterministic plans exist with uncertainty in the push-sense function?*
Angle uncertainties in the push-sense function make some actions non-deterministic. To find deterministic plans, we identify deterministic actions that permit any stable orientation to be reached from any other stable orientation, and enable any set of states with indistinguishable sensor values to be distinguished. The deterministic action ranges are obtained by shrinking the nominal action range for uncertainty. The maximum angle uncertainty ρ_{max} is $(\gamma_{max} + \alpha_{max})$. When the maximum number of indistinguishable stable states is m, and the smallest step semiwidth is β_{min}, if $\beta_{min} > m\rho_{max}$, a deterministic plan exists.

3. *Can we find peg placement regions that permit multi-step plans in the presence of uncertainty?*
A multi-step plan is guaranteed to exist despite uncertainty when there is a non-null intersection of a k-discriminating region with the peg placement region that provides the desired uncertainty bounds in the push-sense function. Within each edge-subset stability region, the bounded uncertainty region exists when the following conditions are met.

(a) Only edges in the edge-subset are stable: We want regions in which the stable edges are guaranteed to be stable and the unstable edges are guaranteed to be unstable. From the uncertainty shrunken edge stability regions of the stable edges and the uncertainty expanded edge stability regions of the unstable edges, we compute the regions in which only the desired set of edges is stable.

(b) The step semiwidths in the push-sense function have minimum magnitude β_{min}: The step semiwidths β_{cw} and β_{ccw} (Figure 7) depend on the peg location relative to the corresponding stable edge. The region of peg positions in the polygon interior bounded by two lines at the edge vertices, each making an inward angle to the normal of β_{min}, satisfies this constraint after it is shrunk for uncertainty.

(c) The transition angle uncertainties have maximum magnitude α_{max}: When the peg is near a vertex, small positional uncertainties of the peg or vertex cause large uncertainties in the transition angle between edges. The transition angle uncertainty exceeds the maximum transition angle uncertainty α_{max} within a circular sector at the vertex of radius $(r_p + r_v)/sin(\alpha_{max})$. The polygon interior region excluding this circular sector expanded for uncertainty guarantees bounded transition angle uncertainty for the vertex.

Within each edge-subset stability region, the intersection of the above regions generated for every vertex and stable edge gives the region that bounds angular uncertainties in the push-sense function. For each edge-subset stability region, we compute the intersection of this bounded uncertainty region with the corresponding k-discriminating region; the union of all these regions is the region of peg placements that permit multi-step orienting of the object with shape and sensor uncertainty.

6 Conclusion

A study of the relationship between actions that modify part configurations and sensory operations that provide limited information on part configurations is important for parts orienting tasks. We have studied a simple task of orienting parts by a sequence of push-align actions and distance measurements. We have seen that both sensorless and sensor-based plans exist, and that the sensor-based plans require fewer steps to orient a part. We have also explored the use of design freedom in locating the compliance center to generate single-step orientation plans. Under certain

conditions, the planning and design solutions can be extended to handle shape uncertainty.

The push-align task has a structure similar to the task of orienting objects by frictionless parallel-jaw grasping using a jaw diameter sensor [16]. We wish to exploit these similarities in the context of finding shortest plans, and orienting parts with different shapes. We would like to extend this work to handle parts with curved edges, and explore part orienting when the grasp point is unknown, as when parts are picked up by a vacuum gripper.

Acknowledgments

This work has benefited from discussions with Mike Erdmann, Ken Goldberg, Reid Simmons, Dafna Talmor, and Manipulation lab members. Mike Erdmann generously provided the polygon intersection code and Yan-bin Jia enhanced it. Discussions with Randy Brost and Ken Goldberg got us started on the shape uncertainty problem. This research was supported by NSF Grant No. IRI-9113208.

References

[1] G. Boothroyd, C. Poli, and L. E. Murch. *Automatic Assembly*. Marcel Dekker, 1982.

[2] R. C. Brost. Automatic grasp planning in the presence of uncertainty. *International Journal of Robotics Research*, 7(1), 1988.

[3] M. Caine. The design of shape interactions using motion constraints. In *1994 IEEE International Conference on Robotics and Automation*, 1994.

[4] J. F. Canny and K. Y. Goldberg. RISC for industrial robotics: Recent results and open problems. In *1994 IEEE International Conference on Robotics and Automation*, 1994.

[5] Y.-B. Chen and D. J. Ierardi. Oblivious plans for orienting and distinguishing polygonal parts. In *The 4th Canadian Conference on Computational Geometry*, 1992.

[6] B. R. Donald. Planning multi-step error detection and recovery strategies. *International Journal of Robotics Research*, 9(1), 1990.

[7] M. A. Erdmann. Understanding action and sensing by designing action-based sensors. In *Sixth International Symposium on Robotics Research*, 1993.

[8] K. Y. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10, 1993.

[9] D. D. Grossman and M. W. Blasgen. Orienting mechanical parts by computer-controlled manipulator. *IEEE Transactions on Systems, Man, and Cybernetics*, 5(5), 1975.

[10] Z. Kohavi. *Switching and Finite Automata Theory*. McGraw-Hill, 1978.

[11] M. Mani and W. Wilson. A programmable orienting system for flat parts. In *North American Manufacturing Research Institute Conference XIII*, 1985.

[12] M. T. Mason. Mechanics and planning of manipulator pushing operations. *International Journal of Robotics Research*, 5(3), 1985.

[13] B. K. Natarajan. Some paradigms for the automated design of parts feeders. *International Journal of Robotics Research*, 8(6), 1989.

[14] M. A. Peshkin and A. C. Sanderson. Planning robotic manipulation strategies for workpieces that slide. *IEEE Journal of Robotics and Automation*, 4(5), 1988.

[15] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

[16] A. S. Rao and K. Y. Goldberg. Shape from diameter: Recognizing polygonal parts with a parallel-jaw gripper. *International Journal of Robotics Research*, 13(1), 1994.

[17] A. A. Requicha. Toward a theory of geometric tolerancing. *International Journal of Robotics Research*, 2(4), 1983.

[18] E. Rich and K. Knight. *Artificial Intelligence*. McGraw-Hill, 1991.

[19] R. H. Taylor, M. T. Mason, and K. Y. Goldberg. Sensor-based manipulation planning as a game with nature. In *Fourth International Symposium on Robotics Research*, 1987.

[20] D. E. Whitney. Quasi-static assembly of compliantly supported rigid parts. *ASME Journal of Dynamic Systems, Measurement, and Control*, 104, 1982.